Implementation of Bourbaki's Elements of Mathematics in Coq: Part Two Ordered Sets, Cardinals, Integers

José Grimm

ISSN 0249-6399 ISRN INRIA/RR--7150--FR+ENG

RESEARCH REPORT
N° 7150
December 2009
Project-Team Marelle

Implementation of Bourbaki's Elements of Mathematics in Coq: Part Two Ordered Sets, Cardinals, Integers

José Grimm*

Project-Team Marelle

Research Report n° 7150 — version 9 — initial version December 2009 — revised version November 2017 — 764 pages

Abstract: We believe that it is possible to put the whole work of Bourbaki into a computer. One of the objectives of the Gaia project concerns homological algebra (theory as well as algorithms); in a first step we want to implement all nine chapters of the book Algebra. But this requires a theory of sets (with axiom of choice, etc.) more powerful than what is provided by Ensembles; we have chosen the work of Carlos Simpson as basis. This reports lists and comments all definitions and theorems of the Chapter "Ordered Sets, Cardinals, Integers".
Version 6 includes the Veblen hierarchy of ordinals, the Schütte function psi, and a bit of theory of models. Version 7 includes rational and real numbers. Versions 8 and 9 include more theorems about ordinal numbers. Version 9 includes Sperner's theorem, and corrects a mistake in the size of one. The code (including some exercises) is available on the Web, under http://www-sop.inria.fr/marelle/gaia.

Key-words: Gaia, Coq, Bourbaki, orders, cardinals, ordinals, integers

Work done in collaboration with Alban Quadrat, started when the author was in the Apics Team.

* Email: Jose.Grimm@inria.fr

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Implémentation de la théorie des ensembles de Bourbaki dans Coq partie 2 Ensembles Ordonnés, cardinaux, nombres entiers

Résumé : Nous pensons qu'il est possible de mettre dans un ordinateur l'ensemble de l'œuvre de Bourbaki. L'un des objectifs du projet Gaia concerne l'algèbre homologique (théeorie et algorithmes); dans une première étape nous voulons implémenter les neuf chapitres du livre Algèbre. Au préalable, il faut implémenter la théorie des ensembles. Nous utilisons l'Assistant de Preuve Coq; les choix fondamentaux et axiomes sont ceux proposés par Carlos Simpson. Ce rapport liste et commente toutes les définitions et théorèmes du Chapitre "Ensembles ordonnés, cardinaux, nombres entiers". Une partie des exercises a été résolue. La version 9 de ce document décrit la bibliothèque à la fin de l'année 2017. Le code est disponible sur le site Web http://www-sop.inria.fr/marelle/gaia.

Mots-clés : Gaia, Coq, Bourbaki, ordre, cardinaux, ordinaux, entiers

Chapter 1

Introduction

1.1 Objectives

Our objective (it will be called the *Bourbaki Project* in what follows) is to show that it is possible to implement the work of N. Bourbaki, "Éléments de Mathématiques"[5], into a computer, and we have chosen the Coq Proof Assistant, see [21, 3]. All references are given to the English version "Elements of Mathematics"[4], which is a translation of the French version (the only major difference is that Bourbaki uses an axiom for the ordered pair in the English version and a theorem in the French one). We start with the first book: theory of sets. It is divided into four chapters, the first one describes formal mathematics (logical connectors, quantifiers, axioms, theorems). Chapters II describes sets, unions, intersections, functions, products, equivalences; Chapter III defines orders, integers, cardinals, limits. The last chapter describes structures. The first part of this report[11] describes Chapter I and Chapter II, we consider here Chapter III.

1.2 Content of this document

This document describes the code found in the files *set5.v*, *set6.v*, *set7.v*, *set8.v*, *set9.v*, and *set10.v*, corresponding to sections 1 to 6 of Chapter III. The first section describes order relations and associated properties (like upper bounds, greatest elements, increasing functions, order isomorphisms). The second section studies well-ordered sets, and introduces the notion of transfinite induction. We show Zermelo's theorem (which is equivalent to the axiom of choice). Section 3 defines cardinals, addition, multiplication and order on cardinals (a cardinal is a representative of a class of equipotent sets; this class is not a set, and the axiom of choice is required). Section 4 defines natural integers as cardinals x such that $x \neq x+1$. It introduces induction on natural integers, so that a natural integer is any cardinal obtained by applying a "finite" number of times $x \mapsto x+1$ to the empty set. More formally, if E is a set containing zero and stable by $x \mapsto x+1$, it contains all natural integers. If E is any set with cardinal x, the set $\mathrm{E} \cup \{\mathrm{E}\}$ has cardinal $x+1$. As a consequence, one can define a mapping $n \mapsto \mathrm{E}_n$ that associates to each natural number n a set E_n of cardinal n (by transfinite induction, one can do this for any cardinal n). This set E_n is called an ordinal in [14] (For Bourbaki, an ordinal is a representative of well-ordered sets). It allows one to define finite cardinals without the use of the axiom of choice. There is no set containing all cardinals (thus no set containing all ordinals) but given a cardinal (or ordinal) a, there is a set containing all cardinals (or ordinals) less than a, and it is well-ordered. Every finite ordinal is a natural integer; infinite ordinals possess strange properties (addition and multiplication are non-commutative) studied in the Exercises. Section 5 studies some properties of integers (for instance division, expansion to base b) and computes the number of elements of various sets (for instance the number of subsets of p elements of a set of n elements, the number of permutations, etc). Section 6 studies infinite sets. If there exists an infinite set, then there exists a set **N** containing all natural integers. An axiom is required in Bourbaki; in Coq, there is an infinite set, namely `nat`, and it is canonically isomorphic to the set of natural integers. We use this isomorphism in section 5 (for instance, the factorial function and binomial coefficient are defined by induction on the Coq type `nat`, then shown to satisfy the Bourbaki definitions). There are few infinite cardinals (i.e., for any cardinal x there is a cardinal y such that $y > x$, for instance 2^x, but one could add an axiom saying that $y > x$ implies $y \geq 2^x$), so that one gets result like: the number of permutations of E is the number of mapping $\mathbf{N} \mapsto \mathrm{E}$, it is also the number of orderings on E (see Exercices).

Section 6 defines direct limits and inverse limits. It is not yet implemented. There are many exercises, two third of them are solved.

In the current version of this document, we use von Neumann ordinals to define cardinals. This means that file *set7.v* contains the definition and basic properties of ordinal numbers (including comparison). An ordinal equipotent to its successor is called infinite; the least infinite ordinal is called ω (it exists since `nat` is infinite). The cardinal of a set is defined to be the least ordinal equipotent to it; a finite ordinal is a finite cardinal, so that $\mathbf{N} = \omega$ is the set of all integers. We moved the definition and basic properties of addition; multiplication and exponentiation from file *set7.v* into file *set8.v*.

Several additional files *sset11.v*, *sset12.v*, *sset13.v*, *sset14.v*, *sset15.v* *sset16.v*, *sset17.v* study properties of ordinal numbers (addition, multiplication, exponentiation, Cantor Normal Form) and also of infinite cardinals (cofinality, regular cardinals, inaccessible cardinals, the Generalized Continuum Hypothesis).

We also introduces the set **Z** in file *ssetz.v*, the set **Q** in files *ssetq1.v* and *ssetqq.v*, and the set **R** in file *ssetr.v*. These definitions in these sets are very different from those given by Bourbaki in other Books.

The reader is invited to read the introduction of the first part. It explains some implementation details (for instance, what is a set? what formulation of the axiom of choice is used?).

1.3 Terminology

Chapter III is much less formal that Chapter II. Let's for instance quote Definition 9 [4, p. 146]: "Two elements of a preordered set E are said comparable if the relation "$x \leq y$ or $y \leq x$" is true. A set E is said to be totally ordered if it is ordered and if any two elements of E are comparable. The ordering in E is then said to be a total ordering and the corresponding order relation a total order relation."

One has to understand this as follows. A preordered set is a pair (E, G) which satisfies the preorder condition. The notation $x \leq y$ stands for $(x, y) \in \mathrm{G}$. An ordered set is a preordered set where additional conditions are required for G. The ordering is G, the corresponding order relation is "$(x, y) \in \mathrm{G}$". By definition, E is uniquely determined by G. Instead of saying that E is totally ordered, one can say: G is a total ordering if it is an ordering and for every x and y in the substrate of G one has "$(x, y) \in \mathrm{G}$ or $(y, x) \in \mathrm{G}$". This non ambiguous, and will be our definition. The following sentence is ambiguous "The set **R** of real numbers is totally ordered", since the order is not specified. Note that a sentence like "($\mathbf{R}, \leq$) is totally ordered" is ambiguous since $\leq$ can denote any ordering. One must say "The set **R** of real numbers is totally ordered by the usual order on real numbers." We shall use the notation $x \leq_{\mathbf{R}} y$; an alternative would be $x \leq y \pmod{\mathbf{R}}$ as in [14].

Bourbaki says: "a well-ordered set is totally ordered". This is a short-hand for: for every (E, Γ), if Γ is a well-ordering on E, then Γ is a total ordering on E. It is impossible to say "for every equivalence relation R we have..." since relations cannot be quantified; there is only one theorem in E.II.6, the chapter on equivalence relations. It is of the form: a correspondence Γ between X and X is an equivalence if and only if... This might explain why Bourbaki defines an order as a correspondence, rather than a graph. As a consequence, there are few criteria (C59 to C63 define normal and transfinite induction).

1.4 Notations

The set of natural numbers is denoted **N** by Bourbaki. In the code it will be `Nat`, in order to distinguish it from the set `nat` of Coq integers (these two sets are naturally isomorphic). This is also the least infinite ordinal, usually written ω by `omega0` in this document. The cardinal product is sometimes denoted $\mathsf{P}_{\iota\in\mathrm{I}} \mathfrak{a}_\iota$.

A lemma whose name starts with `OS_` (respectively, `CS_` and `NS_`) says that some quantity is an ordinal number, a cardinal number, or a natural integer.

A lemma whose name ends with `R`, `S`, `A` or `T` says that some relation is reflexive, symmetric, antisymmetric, or transitive.

A lemma whose name ends with `C`, `A` `D`, or `I` says that an operation is commutative, associative, distributive or involutive.

The cardinal sum is denoted by `csum`, and properties of the sum are given in theorems starting with `csum`. Similarly, the ordinal product is denoted by `oprod`, and properties of the product are given in theorems starting with `oprod`. A suffix 2 is sometimes added in the case of a binary operation. Note that `csum_Cn` states commutativity of the cardinal sum in the case of arbitrary number of arguments.

A suffix `M` may indicate monotony; for instance `csum_Mlele` says how $a+b$ and $a'+b'$ compare when $a \leq a'$ and $b \leq b'$.

We start with set-theoretic notations.

```
Notation "a -s b" := (complement a b) (at level 50).
Notation "a -s1 b" := (a -s (singleton b))  (at level 50).
Notation "a \cup b" := (union2 a b) (at level 50).
Notation "a +s1 b" := (a \cup (singleton b)) (at level 50).
Notation "a \cap b" := (intersection2 a b) (at level 50).
Notation "a *s1 b" := (indexed a b) (at level 50).
Notation "A \times B" := (product A B) (at level 40).
Notation "\Po E" :=   (powerset E) (at level 40).

Notation J := Pair.pair_ctor.
Notation P := Pair.first_proj.
Notation Q := Pair.second_proj.
```

These complicated notations are described in the first part of the report.

```
Notation "{ 'inc' d , P }" :=
  (prop_inc1 d (inPhantom P))
  (at level 0, format "{ 'inc'  d ,  P }") : type_scope.
Notation "{ 'inc' d1 & d2 , P }" :=
  (prop_inc11 d1 d2 (inPhantom P))
  (at level 0, format "{ 'inc'  d1  &  d2 ,  P }") : type_scope.
Notation "{ 'inc' d & , P }" :=
  (prop_inc2 d (inPhantom P))
  (at level 0, format "{ 'inc'  d  & ,  P }") : type_scope.

Notation "{ 'when' d , P }" :=
  (prop_when1 d (inPhantom P))
  (at level 0, format "{ 'when'  d ,  P }") : type_scope.
Notation "{ 'when' d1 & d2 , P }" :=
  (prop_when11 d1 d2 (inPhantom P))
  (at level 0, format "{ 'when'  d1  &  d2 ,  P }") : type_scope.
Notation "{ 'when' d & , P }" :=
  (prop_when2 d (inPhantom P))
  (at level 0, format "{ 'when'  d   & ,  P }") : type_scope.
Notation "{ 'when' : d , P }" :=
  (prop_when22 d (inPhantom P))
  (at level 0, format "{ 'when' :  d ,  P }") : type_scope.

Notation "{ 'compat' f : x / p >-> q }" :=
  (compatible_1 f  (fun x => p) (fun x => q))
  (at level 0,  f at level 99, x ident,
     format "{ 'compat'  f  : x  /  p  >->  q }") : type_scope.
Notation "{ 'compat' f : x / p }" :=
  (compatible_1 f  (fun x => p) (fun x => p))
  (at level 0,  f at level 99, x ident,
    format "{ 'compat'  f  : x  /  p }") : type_scope.
Notation "{ 'compat' f : x y / p >-> q }" :=
  (compatible_2 f  (fun x y => p) (fun x y => q))
  (at level 0,  f at level 99, x ident, y ident,
    format "{ 'compat'  f  : x  y  /  p  >->  q }") : type_scope.
Notation "{ 'compat' f : x y / p }" :=
  (compatible_2 f  (fun x y => p) (fun x y => p))
  (at level 0,  f at level 99, x ident, y ident,
    format "{ 'compat'  f  : x  y  /  p }") : type_scope.
Notation "{ 'compat' f : x & / p >-> q }" :=
  (compatible_3 f  (fun x => p) (fun x => q))
  (at level 0,  f at level 99, x ident,
     format "{ 'compat'  f  : x  &  /  p  >->  q }") : type_scope.
Notation "{ 'compat' f : x & / p }" :=
  (compatible_3 f  (fun x => p) (fun x => p))
  (at level 0,  f at level 99, x ident,
     format "{ 'compat'  f  : x  &  /  p }") : type_scope.
```

Notations for functions and functional objects.

```
Notation "f1 =1g f2" := (same_Vg f1 f2)
Notation "f1 =1f f2" := (same_Vf f1 f2)
Notation "f =1o g" := (forall x, ordinalp x -> f x = g x)
  (at level 70, format "'[hv' f '/ '  =1o  g ']'", no associativity).
Notation "f =2o g" := (forall x y, ordinalp x -> ordinalp y -> f x  y = g x y)
  (at level 70, format "'[hv' f '/ '  =2o  g ']'", no associativity).
```

```
Notation "x \cf y" := (composef x y) (at level 50).
Notation "x \cg y" := (composeg x y) (at level 50).
Notation "f1 \co f2" := (compose f1 f2) (at level 50).
Notation "x \cfP y" := (composablef x y) (at level 50).
Notation "f1 \coP f2" := (composable f1 f2) (at level 50).
Notation "f \ftimes g" := (ext_to_prod f g) (at level 40).
Notation "\Pof f" :=   (extension_to_parts f) (at level 40).
```

```
Notation "\0o" := ord_zero.
Notation "\0c" := card_zero.
Notation "\1o" := ord_one.
Notation "\1c" := card_one.
Notation "\2o" := ord_two.
Notation "\2c" := card_two.
Notation "\3c" := card_three.
Notation "\4c" := card_four.
Notation "\5c" := card_five.
Notation "\6c" := card_six.
Notation "\9c" := card_nine.
Notation "\10c" := card_ten.
```

The following notations introduce some alternate names.

```
Notation "\osup" := union (only parsing).
Notation "\csup" := union (only parsing).
Notation "\omega" := omega_fct.
Notation "\aleph" := omega_fct (only parsing).
```

Comparison of ordinals and cardinals

```
Notation "x <o y" := (ordinal_lt x y) (at level 60).
Notation "x <=o y" := (ordinal_le x y) (at level 60).
Notation "x <<o y" := (ord_negl x y) (at level 60).

Notation "x <=t y" := (order_type_le x y) (at level 60).
Notation "x <=O y" := (order_le x y) (at level 60).

Notation "x =c y" := (cardinal x = cardinal y)
Notation "x <=c y" := (cardinal_le x y) (at level 60).
Notation "x <c y" := (cardinal_lt x y) (at level 60).
Notation "x <=N y" := (Nat_le x y) (at level 60).
Notation "x <N y" := (Nat_lt x y) (at level 60).
```

Operations on cardinals and ordinals

```
Notation "x +c y" := (csum2 x y) (at level 50).
Notation "x *c y" := (cprod2 x y) (at level 40).
Notation "x ^c y" := (cpow x y) (at level 30).
Notation "x -c y" := (cdiff x y) (at level 50).
Notation "x %/c y" := (cquo x y) (at level 40).
Notation "x %%c y" := (crem x y) (at level 40).
Notation "x %|c y" := (cdivides x y) (at level 40).
Notation "x ^<c y" := (cpow_less x y) (at level 30).
Notation "m = n %c[mod d ]" := (m %%c d = n %%c d)
```

```
Notation "x +o y" := (osum2 x y) (at level 50).
Notation "x *o y" := (oprod2 x y) (at level 40).
Notation "x -o y" := (odiff x y) (at level 50).
Notation "x ^o y" := (opow x y) (at level 30).
Notation "x +#o y" := (natural_sum x y) (at level 50).

Notation "x ^O y" := (ord_powa x y) (at level 30).
Notation "x +t y" := (OT_sum2 x y) (at level 50).
Notation "x *t y" := (OT_prod2 x y) (at level 40).
```

These notations are used for **Z**.

```
Notation BZ_val := P (only parsing).
Notation BZ_sg := Q (only parsing).

Notation "\0z" := BZ_zero.
Notation "\1z" := BZ_one.
Notation "\2z" := BZ_two.
Notation "\3z" := BZ_three.
Notation "\4z" := BZ_four.
Notation "\1mz" := BZ_mone.
Notation "x <=z y" := (BZ_le x y) (at level 60).
Notation "x <z y" := (BZ_lt x y) (at level 60).
Notation "x +z y" := (BZsum x y) (at level 50).
Notation "x *z y" := (BZprod x y) (at level 40).
Notation "x -z y" := (BZdiff x y) (at level 50).
Notation "x %/z y" := (BZquo x y) (at level 40).
Notation "x %%z y" := (BZrem x y) (at level 40).
Notation "x %|z y" := (BZdivides x y) (at level 40).
```

These notations are used for **Q**.

```
Notation "\0q" := BQ_zero.
Notation "\1q" := BQ_one.
Notation "\2q" := BQ_two.
Notation "\3q" := BQ_three.
Notation "\4q" := BQ_four.
Notation "\1mq" := BQ_mone.
Notation "\2hq" := BQ_half.

Notation Qnum := P (only parsing).
Notation Qden := Q  (only parsing).
Notation "x <=q y" := (BQ_le x y) (at level 60).
Notation "x <q y" := (BQ_lt x y) (at level 60).
Notation "x +q y" := (BQsum x y) (at level 50).
Notation "x -q y" := (BQdiff x y) (at level 50).
Notation "x *q y" := (BQprod x y) (at level 40).
Notation "x /q y" := (BQdiv x y) (at level 40).
```

These notations are used for **R**.

```
Notation "\0r" := BR_zero.
Notation "\1r" := BR_one.
Notation "\2r" := BR_two.
Notation "\3r" := BR_three.
Notation "\4r" := BR_four.
```

```
Notation "\5r" := BR_five.
Notation "\1mr" := BR_mone.
Notation "\2hr" := BR_half.

Notation "x <=r y" := (BR_le x y) (at level 60).
Notation "x <r y" := (BR_lt x y) (at level 60).
Notation "x +r y" := (BRsum x y) (at level 50).
Notation "x -r y" := (BRdiff x y) (at level 50).
Notation "x *r y" := (BRprod x y) (at level 40).
Notation "x /r y" := (BRdiv x y) (at level 40).
Notation "x ^r y" := (BRnpow x y) (at level 30).
```

Other notations

```
Notation CNF_coefficients := CNF_exponents (only parsing).
Notation "\cf x" := (cofinality x) (at level 49).
Notation "x \Eq y" := (equipotent x y) (at level 50).
Notation "x \Is y" := (order_isomorphic x y) (at level 50).
```

1.5 Tactics

We give here the list of tactics that are defined in the files associated to this document . This is now the tactic that exploits properties of order relations.

```
Ltac order_tac:=
  match goal with
    | H1: gle ?r ?x _ |- inc ?x (substrate ?r)
      => exact: (arg1_sr H1)
    | H1: glt ?r ?x _ |- inc ?x (substrate ?r)
      => move: H1 => [H1 _] ; order_tac
    | H1:gle ?r _ ?x |- inc ?x (substrate ?r)
      => exact: (arf2_sr H1)
    | H1:glt ?r _ ?x |- inc ?x (substrate ?r)
      => move: H1 => [H1 _]; order_tac
    | H: order ?r, H1: inc ?u (substrate ?r) |- related ?r ?u ?u
      => apply/(order_reflexivity H)
    | H: order ?r |- inc (J ?u ?u) ?r
       => apply /(order_reflexivityP H)
    | H: order ?r |- gle ?r ?u ?u
       => apply /(order_reflexivityP H)
    | H1: gle ?r ?x ?y, H2: gle ?r ?y ?x, H:order ?r |-
        ?x = ?y => exact: (order_antisymmetry H H1 H2)
    | H:order ?r, H1:related ?r ?x ?y, H2: related ?r ?y ?x |- ?x = ?y
      => apply (order_antisymmetry H H1 H2)
    | H:order ?r, H1: inc (J ?x ?y) ?r , H2: inc (J ?y ?x) ?r |- ?x = ?y
      => apply (order_antisymmetry H H1 H2)
    | H:order ?r, H1:related ?r ?u ?v, H2: related ?r ?v ?w
      |- related ?r ?u ?w
      => apply (order_transitivity H H1 H2)
    | H:order ?r, H1:gle ?r ?u ?v, H2: gle ?r ?v ?w |- gle ?r ?u ?w
      => apply (order_transitivity H H1 H2)
    | H: order ?r, H1: inc (J ?u ?v) ?r, H2: inc (J ?v ?w) ?r |-
      inc (J ?u ?w) ?r
      => apply (order_transitivity H H1 H2)
```

```
    |  H1: gle ?r ?x ?y, H2: glt ?r ?y ?x, H: order ?r |- _
      => case: (not_le_gt H H1 H2)
    | H1:glt ?r ?x ?y, H2: glt ?r ?y ?x, H:order ?r |- _
      => move: H1 => [H1 _] ; case: (not_le_gt H H1 H2)
    | H1:order ?r, H2:glt ?r ?x ?y, H3: gle ?r ?y ?z |- glt ?r ?x ?z
       => exact: (lt_leq_trans H1 H2 H3)
    | H1:order ?r, H2:gle ?r ?x ?y, H3: glt ?r ?y ?z |- glt ?r ?x ?z
       => exact: (leq_lt_trans H1 H2 H3)
    | H1:order ?r, H2:glt ?r ?x ?y, H3: glt ?r ?y ?z |- glt ?r ?x ?z
       => exact: (lt_lt_trans H1 H2 H3)
    | H1:order ?r, H2:gle ?r ?x ?y |- glt ?r ?x ?y
       => split =>//
    | H:glt ?r ?x ?y |- gle ?r ?x ?y
       => by move: H=> []
end.
```

This is used for intervals.

```
Ltac zztac2 v  := set_extens v ;
   try (move/setI2_P=>[] /Zo_P [pa pb] /Zo_P [pc pd]; apply: Zo_i => //);
   try (move /Zo_P  => [ pa pb]; apply /setI2_P; split; apply: Zo_i => //).
```

Chapter 2

Order relations. Ordered sets

This chapter defines order relations and studies some properties of sets and subsets ordered by a relation. We define the notion of maximal element, greatest element, upper bound, least upper bound. Some ordered sets may be qualified as directed, lattice or totally ordered, or intervals. Functions weakly compatible with the order are called "increasing", and functions strongly compatible are called "order isomorphisms".

2.1 Definition of an order relation

In Bourbaki, there are two kinds of objects: sets and relations. For instance, $\emptyset$ and $\{\emptyset\}$ are two sets, while $\emptyset \subset \{\emptyset\}$ is a relation (that happens to be true), and $\{\emptyset\} \subset \emptyset$ is a relation that happens to be false. The objects $x \subset x$ and $(\forall x)(x \subset x)$ are true relations. The first one contains the letter x (which is a set); the second one does not contain x and is identical to $(\forall y)(y \subset y)$. In a context where x is not a constant, $x \subset x$ is equivalent to $(\forall x)(x \subset x)$, and one can deduce $\mathrm{E} \subset \mathrm{E}$ whenever E is a set (either a constant, a letter, or an expression with variables).

In our implementation of Bourbaki in COQ, we have much more types. $\emptyset$ is a set, and has type `Set`; $\emptyset \subset \{\emptyset\}$ is a relation and has type `Prop`; $x \mapsto \{x\}$ is of type `Set→Set`, in short `fterm`; $x \mapsto x \not\subset x$ is of type `Set→Prop`, in short `property`; $(x, y) \mapsto x \cup y$ is a shorthand for $x \mapsto (y \mapsto x \cup y)$ of type `Set→Set→Set`, in short `fterm2`; $(x, y) \mapsto x \subset y$ is of type `Set→Set→Prop`, in short `relation`. Bourbaki uses first order logic. This really means that in $x \mapsto \mathrm{P}$ or $(\forall x)\mathrm{P}$, the variable x is a set. For instance, "if P then 0 else 1" cannot be considered as a function of P.

Bourbaki says: «Let $\mathrm{R}\{x, y\}$ be a relation, x and y being distinct letters. R is said to be an order relation with respect to the letters x and y (or between x and y) if »

$$(\mathrm{R}\{x, y\} \text{ and } \mathrm{R}\{y, z\}) \implies \mathrm{R}\{x, z\}, \tag{2.1}$$

$$(\mathrm{R}\{x, y\} \text{ and } \mathrm{R}\{y, x\} \implies (x = y), \tag{2.2}$$

$$\mathrm{R}\{x, y\} \implies (\mathrm{R}\{x, x\} \text{ and } \mathrm{R}\{y, y\}). \tag{2.3}$$

Examples: «The relation of equality, $x = y$, is an order relation» (implicitly with respect to x and y, because x comes before y in the notation), «The relation $\mathrm{X} \subset \mathrm{Y}$ is an order relation between X and Y.» «If A is a set, the relation "$\mathrm{X} \subset \mathrm{Y}$ and $\mathrm{X} \subset \mathrm{A}$ and $\mathrm{Y} \subset \mathrm{A}$" is an order relation between subsets of A » (as A is qualified as "set", it is a constant, and the variables are X and Y). So, an order relation is given by a relation (that may depend on some letters a, x, X, whatever) and two such letters. Assume that the chosen letters are u and v. Then $\mathrm{R}\{0, 1\}$ is the relation obtained by replacing u by 0 and v by 1 (substitution is done in parallel so that $\mathrm{R}\{v,u\}$ is R with u and v exchanged). An order relation must satisfy the three relations (2.1), (2.2), (2.3). Relation (2.2) in the case of equality says: $x = y$ and $y = x$ implies $x = y$. To say that this is true means $(\forall x)(\forall y)(x = y \wedge y = x \implies x = y)$. The third example involves three variables A, X and Y; so that we have to choose two of them, for instance A and X and obtain a result of the form; for all Y (or for no Y, or for some Y) the relation is an order. See discussion on page 510. It can happen that R has less than two free variables; it can also happen that z is a free variable in R. In this case the letter z in (2.1) should be replaced by a letter that does not appear in R (and quantification is over the three letters); see 510 what can go wrong. In Chapter II, definition of an equivalence relation, Bourbaki explicitly states that z cannot appear in R.

Definitions. In the last chapter of the first part of this document, we studied equivalence relations, that were reflexive, symmetric and transitive. We also introduced the notion of a *preorder relation*, which is reflexive and transitive and of an *order relation*, which is reflexive, antisymmetric and transitive. Here "relation" means `relation` as explained above, so is a function with two arguments, and $\mathrm{R}(x,y)$ is the result of applying R to x as first argument and y as second argument. The set G of all pairs (x,y) that are related by the relation is called the *graph* of the relation when it exists; otherwise the relation is said to be "without graph". The set E of all x such that there is y such that (x,y) or (y,x) is in G is called the *substrate*. If all $x \in \mathrm{E}$ are related to themselves, the relation is called "reflexive on E"; if it is an order relation, it is called an order relation on E. A *preorder* or an *order* is a graph G such that the relation $(x,y) \in \mathrm{G}$ between x and y is a preorder or order relation. Such a relation has a graph. Any relation that has a graph is of this form. One writes generally $x \prec y$ for a preorder relation and $x \le y$ for an order relation.

Examples: $x = y$ and $x \subset y$ are order relations without graph (there is no set that contains all sets). If $x \le y$ is a preorder or an order relation, the *opposite* relation (the relation $(y,x) \mapsto x \le y$, denoted in general $y \ge x$) is of the same kind. If G is the graph of $x \le y$ then G^{-1} is the graph of $y \le x$; for this reason, G^{-1} is called the *opposite* graph of G.

```
Definition opposite_relation (r:relation) :=  fun x y => r y x.
Definition opp_order := inverse_graph.

Lemma equality_order_r: order_r (fun x y => x = y).
Lemma sub_order_r: order_r sub.
Lemma opposite_preorder_r r:  preorder_r r -> preorder_r (opposite_relation r).
Lemma opposite_order_r r:  order_r r -> order_r (opposite_relation r).
```

Consider a relation $x \le y$ and a set E. The set of all $(x,y) \in \mathrm{E} \times \mathrm{E}$ such that $x \le y$ is called the *graph of* $\le$ *on* E, and is sometimes denoted $\mathrm{E}_{\le}$. We have already shown that this is an order on E if the relation "$x \in \mathrm{E}$ and $y \in \mathrm{E}$ and $x \le y$" is an order relation. We prove here a variant.

```
Lemma order_from_rel1 r x:
  transitive_r r ->
  (forall u v, inc u x -> inc v x -> r u v -> r v u -> u = v) ->
  (forall u, inc u x -> r u u) ->
  order (graph_on r x).
```

If G is a graph, we sometimes write $x \le_{\mathrm{G}} y$ instead of $(x,y) \in \mathrm{G}$, and $x <_{\mathrm{G}} y$ as short for "$x \le_{\mathrm{G}} y$ and $x \ne y$". The Coq notations are '`gle G x y`' and '`glt G x y`'.

```
Definition gle r x y := related r x y.
Definition glt r x y := gle r x y /\ x <> y.

Lemma order_reflexivityP r: order r ->
   forall a, (inc a (substrate r) <-> gle r a a).
Lemma order_antisymmetry r a b:
  order r -> gle r a b -> gle r b a -> a = b.
Lemma order_transitivity r a b c:
  order r -> gle r a b -> gle r b c -> gle r a c.
Lemma lt_leq_trans r x y z:
  order r -> glt r x y -> gle r y z -> glt r x z.
Lemma leq_lt_trans r x y z:
  order r -> gle r x y -> glt r y z -> glt r x z.
Lemma lt_lt_trans r a b c:
  order r -> glt r a b -> glt r b c -> glt r a c.
Lemma not_le_gt r x y:
   order r -> gle r x y -> glt r y x -> False.
Lemma order_exten r r': order r -> order r' ->
  (forall x y, gle r x y <-> gle r' x y) -> (r = r').
```

Some properties of opposite order.

```
Lemma opp_gleP r x y: gle (opp_order r) x y <-> gle r y x.
Lemma opp_gltP r x y: glt (opp_order r) x y <-> glt r y x.
Lemma opp_osr r: order r -> order_on (opp_order r) (substrate r).
```

Inclusion suborder. The relation "$x \in \mathrm{A}$ and $y \in \mathrm{A}$ and $x \subset y$" is an order relation on A. If $\mathrm{A} = \mathfrak{P}(\mathrm{E})$, then $x \in \mathrm{A}$ can be replaced by $x \subset \mathrm{E}$.

```
Definition sub_order :=  graph_on sub.
Definition subp_order E := sub_order (\Po E).

Lemma sub_osr A: order_on (sub_order A) A.
Lemma subp_osr E: order_on (subp_order E) (÷Po E).
Lemma sub_gleP A u v:
  gle (sub_order A) u v <-> [/\ inc u A, inc v A & sub u v].
Lemma subp_gleP E u v:
  gle (subp_order E) u v <-> [/\ sub u E, sub v E & sub u v].
```

Extension ordering for functions. Let E and F be two sets, $\Phi(\mathrm{E},\mathrm{F})$ the set of functions from a subset of E to F, and $\mathrm{S}(f)$, $\mathrm{T}(f)$ and $\mathrm{G}(f)$, the source, target and graph of the function f. Since G is injective on Φ, the relation $\mathrm{G}(f) \subset \mathrm{G}(g)$ induces an order on Φ. Note that $\mathrm{G}(f) \subset \mathrm{G}(g)$ is the same as "g extends f", it is equivalent to $f(x) = g(x)$ for any x in the source of f.

```
Definition extension_order E F :=
  graph_on extends (sub_functions E F).

Lemma extension_osr E F:
  order_on (extension_order E F) (sub_functions E F).
Lemma extension_orderP E F f g:
  gle (extension_order E F) g f <->
  [/\ inc g (sub_functions E F), inc f (sub_functions E F)
    & extends g f].
Lemma extension_order_P1 E F f g:
```

```
  gle (extension_order E F) g f <->
  (inc g (sub_functions E F) /\ inc f (sub_functions E F)
    /\ sub (graph f) (graph g)).
Lemma extension_order_P2 E F f g:
  gle (opp_order (extension_order E F)) g f <->
  [/\ inc g (sub_functions E F), inc f (sub_functions E F)
    & sub (graph f) (graph g)].
Lemma extension_order_pr E F f g:
  gle (opp_order (extension_order E F)) f g ->
  {inc source f, f =1f g}.
```

Coarser partitions. Let E be a set. Recall that a partition ϖ of E is a set of sets whose union is E, the empty set is not in ϖ, the elements of ϖ are mutually disjoint. We shall denote by W_E the set of all partitions of E.

Note that $\varpi \subset \mathfrak{P}(E)$, and $\varpi \in \mathfrak{P}(\mathfrak{P}(E))$. Thus, $x \in \varpi$ implies $x \in \mathfrak{P}(E)$. We can consider the canonical injection $\varpi \to \mathfrak{P}(E)$; this is the function defined on ϖ that maps $x \in \varpi$ to itself. The graph of this function is a partition (in the sense that the union of the elements of the graph is E, and these elements are mutually disjoint).

```
Section Partition.

Definition partitions E :=
  Zo(\Po(\Po E)) (fun z => partition_s z E).

Definition part_fun p E := canonical_injection p (\Po E).

Lemma partition_set_in_PP p E:
  partition_s p E -> inc p (\Po (\Po E)).
Lemma partitionsP p E:
  inc p (partitions E) <-> partition_s p E.
Lemma pfs_f p E: partition_s p E -> function (part_fun p E).
Lemma pfs_V p E a: partition_s p E -> inc a p -> Vf (part_fun p E) a = a.
Lemma pfs_partition y x:
  partition_s y x ->  partition_w_fam (graph (part_fun y x)) x.
```

Recall that X is coarser than Y if for any $y \in Y$ there exists $x \in X$ such that $x \subset y$. We have already shown that this relation satisfies all the properties of an order; we have even found the greatest and least elements.

```
Definition coarser x := graph_on coarser_cs (partitions x).
Lemma coarser_gleP x y y':
  gle (coarser x) y y' <->
  [/\ partition_s y x, partition_s y' x & coarser_cs y y']
Lemma coarser_osr x: order_on (coarser x) (partitions x).
Lemma least_partition_is_least x y:
  nonempty x ->
  partition_s y x -> gle (coarser x) (least_partition x) y.
Lemma greatest_partition_is_greatest x y:
  partition_s y x -> gle (coarser x) y (greatest_partition x).
```

Let ϖ be a partition; consider the set formed of all $A \times A$ with $A \in \varpi$; let $\tilde{\varpi}$ be the union of all these sets. We pretend that this set is the graph of the equivalence associated to the partition. The relation "ϖ coarser than ϖ'" is equivalent to $\tilde{\varpi} \supset \tilde{\varpi}'$. Bourbaki says that this shows that coarser is an order. The nontrivial point is antisymmetry: we must show that $\tilde{\varpi} = \tilde{\varpi}'$ implies $\varpi = \varpi'$. This is a consequence of the fact that the sets $A \times A$ are mutually disjoint. If a and b are in the same element of ϖ and in A and B for ϖ', the pair (a, b) is in $\tilde{\varpi}$, hence in $\tilde{\varpi}'$, hence in $A \times A$ and $B \times B$, so that the intersection of A and B is not empty, and $A = B$.

```
Definition partition_relset_aux y x :=
  Zo (\Po (coarse x)) (fun z => exists2 a, inc a y & z = coarse a).
Definition partition_relset y x :=
  partition_relation (part_fun y x) x.

Lemma prs_is_equivalence y x:
  partition_s y x -> equivalence (part_relset y x).
Lemma partition_relset_pr1 y x a:
  partition_s y x ->
  inc a y -> inc (coarse a) (partition_relset_aux y x).
Lemma partition_relset_pr y x:
  partition_s y x ->
  partition_relset y x = union (partition_relset_aux y x).
Lemma sub_part_relsetX y x:
  partition_s y x -> sub (partition_relset y x) (coarse x).
Lemma partition_relset_order x y y':
  partition_s y x -> partition_s y' x ->
  (sub (partition_relset y' x)(partition_relset y x) <->
    gle (coarser x) y y').
Lemma part_relset_anti x y y':
  partition_s y x -> partition_s y' x ->
  (partition_relset y' x = partition_relset y x) ->
   y = y'.
```

The order structure. Let G be a set. We have shown that relation "$G \circ G = G$ and $G = G^{-1}$" is equivalent to say that G is the graph of an equivalence relation. We give here the corresponding property for an order. Let Δ be the diagonal of the substrate of G. If G is a graph, then G is a preorder if and only if $\Delta \subset G$ and $G \circ G \subset G$. These relations imply $G \circ G = G$. Antisymmetry is $G \cap G^{-1} \subset \Delta$. Reflexivity is also $\Delta \subset G^{-1}$, so that G is an order if and only $G \circ G = G$, and $G \cap G^{-1} = \Delta$. This is Proposition 1 of Bourbaki [4, p. 132].

The "order structure" on a set A is characterized by its object s, whose typification is $s \in \mathfrak{P}(A \times A)$ and the axiom is "$s \circ s = s$ and $s \cap s^{-1} = \Delta_A$". It follows that A is the substrate of s, so that s is an order on A.

```
Lemma preorder_prop1 g:
  sgraph g ->
  sub (diagonal (substrate g)) g -> sub (g \cg g) g ->
  g \cg g = g.
Lemma preorderP g: sgraph g ->
  (preorder g <-> (sub (diagonal (substrate g)) g /\ sub (g \cg g) g)).
Theorem orderP r:
  order r <->
    (r \cg r = r /\  r \cap (opp_order r) = diagonal (substrate r)).
Lemma order_structure s a:
  inc s (\Po (coarse a)) ->
  s \cg s = s ->  s \cap (inverse_graph s) = diagonal a ->
  a = substrate s.
```

2.2 Preorder relations

A non-trivial example of a preorder relation is the following: in the set of all coverings of a set E, the relation "R is coarser than R′" is reflexive and transitive, but neither symmetric nor antisymmetric. For instance, consider a covering R. Assume that there are two subsets X and Y of E such that $X \subset Y$, $Y \in R$, and $X \not\in R$. Let $R' = R \cup \{X\}$. This is a covering which is coarser than R. Moreover, our set X is chosen such that R is coarser than R′ and $R \neq R'$.

Here are some basic properties.

```
Lemma preorder_reflexivity r a:
  preorder r -> (inc a (substrate r) <-> gle r a a).
Lemma opposite_is_preorder1 r:
  preorder r -> preorder (opp_order r).
```

Equivalence associated to a preorder. The relation "$x \prec y$ and $y \prec x$" is an equivalence if $\prec$ is a preorder. Denote it by $\sim$. Then $x \prec y$ is compatible with $x \sim x'$ and $y \sim y'$. This means that if these three relations hold, then we have also $x' \prec y'$. If $\prec$ has a graph G, then $\sim$ has a graph, which is $G \cap G^{-1}$.

```
Definition equivalence_associated_o r := r \cap(opp_order r).
Definition symmetrize (r: relation) := fun x y => r x y  /\ r y x.

Lemma equivalence_preorder r:
  preorder_r r -> equivalence_r (symmetrize r).
Lemma compatible_equivalence_preorder r (s := symmetrize r):
  preorder_r r -> forall x y x' y', r x y -> s x x' -> s y y' -> r x' y'.
Lemma eao_P r x y:
  related (equivalence_associated_o r) x y <-> symmetrize (gle r) x y.
Lemma equivalence_preorder1 r:
  preorder r ->
  equivalence (equivalence_associated_o r).
Lemma equivalence_associated_o_sr r:
  preorder r ->
  substrate (equivalence_associated_o r) = substrate r.
Lemma compatible_equivalence_preorder1 r u v x y:
  preorder r -> related r x y ->
  related (equivalence_associated_o r) x u ->
  related (equivalence_associated_o r) y v->
  related r u v.
```

Order associated to a preorder. Let $\prec$ be a preorder on a set E, $\sim$ the equivalence associated to it, and π be the canonical projection $E \to E/\sim$. Given two objects of the form $u = \pi(x)$ and $v = \pi(y)$ in the quotient, we can compare u and v via $x \prec y$, this is independent of the representatives x and y. This relation has a graph, namely $S \circ G \circ S^{-1}$, where S is the graph of π. This set is also $(\pi \times \pi)\langle G \rangle$, where $\pi \times \pi$ maps $E \times E$ into $(E/\sim) \times (E/\sim)$. This relation is an order on the quotient; it is called the order relation *associated* with $\prec$.

```
Definition order_associated r :=
  let s := graph (canon_proj (equivalence_associated_o r)) in
  (s \cg r) \cg (opp_order s).

Lemma oap_graph r:
```

```
  sgraph (order_associated r).
Lemma compose3_relP s r u v:
  related ((s \cg r) \cg (opp_order s)) u v <->
  exists x y, [/\ related s x u, related s y v & related r x y].


Section OrderAssociated.
Variable (r:Set).
Hypothesis pr: preorder r.

Lemma oap_relP1 u v:
  related (order_associated r) u v <->
  [/\ inc u (quotient (equivalence_associated_o r)),
    inc v (quotient (equivalence_associated_o r)) ,
    exists x y, [/\ inc x u, inc y v & related r x y]].
Lemma oap_relP2 u v:
  related (order_associated r) u v <->
  [/\ inc u (quotient (equivalence_associated_o r)),
    inc v (quotient (equivalence_associated_o r))  &
    forall x y, inc x u -> inc y v -> related r x y].
Lemma oap_osr:
  order_on (order_associated r)(quotient (equivalence_associated_o r)).

Lemma oap_pr:
  order_associated r =
    Vfs ( canon_proj (equivalence_associated_o r) \ftimes
    (canon_proj (equivalence_associated_o r))) r.
End OrderAssociated.
```

2.3 Notation and terminology

The purpose of this section is to explain the abuses of language and notations in sentences like: «Let E be an ordered set. An element $a \in \mathrm{E}$ is said to be the greatest element of E if for all $x \in \mathrm{E}$ we have $x \leq a$. Let X be a subset of E. Any element $x \in \mathrm{E}$ such that $x \leq y$ for all $y \in \mathrm{X}$ is called a lower bound of X in E; an element of E is said to be the infimum of X in E if it is the greatest element of the set of lower bounds of X in E.»

Bourbaki says: «an ordering on a set E is a correspondence $\Gamma = (\mathrm{G},\mathrm{E},\mathrm{E})$ with E as source and as target, and such that the relation $(x, y) \in \mathrm{G}$ is an order relation on E. By abuse of language we shall sometimes refer to the graph G of Γ as an ordering on E.»

An example of abuse of language is the following: «let E be a set ordered by an ordering Γ, with graph G; for each subset A of E, $\mathrm{G} \cap (\mathrm{A} \times \mathrm{A})$ is an ordering on A». Here "ordering" has the two meanings. We shall write G_A as short for $\mathrm{G} \cap (\mathrm{A} \times \mathrm{A})$.

In what follows, we shall use the word "ordering" for Γ and "order" for G. Given an ordering Γ, the source E is $\mathrm{pr}_1(\mathrm{pr}_2(\Gamma))$, and the graph G is $\mathrm{pr}_1(\Gamma)$. On the other hand, if G is a graph, we denote its substrate by S_G and write $x \leq_\mathrm{G} y$ instead of $(x, y) \in \mathrm{G}$. If this is an order relation on S_G, then $(\mathrm{G},\mathrm{S}_\mathrm{G},\mathrm{S}_\mathrm{G})$ is a correspondence and an ordering. Moreover S_G is the set of all x such that $x \leq_\mathrm{G} x$. This means that the two notions of "order" and "ordering" are equivalent, and we shall use only the simple one.

Let G be an order. We say that a is the greatest element of G if $a \in \mathrm{S}_\mathrm{G}$ and for all $x \in \mathrm{S}_\mathrm{G}$ we have $x \leq_\mathrm{G} a$. Let X be a subset of S_G. Any element $x \in \mathrm{S}_\mathrm{G}$ such that $x \leq_\mathrm{G} y$ for all $y \in \mathrm{X}$ is called a lower bound of X for G; an element is said to be the infimum of X for G if it is the greatest element of G_W, where W is the set of lower bounds of X for G. We get the Bourbaki definition by omitting the indices on ≤, and writing E instead of S_G and G.

Bourbaki says «The definitions to be given in the remainder of this section apply to an arbitrary order relation $R\{x, y\}$ between x and y, but will be used mainly in the case where $R\{x, y\}$ is written $x \leq y$. [...] Then $y \geq x$ is synonymous with $x \leq y$. [...] We shall write $x < y$ for the relation "$x \leq y$ and $x \neq y$" [...] The conditions for a relation written $x \leq y$ to be an order relation on a set E are as follows:

(RO_I) The relation "$x \leq y$ and $y \leq z$" implies $x \leq z$.
(RO_{II}) The relation "$x \leq y$ and $y \leq x$" implies $x = y$.
(RO_{III}) The relation $x \leq y$ implies "$x \leq x$ and $y \leq y$".
(RO_{IV}) The relation $x \leq x$ is equivalent to $x \in E$.
If we leave out the condition (RO_{II}), we have the conditions for $x \leq y$ to be a preorder relation on E.»

We implement this statement as:

```
Definition order_axioms r s :=
  [/\ (forall y x z, gle r x y ->  gle r y z -> gle r x z),
      (forall x y, gle r x y ->  gle r y x -> x = y),
      (forall x y, gle r x y ->  (inc x s /\ inc y s)),
      (forall x, gle r x x <-> inc x s) &
       sgraph r].

Lemma axioms_of_order r:  order r <-> (order_axioms r (substrate r)).
```

In Bourbaki's framework, one can state theorems of the form: $\forall\Gamma$, if Γ is an ordering on E with graph G, if $(x, y) \in G$ is written $x \leq y$, and $x < y$ means $x \leq y$ and $x \neq y$, then $(\forall x)(\forall y)(x \in E \implies (x \leq y \iff (x < y \text{ or } x = y)))$. However one cannot quantify over relations. So one has to say: whenever we have a relation between two variables, denoted by $a \leq b$, then whatever x, whatever y, it is true that .... This is called a criterion. Example C58: *Let ≤ be an order relation, and let x, y be two distinct letters. The relation $x \leq y$ is equivalent to "$x < y$ or $x = y$". Each of the relations "$x \leq y$ and $y < z$", "$x < y$ and $y \leq z$" implies $x < z$.*

Bourbaki says: «The first assertion follows from the criterion A $\implies$ ((A and (not B)) or B) (Chapter I, §3, Criterion C24).» In fact, by excluded middle, A is equivalent to (A and (B or not B); using distributivity and simplification shows $\implies$; the converse implication is false. Thus the criterion is false (if $x = y$ then $x \leq x$ only if x belongs to some set E for which the relation is reflexive on E, but E is not mentioned in the criterion, and $x \in E$ is missing.) Note also that the criterion assumes that x and y are distinct letters, but says nothing of z.

Bourbaki writes '«n order to make matters easier and to replace metamathematical criteria by mathematical theorems, we shall usually consider a theory $\mathscr{T}$ which contains the axioms and axiom schemes of the theory of sets, and in addition, two constants E and Γ satisfying the axiom "Γ is an ordering on the set E". We shall denote by $x \leq y$ the relation $y \in \Gamma\langle x\rangle$, and we shall say that E is ordered by the ordering Γ (or by the order relation $y \in \Gamma\langle x\rangle$). [...] In some situations the theories which we shall consider are a little more complicated. We shall leave it to the reader to make explicit the constants and axioms of such theories».

This kind of machinery is quite complicated, and not needed. All that needs to be done is to replace statements as "let E be an ordered set" by "Let G be an ordering, let E denote its substrate and $x \leq y$ the relation $(x, y) \in G$"; in some cases (as in "the least element of E") E has to be replaced by G, that's all.

Order morphisms. We say that f is a *morphism* for two orders denoted $\le_E$ or $\le_F$ on E and F if f is a function from E to F compatible with the orders (if x and y are in E, then $x \le_E y$ is equivalent to $f(x) \le_F f(y)$). Such a function is injective. If x and y are in E, then $x <_E y$ is equivalent to $f(x) <_F f(y)$. Thus $x \le_E y$ implies $f(x) \le_F f(y)$ and $x <_E y$ implies $f(x) <_F f(y)$. If a morphism is bijective, it is called an *isomorphism*. Two orderings are called *isomorphic* if there is an isomorphism.

```
Definition fincr_prop f r r' :=
  forall x y,  gle r x y  -> gle r' (Vf f x) (Vf f y).
Definition fsincr_prop f r r' :=
  forall x y,  glt r x y  -> glt r' (Vf f x) (Vf f y).
Definition fiso_prop f r r' :=
  forall x y, inc x (source f) -> inc y (source f) ->
    (gle r x y <-> gle r' (Vf f x) (Vf f y)).
Definition fsiso_prop f r r' :=
  forall x y, inc x (source f) -> inc y (source f) ->
    (glt r x y <-> glt r' (Vf f x) (Vf f y)).

Definition order_isomorphism f r r' :=
 [/\ order r, order r', bijection_prop f (substrate r) (substrate r')&
  fiso_prop f r r'].

Definition order_morphism f r r' :=
  [/\ order r, order r', function_prop f (substrate r) (substrate r') &
  fiso_prop f r r'].

Definition order_isomorphic r r':=
  exists f, order_isomorphism f r r'.
```

We list some basic properties.

```
Lemma order_morphism_fi f r r':  order_morphism f r r'->
  injection f.
Lemma order_isomorphism_w r r' f:
  order_isomorphism f r r' -> order_morphism f r r'.
Lemma order_isomorphism_ws r r' f:
  bijection f -> order_morphism f r r' -> order_isomorphism f r r'.
Lemma order_morphism_incr f r r': order_morphism f r r' -> fincr_prop f r r'.
Lemma order_morphism_sincr f r r': order_morphism f r r' -> fsincr_prop f r r'.
Lemma order_isomorphism_incr f r r':
  order_isomorphism f r r' -> fincr_prop f r r'.
Lemma order_isomorphism_sincr f r r':
  order_isomorphism f r r' -> fsincr_prop f r r'.
Lemma order_morphism_siso f r r': order_morphism f r r' -> fsiso_prop f r r'.
Lemma order_isomorphism_siso f r r': order_isomorphism f r r' ->
  fsiso_prop f r r'.
```

A morphism is an isomorphism on its range. The composition of two morphisms (isomorphisms) is a morphism (resp., an isomorphism). The inverse of an isomorphism is an isomorphism.

```
Lemma identity_is r: order r ->
  order_isomorphism (identity (substrate r)) r r.
Lemma identity_morphism  r: order r ->
  order_morphism (identity (substrate r)) r r.
```

```
Lemma inverse_order_is r r' f:
  order_isomorphism f r r' -> order_isomorphism (inverse_fun f) r' r.
Lemma compose_order_morphism r r' r'' f f':
  f' \coP f -> order_morphism f r r' -> order_morphism f' r' r'' ->
  order_morphism (f' \co f) r r''.
Lemma compose_order_is r r' r'' f f':
  order_isomorphism f r r' -> order_isomorphism f' r' r'' ->
  order_isomorphism (f' \co f) r r''.
```

We can summarize the previous lemmas as: being order isomorphic is an equivalence relation (this relation has no graph, since every set can be ordered).

```
Lemma orderIR r: order r -> r \Is r.
Lemma orderIS r r': r \Is r' -> r' \Is  r.
Lemma orderIT r' r r'': r \Is r' -> r' \Is r'' -> r \Is r''.
```

2.4 Ordered subsets. Product of ordered sets

Induced order. Let G be a graph, A a set. Define $G_A = G \cap (A \times A)$. If G is an order (or preorder) on E, and $A \subset E$, then G_A is an order (or preorder) on A. It is called the order *induced* by G. If the order relation associated to G is denoted $x \le y$, then the order relation associated to G_A is denoted $x \le_A y$ or $x \le y$ by abuse of notations. By abuse of language, we say that A is an "ordered subset" of E, if we consider that A is ordered by G_A and E is ordered by G.

```
Definition induced_order r a := r \cap (coarse a).

Lemma iorder_gleP r a x y: inc x a -> inc y a ->
  (gle (induced_order r a) x y <-> gle r x y).
Lemma iorder_gle1 r a x y:
  gle (induced_order r a) x y -> gle r x y.
Lemma iorder_gle2 r a x y:
  glt (induced_order r a) x y -> glt r x y.

Lemma iorder_gle3 r a x y:
  gle (induced_order r a) x y -> (inc x a /\ inc y a).
Lemma iorder_gle4 r a x y:
  glt (induced_order r a) x y -> (inc x a /\ inc y a).
Lemma iorder_gle5P r a x y:
  gle (induced_order r a) x y <-> [/\ inc x a, inc y a & gle r x y].
Lemma iorder_gle6P r a x y:
  glt (induced_order r a) x y <-> [/\ inc x a, inc y a & glt r x y]..
```

If A is the substrate of G, then $G_A = G$. If $B \subset A$ then $(G_A)_B = G_B$. Moreover $G_A^{-1} = (G_A)^{-1}$.

```
Lemma iorder_preorder r a:
  sub a (substrate r) ->
  preorder r -> preorder (induced_order r a).
Lemma iorder_osr r a: order r -> sub a (substrate r) ->
  order_on (induced_order r a) a.
Lemma iorder_substrate r:
  order r -> induced_order r (substrate r) = r.
Lemma iorder_opposite r x: order r ->
  commutes_at (induced_order ^~ x) (opp_order) r.
Lemma iorder_trans a b c: sub c b ->
  induced_order (induced_order a b) c = induced_order a c.
```

Example of functional graphs. Let $\Phi(E, F)$ be the set of all mappings of subsets of E into F (this is the union of all $\mathscr{F}(I; F)$, for $I \subset E$). Denote by $\Psi(E, F)$ the set of functional graphs included in $E \times F$; this is the union of all F^I for $I \subset E$. Let $f \mapsto G(f)$ be the function that maps a function to its graph. This is a bijection $\mathscr{F}(I; F) \to F^I$, and extends to a bijection $\Phi(E, F) \to \Psi(E, F)$. This is an order isomorphism, if we compare partial functions by "g extends f" and graphs by $f \subset g$.

```
Definition fgraphs x y :=
  Zo (\Po (x \times y)) fgraph.

Definition graph_of_function x y :=
  Lf graph (sub_functions x y) (fgraphs x y).

Lemma graph_of_function_sub x y z:
  inc z (sub_functions x y) -> sub (graph z) (x \times y).
Lemma set_of_graphs_pr x y z:
  inc z (sub_functions x y) -> inc (graph z) (fgraphs x y).
Lemma graph_of_fonction_f x y:
  function (graph_of_function x y).
Lemma graph_of_function_V x y f:
  inc f (sub_functions x y) -> Vf (graph_of_function x y) f = graph f.
Lemma graph_of_function_fb x y:
  bijective (graph_of_function x y).
Lemma graph_of_function_is x y:
  order_isomorphism (graph_of_function x y)
  (opp_order (extension_order x y))
  (sub_order (fgraphs x y)).
```

Example of partitions. If E is a set, we consider the mapping $\varpi \mapsto \tilde{\varpi}$, that maps a partition to the graph of the associated equivalence. We know that "ϖ coarser than ϖ'" is equivalent to $\tilde{\varpi} \supset \tilde{\varpi}'$. This mapping is an isomorphism on its image (when the source is endowed with the opposite of the coarse relation, and the target with $\subset$). The source is the set of partitions, the target is some subset of $\mathfrak{P}(E \times E)$.

```
Definition graph_of_partition x :=
  Lf(fun y => partition_relset y x) (partitions x) (\Po (coarse x)).

Lemma gop_axiom x:
  lf_axiom (fun y => partition_relset y x)
     (partitions x) (\Po (coarse x)).
Lemma gop_V x y:
  partition_s y x ->
  Vf (graph_of_partition x) y = partition_relset y x.
Lemma gop_morphism x:
  order_morphism (graph_of_partition x) (coarser x)
  (opp_order (subp_order (coarse x))).
```

Example of preorders. On the set of all graphs that are preorders on a set E, we may consider the ordering induced by $\subset$. If $s \subset t$, then s is finer than t. This amounts to say that elements related by s are also related by t.

```
Definition preorder_on r E := preorder r /\ substrate r = E.
Definition preorders x :=  Zo (\Po (coarse x))(preorder_on ^~ x).
```

```
Definition coarser_preorder x := sub_order (preorders x).

Lemma preordersP x z:
  inc z (preorders x) <-> (preorder_on z x).

Lemma coarser_preorder_osr x:
  order_on (coarser_preorder x) (preorders x).
Lemma coarser_preorder_gleP x u v:
  gle (coarser_preorder x) u v <->
  [/\ preorder u, preorder v, substrate u = x, substrate v = x & sub u v].
Lemma coarser_preorder_gleP1 x u v:
  gle (coarser_preorder x) u v <->
  [/\ preorder u, preorder v, substrate u = x, substrate v = x &
  forall a b, inc a x -> inc b x -> related u a b -> related v a b].
```

Product of orders. Consider a family of relations $(G_\iota)_{\iota\in I}$, with substrate E_ι. On the product $\prod E_\iota$, we can consider the relation: $(x_\iota)_\iota$ and $(x'_\iota)_\iota$ are related if for each $\iota \in I$, x_ι and x'_ι are related. This is called the product of the relations.

```
Definition fam_of_substrates g :=
  Lg (domain g) (fun i => substrate (Vg g i)).
Definition fam_of_opp g := Lg (domain g) (fun i => opp_order (Vg g i)).

Definition prod_of_substrates g := productb (fam_of_substrates g).
Definition order_fam g := allf g order.
Definition order_product_r g x x' :=
   forall i, inc i (domain g) -> gle (Vg g i) (Vg x i) (Vg x' i).
Definition order_product g :=
  graph_on (order_product_r g)(prod_of_substrates g).

Lemma fos_graph f: fgraph (fam_of_substrates f).
Lemma fos_d f: domain (fam_of_substrates f) = domain f.
Lemma fos_V f x: inc x (domain f) ->
   Vg (fam_of_substrates f) x = substrate (Vg f x).
Lemma fam_of_opp_sr g: allf g sgraph ->
   fam_of_substrates g =  fam_of_substrates (fam_of_opp g).
Lemma fam_of_opp_or g: order_fam g -> order_fam (fam_of_opp g).
Lemma prod_of_substratesP g x:
 inc x  (prod_of_substrates g) <->
   [/\ fgraph x, domain x = domain g &
    forall i, inc i (domain g) -> inc (Vg x i) (substrate (Vg g i))].
Lemma prod_of_substrates_gi g f:
  (forall i, inc i (domain g) -> inc (f i) (substrate (Vg g i))) ->
  inc (Lg (domain g) f) (prod_of_substrates g).
Lemma prod_of_substrates_p g x i:
  inc x  (prod_of_substrates g) ->
  inc i (domain g) -> inc (Vg i x) (substrate (Vg g i)).
Lemma order_product_gleP g x x':
  (gle (order_product g) x x' <->
   [/\ inc x (prod_of_substrates g),  inc x' (prod_of_substrates g) &
      forall i, inc i (domain g) -> gle (Vg g i) (Vg x i)(Vg x' i)]).
```

If all relations are orders, then the product is an order. It will be called the *product* of the order relations and its graph will be called the *product* of the orders. The graph is $\prod G_\iota$

transported from $\prod(E_\iota \times E_\iota)$ to $\prod E_\iota \times \prod E_\iota$ via the canonical bijection. We get the opposite order by taking the opposite order of each factor.

```
Lemma order_product_osr g:
  order_fam g -> order_on (order_product g) (prod_of_substrates g).
Lemma order_product_opp_osr g: order_fam g ->
    order_on  (order_product (fam_of_opp g)) (prod_of_substrates g) /\
    order_product (fam_of_opp g) = opp_order (order_product g).
Lemma product_order_def g (f := fam_of_substrates g):
  order_fam g ->
  Vfs (prod_of_products_canon f f) (order_product g) = (productb g).
```

If we have two orders $\leq_E$ and $\leq_F$ on E and F, we may consider the product Π of the family formed of these two orders; the substrate of the product is canonically isomorphic to $E \times F$; this isomorphism induces then an order on $E \times F$, such that $(x, y) \leq (x', y')$ whenever $x \leq_E x'$ and $y \leq_F y'$. Note that $\leq$ is just the product of the two relations $\leq_E$ and $\leq_F$ (see first part of this report, where we have shown that it is an order).

```
Definition order_product2 f g := prod_of_relation f g.

Lemma order_product2_P f g x x':
  gle (order_product2 f g) x x' <->
  [/\ inc x ((substrate f) \times (substrate g)),
    inc x' ((substrate f)\times (substrate g)) &
    gle f (P x) (P x') /\ gle g (Q x) (Q x')].
```

Comparing functions. Since F^E is a product, an order on F gives an order on the set of graphs of functions. We have $f \leq g$ whenever $\forall i, f_i \leq g_i$. Similarly, we can compare two functions f and $g : E \to F$ via $\forall x \in E, f(x) \leq g(x)$. This gives a natural order on $\mathscr{F}(E;F)$. The function that associates to a function of $\mathscr{F}(E;F)$ its graph in F^E is an order isomorphism.

```
Definition order_graph_r x r z z' :=
  forall i, inc i x -> gle r (Vg z i) (Vg z' i).

Definition order_graph x y r :=
  graph_on (order_graph_r x r) (gfunctions x y).

Definition order_function_r x y r f g :=
  [/\ function_prop f x y, function_prop g x y &
  forall i, inc i x -> gle r (Vf f i) (Vf g i)].

Definition order_function x y r :=
  graph_on (order_function_r x y r) (functions x y).
```

Now some properties of graph orderings.

```
Lemma order_fam_cst x r: order r -> order_fam (cst_graph x r).
Lemma order_graph_r_P x r z z':
  order_graph_r x r z z' <->
  order_product_r (cst_graph x r) z z'.

Section OrderGraph.
Variables (x y g: Set).
```

```
Hypothesis (sr:  substrate g = y).

Lemma order_graph_pr1:
  gfunctions x y = prod_of_substrates (cst_graph x g).
Lemma order_graph_pr:
  order_graph x y g = order_product (cst_graph x g).
Lemma order_graph_osr: order g ->
   order_on (order_graph x y g) (gfunctions x y).
End OrderGraph.
```

Now some properties of function orderings.

```
Section OrderFunction.
Variables (x y r: Set).
Hypothesis (or: order r) (sr: substrate r = y).
Lemma order_functionP f f':
  gle (order_function x y r) f f' <->
  (inc f (functions x y) /\
    inc f' (functions x y) /\
    forall i, inc i x -> gle r (Vf f i) (Vf f' i)).
Lemma order_function_osr: order_on (order_function x y r)(functions x y).
Lemma function_order_is:
  order_isomorphism (Lf graph (functions x y)(gfunctions x y))
  (order_function x y r)(order_graph x y r).
```

The last theorem can be weakened to: the two ordered sets $\mathscr{F}(\mathrm{E};\mathrm{F})$ and F^{E} are order isomorphic.

```
Lemma order_function_is1: (order_function x y r) \Is (order_graph x y r).
End OrderFunction.
```

2.5 Increasing mappings

Consider two sets E and F ordered by $\leq_{\mathrm{E}}$ and $\leq_{\mathrm{F}}$ and a function $f : \mathrm{E} \to \mathrm{F}$. We say that f is *increasing* if $x \leq_{\mathrm{E}} y$ implies $f(x) \leq_{\mathrm{F}} f(y)$ and *decreasing* if $x \leq_{\mathrm{E}} y$ implies $f(x) \geq_{\mathrm{F}} f(y)$. A function is *strictly increasing* or *strictly decreasing* if $x < y$ implies $f(x) < f(y)$ or $f(x) > f(y)$. A function that is increasing or decreasing is *monotone*.

```
Definition increasing_fun f r r' :=
  [/\ order r, order r', function_prop f (substrate r) (substrate r')
   & fincr_prop f r r'].
Definition decreasing_fun f r r' :=
  [/\ order r, order r', function_prop f (substrate r) (substrate r')
  & forall x y,  gle r x y  -> gle r' (Vf f y) (Vf f x)].

Definition monotone_fun f r r' :=
  increasing_fun f r r' \/  decreasing_fun f r r'.
Definition strict_increasing_fun f r r' :=
  [/\ order r, order r', function_prop f (substrate r) (substrate r')
   & fsincr_prop f r r'].
Definition strict_decreasing_fun f r r' :=
  [/\ order r, order r', function_prop f (substrate r) (substrate r')
  & forall x y,  glt r x y  -> glt r' (Vf f y) (Vf f x)].
Definition strict_monotone_fun f r r' :=
  strict_increasing_fun f r r' \/  strict_decreasing_fun f r r'.
```

Some consequences when we replace one order by its opposite.

```
Lemma increasing_fun_reva f r r':
   increasing_fun f r r' -> decreasing_fun f r (opp_order r').
Lemma increasing_fun_revb f r r':
   increasing_fun f r r' -> decreasing_fun f (opp_order r) r'.
Lemma decreasing_fun_reva f r r':
   decreasing_fun f r r' -> increasing_fun f r (opp_order r').
Lemma decreasing_fun_revb  f r r':
   decreasing_fun f r r' -> increasing_fun f (opp_order r) r'.
Lemma monotone_fun_reva f r r':
   monotone_fun f r r' -> monotone_fun f r (opp_order r').
Lemma monotone_fun_revb  f r r':
   monotone_fun f r r' -> monotone_fun f (opp_order r) r'.
```

Same for strictly monotone.

```
Lemma strict_increasing_fun_reva  f r r':
  strict_increasing_fun f r r' -> strict_decreasing_fun f r (opp_order r').
Lemma strict_increasing_fun_revb f r r',:
  strict_increasing_fun f r r' -> strict_decreasing_fun f (opp_order r) r'.
Lemma strict_decreasing_fun_reva f r r':
  strict_decreasing_fun f r r' -> strict_increasing_fun f r (opp_order r').
Lemma strict_decreasing_fun_revb f r r':
  strict_decreasing_fun f r r' -> strict_increasing_fun f (opp_order r) r'.
Lemma strict_monotone_fun_reva f r r':
  strict_monotone_fun f r r' -> strict_monotone_fun f r (opp_order r').
Lemma strict_monotone_fun_revb f r r':
   strict_monotone_fun f r r' -> strict_monotone_fun f (opp_order r) r'.
```

A strictly increasing function is increasing. An order morphism is strictly increasing.

```
Lemma strict_increasing_w f r r':
   strict_increasing_fun f r r' -> increasing_fun f r r'.
Lemma strict_decreasing_w f r r':
   strict_decreasing_fun f r r' -> decreasing_fun f r r'.
Lemma increasing_compose f g r r' r'':
  increasing_fun f r r' ->  increasing_fun g r' r'' ->
 [/\ g \coP f,
    (forall x, inc x (source f) -> Vf (g \co f) x = Vf g (Vf f x))
    & increasing_fun (g \co f) r r''].

Lemma increasing_compose3 f g h r r' r'' r''':
  strict_increasing_fun f r r' ->  increasing_fun g r' r'' ->
  strict_increasing_fun h r'' r''' ->
    [/\ inc res (functions (source f) (target h)),
      (forall x, inc x (source f) -> Vf res x = Vf h (Vf g(Vf f x))) &
      increasing_fun res r r'''].

Lemma order_isomorphism_increasing f r r':
  order_isomorphism f r r' ->
  strict_increasing_fun f r r'.
Lemma order_morphism_increasing f r r':
  order_morphism f r r' ->
  strict_increasing_fun f r r'.
```

Examples. A constant function is increasing and decreasing (conversely, a function that is increasing and decreasing is constant on all classes of the equivalence relation “x and y are comparable”; it is constant if the set is totally ordered). The identity function is increasing and decreasing on a set ordered by equality (this function is not constant when the set has more than one element). Let E be a set, $f : \mathfrak{P}(E) \to \mathfrak{P}(E)$ the function that maps $X \subset E$ to its complementary. This is an order isomorphism if $\mathfrak{P}(E)$ is ordered by $\subset$ and $\supset$ (different ordering on source and target) and is strictly decreasing if the same ordering is chosen on the source and the target.

```
Lemma constant_fun_increasing f r r':
  order r -> order r' -> substrate r = source f  ->
  substrate r' = target f -> constantfp f ->
  increasing_fun f r r' /\  decreasing_fun f r r'.
Lemma identity_increasing_decreasing x (r := diagonal x) :
  (increasing_fun (identity x) r r /\ decreasing_fun (identity x) r r).
Lemma setC_decreasing E:
  strict_decreasing_fun (Lf (fun X => E -s X)(\Po E)(\Po E))
  (subp_order E) (subp_order E).
Lemma setC_isomorphism E:
  order_isomorphism (Lf (complement E)(\Po E)(\Po E))
  (subp_order E) (opp_order (subp_order E)).
```

Let U_x be the set of upper bounds of $\{x\}$. We have $x \leq y$ if and only if $U_y \subset U_x$. The function $x \mapsto U_x$ is thus strictly decreasing.

```
Definition upper_bounds1 r x :=
  Zo (substrate r)(fun y => gle r x y).
Lemma upper_bounds1_P x y r:
  order r -> inc x (substrate r) -> inc y (substrate r) ->
  (gle r x y <-> sub (upper_bounds1 r y) (upper_bounds1 r x)).
Lemma upper_bounds1_decreasing r:
 order r ->
 strict_decreasing_fun
   (Lf (upper_bounds1 r)(substrate r)(\Po (substrate r)))
   r (subp_order (substrate r)).
```

If a function is injective, monotone implies strictly monotone. If a function is bijective, it is an isomorphism if and only if the function and its inverse are increasing. An isomorphism remains one if the orders on the source and target are replaced by the opposite ones.

```
Lemma strict_increasing_from_injective f r r':
  injection f -> increasing_fun f r r' -> strict_increasing_fun f r r'.
Lemma strict_decreasing_from_injective f r r':
  injection f -> decreasing_fun f r r' -> strict_decreasing_fun f r r'.
Lemma strict_monotone_from_injective f r r':
  injection f -> monotone_fun f r r' -> strict_monotone_fun f r r'.

Lemma order_isomorphism_P f r r':
  order r -> order r' ->
  bijection f -> substrate r = source f -> substrate r' = target f ->
  (order_isomorphism f r r' <->
    (increasing_fun f r r' /\ increasing_fun (inverse_fun f) r' r)).
Lemma order_isomorphism_opposite g r r':
  order_isomorphism g r r' ->
  order_isomorphism g (opp_order r) (opp_order r').
```

Assume that we have two ordered sets E and E′, decreasing functions u and v from E to E′ and E′ to E. Assume $u(v(x)) \geq x$ and $v(u(x')) \geq x'$ for all x and x'. Fix x, and let $y = v(u(v(x)))$. Since v is decreasing, the first relation says $y \leq v(x)$. The second relation says $y \geq v(x)$, so that $y = v(x)$. We deduce Proposition 2 [4, p. 139], that states $u \circ v \circ u = u$ and $v \circ u \circ v = v$.

```
Theorem decreasing_composition u v r r':
  decreasing_fun u r r' -> decreasing_fun v r' r ->
 (forall x, inc x (substrate r) -> gle r (Vf v (Vf u x)) x) ->
  (forall x', inc x' (substrate r') -> gle r' (Vf u (Vf v x')) x') ->
  (u \co v \co u = u /\ v \co u \co v = v).
```

2.6 Maximal and minimal elements

Bourbaki says: if E is a set with a preorder, then $a \in$ E is *minimal* (resp. *maximal*) in E if $x \leq a$ (resp. $x \geq a$) implies $x = a$. Our definition applies to any graph.

```
Definition maximal r a :=
  inc a (substrate r) /\ forall x, gle r a x -> x = a.
Definition minimal r a :=
  inc a (substrate r) /\ forall x, gle r x a -> x = a.
```

Examples. In $\mathfrak{P}$(E), the empty set is the least element for inclusion. If we remove it, minimal elements are singletons. On the set of partial functions ordered by extension, maximal elements are total functions (because non-total functions can be extended).

```
Lemma maximal_opp r: order r -> forall x,
   (maximal (opp_order r) x <-> minimal r x).
Lemma minimal_opp r: order r -> forall x,
   (minimal (opp_order r) x <-> maximal r x).

Lemma minimal_inclusion E y (F:= (\Po E) -s1 emptyset):
  inc y F -> (minimal (sub_order F) y <-> singletonp y).
Lemma maximal_extensionP E F x:
  nonempty F -> inc x (sub_functions E F) ->
  (maximal (opp_order (extension_order E F)) x <-> (source x = E)).
```

2.7 Greatest element and least element

Given an ordering ≤ on a set E, we say that a is a *greatest* or *least* element, if $a \in$ E and for all x, $a \leq x$ (respectively, $x \leq a$) implies $x = a$. By antisymmetry, there is at most one such element. This allows us to define max(E) and min(E) that are such that: if a is a greatest element, then $a =$ max(E) and: if there is a greatest element, then max(E) is a greatest element.

```
Definition greatest r a :=
  inc a (substrate r) /\ forall x, inc x (substrate r) -> gle r x a.
Definition least r a :=
  inc a (substrate r) /\ forall x, inc x (substrate r) -> gle r a x.
Definition has_least r := (exists u, least r u).
Definition has_greatest r := (exists u, greatest r u).
Definition the_least r := select (least r) (substrate r).
Definition the_greatest r := select (greatest r) (substrate r).
```

```
Section GreatestProperties.
Variable (r: Set).
Hypothesis (or: order r).

Lemma unique_greatest: uniqueness (greatest r).
Lemma unique_least: uniqueness (least r).
Lemma the_least_pr: has_least r -> least r (the_least r).
Lemma the_greatest_p: has_greatest r -> greatest r (the_greatest r).
Lemma the_least_pr2 x:  least r x -> the_least r = x.
Lemma the_greatest_pr2 x: greatest r x -> the_greatest r = x.
```

If $\min E = \max E$, then E has a single element.

```
Lemma least_and_greatest x: least r x -> greatest r x ->
  singletonp (substrate r).
Lemma least_minimal a: least r a -> minimal r a.
Lemma greatest_maximal a: greatest r a -> maximal r a.

End GreatestProperties.
```

If E is ordered by $\le$, and $E' \subset E$ is also ordered by $\le$, then $\min E = \min E'$ provided that $\min E$ exists and is an element of E'.

```
Lemma greatest_of_induced r s x:
  order r -> sub s (substrate r) -> inc x s ->
  greatest r x ->
  the_greatest r = the_greatest (induced_order r s).
Lemma least_of_induced r s x:
  order r -> sub s (substrate r) -> inc x s ->
  least r x ->
  the_least r = the_least (induced_order r s).

Lemma least_and_greatest r x: order r ->
  least r x -> greatest r x ->
  singletonp (substrate r).
```

More simple properties.

```
Lemma greatest_opposite a r:
  order r -> greatest r a -> least (opp_order r) a.
Lemma least_opposite a r:
  order r -> least r a -> greatest (opp_order r) a.
Lemma the_greatest_opposite r: order r ->
  (has_least r) ->
  the_greatest (opp_order r) =  the_least r.
Lemma the_least_opposite r: order r ->
  (has_greatest r) ->
  the_least (opp_order r) =  the_greatest r.
Lemma greatest_unique_maximal a b r:
  greatest r a -> maximal r b -> a = b.
Lemma least_unique_minimal a b r:
  least r a -> minimal r b -> a = b.
```

Greatest and least elements of inclusion. If $\mathfrak{S}$ is a subset of $\mathfrak{P}(\mathrm{E})$, ordered by inclusion, the least upper bound and greatest lower bound (defined below) are the intersection and union. This implies the following fact: if there is a greatest element, it is the union, and the union is the greatest element if it is in the set.

```
Lemma least_is_setI s a:
  least (sub_order s) a -> a = intersection s.
Lemma greatest_is_setU s a:
  greatest (sub_order s) a -> a = union s.
Lemma setI_least s:
  inc (intersection s) s ->
  least (sub_order s) (intersection s).
Lemma setU_greatest s:
  inc (union s) s -> greatest (sub_order s) (union s).
Lemma emptyset_least E:
  least (subp_order E) emptyset.
Lemma wholeset_greatest E:
  greatest (subp_order E) E.
```

Greatest and least partial function. On the set of partial functions from E to F, where $f \le g$ means that g extends f, the least element is the empty function. If E is non-empty and F has at least two elements, there is no greatest element (let $x \in \mathrm{E}$, $y \in \mathrm{F}$, a a greatest element, b the constant function with value y; from $b \le a$ we deduce $a(x) = y$; this implies that F has a single element).

```
Lemma least_extension E F:
  least (opp_order (extension_order E F)) (empty_function_tg F).
Lemma greatest_extension E F x:
  greatest (opp_order (extension_order E F)) x ->
  nonempty E -> small_set F.
```

Least equivalence. Let E be a set; consider the subset X of $\mathfrak{P}(\mathrm{E}\times\mathrm{E})$ formed of all preorders (or all equivalences) ordered by $\subset$. Then the diagonal Δ of E is the least element of X (obviously, $\Delta \in \mathrm{X}$; we show here that, if $\mathrm{A} \in \mathrm{X}$, more generally, if A is reflexive with substrate E, then $\Delta \subset \mathrm{A}$).

```
Lemma least_equivalence r:
  reflexivep r -> sub (diagonal (substrate r)) r.
```

Extending an order. Proposition 3 [4, p. 140] in Bourbaki says:*Let* E *be an ordered set and let* E′ *be the disjoint union of* E *and a set* $\{a\}$ *consisting of a single element. Then there exists a unique ordering on* E′ *which induces the given ordering on* E *and for which a is the greatest element of* E′.

What Bourbaki proves is: «assume $a \not\in \mathrm{E}$, and let $\mathrm{E}' = \mathrm{E} \cup \{a\}$. Let G be the graph of the ordering; there is a unique ordering, satisfying the stated conditions; its graph is $\mathrm{G}\cup(\mathrm{E}'\times\{a\})$.»

Assume now that E′ is the "disjoint union" of E and $\{a\}$. This means that E′ is the union of two sets, F and $\{\alpha\}$, where $\alpha \not\in \mathrm{F}$, and there is a canonical bijection $f : \mathrm{E} \to \mathrm{F}$. We show that there is an order (obviously unique) on F that makes f an order isomorphism. It is then obvious that there is a unique order on E′ which induces the given ordering on E, transported by f, and for which α is the greatest element of E′. We shall not prove this fact. However, if we identify isomorphic sets, we get: given an ordering on E, there is a unique ordering (modulo isomorphism) on E', a set that has a greatest element a, such that $E' - \{a\}$ is isomorphic to E. This will be called the "ordinal successor" of the order-type of E.

```
Lemma order_transportation f r (r':= Vfs (f \ftimes f) r) :
  bijection f -> order r -> substrate r = source f ->
  order_isomorphism f r r'. (* 53 *)

Definition order_with_greatest r a :=
  r \cap (((substrate r) +s1 a) *s1 a).
Lemma order_with_greatest_pr
  r a (r':=order_with_greatest r a) :
    order r ->  ~ (inc a (substrate r)) ->
   [/\ order_on r' ((substrate r) +s1 a),
      r = induced_order r' (substrate r) & greatest r' a].
Theorem adjoin_greatest r a E:
  order r -> substrate r = E -> ~ (inc a E) ->
  exists! r', (fun r' => [/\ order_on r' (E +s1 a),
     r = induced_order r' E & greatest r' a]).
```

Cofinality. If r is an order (denoted by $\leq$) on E, we say that a subset A of E is *cofinal* (or *coinitial*) if for all $x \in E$ there is a $y \in A$ such that $x \leq y$ (or $y \leq x$). Bourbaki says «To say that an ordered set E has a greatest element therefore means that E has a cofinal subset consisting of a single element». Note that there is no need to assume that r is an order here.

```
Definition cofinal r a :=
  sub a (substrate r) /\
  (forall x, inc x (substrate r) -> exists2 y, inc y a & gle r x y).
Definition coinitial r a :=
  sub a (substrate r) /\
  (forall x, inc x (substrate r) -> exists2 y, inc y a & gle r y x).
Lemma exists_greatest_cofinalP r:
  (has_greatest r) <->
  (exists2 a, cofinal r a & singletonp a).
Lemma exists_least_coinitialP r:
  (has_least r) <->
  (exists2 a, coinitial r a & singletonp a).
```

2.8 Upper and lower bounds

Given a relation r (an order or a preorder) on a set E denoted by $\leq$ and a set X, an element $x \in E$ is said to be an *upper bound* for r and X if $y \in X$ implies $y \leq x$. A *lower bound* is an element $x \in E$ such that $y \in X$ implies $x \leq y$.

```
Definition upper_bound r X x  :=
  inc x (substrate r) /\ forall y, inc y X -> gle r y x.
Definition lower_bound r X x :=
  inc x (substrate r) /\ forall y, inc y X -> gle r x y.
```

The first properties given here are trivial. If we have an order on E and if X is a subset of E, we can consider the order induced on X; this may have a least element m or a greatest element M. If these quantities exist, they are in X and are an upper or lower bound of X for the relation on E. Converse holds: if X has an upper bound in X, it is M.

```
Lemma opposite_upper_boundP r: order r -> forall X x,
  (upper_bound r X x <-> lower_bound (opp_order r) X x).
Lemma opposite_lower_boundP r: order r -> forall X x,
  (lower_bound r X x <-> upper_bound  (opp_order r) X x).
Lemma smaller_lower_bound x y X r: preorder r ->
  lower_bound r X x -> gle r y x -> lower_bound r X y.
Lemma greater_upper_bound x y X r: preorder r ->
  upper_bound r X x -> gle r x y -> upper_bound r X y.
Lemma sub_lower_bound  x X Y r:
  lower_bound r X x -> sub Y X -> lower_bound r Y x.
Lemma sub_upper_bound x X Y r:
  upper_bound r X x -> sub Y X -> upper_bound r Y x.
Lemma least_elementP X r: order r -> sub X (substrate r) ->
  ((has_least (induced_order r X)) <->
   (exists2 x, lower_bound r X x & inc x X)).
Lemma greatest_elementP X r: order r -> sub X (substrate r) ->
  ( (has_greatest (induced_order r X)) <->
    (exists2 x, upper_bound r X x & inc x X)).
```

We consider now *bounded* sets, that are sets that have a bound.

```
Definition bounded_above r X := exists x, upper_bound r X x.
Definition bounded_below r X := exists x, lower_bound r X x.
Definition bounded_both r X := bounded_above r X /\ bounded_below r X.

Lemma bounded_above_sub X Y r:
  sub Y X -> bounded_above r X  -> bounded_above r Y.
Lemma bounded_below_sub X Y r:
  sub Y X -> bounded_below r X  -> bounded_below r Y.
Lemma bounded_both_sub X Y r:
  sub Y X -> bounded_both r X  -> bounded_both r Y.
Lemma singleton_bounded x r:
  singletonp x -> order r -> sub x (substrate r) -> bounded_both r x.
```

2.9 Least upper bound and greatest lower bound

The Bourbaki definition is: *let* E *be an ordered set and let* X *be a subset of* E. *An element of* E *is said to be the* greatest lower bound *of* X *in* E *if it is the greatest element of the set of lower bounds of* X *in* E (for the ordering induced by that of E). If it exists, it is unique, since there is at most one greatest element. We can avoid introducing the set of lower bounds and its induced ordering by noticing that x is the greatest lower bound if, and only if, it is a lower bound, and if z is another lower bound we have $z \le x$. Similarly, x is the *least upper bound* of X, if it is the least element of the set of upper bounds of X in E. This is equivalent to: x is an upper bound such that if z is another upper bound, then $x \le z$.

```
Definition greatest_induced r X x := greatest (induced_order r X) x.
Definition least_induced r X x := least (induced_order r X) x.
Definition greatest_lower_bound r X x :=
  greatest_induced r (Zo (substrate r) (lower_bound r X)) x.
Definition least_upper_bound r X x :=
  least_induced r (Zo (substrate r) (upper_bound r X)) x.

Lemma glbP r X:
  order r -> sub X (substrate r) -> forall x,
```

```
  (greatest_lower_bound r X x <-> (lower_bound r X x
    /\ forall z, lower_bound r X z -> gle r z x)).
Lemma lubP r X:
  order r -> sub X (substrate r) -> forall x,
  (least_upper_bound r X x <-> (upper_bound r X x
    /\ forall z, upper_bound r X z -> gle r x z)).
```

The greatest lower bound and least upper bound are also called supremum and infimum and denoted by $\sup_E X$ and $\inf_E X$. As usual, the order is $\leq$ and the substrate is E; in some cases the set E is not mentioned. If $X = \{x, y\}$ we often write $\sup(x, y)$ and $\inf(x, y)$. The sup does not always exists, but is unique since there is a unique least element.

```
Lemma supremum_unique X r: order r -> uniqueness (least_upper_bound r X).
Lemma infimum_unique X r: order r -> uniqueness (greatest_lower_bound r X).
Definition infimum r X :=
  the_greatest (induced_order r (Zo (substrate r) (lower_bound r X))).
Definition supremum r X :=
  the_least (induced_order r (Zo (substrate r) (upper_bound r X))).
Definition has_supremum r X :=(exists x, least_upper_bound r X x).
Definition has_infimum r X :=(exists x, greatest_lower_bound r X x).
Definition sup r x y := supremum r (doubleton x y).
Definition inf r x y := infimum r (doubleton x y).

Lemma supremum_pr1 X r:
  has_supremum r X ->
  least_upper_bound r X (supremum r X).
Lemma infimum_pr1 X r:
  has_infimum r X ->
  greatest_lower_bound r X (infimum r X).
Lemma supremum_pr2 r X a: order  r ->
  least_upper_bound r X a -> a = supremum r X.
Lemma infimum_pr2 r X a: order  r ->
  greatest_lower_bound r X a -> a = infimum r X.
Lemma inc_supremum_substrate X r:
  order r -> sub X (substrate r) -> has_supremum r X ->
  inc (supremum r X) (substrate r).
Lemma inc_infimum_substrate  X r:
  order r -> sub X (substrate r) -> has_infimum r X ->
  inc (infimum r X) (substrate r).
Lemma supremum_pr X r:
  order r ->  sub X (substrate r) -> has_supremum r X ->
  (upper_bound r X (supremum r X) /\
    forall z, upper_bound r X z -> gle r (supremum r X) z).
Lemma infimum_pr X r:
  order r ->  sub X (substrate r) -> has_infimum r X ->
  (lower_bound r X (infimum r X) /\
    forall z, lower_bound r X z -> gle r z (infimum r X)).
Lemma sup_pr a b r:
  order r -> inc a (substrate r) -> inc b (substrate r) ->
  has_supremum r (doubleton a b) ->
  [/\ gle r a (sup r a b) ,  gle r b (sup r a b) &
    forall z, gle r a z ->  gle r b z -> gle r (sup r a b) z).
Lemma inf_pr a b r:
  order r -> inc a (substrate r) -> inc b (substrate r) ->
  has_infimum r (doubleton a b) ->
  [/\ gle r (inf r a b) a,  gle r (inf r a b) b &
```

```
    forall z, gle r z a ->  gle r z b -> gle r z (inf r a b)).
Lemma lub_set2 r x y z:
  order r -> gle r x z -> gle r y z ->
  (forall t, gle r x t -> gle r y t -> gle r z t) ->
  least_upper_bound r (doubleton x y) z.
Lemma glb_set2 r x y z:
  order r -> gle r z x  -> gle r z y ->
  (forall t, gle r t x -> gle r t y -> gle r t z) ->
  greatest_lower_bound r (doubleton x y) z.
```

We show here the following claim: if a subset X of E has a greatest element a, then a is the least upper bound of X in E.

```
Lemma greatest_is_sup r X a:
  order r -> sub X (substrate r) ->
  greatest_induced r X a -> least_upper_bound r X a.
Lemma least_is_inf r X a:
  order r -> sub X (substrate r) ->
  least_induced r X a -> greatest_lower_bound r X a.
```

The roles of inf and sup are exchanged if we replace the order by its opposite.

```
Lemma inf_sup_oppP r X:
  order r -> sub X (substrate r) -> forall a,
  (greatest_lower_bound r X a <-> least_upper_bound (opp_order r) X a).
Lemma sup_inf_oppP r X:
  order r -> sub X (substrate r) -> forall a,
  (least_upper_bound r X a <-> greatest_lower_bound (opp_order r) X a).
Lemma sup_inf_opp r X:
  order r -> sub X (substrate r) -> has_supremum r X ->
  (has_infimum (opp_order r) X /\ infimum (opp_order r) X = supremum r X).
Lemma inf_sup_opp r X:
  order r -> sub X (substrate r) -> has_infimum r X ->
  (has_supremum (opp_order r) X /\ supremum (opp_order r) X = infimum r X).
```

Examples. We study the sup and inf of the empty set.

```
Lemma set_of_lower_bounds_set0 r :
  Zo (substrate r) (lower_bound r emptyset) = substrate r.
Lemma set_of_upper_bounds_set0 r:
  Zo (substrate r) (upper_bound r emptyset) = substrate r.
Lemma lub_set0 r: order r -> forall x,
  (least_upper_bound r emptyset x = least r x).
Lemma glb_set0 r: order r -> forall x;
  greatest_lower_bound r emptyset x = greatest r x.
```

Case of set inclusion. If $\mathfrak{S}$ is a subset of $\mathfrak{P}(E)$, then the upper and lower bounds of $\mathfrak{S}$ are the union and intersection, as claimed before. If $\mathfrak{S}$ is empty, the intersection is empty, and the greatest lower bound is the greatest element, namely E. Assume $\mathfrak{S} \subset \mathfrak{F}$ and $\mathfrak{F} \subset \mathfrak{P}(E)$; then the upper and lower bounds of $\mathfrak{S}$ in $\mathfrak{F}$ are the union and intersection, provided that these elements are in $\mathfrak{F}$.

```
Lemma setI_inf s E: sub s (\Po E) ->
  greatest_lower_bound (subp_order E) s
```

```
  (Yo (nonempty s) (intersection s) E).
Lemma setU_sup s E: sub s (\Po E) ->
  least_upper_bound (subp_order E) s (union s).
Lemma setU_sup1 s F E:
  sub F (\Po E) ->
  sub s F -> inc (union s) F ->
  least_upper_bound (sub_order F) s (union s).
Lemma setI_inf1 s F E (T := (Yo (nonempty s) (intersection s) E)):
  sub F (\Po E) ->
  sub s F -> inc T F ->
  greatest_lower_bound (sub_order F) s T.
```

Case of function extension ordering. Consider the set $\Phi(\mathrm{E},\mathrm{F})$ of partial functions from E to F, ordered by f extends g. A set of functions has a least upper bound only if, for any two functions u and v in the set, we have $u(x) = v(x)$ for any x in the intersection of the sources of u and v. If this condition holds, there is a unique function f defined in the union of the source, that agrees with each v. This function is then the supremum.

```
Lemma sup_extension_order1 E F T f:
  sub T (sub_functions E F) ->
  least_upper_bound (opp_order (extension_order E F)) T f ->
  forall u v x, inc u T -> inc v T -> inc x (source u) -> inc x (source v) ->
    Vf u x = Vf v x.
Lemma sup_extension_order2 E F T:
  sub T (sub_functions E F) ->
  (forall u v x, inc u T -> inc v T -> inc x (source u) -> inc x (source v) ->
    Vf u x = Vf v x) ->
  exists x, [/\ least_upper_bound (opp_order (extension_order E F)) T x,
     (source x = unionb (Lg T source)),
     (Imf x = unionb (Lg T (fun u => (Imf u)))) &
     (graph x) = unionb (Lg T graph)].
```

Supremum of functions. If f is a function with source A and if its target is an ordered set, the supremum of the image $f\langle \mathrm{A}\rangle$ is denoted by $\sup_{x\in\mathrm{A}} f(x)$. The infimum is denoted by $\inf_{x\in\mathrm{A}} f(x)$.

```
Definition sup_funp r f := least_upper_bound r (Imf f).
Definition inf_funp r f := greatest_lower_bound r (Imf f).

Lemma sup_funP r f: order r -> substrate r = target f ->
  function f -> forall x,
  (sup_funp r f x <-> [/\ inc x (target f),
    (forall a, inc a (source f) -> gle r (Vf f a) x)
    &forall z, inc z (target f) -> (forall a, inc a (source f)
      -> gle r (Vf f a) z) -> gle r x z]).
Lemma inf_funP r f: order r -> substrate r = target f ->
  function f -> forall x,
  (inf_funp r f x <->
   [/\ inc x (target f),
     (forall a, inc a (source f) -> gle r x (Vf f a))
    & forall z, inc z (target f) -> (forall a, inc a (source f)
     -> gle r z (Vf f a)) -> gle r z x]).
```

Supremum of functional graphs. The supremum of a functional graph is the supremum of its range.

```
Definition sup_graphp r f := least_upper_bound r (range f).
Definition inf_graphp r f := greatest_lower_bound r (range f).
Definition has_sup_graph r f := has_supremum r (range f).
Definition has_inf_graph r f := has_infimum r (range f).
Definition sup_graph r f := supremum r (range f).
Definition inf_graph r f := infimum r (range f).
```

Here are the characteristic properties.

```
Lemma sup_graph_pr1 r f:
  order r -> sub (range f) (substrate r) -> has_sup_graph r f ->
  least_upper_bound r (range f) (sup_graph r f).
Lemma inf_graph_pr1 r f:
  order r -> sub (range f) (substrate r) -> has_inf_graph r f ->
  greatest_lower_bound r (range f) (inf_graph r f).

Lemma sup_graphP r f: order r -> sub (range f) (substrate r) ->
  fgraph f -> forall x,
  (sup_graphp r f x <-> [/\ inc x (substrate r),
   (forall a, inc a (domain f) -> gle r (Vg f a) x)
   & forall z, inc z (substrate r) -> (forall a, inc a (domain f)
     -> gle r (Vg f a) z) -> gle r x z]).
Lemma inf_graphP r f: order r -> sub (range f) (substrate r) ->
  fgraph f -> forall x,
  (inf_graphp r f x <->  [/\ inc x (substrate r),
   (forall a, inc a (domain f) -> gle r x (Vg f a))
   & forall z, inc z (substrate r) -> (forall a, inc a (domain f)
     -> gle r z (Vg f a)) -> gle r z x]).
```

Monotonicity properties. Assume that $\mathrm{A} \subset \mathrm{E}$ is a set that has an infimum and a supremum. If A is empty, we know that these elements are the least and greatest elements; otherwise, if $y \in \mathrm{A}$ we have $\inf \mathrm{A} \le y \le \sup \mathrm{A}$, hence $\inf \mathrm{A} \le \sup \mathrm{A}$. This is Proposition 4 [4, p. 142].

```
Theorem compare_inf_sup1 r A: order r -> sub A (substrate r) ->
  has_supremum r A -> has_infimum r A ->
  A = emptyset ->
  (greatest r (infimum r A) /\  least r (supremum r A)).
Theorem compare_inf_sup2 r A: order r -> sub A (substrate r) ->
  has_supremum r A -> has_infimum r A ->
  nonempty A -> gle r (infimum r A) (supremum r A).
```

Proposition 5 [4, p. 142] says that sup is increasing and inf is decreasing (as a function from $\mathfrak{P}(\mathrm{E})$ into E, where $\mathfrak{P}(\mathrm{E})$ is ordered by inclusion). Of course, these are only partial functions. As a corollary, consider a family $(x_\iota)_{\iota \in \mathrm{I}}$ and $\mathrm{J} \subset \mathrm{I}$; we have $\sup_{\iota \in \mathrm{J}} x_\iota \le \sup_{\iota \in \mathrm{I}} x_\iota$ if both quantities are defined. Note that the first term is the supremum of the restriction of the family to J.

```
Theorem sup_increasing r A B: order r -> sub A (substrate r) ->
  sub B (substrate r) -> sub A B ->
  has_supremum r A -> has_supremum r B ->
```

```
  gle r (supremum r A) (supremum r B).
Theorem inf_decreasing r A B: order r -> sub A (substrate r) ->
  sub B (substrate r) -> sub A B ->
  has_infimum r A -> has_infimum r B ->
  gle r (infimum r B) (infimum r A) .
Lemma sup_increasing1 r f j:
  order r -> fgraph f -> sub (range f) (substrate r) -> sub j (domain f) ->
  has_sup_graph r f  -> has_sup_graph r (restr f j) ->
  gle r (sup_graph r (restr f j)) (sup_graph r f).
Lemma inf_decreasing1 r f j:
  order r -> fgraph f -> sub (range f) (substrate r) -> sub j (domain f) ->
  has_inf_graph r f  -> has_inf_graph r (restr f j) ->
  gle r (inf_graph r f) (inf_graph r (restr f j)) .
```

Proposition 6 [4, p. 143] says that if f and g are two functions of type $\mathrm{F} \to \mathrm{E}$, if $f(x) \le g(x)$ for all $x \in \mathrm{F}$ then $\sup f \le \sup g$, provided that both quantities are defined. In fact, it is stated as:

$$(\forall\iota \in \mathrm{I}, x_\iota \le y_\iota) \implies \sup_{\iota\in \mathrm{I}} x_\iota \le \sup_{\iota\in \mathrm{I}} y_\iota \text{ and } \inf_{\iota\in \mathrm{I}} x_\iota \le \inf_{\iota\in \mathrm{I}} y_\iota.$$

```
Lemma sup_increasing2 r f f':
  order r -> fgraph f ->  fgraph f' ->  domain f = domain f' ->
  sub (range f) (substrate r) -> sub (range f') (substrate r) ->
  has_sup_graph r f -> has_sup_graph r f' ->
  (forall x , inc x (domain f) -> gle r (Vg f x) (Vg f' x)) ->
  gle r (sup_graph r f) (sup_graph r f').
Lemma inf_increasing2 r f f':
  order r -> fgraph f ->  fgraph f' ->  domain f = domain f' ->
  sub (range f) (substrate r) -> sub (range f') (substrate r) ->
  has_inf_graph r f  ->  has_inf_graph r f' ->
  (forall x , inc x (domain f) -> gle r (Vg f x) (Vg f' x)) ->
  gle r (inf_graph r f) (inf_graph r f').
```

Associativity properties. Proposition 7 [4, p. 143] is the following. Consider a family $(x_\iota)_{\iota\in\mathrm{I}}$, and let $(\mathrm{J}_\lambda)_{\lambda\in\mathrm{L}}$ be a covering[1] of I. The family $(x_\iota)_{\iota\in\mathrm{J}_\lambda}$ is the restriction of (x_ι) to J_λ; we assume that it has a supremum $\sup_{\iota\in\mathrm{J}_\lambda} x_\iota$, and we consider the family $(\sup_{\iota\in\mathrm{J}_\lambda} x_\iota)_{\lambda\in\mathrm{L}}$. This family has a supremum if and only if $(x_\iota)_{\iota\in\mathrm{I}}$ has one, and the values are the same. The second equality in (2.4) is true under similar conditions; the proof is similar, but a shorter proof is obtained by considering opposite orders (and replace inf by sup).

$$\sup_{\iota\in\mathrm{I}} x_\iota = \sup_{\lambda\in\mathrm{L}}\big(\sup_{\iota\in\mathrm{J}_\lambda} x_\iota\big), \qquad \inf_{\iota\in\mathrm{I}} x_\iota = \inf_{\lambda\in\mathrm{L}}\big(\inf_{\iota\in\mathrm{J}_\lambda} x_\iota\big). \tag{2.4}$$

The first lemma here says that if x is a least upper bound for one family, it is also the least upper bound for the other one. Finally, since the supremum is a least upper bound, we get the result by uniqueness.

```
Section supAssoc.
Variables r f c: Set.
Hypothesis (or:order r) (fgf: fgraph f)
   (rf: sub (range f) (substrate r)) (df: domain f = unionb c).
```

[1] In fact, the union of J_λ has to be exactly I

```
Lemma sup_A x:
  (forall l, inc l (domain c) -> has_sup_graph r (restr f (Vg c l))) ->
  (sup_graphp r f x  <->
   sup_graphp r (Lg (domain c) (fun l => sup_graph r (restr f (Vg c l)))) x).
Lemma inf_A x:
  (forall l, inc l (domain c) -> has_inf_graph r (restr f (Vg c l))) ->
  (inf_graphp r f x <->
   inf_graphp r (Lg (domain c) (fun l => inf_graph r (restr f (Vg c l)))) x).

Lemma sup_A1:
  (forall l, inc l (domain c) -> has_sup_graph r (restr f (Vg c l))) ->
  (has_sup_graph r f <->
   has_sup_graph r (Lg (domain c) (fun l => sup_graph r (restr f (Vg c l))))).
Lemma inf_A1:
  (forall l, inc l (domain c) -> has_inf_graph r (restr f (Vg c l))) ->
  (has_inf_graph r f <->
   has_inf_graph r (Lg (domain c) (fun l => inf_graph r (restr f (Vg c l))))).

Theorem sup_A2:
  (forall l, inc l (domain c) -> has_sup_graph r (restr f (Vg c l))) ->
  ((has_sup_graph r f <->
    has_sup_graph r (Lg (domain c) (fun l => sup_graph r (restr f (Vg c l)))))
   /\
  (has_sup_graph r f -> sup_graph r f =
    sup_graph r (Lg (domain c) (fun l => sup_graph r (restr f (Vg c l)))))).
Theorem inf_A2:
  (forall l, inc l (domain c) -> has_inf_graph r (restr f (Vg c l))) ->
  ((has_inf_graph r f <->
    has_inf_graph r (Lg (domain c) (fun l => inf_graph r (restr f (Vg c l)))))
    /\
  (has_inf_graph r f -> inf_graph r f =
    inf_graph r (Lg (domain c) (fun l => inf_graph r (restr f (Vg c l)))))).
End supAssoc.
```

Corollary. Let $(x_{\lambda\mu})_{(\lambda,\mu)\in L\times M}$ be a double family of elements of an ordered set E such that for each $\mu \in M$ the family $(x_{\lambda\mu})_{\lambda\in L}$ has a least upper bound in E. This family is the restriction of the double family to $L \times \{\mu\}$. For the double family to have a least upper bound in E it is necessary and sufficient that $(\sup_{\lambda\in L} x_{\lambda\mu})_{\mu\in M}$ has a least upper bound, and the bounds are the same. The second equality in (2.5) holds under similar conditions.

$$\sup_{(\lambda,\mu)\in L\times M} x_{\lambda\mu} = \sup_{\mu\in M}\big(\sup_{\lambda\in L} x_{\lambda\mu}\big), \qquad \inf_{(\lambda,\mu)\in L\times M} x_{\lambda\mu} = \inf_{\mu\in M}\big(\inf_{\lambda\in L} x_{\lambda\mu}\big). \tag{2.5}$$

```
Definition partial_fun f x m := restr f (x *s1 m).

Lemma sup_A3 r f x y:
  order r -> fgraph f ->  sub (range f) (substrate r) ->
  domain f = x \times y ->
  (forall m, inc m y -> has_sup_graph r (partial_fun f x m)) ->
  ((has_sup_graph r f <->
    has_sup_graph r (Lg y (fun m => sup_graph r  (partial_fun f x m)))) /\
  (has_sup_graph r f -> sup_graph r f =
    sup_graph r (Lg y (fun m => sup_graph r (partial_fun f x m))))).
Lemma inf_A3 r f x y r:
  order r -> fgraph f ->  sub (range f) (substrate r) ->
```

```
  domain f = x \times y ->
  (forall m, inc m y -> has_inf_graph r (partial_fun f x m)) ->
  ((has_inf_graph r f <->
    has_inf_graph r (Lg y (fun m => inf_graph r  (partial_fun f x m)))) /\
  (has_inf_graph r f -> inf_graph r f =
    inf_graph r (Lg y (fun m => inf_graph r (partial_fun f x m))))).
```

Case of a product. Proposition 8 [4, p. 144] says that if we have a family of ordered sets E_ι, a subset A of $\prod E_\iota$, and if $A_\iota = pr_\iota A$, then A has a least upper bound of the form $(x_\iota)_\iota$ if and only if each A_ι has one, and there is equality; a similar property holds for the greatest lower bound.

$$\sup A = (\sup A_\iota)_{\iota\in I} = \big(\sup_{x\in A} pr_\iota x\big)_{\iota\in I}, \qquad \inf A = (\inf A_\iota)_{\iota\in I} = \big(\inf_{x\in A} pr_\iota x\big)_{\iota\in I}.$$

```
Lemma sup_in_product_aux g A (f := fam_of_substrates g)
  (Ai:= fun i => (Vfs (pr_i f i) A)):
  order_fam g -> sub A (productb f) ->
  forall i, inc i (domain g) -> sub (Ai i) (substrate (Vg g i)).

Theorem sup_in_product g A  (* 77 *)
  (f := fam_of_substrates g)
  (Ai:= fun i => (Vfs (pr_i f i) A))
  (has_sup := forall i, inc i (domain g) -> has_supremum (Vg g i) (Ai i)):
    order_fam g -> sub A (productb f) ->
    ((has_sup <-> has_supremum (order_product g) A) /\
      (has_sup -> supremum (order_product g) A =
        Lg (domain g) (fun i => supremum (Vg g i) (Ai i)))).

Theorem inf_in_product g A
  (f := fam_of_substrates g)
  (Ai:= fun i => (Vfs (pr_i f i) A))
  (has_inf := forall i, inc i (domain g) -> has_infimum (Vg g i) (Ai i)):
    order_fam g -> sub A (productb f) ->
    ((has_inf <-> has_infimum (order_product g) A) /\
      (has_inf -> infimum (order_product g) A =
        Lg (domain g) (fun i => infimum (Vg g i) (Ai i)))).
```

Case of induced ordering. Proposition 9 [4, p. 144] assumes that E is an ordered set, F is a subset of E and A is a subset of F. It can happen that one of $\sup_E A$ and $\sup_F A$ exists, but not the other; they may be unequal. If the objects exist we have

$$\sup_E A \le \sup_F A, \qquad \inf_E \ge \inf_F A \qquad (F \subset E).$$

If $\sup_E A$ exists and is in F, it is $\sup_F A$.

```
Theorem sup_induced1 r A F: order r -> sub F (substrate r) -> sub A F ->
  has_supremum r A -> has_supremum (induced_order r F) A ->
  gle r (supremum r A) (supremum (induced_order r F) A).
Theorem inf_induced1 r A F: order r -> sub F (substrate r) -> sub A F ->
  has_infimum r A -> has_infimum (induced_order r F) A ->
  gle r  (infimum (induced_order r F) A) (infimum r A).
Theorem sup_induced2 r A F: order r -> sub F (substrate r) -> sub A F ->
  has_supremum r A -> inc (supremum r A)  F ->
  (has_supremum (induced_order r F) A /\
    supremum r A = supremum (induced_order r F) A).
```

```
Theorem inf_induced2 r A F: order r -> sub F (substrate r) -> sub A F ->
  has_infimum r A -> inc (infimum r A)  F ->
  (has_infimum (induced_order r F) A /\
    infimum r A = infimum (induced_order r F) A).
```

2.10 Directed sets

An ordered set is said left or right *directed* if every doubleton is bounded (above or below).

```
Definition right_directed_prop r :=
  forall x y, inc x (substrate r) -> inc y (substrate r) ->
            exists z, gle r x z /\ gle r y z.
Definition right_directed r :=
  order r /\ forall x y, inc x (substrate r) -> inc y (substrate r) ->
    bounded_above r (doubleton x y).
Definition left_directed r :=
  order r /\ forall x y, inc x (substrate r) -> inc y (substrate r) ->
    bounded_below r (doubleton x y).
```

We rewrite the definition as: for all x and y there is a z such that $x \le z$ and $y \le z$. A set that has a greatest element is right directed. A product of directed sets is directed[2]. A cofinal set of a directed set is directed for the induced order.

```
Lemma right_directedP r:
  right_directed r <-> (order r /\ right_directed_prop r).
Lemma left_directedP r:
  left_directed r <-> (order r /\
    forall x y, inc x (substrate r) -> inc y (substrate r) -> exists z,
       gle r z x /\ gle r z y).

Lemma greatest_right_directed  r: order r ->
  has_greatest r -> right_directed r.
Lemma least_left_directed r: order r ->
  has_least r -> left_directed r.
Lemma opposite_right_directedP r: sgraph r ->
  (right_directed r <-> left_directed(opp_order r)).
Lemma opposite_left_directedP r: sgraph r ->
  (left_directed r <-> right_directed(opp_order r)).
Lemma setX_right_directed g:
  order_fam g -> (allf g right_directed) ->
  right_directed (order_product g).
Lemma setX_left_directed g:
  order_fam g -> (allf g left_directed) ->
  left_directed (order_product g).
Lemma cofinal_right_directed r A:
  right_directed r -> cofinal r A -> right_directed (induced_order r A).
Lemma coinitial_left_directed  r A:
  left_directed r -> coinitial r A -> left_directed (induced_order r A).
```

Proposition 10 [4, p. 145] says that in a right directed set, a maximal element is the greatest element.

[2]This requires the axiom of choice

```
Theorem right_directed_maximal  r x:
  right_directed r -> maximal r x -> greatest r x.
Theorem left_directed_minimal r x:
  left_directed r -> minimal r x -> least r x.
```

2.11 Lattices

A *lattice* is an ordered set on which each pair has a least upper bound and a greatest lower bound.

```
Definition lattice r :=  order  r /\
  forall x y, inc x (substrate r) -> inc y (substrate r) ->
    (has_supremum r (doubleton x y) /\ has_infimum r (doubleton x y)).

Lemma lattice_sup_pr r a b:
  lattice r -> inc a (substrate r) -> inc b (substrate r) ->
  [/\ gle r a (sup r a b),  gle r b (sup r a b) &
    forall z, gle r a -> gle r b z -> gle r (sup r a b) z].
Lemma lattice_inf_pr r a b:
  lattice r -> inc a (substrate r) -> inc b (substrate r) ->
  [/\ gle r (inf r a b) a,  gle r (inf r a b) b
    forall z, gle r z a ->  gle r z b -> gle r z (inf r a b)].
```

The power set is a lattice for inclusion. In fact each set has a supremum and an infimum.

```
Lemma inf_inclusion A x y: sub x A -> sub y A  ->
  greatest_lower_bound (subp_order A) (doubleton x y) (x \cap y).
Lemma sup_inclusion A x y: sub x A -> sub y A  ->
  least_upper_bound (subp_order A) (doubleton x y) (x \cup y).
Lemma setP_lattice A: lattice (subp_order A).
Lemma setP_lattice_pr A x y (r := subp_order A):
    inc x (\Po A) -> inc y (\Po A) ->
    (sup r x y = x \cup y /\ inf r x y = x \cap y).
```

The product of lattices is a lattice. This is an easy consequence of Proposition 8, and the fact that $\mathrm{pr}_\iota A$ is a doubleton if A is a doubleton. A lattice is a directed set.

```
Lemma setX_lattice g:
  order_fam g -> (allf g lattice) ->
  lattice (order_product g).
Lemma lattice_directed  r:
  lattice r -> (right_directed r /\ left_directed r).
```

Other examples. The set of integers, with the order "x divides y" is a lattice. The set of subgroups of a group (ordered by inclusion) is a lattice. The set of topologies on a set is a lattice. The set of real functions on an interval is a lattice. (We shall not prove these properties). The opposite of a lattice is a lattice.

```
Lemma lattice_opposite r: lattice r -> lattice (opp_order r).
```

2.12 Totally ordered sets

Two elements of an ordered set E are said comparable if the relation "$x \le y$ or $y \le x$" is true. A set E is said to be *totally ordered* if is is ordered and if any two elements of E are comparable.

```
Definition ocomparable r x y := gle r x y \/ gle r y x.
Definition total_order r :=
  order r /\ {inc  (substrate r) &, (forall x y, ocomparable r x y)}.
```

We know that G is an order if $G \circ G = G$ and $G \cap G^{-1} = \Delta_E$. It is total if moreover $G \cup G^{-1} = E \times E$, where E is the substrate of G. If the relation $\le$ is total, then for any x, y in E we have $x < y$ or $x > y$ or $x = y$; we also have $x < y$ or $y \le x$. A subset of a totally ordered set is totally ordered. A small set is totally ordered. The opposite of a totally ordered set is totally ordered.

```
Lemma total_orderP r:
  total_order r <->
  [/\ r \cg r = r,
    r \cap (inverse_graph r) = diagonal (substrate r) &
    r \cup (inverse_graph r) =  coarse (substrate r) ].
Lemma total_order_pr1 r x y:
  total_order r ->  inc x (substrate r) -> inc y (substrate r) ->
   [\/ glt r x y, glt r y x | x = y ].
Lemma total_order_pr2  r x y:
  total_order r ->  inc x (substrate r) -> inc y (substrate r) ->
    (glt r x y \/ gle r y x).

Lemma total_order_sub r x:
  total_order r ->  sub x (substrate r) -> total_order (induced_order r x).
Lemma total_order_opposite r:
 total_order r -> total_order (opp_order r).
```

If $x \le y$, then $\sup(x, y) = y$ and $\inf(x, y) = x$, hence a totally ordered set is a lattice. Consider a doubleton $X = \{a, b\}$ and $E = \mathfrak{P}(X)$ ordered by inclusion. Assume $a \ne b$. Then $\{a\}$ and $\{b\}$ are non-comparable. Thus E is a non-totally ordered lattice.

```
Lemma sup_comparable r x y: gle r x y ->
  order r -> least_upper_bound r (doubleton x y) y.
Lemma inf_comparable r x y: gle r x y ->
  order r -> greatest_lower_bound r (doubleton x y) x.
Lemma sup_comparable1 r x y: order r -> gle r x y -> sup r x y = y.
Lemma inf_comparable1 r x y: order r -> gle r x y -> inf r x y = x.
Lemma infimum_singleton r x:
   order r -> inc x (substrate r) -> infimum r (singleton x) = x.
Lemma supremum_singleton r x:
   order r -> inc x (substrate r) -> supremum r (singleton x) = x.
Lemma total_order_lattice  r:  total_order r -> lattice r.
Lemma total_order_counterexample (r := (subp_order C2)):
  lattice r /\ ~ (total_order r).
Lemma total_order_directed r:
  total_order r -> (right_directed r /\ left_directed r).
```

Obviously, sup and inf are commutative. If E has a least (or greatest) element e, we can always compute $\sup(x, e)$ and $\inf(x, e)$.

```
Lemma inf_C r x y: inf r x y = inf r y x.
Lemma sup_C r x y: sup r x y = sup r y x.
Lemma least_greatest_pr r (E := substrate r): order r ->
 [/\ (has_least r ->
         forall a, inc a E -> sup r (the_least r) a = a),
    (has_greatest r ->
      forall a, inc a E -> inf r a (the_greatest r) = a),
    (has_least r ->
      forall a, inc a E -> inf r (the_least r) a = (the_least r))&
    (has_greatest r ->
      forall a, inc a E -> sup r a (the_greatest r) = (the_greatest r))].
```

We show here additional properties of a lattice. In particular, we have associativity of sup and inf. Note that $\sup(\inf(x,y),y) = y$, and $\inf(\inf(x,y),y) = \inf(x,y)$, since $\inf(x,y) \le y$. If X has a supremum, then $X \cup \{a\}$ has a supremum.

```
Section LatticeProps.

Variables (r: Set).
Hypothesis lr: lattice r.
Let E := substrate r.

Lemma lattice_props:
    ( (forall x y, inc x E->  inc y E -> inc (sup r x y) E)
      /\ (forall x y, inc x E->  inc y E -> inc (inf r x y) E)
      /\ (forall x y, inc x E->  inc y E -> sup r (inf r x y) y = y)
      /\ (forall x y, inc x E->  inc y E -> inf r (sup r x y) y = y)
      /\ (forall x y z, inc x E->  inc y E -> inc z E ->
        sup r x (sup r y z) = sup r (sup r x y) z)
      /\ (forall x y z, inc x E->  inc y E -> inc z E ->
        inf r x (inf r y z) = inf r (inf r x y) z)
      /\ (forall x, inc x E -> sup r x x = x)
      /\ (forall x, inc x E -> inf r x x = x)
      /\ (forall x y, inc x E->  inc y E -> sup r (sup r x y) x = sup r x y)
      /\ (forall x y, inc x E->  inc y E -> inf r (inf r x y) x = inf r x y)
).
Lemma sup_monotone a b c:
   inc a E -> gle r b c- > gle r (sup r a b) (sup r a c).
Lemma inf_monotone a b c:
   inc a E -> gle r b c-> gle r (inf r a b) (inf r a c).
Lemma lattice_finite_sup1 X x a:
  sub X E -> least_upper_bound r X x -> inc a E ->
  least_upper_bound r (X +s1 a) (sup r x a).
Lemma lattice_finite_inf1 X x a:
  sub X E -> greatest_lower_bound r X x -> inc a E ->
  greatest_lower_bound r (X +s1 a) (inf r x a).

End LatticeProps.
```

We say that a lattice is distributive if there is a distributive law between inf and sup (there is a symmetry between the two operators). We give here equivalent properties.

```
Definition distributive_lattice1 r :=
  forall x y z, inc x (substrate r) ->inc y (substrate r) ->
    inc z (substrate r) ->
    sup r x (inf r y z) = inf r (sup r x y) (sup r x z).
```

```
Definition distributive_lattice2 r :=
  forall x y z, inc x (substrate r) ->inc y (substrate r) ->
    inc z (substrate r) ->
    inf r x (sup r y z) = sup r (inf r x y) (inf r x z).
Definition distributive_lattice3 r :=
  forall x y z, inc x (substrate r) ->inc y (substrate r) ->
    inc z (substrate r) ->
    sup r (inf r x y) (sup r (inf r y z) (inf r z x)) =
    inf r (sup r x y) (inf r (sup r y z) (sup r z x)).
Definition distributive_lattice4 r :=
  forall x y z, inc x (substrate r) ->inc y (substrate r) ->
    inc z (substrate r) ->
    gle r z x -> sup r z (inf r x y) = inf r x (sup r y z).
Definition distributive_lattice5 r:=
  forall x y z, inc x (substrate r) ->inc y (substrate r) ->
    inc z (substrate r) ->
    gle r (inf r z (sup r x y)) (sup r x (inf r y z)).
Definition distributive_lattice6 r :=
  forall x y z, inc x (substrate r) ->inc y (substrate r) ->
    inc z (substrate r) ->
    inf r (sup r x y) (sup r z (inf r x y))
    = sup r (inf r x y) (sup r (inf r y z) (inf r z x)).
```

A total order is distributive.

```
Lemma total_order_dlattice r:
  total_order r -> distributive_lattice1 r.

Section DistributiveLattice.
Variable r: Set.
Hypothesis lr: lattice r.

Lemma distributive_lattice_prop1:
  ( (distributive_lattice1 r -> distributive_lattice3 r) /\
    (distributive_lattice2 r -> distributive_lattice3 r)).
Lemma distributive_lattice_prop2:
  [/\ (distributive_lattice3 r -> distributive_lattice4 r),
    (distributive_lattice3 r -> distributive_lattice1 r) &
    (distributive_lattice3 r -> distributive_lattice2 r)].
Lemma distributive_lattice_prop3:
  (distributive_lattice3 r <-> distributive_lattice5 r).
Lemma distributive_lattice_prop4:
  (distributive_lattice3 r <-> distributive_lattice6 r).
End DistributiveLattice.
```

Proposition 11 [4, p. 147] says that if f is strictly monotone and the ordering on the source is total, then f is injective. If f is strictly increasing, it is a morphism (an isomorphism onto the image). If f is injective and increasing, it is a morphism.

```
Theorem total_order_monotone_injective f r r':
  total_order r -> strict_monotone_fun f r r' -> injection f.
Theorem total_order_increasing_morphism f r r':
  total_order r -> strict_increasing_fun f r r' -> order_morphism f r r'.
Lemma total_order_morphism f r r':
  total_order r -> order r' ->
```

```
  injection f ->  substrate r = source f -> substrate r' = target f ->
  {inc source f &, fincr_prop f r r'} ->
  order_morphism f r r'.
Lemma total_order_isomorphism f r r':
  total_order r -> order r' ->
  bijection f ->  substrate r = source f -> substrate r' = target f ->
  {inc source f &, fincr_prop f r r'} ->
  order_isomorphism f r r'.
```

Proposition 12 [4, p. 147] says that in a totally ordered set E, an element x is the least upper bound of a subset X if and only if it is an upper bound and, for all $y < x$, there is a $z \in X$ such that $y < z$ and $z \leq x$.

```
Theorem sup_in_total_order r X x: total_order r -> sub X (substrate r)->
  (least_upper_bound r X x <-> (upper_bound r X x /\
    (forall y, glt r y x -> exists z, [/\ inc z X, glt r y z & gle r z x]))).
Theorem inf_in_total_order r X x: total_order r -> sub X (substrate r)->
  (greatest_lower_bound r X x <-> (lower_bound r X x /\
    (forall y, glt r x y -> exists z, [/\ inc z X, glt r z y & gle r x z]))).
```

2.13 Intervals

There are many definitions of an *interval*. The set of all x such that $a \leq x \leq b$ is called the closed interval and denoted $[a, b]$; the set of all x such that $a < x < b$ is called the open interval and denoted $]a, b[$; intervals can be semi open. One can drop one of the conditions, for instance the set of all x such that $x < b$ is denoted by $]\leftarrow, b[$, this is an unbounded interval.

The letters o, c, u stand for open, close, and unbounded.

```
Definition interval_oo r a b :=
  Zo(substrate r)(fun z => glt r a z /\ glt r z b).
Definition interval_oc r a b :=
  Zo(substrate r)(fun z => glt r a z /\ gle r z b).
Definition interval_ou r a  :=
  Zo (substrate r) (fun z  => glt r a z).
Definition interval_co r a b :=
  Zo(substrate r)(fun z => gle r a z /\ glt r z b).
Definition interval_cc r a b :=
  Zo(substrate r)(fun z => gle r a z /\ gle r z b).
Definition interval_cu r a  := Zo (substrate r) (fun z  => gle r a z).
Definition interval_uo r b := Zo (substrate r) (fun z  =>  glt r z b).
Definition interval_uc r b := Zo (substrate r) (fun z => gle r  z b).
Definition interval_uu r := Zo (substrate r) (fun z => True).

Definition closed_interval r x := exists a b,
  [/\ inc a (substrate r), inc b (substrate r), gle r a b &
  x = interval_cc r a b].
Definition open_interval r x := exists a b,
  [/\ inc a (substrate r), inc b (substrate r), gle r a b &
  x = interval_oo r a b].
Definition semi_open_interval r x := exists a b,
  [/\ inc a (substrate r), inc b (substrate r),  gle r a b &
  (x = interval_oc r a b \/ x = interval_co r a b)].
Definition bounded_interval r x := closed_interval r x \/
```

```
  open_interval r x \/ semi_open_interval r x.
Definition left_unbounded_interval r x :=
  exists2 b, inc b (substrate r)& (x = interval_uc r b \/ x = interval_uo r b).
Definition right_unbounded_interval r x :=
  exists2 a, inc a (substrate r)& (x = interval_cu r a \/ x = interval_ou r a).
Definition unbounded_interval r x :=
  left_unbounded_interval r x \/ right_unbounded_interval r x  \/
  x = interval_uu r.
Definition interval r x :=
  bounded_interval r x  \/ unbounded_interval r x.
```

A non-empty interval $[a,b]$ has a least and greatest elements, which are a and b respectively.

```
Lemma the_least_interval r a b: order r ->
  gle r a b ->   the_least (induced_order r (interval_cc r a b)) = a.
Lemma the_greatest_interval r a b: order r ->
  gle r a b -> the_greatest (induced_order r (interval_cc r a b)) = b.
```

A closed interval is never empty; however $[a,a[$, $]a,a]$ and $]a,a[$ are empty.

```
Lemma nonempty_closed_interval r x:
  order r -> closed_interval r x -> nonempty x.
Lemma singleton_interval r a:
  order r -> inc a (substrate r) -> singletonp (interval_cc r a a).
Lemma empty_interval r a:
  order r -> inc a (substrate r) ->
  [/\ empty (interval_co r a a),  empty (interval_oc r a a) &
    empty (interval_oo r a a)].
```

The only non trivial result here is Proposition 13 [4, p. 148] that says that, in a lattice, the intersection of two intervals is an interval.

Let's say that an interval is of type L if it is left unbounded, of type R if it is right unbounded (the interval $\mathrm{U} =]\leftarrow,\rightarrow[$ is of both types, and $\mathrm{U}\cap\mathrm{X} = \mathrm{X}$ for any interval X). Obviously, each interval is the intersection of two intervals of type L and R (if the interval is unbounded, consider intersection with U). The intersection of two intervals is thus of the form $(\mathrm{L}_1\cap\mathrm{R}_1)\cap(\mathrm{L}_2\cap\mathrm{R}_2) = (\mathrm{L}_1\cap\mathrm{L}_2)\cap(\mathrm{R}_1\cap\mathrm{R}_2)$. This is of the form $\mathrm{L}_3\cap\mathrm{R}_3$.

```
Definition lu_interval r x :=
   x = interval_uu r \/ left_unbounded_interval r x.
Definition ru_interval r x :=
   x = interval_uu r \/ right_unbounded_interval r x.
Lemma setI_i1  r x:
  interval r x -> x \cap (interval_uu r) = x.
Lemma setI_i2 r x:
  interval r x ->
  (exists u v, [/\ lu_interval r u, ru_interval r v &
    u \cap v = x]).
Lemma setI_i3  r x y: lattice r ->
  left_unbounded_interval r x -> left_unbounded_interval r y ->
  left_unbounded_interval r (x \cap y).
Theorem setI_interval r x y:
  lattice r -> interval r x -> interval r y ->
  interval r (x \cap y).
```

Chapter 3

Well-ordered sets

This chapter defines the notion of a well-ordering, and the lexicographic ordering of a product of ordered sets. We show Zermelo's theorem (there exists a well-ordering) and Zorn's lemma (every inductive ordered set has a maximal element). These theorems are equivalent to the axiom of choice, thus are non-constructive. We introduce the principle of transfinite induction: given a well-ordered set and a term T, there exists a unique function f such that $f(x)$ is $\mathrm{T}(f'_x)$, where f'_x is the restriction of f to the set of all y such that $y < x$.

3.1 Segments of a well-ordered set

The Bourbaki definition is the following:

«A relation $\mathrm{R}\{x, y\}$ is said to be a *well-ordering relation* between x and y if R is an order relation between x and y and if for each non-empty set E on which $\mathrm{R}\{x, y\}$ induces an order relation, E, ordered by this relation, has a least element. A set E ordered by an ordering Γ is said to be *well-ordered* if the relation $y \in \Gamma\langle x\rangle$ is a well-ordering between x and y; Γ is then said to be a well-ordering on E.»

Recall that an ordering Γ is a correspondence, whose graph G is an order. The relation $y \in \Gamma\langle x\rangle$ is the same as $(x, y) \in \mathrm{G}$. We shall not consider Γ in what follows. The condition "$\mathrm{R}\{x, y\}$ induces an order relation on E" is equivalent to "if $x \in \mathrm{E}$ then $\mathrm{R}\{x, x\}$". We can then restate the definitions as:

A relation $x \le y$ is a well-ordering, if it is an ordering, and whenever E is a non-empty set such that $x \in \mathrm{E}$ implies $x \le x$, then $\le_{\mathrm{E}}$ has a least element. A graph G is a well-order on X if it is an order on X, and for any non-empty subset E of X, G_{E} has a least element. Here $\le_{\mathrm{E}}$ is the order on E, induced by $\le$ and G_{E} is the order on E induced by G. These two definitions are related by: if $\le$ is a well-order relation, if $x \in \mathrm{E}$ implies $x \le x$, then $\le_{\mathrm{E}}$ is a well-order on E.

```
Definition worder_r (r: relation) :=
  order_r r /\ forall x, {inc x, reflexive_r r} -> nonempty x ->
    has_least (graph_on r x).
Definition worder r :=
  order r /\ forall x, sub x (substrate x) -> nonempty x ->
    has_least (induced_order r x).
Definition worder_on G E := worder G /\ substrate G = E.
```

```
Lemma worder_or r: worder r -> order r.
```

```
Lemma wordering_pr r x:
  worder_r r -> {inc x, reflexive_r r} ->
  worder_on (graph_on r x) x.
Lemma worder_prop r x: worder r -> sub x (substrate r) -> nonempty x ->
  exists2 a, inc a x & (forall b, inc b x -> gle r a b).
Lemma worder_prop_rev r: order r ->
  (forall x, sub x (substrate r) -> nonempty x ->
    exists2 a, inc a x & (forall b, inc b x -> gle r a b)) ->
  worder r.
Lemma worder_invariance r r':
  r \Is r' -> worder r -> worder r'.
```

Examples. The empty set is a well-order on the empty set (note that any order on $\emptyset$ is $\emptyset$).

```
Lemma set0_osr: order_on emptyset emptyset.
Lemma set0_wor: worder_on emptyset emptyset.
Lemma empty_substrate_zero x: substrate x = emptyset -> x = emptyset.
```

There is a unique order on a singleton $\{x\}$, namely $\{(x,x)\}$. Thus all singletons are order-isomorphic.

```
Lemma set1_wor x: worder_on (singleton (J x x)) (singleton x).
Lemma set1_order_is0 r x:
  order r -> substrate r = singleton x ->  r = singleton (J x x).
Lemma set1_order_is x y:
  (singleton (J x x)) \Is (singleton (J y y)).
Lemma set1_order_is1 r:
  order r -> singletonp (substrate r) ->
  exists x, r = singleton (J x x).
Lemma set1_order_is2 r r':
  order r -> order r' ->
  singletonp (substrate r) -> singletonp (substrate r') ->
  r \Is r'.
Lemma worder_set1 r e: order_on r (singleton e) -> worder r.
```

Bourbaki notes that a totally ordered set with two elements is well-ordered. In fact, it contains a and b such that $a < b$, so that the graph is the set with three elements (a,a), (a,b) and (b,b). All total orders on sets with two elements are isomorphic. We consider here the case where a and b are the elements of the doubleton `C2`.

```
Definition canonical_doubleton_order :=
  (tripleton (J C0 C0) (J C1 C1) (J C0 C1)).

Lemma cdo_gleP x y:
  gle canonical_doubleton_order x y <->
  [\/ (x = C0 /\ y = C0),  (x = C1 /\ y = C1) | (x = C0 /\ y = C1)].
Lemma cdo_wor: worder_on canonical_doubleton_order C2.
```

Basic properties. A well-order on E is total (any subset with two elements has a least element, thus the elements are comparable). Every subset of E bounded above has a supremum; every non-empty subset has a least element; every subset is well-ordered by the induced ordering. Moreover, adjoining a greatest element to a well-order yields a well-order.

```
Lemma worder_total r: worder r -> total_order r.
Lemma worderr_total r x y: worder_r r -> r x x -> r y y ->
    (r x y \/ r y x).
Lemma worder_hassup r A: worder r -> sub A (substrate r) ->
  bounded_above r A -> has_supremum r A.

Lemma induced_wor r A: worder r -> sub A (substrate r) ->
  worder (induced_order r A).
Lemma worder_adjoin_greatest r a: worder r-> ~ (inc a (substrate r)) ->
  worder (order_with_greatest r a).
Lemma worder_least r: worder r -> nonempty (substrate  r) ->
  has_least r.
```

Existence of a well-order. In 1908, Ernst Zermelo presented an alternative, simpler proof of his theorem (see [23, pages 183-189]), (see also page 501). The ordering depends explicitly on `rep`, the axiom of choice.

```
Definition segment_r r x:= interval_cu r x.
Lemma segment_rP r x y: inc y (segment_r r x) <-> gle r x y.

Definition Zermelo_like r:= worder r /\
  forall x, inc x (substrate r) -> rep (segment_r r x) = x.
Definition Zermelo_chain E F :=
  let p :=  fun a => a -s1 (rep a) in
   [/\ sub F (\Po E), inc E F,
    (forall A, inc A F -> inc (p A) F)
    & (forall A, sub A F -> nonempty A -> inc (intersection A) F)].
Definition worder_of E :=
  let om :=  intersection (Zo (\Po (\Po E)) (Zermelo_chain E)) in
  let d:= fun x => intersection (Zo om (sub x)) in
  let R := fun x => d (singleton x) in
  graph_on (fun x y => (sub (R y) (R x))) E.

Lemma Zermelo_ter E (r := worder_of E):
   worder_on r E /\ Zermelo_like r.
```

Uniqueness properties of isomorphisms. Consider two ordered sets E, E′, two strictly increasing functions f and g, that map E onto a same subset of E′. If E is well-ordered then $f = g$. For otherwise, there would be a least x such that $f(x) \neq g(x)$. Since f and g have the same range, there is y such that $f(x) = g(y)$ and z such that $g(x) = f(z)$. By definition of x, if $y < x$, then $f(y) = g(y)$, and by injectivity of f, we get $x = y$, absurd. Thus $x < y$ and $g(x) < g(y)$. This is $g(x) < f(x)$. The same argument says $f(x) < g(x)$, absurd.

Example. The order isomorphisms $\mathbf{Z} \to \mathbf{Z}$ are all functions of the form $x \mapsto x + a$. These functions are obviously order isomorphisms. Consider an isomorphism g, let $a = g(0)$ and $f(x) = x + a$. If $x < 0$ then $g(x) < a$ and if $x > 0$ then $g(x) > a$. Since g is surjective, every $y \geq a$ has the form $y = g(x)$ for $x \geq 0$. Since $\mathbf{N}$ is well-ordered and the restrictions of f and g to $\mathbf{N}$ have the same range, we deduce $f(x) = g(x)$ for $x \geq 0$. Consider now $x \mapsto -g(-x)$ for $x > 0$. This is a strictly increasing function, thus $f = g$ (see section 8.7).

```
Lemma strict_increasing_extens f g r r’:
  strict_increasing_fun f r r’-> strict_increasing_fun g r r’ -> worder r ->
  Imf f = Imf g ->
```

```
  f = g.
Lemma iso_unique r r' f f':
  worder r -> order_isomorphism f r r' ->  order_isomorphism f' r r' ->
  f = f'.
```

Segments. A *segment*[1] S in an ordered set E is such that, if $x \in \mathrm{S}$ and $y \leq x$, then $y \in \mathrm{S}$. If $x \in \mathrm{E}$, the set of all y such that $y < x$ is a segment, it is called the *segment with endpoint* x, and denoted $]\leftarrow, x[$ or S_x. The set of all y such that $y \leq x$ is a also a segment, it is denoted $]\leftarrow, x]$.

```
Definition segmentp r s :=
  sub s (substrate r) /\ forall x y, inc x s -> gle r y x -> inc y s.
Definition segment r x := interval_uo r x.
Definition segmentc r x := interval_uc r x.
```

We list some properties of segments of an ordered sets. Note that $\emptyset$, E, the union of segments, and the intersection of segments, are segments.

```
Lemma lt_in_segment r s x y:
  segmentp r s ->  inc x s -> glt r y x -> inc y s.
Lemma inc_segment r x y: inc y (segment r x) -> glt r y x.
Lemma not_in_segment  r x: ~ inc x (segment r x).
Lemma sub_segment r x: sub (segment r x) (substrate r).
Lemma sub_segment1 r s:  segmentp r s -> sub s (substrate r).
Lemma sub_segment2 r x y: sub (segment (induced_order r x) y) x.
Lemma segment_inc r x y: glt r y x -> inc y (segment r x).
Lemma segmentP r x y:   inc y (segment r x) <-> glt r y x.
Lemma segmentcP r x y:  inc y (segmentc r x) <-> gle r y x.
Lemma inc_bound_segmentc r x: order r -> inc x (substrate r) ->
  inc x (segmentc r x).
Lemma lt_in_segment2 r x s y:
  segmentp r s -> inc x s -> inc y (segment r x) -> inc y s.
Lemma sub_segmentc r x: sub (segmentc r x) (substrate r).
Lemma segmentc_pr r x: order r -> inc x (substrate r) ->
  (segment r x) +s1 x = segmentc r x.
Lemma set0_segment r: segmentp r emptyset.
Lemma substrate_segment r: segmentp r (substrate r).
Lemma setI_segment r s:
  (alls s (segmentp r)) -> segmentp r (intersection s).
Lemma setU_segment r s:
  (alls s (segmentp r)) -> segmentp r (union s).
Lemma setUf_segment r j s:
  (alls j (fun x => segmentp r (s x))) segmentp r (unionf j s).
Lemma subsegment_segment r s s': order r ->
  segmentp r s -> segmentp (induced_order r s) s' -> segmentp r s'.
Lemma segment_segment r x: order r -> segmentp r (segment r x).
Lemma segmentc_segment r x: order r -> segmentp r (segmentc r x).
```

Proposition 1 [4, p. 149] says: in a well-ordered set E, a segment is either E, or has the form S_x. Note that, if $x \in \mathrm{E}$, then E is non-empty and has a least element a and $\mathrm{S}_x = [a, x[$.

```
Theorem well_ordered_segment r s: worder r -> segmentp r s ->
  s = substrate r \/ (exists2 x, inc x (substrate r) & s = segment r x).
```

[1] This is sometimes called an "initial segment"

```
Lemma segment_alt r x a: least r a ->
  segment r x = interval_co r a x.
Lemma segment_alt1 r s: worder r -> segmentp r s ->
  s = substrate r \/ (exists x, exists a, s = interval_co  r a x).
```

Some useful lemmas. We consider a well-ordered set. If S and S′ are segments, then S ⊂ S′ or S′ ⊂ S. If S ⊂ S′, if $x \in$ S, the segments with endpoint x in S or S′ coincide. If $x \le y$, then $S_x \subset S_y$ and $S_x \times S_x \subset S_y \times S_y$. If $z \le y$ and $y \in S_x$ then $z \in S_x$. The set $]\leftarrow, x]$ is a segment. If S is a segment and $x \in$ S, then S_x is the segment with endpoint x for the order induced on S. It is also the segment with endpoint x for the order induced on $]\leftarrow, y]$ or $]\leftarrow, y[$ if $x < y$.

```
Lemma segment_monotone r x y: order r -> gle r x y ->
  sub (segment r x)  (segment r y).
Lemma segment_dichot_sub r x y:
  worder r -> segmentp r x -> segmentp r y ->
 (sub x y \/ sub y x).
Lemma le_in_segment r x y z: order r -> inc x (substrate r) ->
  inc y (segment r x) -> gle r z y -> inc z (segment r x).
Lemma coarse_segment_monotone r x y: order r -> gle r x y ->
  sub (coarse (segment r x)) (coarse (segment r y)).
  segmentp r (segment_c r x).
Lemma segment_induced_a r s x:
  segmentp r s -> inc x s ->
  segment (induced_order r s) x = segment r x.
Lemma segment_induced r a b: order r ->  glt r b a ->
  segment (induced_order r (segment r a)) b = segment r b.
Lemma segment_induced1 r a b: order r ->  glt r b a ->
  segment (induced_order r (segmentc r a)) b = segment r b.
```

In a totally ordered set E, the union of all segments is E (when E has no greatest element) or E − $\{a\}$ (if a is the greatest element of E).

```
Definition segmentss r:=
  fun_image (substrate r) (segment r).

Lemma union_segments r (E := substrate r)(A := union (segmentss r)):
   total_order r ->
   ( (forall x, ~ (greatest r x)) -> A = E)
   /\ (forall x, greatest r x -> A = E -s1 x).
```

Well-ordering on the set of segments. Consider first a totally ordered set. Then $x \mapsto S_x$ is strictly increasing (when the target is ordered by inclusion) hence injective.

```
Lemma segment_monotone1 r x y: total_order r ->
  inc x (substrate r) ->  inc y (substrate r) ->
  sub (segment r x)(segment r y) -> gle r x y.
Lemma segment_injective r : total_order r ->
  {inc (substrate r) &,  injective (segment r) }.
```

Consider now a well-ordered set E. The previous lemma says that the set of all S_x is isomorphic to E, thus well-ordered. Let E* be the set of all segments. This is the set of all S_x to which a greatest element has been added. Thus E* is well-ordered. This is Proposition 2 in [4, p. 149].

```
Definition segments r:=
  (segmentss r) +s1  (substrate r).
Definition segments_iso r:=
  Lf(segment r) (substrate r) (segmentss r).

Lemma inc_segmentsP r: worder r -> forall x
  (segmentp r x <-> inc x (segments r)).
Lemma segmentc_insetof r x: worder r -> inc (segmentc r x) (segments r).
Lemma segment_insetof r x: worder r -> inc (segment r x) (segments r).
Lemma sub_segments r x: worder r ->
  inc x (segments r) -> sub x (substrate r).
Theorem segments_iso_is r: worder r ->
  order_isomorphism (segments_iso r) r (sub_order (segmentss r)).
Theorem segments_worder r: worder r ->
  worder (sub_order (segments r)).
```

Common order extension. We state Lemma 1 [4, p. 150]. *Let* $(X_\alpha)_{\alpha\in A}$ *be a family of ordered sets, directed with respect to the relation* $\subset$. *Suppose that, for each pair of indices* (α,β) *such that* $X_\alpha \subset X_\beta$, *the ordering induced on* X_α *by that of* X_β *is identical with the given ordering on* X_α. *Under these conditions there exists a unique ordering on the set* $E = \bigcup_{\alpha\in A} X_\alpha$ *which induces the given ordering on each* X_α.

If (X_α) is a family of orders, we denote the substrate by E_α and the order by G_α. We say that the family is monotone if, whenever $E_\alpha \subset E_\beta$, then G_β, restricted to E_α, is G_α. We say that the family is directed if for all α and β, there is γ such that $E_\alpha \subset E_\gamma$ and $E_\beta \subset E_\gamma$. We also consider a stronger condition: E_α is a segment of E_β or E_β is a segment of E_α.

```
Definition worder_fam g :=  allf g worder.
Definition order_extends r r' := r = induced_order r' (substrate r).
Definition monotone_order_fam g :=
  forall a b, inc a (domain g) -> inc b (domain g) ->
    sub (substrate (Vg g a)) (substrate (Vg g b)) ->
    order_extends (Vg g a) (Vg g b)

Definition common_extension_order g h:=
  [/\ order h, substrate h = unionf (domain g) (fun a => substrate (Vg g a))
  & (forall a, inc a (domain g) -> order_extends (Vg g a) h)].
Definition common_extension_order_axiom g :=
  [/\ order_fam g,
  (forall a b, inc a (domain g) -> inc b (domain g )-> exists c,
    [/\ inc c (domain g), sub (substrate (Vg g a)) (substrate (Vg g c))
      & sub (substrate (Vg g b)) (substrate (Vg g c))])
  & monotone_order_fam g].
Definition common_worder_axiom g:=
  [/\  worder_fam g,
  (forall a b, inc a (domain g) -> inc b (domain g) ->
    segmentp (Vg g a) (substrate (Vg g b))
    \/ segmentp (Vg g b) (substrate (Vg g a)))
  & monotone_order_fam g].
```

Existence and uniqueness are easy to prove.

```
Lemma order_merge1 g :
  common_extension_order_axiom g -> common_extension_order g (unionb g).
```

```
Lemma order_merge2 g:  common_extension_order_axiom g ->
  uniqueness (common_extension_order g).
```

We consider now Proposition 3 [4, p. 149]. It says *Let* $(X_\iota)_{\iota\in I}$ *be a family of well-ordered sets such that for each pair of indices* (ι,κ) *one of the sets* X_ι, X_κ *is a segment of the other. Then there exists a unique ordering on the set* $E = \bigcup_{\iota\in I} X_\iota$ *which induces the given ordering on each of the* X_ι. *Endowed with this ordering,* E *is a well-ordered set. Every segment of* X_ι *is a segment of* E*; for each* $x \in X_\iota$*, the segment with endpoint* x *in* X *is equal to the segment with endpoint* x *in* E*; and each segment of* E *is either* E *itself or a segment of one of the* X_ι.

Existence and uniqueness follows from the previous case.

```
Lemma order_merge3 g:
  common_worder_axiom g -> common_extension_order_axiom g.
Lemma order_merge4 g:
  common_worder_axiom g -> common_extension_order g (unionb g).
Lemma order_merge5 g:  common_worder_axiom g ->
  uniqueness (common_extension_order g).
```

Let x be in the E, say $x \in E_\alpha$. If $y \le_\alpha x$ then $y \le x$, where $\le$ is the order of E. Conversely, if $y \le x$, there is some β such that $y \in E_\beta$ and $y \le_\beta x$; but E_β is a substrate of E_α, or the converse. In any case this implies $y \in E_\alpha$ and $y \le_\alpha x$, thus the result.

```
Theorem worder_merge g  (G := unionb g):
  common_worder_axiom g ->
 [/\  (common_extension_order g G),
   worder G,
   (forall a x, inc a (domain g) -> segmentp (Vg g a) x
     -> segmentp G x),
   (forall a x, inc a (domain g) ->  inc x (substrate (Vg g a)) ->
     segment (Vg g a) x = segment G x)
   & (forall x, segmentp G x ->
     x = substrate G \/ exists2 a, inc a (domain g) & segmentp (Vg g a) x)].
```

3.2 The principle of transfinite induction

The next result is Lemma 2 [4, p. 151]. It says that, given a well-ordered set E and a set $\mathfrak{S}$ of segments of E, stable by union and by successor (i.e., if $]\leftarrow, x[\in \mathfrak{S}$ then $]\leftarrow, x] \in \mathfrak{S}$), then all segments are in $\mathfrak{S}$. In particular $E \in \mathfrak{S}$.

Proof. Assume, by contradiction, that the set of segments not in $\mathfrak{S}$ is non-empty. There is then a least element for inclusion, say Y. Assume first that Y has a greatest element. Then $Y =]\leftarrow, a]$, but $]\leftarrow, a[\in \mathfrak{S}$ by minimality. Assume that Y has no greatest element. Since Y is a well-ordered set, it is the union of all S_x for $x \in Y$. By minimality, $S_x \in \mathfrak{S}$; and the union is also in $\mathfrak{S}$.

```
Section TransfinitePrinciple.
Variables r s: Set.
Hypothesis wor: worder r.
Hypothesis u_stable: forall s', sub s' s -> inc (union s') s.
Hypothesis adj_stable:
 (forall x, inc x (substrate r)-> inc (segment r x) s -> inc (segmentc r x) s).
```

```
Lemma transfinite_principle1 x:  segmentp r x -> inc x s.
Lemma transfinite_principle2: inc (substrate r) s.
End TransfinitePrinciple.
```

We deduce [4, p. 151] C59. (Principle of transfinite induction). *Let* $\mathrm{R}\{x\}$ *be a relation in* $\mathcal{T}$ *(x not being a constant of* $\mathcal{T}$*) such that the relation*

$$(x \in \mathrm{E} \textit{ and } (\forall y)(y \in \mathrm{E} \textit{ and } y < x) \implies \mathrm{R}\{y\}) \implies \mathrm{R}\{x\}$$

is a theorem in $\mathcal{T}$*. Under these conditions, the relation* $(x \in \mathrm{E}) \implies \mathrm{R}\{x\}$ *is a theorem in* $\mathcal{T}$*.*

```
Theorem transfinite_principle r (p:property) (E:= substrate r):
  worder r ->
  (forall x, inc x E -> (forall y, inc y E -> glt r y x -> p y) -> p x) ->
  forall x, inc x E -> p x.
```

Definition by transfinite induction. In what follows we consider a well-ordered set E, and for $x \in \mathrm{E}$ the segment S_x (formed of elements $< x$). If f is a function we denote by $f^{(x)}$ the surjective restriction of f to S_x. Then:
Criterion C60 (Definition of a mapping by transfinite induction) [4, p. 151]: *Let u be a letter,* $\mathrm{T}\{u\}$ *a term in the theory* $\mathcal{T}$ *(in which* E *is a set well-ordered by a relation denoted* $\le$*). There exists a set* U *and a mapping* f *of* E *onto* U *such that for all* $x \in \mathrm{E}$ *we have* $f(x) = \mathrm{T}\{f^{(x)}\}$*. Furthermore the set* U *and the mapping* f *are uniquely determined by these conditions.*

We can convert the criterion into a theorem by assuming that T is of type `Set→Set`, so that $\mathrm{T}\{u\}$ can be rewritten as $\mathrm{T}(u)$, and the condition becomes $f(x) = \mathrm{T}(f^{(x)})$. Note that f is a mapping of E onto U says that f is a surjection and U is the target. So, if f exists, then U exists, if f is unique, then U is unique. In that follows we do not mention U. Note that a surjective function is uniquely determined by its graph, so that we consider a variant of C60, where g is a graph; moreover T takes a graph as argument. Denote by $g_{(x)}$ the restriction of a graph g to S_x. In this case, C60 says; there is a unique functional graph with domain E such that $g(x) = \mathrm{T}(g_{(x)})$ on E.

We explain here how to get a surjective function from a graph.

```
Definition fgraph_to_fun f:= triple (domain f) (range f) f.

Lemma fgraph_to_fun_ev f x: Vf (fgraph_to_fun f) x = Vg f x.
Lemma fgraph_to_fun_source f: source (fgraph_to_fun f) = domain f.
Lemma fgraph_to_fun_fs f: fgraph f -> surjection (fgraph_to_fun f).
Lemma fgraph_to_fun_restr f s:
   function f -> sub s (source f) ->
   restriction1 f s =  fgraph_to_fun (restr (graph f) s).
```

We show equivalence of the two variantqs of C60.

```
Definition transfinite_def r (T: fterm) f:=
  [/\ surjection f, source f = substrate r &
  forall x, inc x (substrate r) -> Vf f x = p (restriction1 f  (segment r x))].
Definition transfiniteg_def r (T: fterm) f :=
  [/\ fgraph f, domain f = substrate r &
  forall x, inc x (substrate r) -> Vg f x = T (restr_to_segment r x f)].

Lemma transfinite_def_prop1 r T f:
```

```
  transfiniteg_def r T f <->
  transfinite_def r (T \o graph)  (fgraph_to_fun f).
Lemma transfinite_def_prop2 r T f:
  transfinite_def r T f ->
  transfiniteg_def r (T \o fgraph_to_fun)  (graph f).
```

Uniqueness is easy (consider the least element for which the functions differ).

```
Lemma transfiniteg_unique r T : worder r ->
  uniqueness (transfiniteg_def r T).
Lemma transfinite_unique r T : worder r ->
  uniqueness (transfinite_def r T).
```

Existence. We fix here the well-ordered set E, the functional term T, and consider the property $\mathscr{I}(\mathrm{S}, f)$ that says that S is a segment of E, f a functional graph and $f(x) = \mathrm{T}(f_{(x)})$ whenever $x \in \mathrm{S}$. If S is the segment with endpoint x, $\mathrm{S}' = \mathrm{S} \cup \{x\}$, there is a graph f_+ such that $\mathscr{I}(\mathrm{S}', f_+)$ holds. If $\mathfrak{S}$ is a set of segments and $\mathscr{I}(\mathrm{S}, f_\mathrm{S})$ holds for every $\mathrm{S} \in \mathfrak{S}$, then $\mathscr{I}(\bigcup \mathfrak{S}, \bigcup f_\mathrm{S})$ holds. (proof: consider two segments S and S′, $x \in \mathrm{S} \cap \mathrm{S}'$: we may assume $\mathrm{S} \subset \mathrm{S}'$; then, by uniqueness f_S is the restriction of $f_{\mathrm{S}'}$ to S: hence $f_\mathrm{S}(x) = f_{\mathrm{S}'}(x)$; this shows that $\bigcup f_\mathrm{S}$ is a functional graph).

```
Lemma transfinite_aux2 r T s (tdf: fterm) : worder r -> (* 58 *)
  (alls s (segmentp r)) ->
  (forall z, inc z s ->  transfiniteg_def (induced_order r z) T (tdf z)) ->
  let f := (unionf s tdf) in
  transfiniteg_def (induced_order r (union s)) T f.
Lemma transfinite_aux3 r T x g:
  worder r ->  inc x (substrate r) ->
  transfiniteg_def (induced_order r (segment r x)) T g ->
  transfiniteg_def (induced_order r (segmentc r x)) T
    (g +s1 J x (T (restr g (segment r x)))).
```

Let $\mathfrak{S}$ be the set of all segments S such that there is f such that $\mathscr{I}(\mathrm{S}, f)$ holds. We use the axiom of choice, define a functional term f_S such that so that $\mathscr{I}(\mathrm{S}, f_\mathrm{S})$ holds for $\mathrm{S} \in \mathfrak{S}$. Then $\mathfrak{S}$ is stable by union, hence contains E, and $\mathscr{I}(\mathrm{E}, f_\mathrm{E})$ holds. This function f_E is the solution to the problem.

```
Definition transfiniteg_defined r T:= choose (fun f => transfiniteg_def r T f).

Lemma transfinite_exists1 r T:
   worder r -> exists f, (transfiniteg_def r T f).
Lemma transfinite_pr1 r T: worder r ->
  transfiniteg_def r T (transfiniteg_defined r T).
Lemma transfinite_pr2 r x T:
  worder r -> transfiniteg_def r T x -> transfiniteg_defined r T = x.
```

Application: Consider two well-ordered sets E and E′. There is an order morphism $\mathrm{E} \to \mathrm{E}'$ whose image is a segment of E′, or there is an order morphism $\mathrm{E}' \to \mathrm{E}$ whose image is a segment of E. Proof. Define by transfinite induction a graph f such that $f(x) = \inf(\mathrm{E}' - f\langle \mathrm{S}_x \rangle)$. Let A be the set of all x such that $f\langle \mathrm{S}_x \rangle$ is a strict subset of E′. If $x \in \mathrm{A}$ then $f(x)$ is the least element of E′ not of the form $f(y)$ for $y < x$. It follows that A is a segment of E, that f is strictly increasing on A, and $f\langle \mathrm{A} \rangle$ is a segment of E′. Assume $\mathrm{A} = \mathrm{E}$; the property holds by converting f into a function $\mathrm{E} \to \mathrm{E}'$. Otherwise, there is a least element b in E not in A. We have $f\langle S_b\rangle = E'$; in order terms E' is the set of all $f(x)$ for $x \in A$. So f, restricted to A, can be considered as an order isomorphism $A \to E'$, its inverse is an order isomorphism $E' \to A$, that can be converted into an order morphism $E' \to E$ whose image is A. Note: the value of $f(b)$ is irrelevant in this case,

```
Lemma isomorphism_worder_exists r r': worder r -> worder r' ->
  let iso:= (fun u v f =>
    segmentp v (Imf f) /\ order_morphism f u v) in
  (exists f,iso r r' f) \/ (exists f, iso r' r f). (* 97 *)
```

Criterion C60 in the case of a function follows from the case of a graph.

Bourbaki notes that in a situation, « where there exists a set F such that *for every mapping h of a segment of* E *onto a subset of* F *we have* $T\{h\} \in F$ then the set U obtained by applying C60 is a *subset of* F » Proof: redo the proof of C60 with a more restrictive property than $\mathscr{I}(S, f_S)$. Simpler proof: check by transfinite induction that the function defined by C60 satisfies $f(x) \in F$, whatever x. Note: yin this special case, the axiom of choice is not needed.

```
Definition transfinite_defined r T:=
  fgraph_to_fun (transfiniteg_defined r (fun f => T (fgraph_to_fun f))).

Lemma transfinite_defined_pr r T: worder r ->
  transfinite_def r T  (transfinite_defined r T).
Lemma transfinite_pr r x T:
   worder r -> transfinite_def r T x -> transfinite_defined r T = x.
Theorem transfinite_definition r T:
  worder r -> exists! f, (transfinite_def r T f).
Theorem transfinite_definition_stable r T F:
  worder r ->
  (forall f, function f -> segmentp r (source f) -> sub (target f) F ->
    inc (T f) F) ->
  sub (target (transfinite_defined r T)) F.
```

Given a well-ordering r on E, we may consider the function f, defined by transfinite induction with the functional "target" ; note that its graph is defined by transfinite induction via "range". We restate here uniqueness in the following way: assume that $g(x)$ a functional term such that that $g(x) = \{g(t), t < x\}$, for $x \in E$. Then $f(x) = g(x)$ on R.

```
Definition ordinal_iso r := transfinite_defined r target.
Definition ordinal_isog r := transfiniteg_defined r range.

emma transdef_tg0 r (f := ordinal_isog r): worder r ->
  forall x, inc x (substrate r) ->
  Vg f x = direct_image f (segment r x).
Lemma transdef_tg1 r  (f := ordinal_iso r):  worder r ->
  forall x, inc x (substrate r) ->
  Vf f x = Vfs f (segment r x).
Lemma transdef_tg2 r f:
  worder r -> surjection f -> source f = substrate r ->
  (forall x, inc x (substrate r) -> Vf f x = Vfs f (segment r x)) ->
  f = ordinal_iso r.
Lemma transdef_tg3 r (f: fterm):
  worder r ->
  (forall x, inc x (substrate r) -> f x = fun_image (segment r x) f) ->
  forall x, inc x (substrate r) -> f x =  Vf (ordinal_iso r) x.
```

3.3 Zermelo's theorem

We show here that every set E is the substrate of a well-order (we have already given a proof of it; the following uses the original arguments of Zermelo). The proof depends on some mysterious set $\mathfrak{S}$ (a set of subsets of E) and a mysterious function $p(X)$ defined on $\mathfrak{S}$.

If G and G′ are two orders on E and E′ (denoted $\le$ and $\le'$), we denote by $G \otimes G'$ the set of all x in E and E′ such that the set S of y such that $y < x$ is the set of all y such that $y <' x$; moreover, we assume that $a \le b$ is equivalent to $a \le' b$ for a and b in S.

Obviously $G \otimes G' = G' \otimes G$, and this is a segment for $\le$ and $\le'$.

```
Definition common_ordering_set r r' :=
  Zo ((substrate r) \cap (substrate r'))
  (fun x => segment r x = segment r' x /\
    induced_order r (segment r x) = induced_order r' (segment r' x)).

Lemma Zermelo_aux0 r r':
  common_ordering_set r r' = common_ordering_set r' r.
Lemma Zermelo_aux1 r r': worder r ->  worder r' ->
   segmentp r (common_ordering_set r r').
Lemma Zermelo_aux2 r r' v: worder r ->  worder r' ->
  v = common_ordering_set r r' ->  sub(induced_order r v)(induced_order r' v).
```

Let Q(G) denote the following property: G is a well-ordering on a set A such that $A \subset E$ and for any $a \in A$, the segment S_a for G satisfies $S_a \in \mathfrak{S}$ and $p(S_a) = a$.

```
Definition Zermelo_axioms E p s r:=
  [/\ worder r,
  sub (substrate r) E,
  (forall x, inc x (substrate r) -> inc (segment r x) s) &
  (forall x, inc x (substrate r) -> p (segment r x)  = x)].
```

Given G and G′ two ordering on A and A′ we write $q(G, G')$ as short for: $A \subset A'$, on A, the two orders G and G′ coincide, and A is segment of G′.

If both Q(G) and Q(G′) hold, then one of $q(G, G')$ and $q(G', G)$ hold. Let $V = G \otimes G'$. This is a segment for G; the result is clear if $V = A$. Otherwise assume $V = S_x$; we get $x = p(V)$. Since V is also a segment of G′, we have $V = A'$ (and the result holds) or $V = S_y$ (for some $y \in B$) and $y = p(V)$. This implies $x = y \in V$, absurd.

```
Lemma Zermelo_aux3 E s p r r':
  let q := fun r r' => [/\ sub (substrate r) (substrate r'),
     r = induced_order r' (substrate r) & segmentp r' (substrate r)] in
    Zermelo_axioms E p s r -> Zermelo_axioms E p s r' ->
    q r r' \/ q r' r.
```

Let G be a well-ordering on A and a an element not in the substrate, G_a the extension of G that makes a the greatest element. This is a well-ordering, and a segment S_x of G_A is either A (if $x = a$) or S_x (as a segment of G).

```
Lemma Zermelo_aux4 r a (owg := order_with_greatest r a):
  worder r ->  ~ (inc a (substrate r)) ->
    [/\ worder owg,  segment owg a = (substrate r) &
      forall x, inc x (substrate owg) -> x = a \/
      segment owg x = segment r x].
```

We assume now that E is a set, $\mathfrak{S} \subset \mathfrak{P}(E)$ and $p(X) \not\in X$ whenever $X \in \mathfrak{S}$. Consider the set $\mathfrak{M}$ of all G that satisfy Q. Lemma `Zermelo_aux3` says that the union G of these orderings is an ordering. It satisfies Q and its substrate M is not in $\mathfrak{S}$ (for otherwise, we could chose $a = p(M)$, and extend G with a as greatest element (because $a \not\in M$); this extension satisfies Q, absurd).

```
Lemma Zermelo_aux E s p: sub s (\Po E) ->  (* 59 *)
  (forall x, inc x s -> inc (p x) E) /\ ~ (inc (p x) x)) ->
  exists2 r, Zermelo_axioms E p s r & (~ (inc (substrate r) s)).
```

Let now $\mathfrak{S}$ be the set of all subsets of E but E itself. If p is the representative of the complement of x in E (this exists, by the axiom of choice), the order defined by the lemma `Zermelo_aux` has its substrate in $\mathfrak{P}(E) - \mathfrak{S}$. Thus, it is a well-ordering on E. This is Theorem 1 [4, p. 153].

```
Theorem Zermelo E: exists r, worder_on r E.
```

3.4 Inductive sets

An ordered set is said to be *inductive* if every totally ordered subset of E has an upper bound in E. More precisely, let r be an order and E its substrate, then every subset X of E, for which the order induced by r is total, has an upper bound for r. The set $\Phi(A,B)$ of partial functions is inductive, see page 508.

```
Definition inductive r :=
  forall X, sub X (substrate r) -> total_order (induced_order r X) ->
    exists x, upper_bound r X x.

Lemma inductive_graphs a b:
  inductive (opp_order (extension_order a b)).
```

Consider an ordered set E; assume that each well-ordered subset of E is bounded above. Let $\mathfrak{S}$ be the set of sets $S \subset E$ that have a strict upper bound. We choose for $p(S)$ an upper bound of S that is not in S. By `Zermelo_aux`, there is a well-ordering G satisfies Q, i.e. G is a well-ordering of a subset M of E, and if S_x denotes the segment of G with endpoint x, we have $S_x \in \mathfrak{S}$ and $p(S_x) = x$. This last condition says that if $y < x$ for G, it is true for the ordering on E; hence G is the restriction to M of the ordering of E. By assumption, M is bounded, say by m. This element is maximal (this is Proposition 4 [4, p. 154]).

Theorem 2 [4, p. 154] says that every inductive ordered set has a maximal element. This is a trivial consequence of the previous result.

```
Theorem Zorn_aux r: order  r ->
  (forall s, sub s (substrate r) -> worder (restriction_order r s) ->
    (bounded_above r s)) ->
  exists a, maximal r a.
Theorem Zorn_lemma r: order  r -> inductive r ->
  exists a, maximal r a.
```

Corollary. If E is inductive, $a \in E$, F is the set of all $x \geq a$, then F is inductive (if X is a totally ordered set in F, then $X \cup \{a\}$ is totally ordered; an upper bound m is in F since it satisfies

$a \leq m$). Hence there is a maximal element m such that $a \leq m$. Second corollary: if $\mathfrak{F}$ is a subset of the powerset of E such that for every subset $\mathfrak{G}$ of $\mathfrak{F}$ which is totally ordered by inclusion, the union (resp. intersection) of the sets of $\mathfrak{G}$ belongs to $\mathfrak{F}$, then $\mathfrak{F}$ has a maximal or minimal element.[2]

```
Lemma inductive_max_greater r a: order r ->  inductive r ->
  inc a (substrate r) ->
  exists2 m, maximal r m & gle r a m.
Lemma setP_inductive  A F: sub A (\Po F) ->
  (forall S, (forall x y, inc x S -> inc y S -> sub x y \/ sub y x) ->
    sub S A ->inc (union S) A) ->
  inductive (sub_order A).
Lemma setP_maximal A F: sub A (\Po F) ->
  (forall So, (forall x y, inc x So -> inc y So -> sub x y \/ sub y x) ->
    sub So A -> inc (union So) A) ->
  exists a, maximal (sub_order A) a.
Lemma setP_minimal A F: sub A (\Po F) -> nonempty A ->
  (forall So, (forall x y, inc x So -> inc y So -> sub x y \/ sub y x) ->
    sub So A -> nonempty So -> inc (intersection So) A) ->
  exists a, minimal (sub_order A) a.
```

3.5 Isomorphisms of well-ordered sets

Assume that E and F are two well-ordered sets. We show Theorem 3 [4, p. 155]: Let $\mathrm{I}(u, v, f)$ be the property that f is an order isomorphism from u onto a segment w of v. We claim that there exists a unique f such that $\mathrm{I}(\mathrm{E}, \mathrm{F}, f)$, or there exists a unique f such that $\mathrm{I}(\mathrm{F}, \mathrm{E}, f)$. Note: The two cases are not excluded; in that case, E and F are order-isomorphic.

In order to show uniqueness we start with a lemma: if f is increasing and g is strictly increasing, if the image of f is a segment of F, then $f(x) \leq g(x)$ for all x. The proof is by contradiction. If a is the least element such that $g(a) < f(a)$, since the image of f is a segment there is a z such that $g(a) = f(z)$. Since f is increasing this gives $z < a$, hence $f(z) \leq g(z) < g(a)$, absurd.

```
Lemma increasing_function_segments r r’ f g:
  worder r -> worder r’ ->
  increasing_fun f r r’ -> strict_increasing_fun g r r’->
  segmentp r’ (Imf f) ->
  forall x, inc x (source f) -> gle r’ (Vf f x) (Vf g x).
Lemma isomorphism_worder_unique r r’ x y:
  worder r -> worder r’ ->
  segmentp r’ (Imf x) ->  segmentp r’ (Imf y) ->
  order_morphism x r r’ -> order_morphism y r r’ ->
  x = y.
```

Given a totally ordered subset X of $\mathfrak{F}$, we can apply lemma `sup_extension_order2`, that says that there exists a function f that extends all elements in X; we know that the source and range of f are the union of the sources and ranges of the elements of X, hence are segments. Given a and b in the source of f, there is a function g that is defined for both a and b (because X is totally ordered); since $a \leq b$ is equivalent to $g(a) \leq g(b)$ and $f(a) = g(a)$ and $f(b) = g(b)$

[2] in the case of intersection, we assume $\mathfrak{G}$ nonempty; for otherwise the intersection is empty, $\emptyset \in \mathfrak{F}$, and the result is trivial.

we deduce that $a \leq b$ is equivalent to $f(a) \leq f(b)$. As a consequence, f is increasing and hence is a morphism. Consider now a maximal element f. If the source of f is E, then I(E, F, f) is true. If the range of f is F, then f^{-1} is a bijection from F onto a subset of E, hence I(F, E, f^{-1}). Otherwise, if a is the least element of E not in the source of f and b the least element not in the range of b, we can extend f to a function g by saying $g(a) = b$. This function is in $\mathfrak{F}$. This contradicts the maximality of f.

```
Theorem isomorphism_worder r r': (* 160 *)
 worder r -> worder r' ->
 let iso:= (fun u v f =>
    segmentp v (Imf f) /\ order_morphism f u v) in
  (exists! f,iso r r' f) \/ (exists! f, iso r' r f).
```

Corollary 1. The only isomorphism from a well-ordered set into a segment of itself is the identity.

```
Lemma unique_isomorphism_onto_segment r f: worder r ->
  segmentp r (Imf f) ->  order_morphism f r r ->
  f = identity (substrate r).
```

Corollary 2. If E and F are two well-ordered sets, f an isomorphism of E onto a segment of F, g an isomorphism of F onto a segment of E, then f and g are inverse bijections.

```
Lemma bij_pair_isomorphism_onto_segment r r' f f':
  worder r -> worder r' ->
  segmentp r' (Imf f) ->  order_morphism f r r' ->
  segmentp r (Imf f') ->  order_morphism f' r' r ->
  (order_isomorphism f r r' /\ order_isomorphism f' r' r /\
    f = inverse_fun f').
```

Finally, we show that every subset of a well-ordered set is isomorphic to a segment of E.

```
Lemma isomorphic_subset_segment r a:
  worder r -> sub a (substrate r) ->
  exists w, exists f, segmentp r w /\
    order_isomorphism f (induced_order r a) (induced_order r w).
```

If f is injective (resp. bijective) and satisfies $f(x) \leq f(y)$ then f is an order morphism (resp. isomorphism) if the source is totally ordered. We refine the theorem about well-ordered set isomorphisms: given two well-ordered sets E and E′, either E is isomorphic to E′, or E isomorphic to a strict segment of E′ or E′ isomorphic to a strict segment of E. There is an other variant: either E is isomorphic to a strict of E′ or E′ isomorphic to a strict segment of E. Two isomorphic segments of E are equal.

```
Lemma segments_iso1 a A B: worder a -> sub A B ->
  segmentp a A -> segmentp a B ->
  (induced_order a B) \Is (induced_order a A) -> A = B.
Lemma segments_iso2 a A B: worder a ->
   inc A (segments a) -> inc B (segments a)  ->
   (induced_order a A)  \Is (induced_order a B) -> A = B.
Lemma isomorphism_worder2 r r':
   worder r -> worder r' ->
  [\/ r \Is r',
```

```
   (exists2 x, inc x (substrate r) & (induced_order r (segment r x)) \Is r') |
   (exists2 x, inc x (substrate r')& (induced_order r' (segment r' x)) \Is r)].
Lemma isomorphism_worder3 r r': worder r -> worder r' ->
  (exists f, segmentp r' (Imf f) /\ order_morphism f r r')
  \/ (exists2 x, inc x (substrate r) & (induced_order r (segment r x)) \Is r').
```

3.6 Lexicographic products

Given a family of orders $(G_\iota)_{\iota\in I}$, with substrate X_ι, and order relation $\leq_\iota$, and a well-order relation $\leq_I$ on I, we consider the relation $x \leq y$ on the product $\prod_{\iota\in I} X_\iota$ defined by "$x = y$ or, if ι is the least index (for the relation $\leq_I$) such that $x_\iota \neq y_\iota$ then $x_\iota \leq_\iota y_\iota$". The graph of this relation will be called the *lexicographic product* of the family.

```
Definition order_prod_r r g :=
  fun x x' =>
    forall j, least (induced_order r (Zo (domain g)
      (fun i =>  Vg x i <> Vg x' i))) j -> glt (Vg g j) (Vg x j)(Vg x' j).

Definition orprod_ax r g:=
  [/\ worder r, substrate r = domain g & order_fam g].
Definition order_prod r g :=
  graph_on (order_prod_r r g)(prod_of_substrates g).
```

We have $x < y$ if there is an index i such that $x_i <_i y_i$, and $x_j = y_j$ whenever $j <_I i$ (this characterization is more convenient than the definition).

```
Lemma orprod_sr r g:
  orprod_ax r g ->
  substrate(order_prod r g) = prod_of_substrates g.
Lemma prod_of_substrates_pr i z g:
  inc i (domain g) -> inc z (prod_of_substrates g) ->
  inc (Vg i z) (substrate (Vg i g)).
Lemma orprod_gle1P r g:
  orprod_ax r g -> forall x x',
  (related (order_prod r g) x x' <->
  [/\ inc x (prod_of_substrates g),  inc x' (prod_of_substrates g) &
    forall j, least (induced_order r (Zo (domain g)
      (fun i =>  Vg x i <> Vg x' i))) j -> glt (Vg g j) (Vg x j)(Vg x' j)]).
Lemma orprod_gleP r g:
  orprod_ax r g -> forall x x',
  (gle (order_prod r g) x x' <->
  [/\ inc x (prod_of_substrates g), inc x' (prod_of_substrates g) &
    (x= x' \/ exists j, [/\ inc j (substrate r),
      glt (Vg g j) (Vg x j) (Vg x' j) &
      forall i, glt r i j -> Vg x i = Vg x' i])]).
```

The lexicographic product is an order.

```
Lemma orprod_osr r g:
  orprod_ax r g -> order_on (order_prod r g) (prod_of_substrates g).
Lemma orprod_sr r g:
  orprod_ax r g -> substrate(order_prod r g) = prod_of_substrates g.
Lemma orprod_or r g:
  orprod_ax r g -> order (order_prod r g).
```

If all orders are total so is the lexicographic product.

```
Lemma orprod_total r g:
  orprod_ax r g ->
  (allf g total_order) ->
  total_order (order_prod r g).
```

The ordinal sum. Consider as above a family $(G_\iota)_{\iota\in I}$ of orders, with substrate X_ι and order relation $\le_\iota$, and an order relation $\le_I$ on I. Let $E = \sum_{\iota\in I} X_\iota$ be the disjoint union. This is the set of all pairs (a, b) where $b \in I$ and $a \in X_b$.

```
Definition sum_of_substrates g := disjointU (fam_of_substrates g).

Lemma du_index_pr1 g x: inc x (sum_of_substrates g) ->
  [/\ inc (Q x) (domain g),  inc (P x) (substrate (Vg g (Q x))) & pairp x].
Lemma disjoint_union_pi1 g x y:
  inc y (domain g) -> inc x (substrate (V g y)) ->
  inc (J x y) (sum_of_substrates g).
Lemma canonical2_substrate r r':
  fam_of_substrates (variantLc r r') = Lvariantc (substrate r) (substrate r').
```

We consider the relation $x \le y$ defined on E by “either $x_2 <_I y_2$, or $x_2 = y_2 = \lambda$ and then $x_1 \le_\lambda y_1$” where $x = (x_1, x_2)$ and $y = (y_1, y_2)$. In Exercise 1.3 (page 519) it is assumed $X_j \neq \emptyset$, but this forbids zero in a sum. We show here that this definition induces an order on the disjoint union.

```
Definition orsum_ax r g:=
  [/\ order r, substrate r = domain g & order_fam g].

Definition order_sum_r r g x x' :=
    (glt r (Q x) (Q x') \/ (Q x = Q x' /\ gle (V g (Q x)) (P x) (P x'))).
Definition order_sum r g :=
  graph_on (order_sum_r r g) (sum_of_substrates g).

Section OrderSumBasic.
Variables r g: Set.
Hypothesis osa: orsum_ax r g.

Lemma orsum_or: order (order_sum r g).
Lemma orsum_sr:
  substrate (order_sum r g) = sum_of_substrates g.
Lemma orsum_osr: order_on (order_sum r g) (sum_of_substrates g).

Lemma orsum_gleP x x':
  gle (order_sum r g) x x' <->
  [/\ inc x (sum_of_substrates g), inc x' (sum_of_substrates g) &
    order_sum_r r g x x'].
Lemma orsum_gle1 x x':
  gle (order_sum r g) x x' ->
  (glt r (Q x) (Q x') \/ (Q x = Q x' /\ gle (V (Q x) g) (P x) (P x'))).
Lemma orsum_gle2  a b a' b':
  gle (order_sum r g) (J a b) (J a' b')  ->
  (glt r b b' \/ (b = b' /\ gle (V g b) a a')).
Lemma orsum_gle_id x x':
```

```
  gle (order_sum r g) x x' -> gle r (Q x) (Q x').
End OrderSumBasic.
```

We consider now the case of the sum and product of two sets. This operation is non-commutative, and we shall use our canonical doubleton as ordering. (The canonical doubleton has two elements `C0` and `C1`, also known as 0 and 1, ordered by $0 < 1$). Note the ordering of the product: it is so that $x + x = x \cdot 2$.

```
Definition order_prod2 r r' :=
  order_prod canonical_doubleton_order (variantLc r' r).

Definition order_sum2 r r' :=
  order_sum canonical_doubleton_order (variantLc r r').

Lemma order_sp_axioms r r':
  order r -> order r' -> order_fam  (variantLc r r').
Lemma cdo_glt1: glt canonical_doubleton_order C0 C1.

Section OrderSum2Basic.
Variables r r': Set.
Hypotheses (or: order r) (or': order r').

Lemma orsum2_osr:
  order_on (order_sum2 r r') (canonical_du2 (substrate r) (substrate r')).
Lemma orprod2_osr:
  order_on (order_prod2 r r')(product2 (substrate r')  (substrate r)).
Lemma orsum2_or: order (order_sum2 r r').
Lemma orprod2_or: order (order_prod2 r r').
Lemma orsum2_sr:
  substrate (order_sum2 r r') = canonical_du2 (substrate r) (substrate r').
Lemma orprod2_sr:
  substrate (order_prod2 r r') = product2 (substrate r') (substrate r).
```

The ordering on $\mathrm{E}_1 + \mathrm{E}_2$ is defined by $x \leq_s y$ if and only if either $\mathrm{pr}_2 x = \mathrm{pr}_2 y = \alpha$ and $\mathrm{pr}_1 x \leq \mathrm{pr}_1 y$ (in E_1), or $\mathrm{pr}_2 x = \mathrm{pr}_2 y = \beta$ and $\mathrm{pr}_1 x \leq \mathrm{pr}_1 y$ (in E_2), or $\mathrm{pr}_2 x = \alpha$ and $\mathrm{pr}_2 y = \beta$. Note that $u = \beta$ can be replaced by $u \neq \alpha$.

In the case of a product $\mathrm{E} \cdot \mathrm{I}$, we have $x < y$ if either $x_\alpha < y_\alpha$ in I or if $x_\alpha = y_\alpha$, and $x_\beta < y_\beta$ in E.

```
Lemma orsum2_gleP x x':
  gle (order_sum2 r r') x x' <->
  [/\ inc x (canonical_du2 (substrate r) (substrate r')),
    inc x' (canonical_du2 (substrate r) (substrate r')) &
    ([/\ Q x = C0, Q x' = C0 & gle r (P x) (P x')]
      \/ [/\ Q x <> C0, Q x' <> C0 & gle r' (P x)
Lemma orsum2_gle_spec  x x':
  inc x (substrate r) -> inc x' (substrate r') ->
  glt (order_sum2 r r') (J x C0) (J x' C1).
Lemma orsum2_gleP x x':
  gle (order_sum2 r r') x x' <->
  [/\ inc x (canonical_du2 (substrate r) (substrate r')),
    inc x' (canonical_du2 (substrate r) (substrate r')) &
    [\/ [/\ Q x = C0, Q x' = C0 & gle r (P x) (P x')],
        [/\ Q x <> C0, Q x' <> C0 & gle r' (P x) (P x')] |
```

```
        (Q x = C0 /\ Q x' <> C0)]].
End OrderSum2Basic.
```

Exercise 2.10 (page 578) says that $E \cdot I$ is isomorphic to the sum $\sum_{\iota \in I} E_\iota$ where each E_ι is equal to E. The isomorphism is simply $x \mapsto (x_\beta, x_\alpha)$.

```
Lemma order_prod_pr r r': order r -> order r' ->
  (order_prod2 r r') \Is (order_sum r' (cst_graph (substrate r') r)).
```

If I and each E_i are well-ordered, so is the ordinal sum. The sum of two totally ordered sets is totally ordered (for the converse, see Exercises).

```
Lemma orsum2_totalorder r r':
  total_order r -> total_order r' -> total_order (order_sum2 r r').
Lemma orprod2_totalorder r r':
  total_order r -> total_order r' -> total_order (order_prod2 r r').

Lemma orsum_wor r g:
  worder r -> substrate r = domain g ->
  worder_fam g ->
  worder (order_sum r g).
```

Chapter 4

Equipotent Sets. Cardinals

Bourbaki denotes by Eq(X, Y) the property that there is a bijection between X and Y and denotes by Card(X) the set $\tau_Z(\mathrm{Eq}(X,Z))$. He calls this *the cardinal of* X. He does not define "a cardinal". The only possible interpretation of "$\mathfrak{x}$ is a cardinal" is "the object $\mathfrak{x}$ is of the form Card(E) for some set E". Using a specific font for cardinals suggests that a cardinal is some special object, and that we should perhaps introduce a type for these cardinals. If $A = \{\emptyset\}$ and $\mathfrak{a}$ denotes the cardinal of A, it is impossible to prove $\mathfrak{a} = A$ or $\mathfrak{a} \neq A$ (non-definiteness of τ).

The cardinal of a set is sometimes called the *the power of* X. It is interesting to notice that no name is given to the notation $\mathfrak{a}^{\mathfrak{b}}$ when this means the cardinal of the set of mappings from one set into another (the operation is nevertheless called "exponentiation of cardinals"). The term "power" is used only in the phrase "power of the continuum", where it means the cardinal of the set of real numbers, or, equivalently, the cardinal of $\mathfrak{P}(\mathbf{N})$ (where $\mathbf{N}$ is the set of natural integers, defined in Chapter 6). For us, the term "power" will only be used to denote $\mathfrak{a}^{\mathfrak{b}}$.

One can define addition and multiplication of cardinals. For instance, $A\times B$ is equipotent to $A'\times B'$ when A is equipotent to A′ and B is equipotent to B′. Thus $\mathrm{Card}(A\times B) = \mathrm{Card}(A'\times B')$ if $\mathrm{Card}(A) = \mathrm{Card}(A')$ and $\mathrm{Card}(B) = \mathrm{Card}(B')$. For instance, if $\mathfrak{a}$ is as above then $\mathfrak{a}.\mathfrak{a} = \mathfrak{a}$. In order to prove properties of these operations (like associativity, commutativity), one needs the notion of a family of cardinals, which is some f that associates to each element i of a given set I a cardinal f_i. Technically, f is a functional graph, and its range is a set. This means that we can consider a set of cardinals, so that a cardinal is a set. One could define the cardinal product $\mathfrak{a}.\mathfrak{b}$ as the cardinal of $A\times B$, whenever $\mathrm{Card}(A) = \mathfrak{a}$ and $\mathrm{Card}(B) = \mathfrak{b}$. Since $\mathfrak{a}$ and $\mathfrak{b}$ are sets that satisfy these conditions, it is simpler to define it as the cardinal of $\mathfrak{a}\times\mathfrak{b}$. This definition then makes sense for any two sets; moreover A.B = B.A, even when A and B are not cardinals.

In the Exercises, Bourbaki denotes by $\mathrm{Is}(\Gamma,\Gamma')$ the property that Γ and Γ' are ordered sets, and there is an order isomorphism between Γ and Γ', he denotes by $\mathrm{Ord}(\Gamma)$ the ordered set $\tau_\Delta(\mathrm{Is}(\Gamma,\Delta))$, and calls it the *order-type* of Γ. He defines an *ordinal* as the order-type of a well-ordered set (the cardinal of any set is a cardinal, the order-type of an ordered set is not always an ordinal). Ordinals are generally denoted by lower-case Greek letters such as ω. The ordinal sum and lexicographic product of orderings induce two operations (sum and product) on the family of order-types, denoted by $\lambda+\mu$ and $\lambda\mu$. These operations are non-commutative: for instance $\lambda+1$ and $1+\lambda$ correspond to the orderings obtained by adjoining a greatest and a least element, respectively. The relation "there is an order isomorphism between Γ and a sub-ordering of Γ'", denoted by $\Gamma \prec \Gamma'$, is a preorder. The sum and product of ordinals are ordinals, and the relation $\lambda \prec \mu$ is a well-ordering, compatible with the two operations. Let $A = \{\emptyset\}$ be as above; there is a unique ordering on this set, which is a well-ordering. Let λ be its ordinal. The support of λ is a singleton, but it is undecidable whether or not the support is A.

In 1923 von Neumann noted that the axiom of choice (i.e. the use of τ) is not needed for defining ordinals. To each well-ordered set $(E, \leq)$ is uniquely associated a numeration, and hence a set F, with a natural ordering $o(F)$, and $(F, o(F))$ is isomorphic to $(E, \leq)$. The von Neumann ordinal of the ordered set $(E, \leq)$ is the set F. It is equipotent to E. Zermelo's theorem asserts that any set E has a well-ordering, so that one can consider the least ordinal equipotent to E. This is called the von Neumann cardinal of E. Let $A = \{\emptyset\}$ be as above; the von Neumann ordinal of the unique ordering of A is A itself, and the von Neumann cardinal of A is A.

In this chapter, we shall use the von Neumann point of view. We shall introduce the notion of finite set, and see that any finite ordinal is a cardinal. In a future chapter, we shall see that if X is an infinite set, then X and $X \times X$ are equipotent. This can be restated as: if A is an infinite cardinal, then $A = A.A$. This result is equivalent to the axiom of choice: if there is no choice function on X, then X has no cardinal, according to von Neumann.

This chapter starts with a study of some properties of ordinals. Sections 1 to 6 correspond to §3.1 to §3.6 of Chapter III of Bourbaki.

Ordinals. Consider a relation between x and y denoted here $x < y$. We say that it is *irreflexive* if $x < x$ is false. We say that the relation is *asymmetric* if at least one of $x < y$, $y < x$ is false. An asymmetric relation is obviously antisymmetric and irreflexive. We say that the relation is irreflexive on E or asymmetric on E if the previous relations are true for $x \in E$, and $y \in E$. We say that the relation is a *strict well-ordering* (on E) if it is asymmetric and if "$a < b$ or $a = b$" is a well-ordering (on E).

If E is any set, we denote by $o(E)$ the relation "$x \in E$ and $y \in E$ and $x \subset y$". We know that this is an order on E. We denote by $o'(E)$ the relation "$x \in E$ and $y \in E$ and $x \in y$ or $x = y$". This is an order on E if $\in$ is transitive and antisymmetric

A set E is said to be *transitive* if $a \in b$ and $b \in E$ implies $a \in E$, it is *irreflexive* if $E \not\in E$, it is *decent* if all elements of E are irreflexive, it is *asymmetric* if one of $x \in y$ and $y \in x$ is false. If E is transitive and asymmetric, then $o'(E)$ is an order on E.

The set $x \cup \{x\}$, denoted by x^+, will be called the ordinal successor of x. If x is irreflexive, then $x \neq x^+$.

```
Definition transitive_set E:= forall x, inc x E -> sub x E.
Definition decent_set E := forall x, inc x E -> ~ (inc x x).
Definition trans_dec_set E := transitive_set E /\ decent_set E.
Definition asymmetric_set E :=
  forall  x y, inc x E -> inc y E -> inc x y -> inc y x -> False.
Definition ordinal_oa :=  graph_on (fun a b => inc a b \/ a = b).
Definition ordinal_o := sub_order.
Definition osucc x := x +s1 x.
```

We prove here that if X is set of transitive and decent sets, so is the union and intersection of the family, as well as the successor of each member.

```
Lemma succ_i x: inc x (osucc x).
```

```
Lemma transitive_setU x: alls x transitive_set -> transitive_set (union x).
Lemma trans_dec_setU x: alls x trans_dec_set -> trans_dec_set (union x).
Lemma trans_dec_setI x: alls x trans_dec_set -> trans_dec_set (intersection x).
Lemma trans_dec_succ y: trans_dec_set y -> trans_dec_set (osucc y).
```

Definition of ordinals. There are different ways of defining an ordinal E.

- The definition of Krivine [14] is very basic. It is: E is transitive, $\in$ is transitive, E is asymmetric, and $\in$ satisfies the properties of a well-ordering.
- An equivalent form of above: E is asymmetric and transitive, $o'(\mathrm{E})$ is a well-ordering.
- If E is an ordinal, then $o(\mathrm{E})$ and $o'(\mathrm{E})$ are the same orderings.
- The definition of von Neumann (see [23]) is: $o(\mathrm{E})$ is a well-ordering, and $\mathrm{S}_x = x$ for $x \in \mathrm{E}$, where S_x is the segment with endpoint x. An easy consequence is that $o(\mathrm{E}) = o'(\mathrm{E})$.
- The Bourbaki definition (that we shall adopt) is: any transitive subset of E is either E or an element of E.

```
Definition ordinalp X:=
  forall Y, sub Y X -> transitive_set Y -> Y <> X -> inc Y X.

Notation "f =1o g" := (forall x, ordinalp x -> f x = g x)
 (at level 70, format "'[hv' f '/ '  =1o  g ']'", no associativity).
Definition ordinal_fam g := allf g ordinalp.
Definition ordinal_set E := alls E ordinalp.
```

It is easy to show that if x is an ordinal, then x is transitive, decent, irreflexive, and its successor is an ordinal.

```
Lemma OS_succ x: ordinalp x ->  ordinalp (osucc x).
Lemma ordinal_trans_dec x:  ordinalp x -> trans_dec_set x.
Lemma ordinal_transitive x: ordinalp x -> transitive_set x.
Lemma ordinal_decent x:  ordinalp x -> decent_set x.
Lemma ordinal_irreflexive x: ordinalp x -> ~ (inc x x).
```

As a consequence, if y is an ordinal, then $x \in y$ implies that x is a strict subset of y. The converse holds if x is transitive (in particular if x is an ordinal). We deduce: if z is an ordinal, then $x \in z^+ \implies x \subset z$, and if x is an ordinal, then $x \in z^+ \iff x \subset z$.

```
Lemma ordinal_sub x y:
  ordinalp x -> ordinalp y -> sub x y ->
  x = y \/ inc x y.
Lemma ordinal_sub2 x y: ordinalp y ->
  inc x y -> ssub x y.
Lemma ordinal_sub3 x y: ordinalp y ->
  inc x (osucc y) -> sub x y.
Lemma ordinal_sub4P x y: ordinalp x ->  ordinalp y ->
  (sub x y <-> inc x (osucc y)).
```

Consider a non-empty set X of ordinals, and its intersection Y. This is a transitive and decent set. The relation $Y \not\in Y$ says that for some $A \in X$ we have $Y \not\in A$. Since Y is a transitive subset of A we get $Y = A$. We restate this as $Y \in X$. In the case where X has two elements, x and y, this says that $x \cap y$ is either x or y, or equivalently that $x \subset y$ or $y \subset x$. It follows one of $x \in y$, $y \in x$, $x = y$.

```
Lemma ordinal_setI x: nonempty x -> ordinal_set x ->
  inc (intersection x) x.
Lemma ordinal_trichotomy x y:
  ordinalp x -> ordinalp y ->
  [\/ inc x y, inc y x | x = y].
```

A transitive set X whose elements are ordinals is an ordinal. The elements of an ordinal are ordinals. We deduce that if x^+ is an ordinal, then x is an ordinal.

```
Lemma ordinal_pr x: transitive_set x -> ordinal_set x -> ordinalp x.
Lemma ordinal_hi x y: ordinalp x -> inc y x -> ordinalp y.
Lemma oset_ordinal x: ordinalp x -> ordinal_set x.
Lemma OS_succr x: ordinalp (osucc x) -> ordinalp x.
```

Consequences. We say that a property p is " not collectivizing" when there is no set E containing all objects satisfying p, this is the same as: there is no set E formed of exactly those objects satisfying p. To be an ordinal is not collectivizing, for the alleged set of all ordinals is itself an ordinal (it is a transitive set of ordinals) so is irreflexive, thus not a member of itself.

```
Definition non_coll (p: property) := ~ exists E, forall x, inc x E <-> p x.

Lemma non_collP p: non_coll p <->  ~ exists E, forall x, p x -> inc x E.
Lemma non_collectivizing_ordinal: non_coll ordinalp.
```

Let X be an ordinal, p some property satisfied by X, so that the set of all $x \in X^+$ that satisfy p is non-empty. The intersection y of this set is the least ordinal satisfying p (we have not yet introduced an order on the class of ordinals, so we state: y is an ordinal that satisfies p, and if z is an ordinal that satisfies p, then $y \subset z$). An ordinal is asymmetric. If $x^+ = y^+$ then $x = y$. If a and b are two elements of an ordinal x, then $a < b$ (for the relation $o(x)$) is equivalent to $a \in b$. The two orderings $o(x)$ and $o'(x)$ are the same, and are well-orderings.

```
Definition least_ordinal (p: property) x:= intersection (Zo (osucc x) p).

Lemma ordinal_asymmetric E: ordinalp E -> asymmetric_set E.
Lemma ord_succ_inj  { when ordinalp &,  injective osucc }.
Lemma ordo_leP x a b:
  gle (ordinal_o x) a b <-> [/\ inc a x, inc b a & sub a b].
Lemma ordo_ltP x a b: ordinalp x -> inc a x -> inc b x ->
   (glt (ordinal_o x) a b <-> inc a b).
Lemma ordinal_same_wo x: ordinalp x ->
  ordinal_oa x = ordinal_o x.
Lemma oset_sub x y: ordinal_set x -> sub y x -> ordinal_set y.
Lemma oset_funI x h:
  (forall t, inc t x -> ordinalp (h t)) -> ordinal_set (fun_image x h).
Lemma ordinal_o_wor x: ordinalp x -> worder (ordinal_o x).
Lemma ordinal_worder x: ordinalp x ->  worder (ordinal_oa x).
Lemma least_ordinal1 x (p: property) (y:= least_ordinal p x) :
  ordinalp x -> p x ->
   [/\ ordinalp y, p y & forall z, ordinalp z -> p z -> sub y z].
```

Alternate definition. Let E be a transitive and decent set. Let S_x be the segment for $o'(E)$ with endpoint x. This is the set of all $y \in E$ such that $y \in x$ and $x \neq y$. Since E is decent, the condition $x \neq y$ is unnecessary. Thus $S_x = x \cap E$. Since E is transitive, $x \in E$ implies $x \subset E$ and $x \cap E = x$. Thus $S_x = x$.

We deduce: if E is an ordinal and $x \in E$, then the segment S_x for $o(E)$ with endpoint x is equal to x.

We also deduce: if E is transitive and asymmetric, if $o'(E)$ is a well-order, then E is an ordinal (the converse holds also).

```
Lemma ordinal_segment1 E x: trans_dec_set E ->
  inc x E -> segment (ordinal_oa E) x = x.
Lemma ordinal_segment E x: ordinalp E ->
  inc x E -> segment (ordinal_o E) x = x.
Lemma ordinal_pr1 E:
  ordinalp E <->
  [/\ transitive_set E, worder (ordinal_oa E) & asymmetric_set E].

Lemma ordinal_o_sr x: substrate (ordinal_o x) = x.
Lemma ordinal_o_or x: order (ordinal_o x).
Lemma ordinal_o_tor x: ordinalp x -> total_order (ordinal_o x).
```

Well-orders and ordinals. We shall prove the following: for any well-ordered set E, there exists a unique function f, whose target X is an ordinal, such that f is an order isomorphism (where the target is ordered by $o(X)$). This is a non-trivial result due to von Neumann.

Uniqueness. We show that an order-isomorphism $f : o(X) \to o(Y)$, where X and Y are ordinals has to be the identity function. For otherwise, there would be a least element $x \in X$ such that $f(x) \neq x$. Then $f(y) = y$ for $y \in x$. This implies that x is a subset of Y. It is transitive and cannot be Y (by injectivity of f), thus is an element of Y, so that $x = f(z)$ for some z. Each of the three alternatives $x \in z$, $x = z$ and $z \in x$ leads to a contradiction.

Existence. Consider a set X well-ordered by $\leq$. Let f be the function such that $f(x)$ is the set of all $f(t)$ where $t < x$ (in particular, if x is the least element of X, then $f(x)$ is empty). This function has been introduced in the previous chapter. By definition, $y < x$ implies $f(y) \in f(x)$, and $y \leq x$ implies $f(y) \subset f(x)$. If $a \in X$ and if x is in $f(a)$ then $x = f(b)$ for some $b < a$ thus $f(b) \subset f(a)$. We deduce that $f(a)$ is an ordinal. The same argument says that the target of f is an ordinal. If $f(a) \in f(a)$, then $f(a) = f(b)$ for some $b < a$, and $f(b) \in f(b)$; thus there no least such a, and since X is well-ordered, there is no such a. Thus, the image of f is decent; but since the source is totally ordered, it says that f is injective. Thus, f is an order isomorphism.

```
Lemma ordinal_isomorphism_unique  x y f:
  ordinalp x -> ordinalp y ->
  order_isomorphism f (ordinal_o x) (ordinal_o y) ->
  (x = y /\ f = identity x).
Lemma ordinal_isomorphism_exists r (f := ordinal_iso r): (* 59 *)
  worder r ->
  ordinalp (target f) /\ order_isomorphism f r (ordinal_o (target f)).
```

We shall denote by ord(E) the target of the isomorphism appearing in the existence theorem, and call it *the ordinal* of the well-ordered set E. Note that for Bourbaki, the ordinal of E, denoted by Ord(E), is a quantity that shares the same properties as $o(\text{ord}(E))$.

The existence theorem says: if E is a well-ordered set, then ord(E) is an ordinal, and E is isomorphic to $o(\mathrm{ord(E)})$. The uniqueness theorem can be expressed as: two isomorphic well-orders have the same ordinal, or as: if x and y are two ordinals such that $o(x)$ and $o(y)$ are isomorphic, then $x = y$.

```
Definition ordinal r := target (ordinal_iso r).

Lemma OS_ordinal r: worder r -> ordinalp (ordinal r).
Lemma ordinal_o_is r: worder r ->  r \Is  (ordinal_o (ordinal r)).
Lemma ordinal_o_o x: ordinalp x ->
  ordinal (ordinal_o x) = x.
Lemma ordinal_isu x y:  ordinalp x -> ordinalp y ->
  (ordinal_o x) \Is (ordinal_o y) -> x = y.
Lemma ordinal_o_isu1 r r': worder r -> worder r' ->
  r \Is r' -> ordinal r = ordinal r'.
Lemma ordinal_o_isu2 r x: worder r -> ordinalp x ->
  r \Is (ordinal_o x) -> ordinal r = x.
```

If r is a well-order and α its ordinal, there is a unique order isomorphism $o(\alpha) \to r$, the inverse of the isomorphism introduced above.

```
Definition the_ordinal_iso r := inverse_fun (ordinal_iso r).

Lemma the_ordinal_iso1 r : worder r ->
  order_isomorphism (the_ordinal_iso r) (ordinal_o (ordinal r)) r.
Lemma the_ordinal_iso2 r g:
  worder r -> order_isomorphism g (ordinal_o (ordinal r)) r ->
  g = the_ordinal_iso r.
```

Comparing orders. Let $r \preceq_{\mathrm{ord}} r'$ be the relation "r and r' are orders and there is an order morphism $f : r \to r'$". Let E and E′ be the substrates of the orders, and X the range of f; we have $\mathrm{X} \subset \mathrm{E}'$, let's order it by the ordering induced by r'. Then f can be considered as an order isomorphism $\mathrm{E} \to \mathrm{X}$. Conversely, if $\mathrm{X} \subset \mathrm{E}$ is ordered by r', and f is an order isomorphism $\mathrm{E} \to \mathrm{X}$, we can consider it as an order morphism $r \to r'$. An important property is that $r \preceq_{\mathrm{ord}} r'$ remains true if one ordering is replaced by an isomorphic one.

```
Definition order_le r r' :=
  [/\ order r, order r' &
  exists f x,
    sub x (substrate r') /\ order_isomorphism f r (induced_order r' x)].

Lemma order_leR x: order x -> order_le x x.
Lemma order_le_alt r r':
  order r -> order r' ->  (exists f, order_morphism f r r') ->
  order_le r r'.
Lemma order_le_alt2 f r r' x: order r -> order r' ->
  sub x (substrate r') ->
  let F := (Lf (Vf f) (substrate r') (substrate r')) in
    order_isomorphism f r (induced_order r' x)
    -> (order_morphism F r r' /\ Imf F = x).
Lemma order_le_compatible r r' r1 r1':  (* 51 *)
  r \Is r1 -> r' \Is r1' ->
  (order_le r r' <-> order_le r1 r1').
```

Let's write $x \preceq_{\mathrm{Ord}} y$ if x and y are ordinals and $o(x) \preceq_{\mathrm{ord}} o(y)$. If r and r' are two well-orders, $x \preceq_{\mathrm{Ord}} y$ is equivalent to $\mathrm{ord}(x) \preceq_{\mathrm{ord}} \mathrm{ord}(y)$. Since any subset of a well-ordered set is isomorphic to a segment, we get: if x and y are ordinals, $x \preceq_{\mathrm{Ord}} y$ is the same as: there is an order morphism $o(x) \to o(y)$ whose range is a segment. Moreover $x \prec_{\mathrm{Ord}} y$ is the same as: there is an order morphism $o(x) \to o(y)$ whose range is an initial segment S_t of $o(y)$ for some $t \in y$.

```
Definition ordinal_leD1 r r' :=
  [/\ ordinalp r, ordinalp r' & order_le (ordinal_o r)(ordinal_o r')].
Definition ordinal_ltD1 r r' :=  ordinal_leD1 r r' /\ r <> r'.

Lemma order_le_compatible1 r r':
  worder r -> worder r' ->
  (order_le r r' <-> ordinal_leD1 (ordinal r) (ordinal r')).
Lemma ordinal_le_P x x':
   ordinal_leD1 x  x' <->
   [/\ ordinalp x, ordinalp x' &
     exists f S,
      segmentp (ordinal_o x') S /\
      order_isomorphism f (ordinal_o x) (induced_order (ordinal_o x') S)].
Lemma ordinal_le_P1 x x':
   ordinal_leD1 x x' <->
   [/\ ordinalp x, ordinalp x' &
     exists2 f, segmentp (ordinal_o x') (Imf f) &
         order_morphism f (ordinal_o x)(ordinal_o x')].
Lemma ordinal_lt_P1 x x':
   ordinal_ltD1 x x' <->
   [/\ ordinalp x, ordinalp x' &
     exists f y,
        [/\ inc y x',
        Imf f = segment (ordinal_o x') y
        & order_morphism f (ordinal_o x) (ordinal_o x')]].
Lemma ordinal_lt_P x x':
  (ordinal_ltD1 x x') <->
   [/\ ordinalp x, ordinalp x' &
     exists f y',
        inc y' x' /\
      order_isomorphism f (ordinal_o x)
       (induced_order (ordinal_o x') (segment (ordinal_o x') y') )].
Lemma ordinal_lt_pr2 a b:
  worder b ->  (ordinal_ltD1 a (ordinal b))  ->
     exists f x,
       [/\ inc x (substrate b),
        (Imf f) = segment b x & order_morphism f (ordinal_o a) b].
```

If x and y are two ordinals, such that $x \subset y$, then the canonical injection is an order morphism $o(x) \to o(y)$, so that $x \preceq_{\mathrm{Ord}} y$. Conversely, this says that there is an order morphism whose range is a segment of $o(y)$. This range is an ordinal, thus is equal to x, so that $x \subset y$.

```
Lemma ordinal_le_P0 x y:
  ordinal_leD1 x y <-> [/\ ordinalp x, ordinalp y & sub x y].
```

Comparing ordinals. Given two ordinals x and y, we may compare the associated orders $o(x)$ and $o(y)$. This relation is obviously a preorder relation. Proving antisymmetry is a bit harder. However, we have seen that it is the same as $x \subset y$. Thus, we introduce $x \le_{\text{ord}} y$ as short for "x and y are ordinals and $x \subset y$". This is obviously an order relation.

```
Definition ordinal_le x y :=
  [/\ ordinalp x, ordinalp y & sub x y].
Definition ordinal_lt x y := ordinal_le x y /\ x <> y.
Definition ole_on x := graph_on ordinal_le x.

Notation "x <=o y" := (ordinal_le x y) (at level 60).
Notation "x <o y" := (ordinal_lt x y) (at level 60).
```

Now $x <_{\text{ord}} y$ becomes"x and y are ordinals such that $x \in y$", it is a well-ordering. The intersection of a set of ordinals is the least element, so that $\le_{\text{ord}}$ is a well-ordering of the ordinals.

```
Lemma oltP0 x y:
  x <o y <-> [/\ ordinalp x, ordinalp y & inc x y].
Lemma oltP a: ordinalp a -> forall x, (x <o a <-> inc x a).
Lemma olt_i x y: x <o y -> inc x y.
Lemma oleP a x: ordinalp a ->
  (x <=o a <-> inc x (osucc a)).
Lemma least_ordinal0 E (x:= intersection E): ordinal_set E -> nonempty E ->
  [/\ ordinalp x, inc x E & forall y, inc y E -> x <=o y].

Theorem wordering_ole: worder_r ordinal_le.
Lemma wordering_ole_pr x:
  ordinal_set x -> worder_on (ole_on x) x.
```

Let's state some properties of $\le_{\text{ord}}$.

```
Lemma ole_order_r: order_r ordinal_le.
Lemma oleR x: ordinalp x -> x <=o x.
Lemma oleT y x z: x <=o y -> y <=o z -> x <=o z.
Lemma oleA x y: x <=o y -> y <=o x -g>  x = y.
Lemma ole_eqVlt a b : a <=o b -> (a = b \/ a <o b).
Lemma oleNgt x y: x <=o y -> ~(y <o x).
Lemma oltNge x y: x <o y -> ~ (y <=o x).
Lemma ole_ltT b c a: a <=o b -> b <o c -> a <o c.
Lemma ole_ltT b a c: a <=o b -> b <o c -> a <o c.
Lemma olt_ltT b a c: a <= b -> b <o c -> a <o c.
Lemma oleT_el a b:
  ordinalp a -> ordinalp b ->  a <=o b \/ b <oa.
Lemma oleT_ell a b:
  ordinalp a -> ordinalp b -> [\/ a = b, a <o b | b <o a].
Lemma oleT_ee a b:
  ordinalp a -> ordinalp b -> a <=o b \/ b <=o a.
Lemma oleT_si a b:
  ordinalp a -> ordinalp b -> (sub a b \/ inc b a).
```

We define here the minimum and maximum of two ordinals. These quantities exist as the ordering of ordinals is total. We show associativity and distributivity (all properties of a lattice are valid for ordinals, even though there is no set of all ordinals).

```
Definition omax x y:=  Yo (x <=o y) y x.
```

```
Definition omin x y:=  Yo (x <=o y) x y.

Lemma OS_omax x y: ordinalp x -> ordinalp y -> ordinalp(omax x y).
Lemma OS_omin x y: ordinalp x -> ordinalp y -> ordinalp(omin x y).
Lemma omax_xy x y: x <=o y -> omax x y = y.
Lemma omax_yx x y: y <=o x -> omax x y = x.
Lemma omin_xy x y: x <=o y -> omin x y = x.
Lemma omin_yx x y: y <=o x -> omin x y = y.

Lemma omax_p1 x y:  ordinalp x -> ordinalp y ->
   x <=o (omax x y) /\ y <=o (omax x y).
Lemma omin_p1 x y:  ordinalp x -> ordinalp y ->
   (omin x y)  <=o x /\ (omin x y) <=o y.
Lemma omax_p0 x y z: x <=o z -> y <=o z -> (omax x y) <=o z.
Lemma omin_p0 x y z: z <=o x -> z <=o y -> z <=o (omin x y).

Lemma omaxC x y: ordinalp x -> ordinalp y -> omax x y = omax y x.
Lemma ominC x y: ordinalp x -> ordinalp y -> omin x y = omin y x.
Lemma omaxA x y z: ordinalp x -> ordinalp y -> ordinalp z ->
    omax x (omax y z) = omax (omax x y) z.
Lemma ominA x y z: ordinalp x -> ordinalp y -> ordinalp z ->
    omin x (omin y z) = omin (omin x y) z.
Lemma ominmax x y z:
  ordinalp x -> ordinalp y -> ordinalp z ->
  omin x (omax y z) = omax (omin x y) (omin x z).
Lemma omaxmin x y z:
  ordinalp x -> ordinalp y -> ordinalp z ->
  omax x (omin y z) = omin (omax x y) (omax x z).
```

Let r and r' be two orderings. Then r is a subset of r' if and only if $x \le y$ (for r) implies $x \le y$ (for r'). If this condition holds, then $\mathrm{E} \subset \mathrm{E}'$ where E and E$'$ are the substrates of r and r'. If r is total, then r is the order induced by r' on E; if moreover r and r' are well-orders, we have $\mathrm{ord}(r) \le \mathrm{ord}(r')$.

Assume further that $r \neq r'$ and that E is a segment of r'. Then r and r' are not isomorphic (an isomorphism $f : \mathrm{E}' \to \mathrm{E}$ can be extended to a morphism $\mathrm{E}' \to \mathrm{E}'$ with range E. Since E is a segment of E$'$ this extension has to be the identity function by uniqueness of isomorphism on a segment, so that $\mathrm{E} = \mathrm{E}'$, thus $r = r'$). It follows $\mathrm{ord}(r) < \mathrm{ord}(r')$.

```
Lemma order_cp0  r r': order r -> order r' ->
  ((sub r r') <-> (forall x y, gle r x y -> gle r' x y)).
Lemma order_cp1 x y: worder x -> worder y -> sub x y ->
  (sub (substrate x) (substrate y) /\ x = (induced_order y (substrate x))).
Lemma order_cp2 x y: worder x -> worder y -> sub x y ->
  ordinal x <=o ordinal y.
Lemma order_cp3 x y: worder x -> worder y ->
   ssub x y -> (segmentp y (substrate x)) -> ordinal x <o ordinal y.
```

We can restate one of the previous results as: Let p be a property, satisfied by at least some x. Then there is a least y (for $\le_{\mathrm{ord}}$) that satisfies p. Assume that p is false for some ordinal; then there exists y, that does not satisfy p, such that if $z <_{\mathrm{ord}} y$, then $p(z)$ holds.

```
Lemma least_ordinal4 x (p: property)  (y := least_ordinal p x):
 ordinalp x -> p x ->
   [/\ ordinalp y, p y & (forall z, ordinalp z -> p z -> y <=o z) ].
```

```
Lemma least_ordinal2 (p: property) x:
  (forall y, ordinalp y -> (forall z, z <o y -> p z) -> p y) ->
  ordinalp x -> p x.
Lemma least_ordinal3 x (p: property) (y := least_ordinal (fun z => (~ p z)) x):
  ordinalp x -> ~ (p x) ->
  [/\ ordinalp y, ~(p y) & (forall z, z <o y -> p z)].
Lemma least_ordinal6 x (p:property) (y :=least_ordinal (fun z => ~ p z) x):
  ordinalp x ->
  p x \/ [/\ ordinalp y, forall z, inc z y -> p z & ~ p y].
```

Ordinal supremum. The union U of a family of ordinals is an ordinal, it is the least upper bound of the family (in the sense that $U = \sup_E X$, where E is any set of ordinals containing U and the elements of X). We shall sometimes use the notation \osup for it[1].

```
Notation "\osup" := union (only parsing).
Notation "\csup" := union (only parsing).
Definition opred := union.
Definition cpred := union.
Definition ordinal_ub E x:= forall i, inc i E -> i <=o x.

Lemma OS_sup E: ordinal_set E ->
  ordinalp (\osup E).
Lemma ord_sup_ub E: ordinal_set E -> ordinal_ub E (\osup E).
Lemma ord_ub_sup E y: ordinalp y -> ordinal_ub E y ->
  \osup E <=o y.
Lemma ord_sup_prop E: ordinal_set E ->
  exists! x, ordinalp x /\
   (forall y,  x <=o y <-> (ordinalp y /\ ordinal_ub E y)).
```

Consider a property p such that, whenever x is an ordinal, there is y such that $x \le y$ and $p(y)$ holds. Then p is non-collectivizing. Proof. Assume that there is E such that $x \in E$ is equivalent to p. Let F be the set of ordinals in E, x the supremum of F and $z = x^+$; now z is an ordinal and there is y such that $z \le y$ and $p(y)$. Now y is an ordinal and $y \in F$, so $y \le x$. This contradicts $x < x^+$.

```
Lemma unbounded_non_coll (p:property):
  (forall x, ordinalp x -> exists2 y, x <=o y & p y) -> non_coll p.
Lemma non_collectivizing_ordinal_bis: non_coll ordinalp.
```

The empty set is the least ordinal; we shall write 0_o instead of $\emptyset$ (later on, we shall see that the empty set is also the least cardinal, and write 0_c for it). If $0 < x$ then $0 \in x$; conversely, if $0 \in x$ and x is an ordinal, then $0 < x$. Note that, if x is an non-empty ordinal, then $0 < x$.

```
Definition ord_zero := emptyset.
Notation "\0o" := ord_zero.

Lemma OS0: ordinalp \0o.
Lemma ole0x x: ordinalp x -> \0o <=o x.
Lemma ole0 x: x <=o \0o -> x = \0o.
Lemma ord_ne0_pos x: ordinalp x -> x <> \0o -> \0o <o x.
Lemma olt0 x: x <o \0o -> False.
```

[1] We shall see below that this is also the cardinal supremum, and the ordinal predecessor, so that we give three names to this object.

Successor and comparison. For every ordinal x, there is a successor x^+ such that $x < x^+$ and for no y do we have $x < y < x^+$.

```
Lemma osucc_pr0 x (z:= osucc x): ordinalp x ->
  x <o z /\ (forall w, x <o w -> z <=o w).
Lemma oltS a: ordinalp a -> a <o (osucc a).
Lemma oltSleP a b: a <o (osucc b) <-> a <=o b.
Lemma oleSltP a b: a <o b <-> (osucc a) <=o b.
Lemma oleSSP x y: (osucc x <=o osucc y) <-> x <=o y.
Lemma oltSSP x y: (osucc x <o osucc y) <->  (x <o y).
```

Limit ordinals. Consider a well-ordering $\le$, an element x of the substrate. The least element of $]x, \to[$ is called the successor of x. If $\le$ is a relation without graph, it can happen that $]x, \to[$ is not a set (this is the case for ordinal comparison). Assume that $A_y =]x, y]$ is a set (this holds when $]\leftarrow, y[$ is a set, thus in the case of ordinals). If A_y is always empty, then x is the greatest element. If $x < y$, then A_y is non-empty and has a least element z_y, such that $z_y \le t$ is equivalent to $x < t$; in particular, this is independent of y, this is called the successor of x.

Consider now some y such that $B =]\leftarrow, y[$ is a set. If B is empty, then y is the least element. Let C be $B \cup \{y\}$. Every subset X of B, included B itself, is bounded by y in C, thus has a supremum $\sup X$ that belongs to C (we could take a larger set for C, this does not change the value of the supremum). Let $x = \sup B$. If $x \neq y$, then $x \in B$, x is the greatest element of C, and $x \le t \le y$ implies that t is either x or y. Moreover, for every t we have $t \le x$ or $y \le t$. In this case y is the sucessor of x, and x is called the predecessor of y. Otherwise (if $x = y$ and B is nonempty) we say that y is a limit element. Consider a subset B' of B and its supremum x'. If B' is finite, then B' has a greatest element, this means $x' \in B$ and $x' < y$. In order for x' to be y it is necessary for B' to be infinite. There is least ordinal β such that β is the ordinal of some subset B' of B (well-ordered by $\le$) such that $\sup B' = y$; it is called the cofinality of B. This cofinality is infinite. However, if B is countable, then $\beta = \omega$; in this case, there is a strictly increasing sequence $(y_i)_{i\in \mathbf{N}}$ of quantities $< y$ such that the supremum is equal to y. One says that y is the limit of the y_i. This explains why y is called "limit". The terms "finite", "infinite", "cardinal", as well as the set of integers $\mathbf{N} = \omega$ will be introduced later (see page 110); the case where B is countable is not handled[2].

The comparison relation of ordinals and cardinals is a well-ordering without a graph, but the previous discussion applies: there is no greatest element, and every ordinal x has a *successor* namely x^+ (since x is irreflexive, it is a strict subset of x^+, and $x < a \le x^+$ says that a is a subset of $x \cup \{x\}$, that contains all elements of x, and $a \neq x$. It follows $a = x^+$). On the other hand $B = y$ and the supremum of B is the union of y. We shall call it the *predecessor* of y, and denote it by x^-.

We say that y is a successor if it has the form x^+. The negation of this property is $y = y^-$. We say that y is a *limit ordinal* if it is neither zero, nor a successor. We can restate this as: y is an ordinal such that $0 \in y$ and $x^+ \in y$ whenever $x \in y$; or equivalently, y is an ordinal such that $y \neq 0$ and $x^+ < y$ whenever $x < y$. If y is limit, it is equal to its predecessor (by definition), but if y is a successor, it is the successor of its predecessor. Moreover, the predecessor of x^+ is x.

```
Definition osuccp x := exists2 y, ordinalp y & x = osucc y.
```

[2] We define cofinality only for ordinals; extending it to well-ordered sets is easy; the non-trivial point is to show that the cofinality is a cardinal; since it is infinite and countable, it has to be ω

```
Definition limit_ordinal x:=
  [/\ ordinalp x, inc \0o x & (forall y, inc y x -> inc (osucc y) x)].

Lemma osuccp_pr a: ordinalp a -> (exists b, a = osucc b) ->
  osuccp a.
Lemma opred_le x: ordinalp x ->  opred x <=o x.
Lemma osuccVidpred x: ordinalp x ->
  exactly_one (osuccp x) (x = opred x).

Lemma limit_nonsucc x: limit_ordinal x -> x = opred x.
Lemma limit_pos x: limit_ordinal x -> \0o <o x.
Lemma limit_nz x: limit_ordinal x -> x <> \0o.

Lemma limit_ordinal_P0 x: ordinalp x ->
  ((limit_ordinal x) <-> (nonempty x /\ x = opred x)).
Lemma ordinal_trichot x: ordinalp x ->
   [\/ x = \0o, osuccp x | limit_ordinal x].
Lemma limit_ordinal_P x:
  limit_ordinal x <->
  (\0o <o x /\ forall t, t <o x -> osucc t <o x).
Lemma succo_K y: ordinalp y -> opred (osucc y) = y.
Lemma predo_K x: osuccp x -> osucc (opred x) = x.
```

Cardinalities of successors. We show here that X^+ and Y^+ are equipotent if and only if X and Y are equipotent. Moreover if $X \subset Y$ and X is equipotent to X^+, then Y is equipotent to Y^+. Here X^+ denotes any set $X \cup \{a\}$ such that $a \notin X$. Sketch of the proof: if f is a bijection $X \to Y$ it is easy to extend to a bijection $X^+ \to Y^+$. Conversely, let f be a bijection $X \cup \{a\} \to Y \cup \{b\}$. Assume $f(a') = b$. We may swap the values of a and a', thus assume $f(a) = b$. Then by restriction we get a bijection $X \to Y$. Assume $X \subset Y$, and f is a bijection $X \cup \{a\} \to X$. Define $g(x)$ by: $f(x)$ (when $x \in X$), $f(a)$ (when $x = b$) and x (otherwise). This is a bijection $Y \cup \{b\} \to Y$.

```
Section SuccProp.
Variables (x y a b: Set).
Hypotheses (nax: ~ inc a x) (nby: ~ inc b y).

Lemma setU1_injective_card1:
  sub x y -> x \Eq (x +s1 a) ->  y \Eq (y +s1 b).
Lemma setU1_injective_card2: (* 50 *)
   ((x +s1 a) \Eq (y +s1 b) <-> x \Eq y).
End SuccProp.
```

Infinite ordinals. Let's say that an ordinal is *infinite* if it is equipotent to its successor, and *finite* otherwise (for simplicity, any set equipotent to its successor is called infinite; so that, if x is a set such that $x = \{x\}$, it will be called infinite, though its cardinal is one).

One of the previous results says: if $x \subset y$ and x is infinite, then y is infinite. Thus, if y is finite so is x. The other result says that x is infinite if and only if is successor is infinite. Thus x is finite if and only if its successor is finite.

```
Definition infinite_o u := u \Eq (osucc u).
Definition finite_o u := ordinalp u /\ ~ (infinite_o u).

Lemma succ_injective_oP x y: ordinalp x -> ordinalp y ->
  ((osucc x) \Eq (osucc y) <->  x \Eq y).
```

```
Lemma OIS_in_inf x y: ordinalp y ->
  inc x y -> infinite_o x -> infinite_o y.
Lemma finite_o_increasing x y:
  inc x y -> finite_o y -> finite_o x.
Lemma infinite_oP x: ordinalp x ->
  (infinite_o (osucc x) <-> infinite_o x).
Lemma finite_oP x: finite_o (osucc x) <-> finite_o x.
```

A limit ordinal x is infinite. Proof: let x be a limit ordinal. There is then a least limit ordinal ω. Define $f(x)$ on ω^+ by $f(x) = x^+$ if $x \in \omega$ and $f(\omega) = \emptyset$. Since ω is limit, we have $f(x) \in \omega$ for all x. This is an injective function. Since ω is the least limit ordinal, the function is surjective, thus bijective. Thus ω is infinite. It follows that x is infinite.

```
Lemma OIS_limit x: ordinal_limit x -> infinite_o x.
```

Cardinals. The least ordinal equipotent to a set X is called the *von Neumann cardinal* of X. By Zermelo's theorem, there is a well-ordering r on X; the ordinal of r is equipotent to X, so that every set has a cardinal. It will be denoted by card(X). Bourbaki uses the notation Card(X) to mean some set equipotent to X; it satisfies the property that two sets have the same cardinal if and only if they are equipotent.

```
Definition cardinal x :=
  (least_ordinal (equipotent x) (ordinal (worder_of x))).
Definition cardinal_prop x y :=
  [/\ ordinalp y,  x \Eq y &
  (forall z, ordinalp z -> x \Eq z -> sub y z)].
Lemma cardinal_unique x y z:
  cardinal_prop x y -> cardinal_prop x z -> y = z.
Lemma cardinal_pr1 x: cardinal_prop x (cardinal x).
```

The following piece of code make `cardinal` provably equal to `cardinalV`, but Coq will never replace one definition by the other; this makes some proofs faster. Not used any more.

```
(*
Module Type CardinalSig.
Parameter cardinal : Set -> Set.
Axiom cardinalE: cardinal = cardinalV.
End CardinalSig.

Module Cardinal: CardinalSig.
Definition cardinal := cardinalV.
Lemma cardinalE: cardinal = cardinalV. Proof. by []. Qed.
End Cardinal.

Notation cardinal := Cardinal.cardinal.
*)
```

We say that x is a cardinal if x if there exists a set y such that x is the cardinal of y. In this case, x is also the cardinal of x. So we say: a set x is called a *cardinal* if it is an ordinal, and whenever z is an ordinal equipotent to x we have $x \subset z$ (or $x \leq_{\text{ord}} z$). In this case $\text{card}(x) = x$. We write $x =_c y$ instead of $\text{card}(x) = \text{card}(y)$.

```
Definition cardinalp x:=
```

```
  ordinalp x /\ (forall z, ordinalp z -> x \Eq z -> sub x z).
Definition cardinal_set X := alls X cardinalp.
Definition cardinal_fam x := allf x cardinalp.

Notation "x =c y" := (cardinal x = cardinal y)
  (at level 70, format "'[hv' x '/ '  =c  y ']'", no associativity).

Lemma card_card x: cardinalp x -> cardinal x = x.
```

Some trivial properties. Proposition 1 [4, p. 158] states that X and Y are equipotent if and only if they have the same cardinal. This is restated as $\mathrm{Eq}(X,Y) \iff X =_c Y$. One implication is trivial since any set is equipotent to its cardinal. The converse depends on the definition of a cardinal. In the von Neumann case, if Z is any ordinal, X equipotent to Z implies $\mathrm{card}(X) \subset Z$. Take $Z = \mathrm{card}(Y)$, this gives $\mathrm{card}(X) \subset \mathrm{card}(Y)$, thus equality.

```
Lemma CS_cardinal x: cardinalp (cardinal x).
Lemma OS_cardinal x: cardinalp x ->  ordinalp x.
Lemma oset_cset E: cardinal_set E -> ordinal_set E.
Lemma card_ord_le x: ordinalp x -> cardinal x <=o x.
Lemma double_cardinal x: cardinal x =c x.
Lemma cardinalP x:
  cardinalp x <-> (ordinalp x /\ forall z, inc z x -> ~ (x \Eq z)).
Theorem card_eqP x y: x =c y <-> x \Eq y.
```

We restate some lemmas that as that A is equipotent to B as A and B have the same cardinal.

```
Lemma card_bijection f: bijection f -> source f =c target f.
Lemma cardinal_image f x :  sub x (source f) -> injection f ->
  Vfs f x =c x.
Lemma cardinal_fun_image t g : {inc t &, injective g} -> fun_image t g =c  t.
Lemma cardinal_range f: injection f -> Imf f =c source f.
Lemma cardinal_indexed a b: a *s1 b =c a.
Lemma cardinal_indexedr a b: indexedr b a =c a.
```

We define here zero, one and two as $\emptyset$, $\{\emptyset\}$ and $\{\emptyset, \{\emptyset\}\}$. These quantities will be denoted by 0_c, 1_c and 2_c, and have 0_o, 1_o and 2_o as alternate names. They have already been used under the names C0, C1, and C2. Unfolding definitions shows that $1_o = 0_o^+$ and $2_o = 1_o^+$ so that these quantities are finite ordinals.

The Bourbaki definition of one is $1_b = \mathrm{Card}(\{\emptyset\}) = \mathrm{Card}(1_c)$. This means that 1_b is some set with one element, but could be any singleton. With our definition, 1_c is a cardinal, so that $1_b = \mathrm{card}(1_c)$ holds. On the other hand, $0_b = \mathrm{Card}(\emptyset)$, thus is $\emptyset$, since this is the only set with zero elements.

```
Definition card_zero := emptyset.
Definition card_one := singleton emptyset.
Definition card_two := doubleton emptyset (singleton emptyset).
Definition ord_one := card_one.
Definition ord_two := card_two.

Notation "\0c" := card_zero.
Notation "\1c" := card_one.
Notation "\2c" := card_two.
```

```
Notation "\1o" := ord_one.
Notation "\2o" := ord_two.

Corollary constants_v: (C0 = \0c /\ C1 = \1c /\ C2 = \2c).
Lemma osucc_zero: osucc \0o = \1o.
Lemma osucc_one: osucc \1o = \2o.

Lemma OS1: ordinalp \1o.
Lemma OS2: ordinalp \2o.

Lemma cardinal_set0: cardinal emptyset = \0c.
Lemma card_nonempty x:
  cardinal x = \0c -> x = emptyset.
Lemma card_nonempty0 x: x <> emptyset -> cardinal x <> \0c.
Lemma card_nonempty1 x:
  nonempty x -> cardinal x <> \0c.
Lemma card1_nz: \1c <> \0c.
Lemma card2_nz: \2c <> \0c.
Lemma card_12: \1c <> \2c.

Lemma finite_zero: finite_o \0o.
Lemma finite_one: finite_o \1o.
Lemma finite_two: finite_o \2o.
```

The Cantor-Bernstein Theorem. If F is a subset of E, and f an injection $\mathrm{E} \to \mathrm{F}$, then E is equipotent to F. (Let D be the smallest set invariant by f that contains $\mathrm{E} - \mathrm{F}$. The bijection is the function that is f on D, the identity elsewhere). It follows that if there is an injection $f : \mathrm{X} \to \mathrm{Y}$ and an injection $g : \mathrm{Y} \to \mathrm{X}$, then there is a bijection $\mathrm{X} \to \mathrm{Y}$ (let Z be the image of g, it is equipotent to Y since g is injective; it is equipotent to X, being a subset of X, and $g \circ f$ is an injection $\mathrm{X} \to \mathrm{Z}$). This property is stated without proof by Cantor in [7, §2, Th B], proved by various authors as Bernstein, Schröder, etc.

```
Lemma Cantor_Bernstein1 E F f:
  sub F E ->
  (forall x, inc x E -> inc (f x) F) ->
  {inc E &, injective f}
  E \Eq F.

Theorem CantorBernstein X Y:
  (exists f, injection_prop f X Y) ->
  (exists f, injection_prop f Y X) ->
  (exists f, bijection_prop f X Y).
```

Assume that x is a finite ordinal. Then it is not a limit ordinal, thus is either 0 or a successor y^+ and y is finite. Conversely, 0 and x^+ are finite ordinals. Moreover x is a cardinal: it suffices to show that y^+ is a cardinal if y^+ is a finite ordinal. Assume that there is $z \in y^+$ equipotent to y^+. We get a bijection $y^+ \to z$, and by restriction, an injection $y \to z$. The Cantor Bernstein theorem then says that y and z are equipotent, thus y equipotent to y^+, absurd as y is finite. We deduce that 0, 1 and 2 are cardinals.

```
Lemma finite_succ x:
  finite_o x <-> (x = emptyset \/ (exists2 y, finite_o y & x = osucc y)).

Lemma CS_osucc x: finite_o x -> cardinalp (osucc x).
```

```
Lemma CS_finite_o x: finite_o x -> cardinalp x.
Lemma gfunctions_empty X: (gfunctions emptyset X) = \1c.

Lemma CS0: cardinalp \0c.
Lemma CS1: cardinalp \1c.
Lemma CS2: cardinalp \2c.
```

Finite sets. We say that a cardinal is *finite* or *infinite* if it is finite or infinite as an ordinal, and we say that a set is finite or infinite if its cardinal is finite or infinite. As noted above, a finite ordinal is a cardinal. A finite cardinal is called a *natural integer*. The *cardinal successor* of x is the cardinal of the ordinal successor of x. A cardinal cannot be both finite and infinite. In particular, an infinite cardinal is non-empty. Note that if x is an infinite cardinal, it is equal to its successor, thus is a limit ordinal.

```
Definition finite_c  :=  finite_o.
Definition infinite_c a := cardinalp a /\ infinite_o a.
Definition finite_set E := finite_c (cardinal E).
Definition infinite_set E := infinite_o (cardinal E).
Definition csucc x := cardinal (osucc x).

Lemma csucc_inf x: infinite_c x <-> csucc x = x.
Lemma infinite_dichot1 x: finite_c x -> infinite_c x -> False.
Lemma infinite_dichot2 x:
  finite_c (cardinal x) -> infinite_set x -> False.
Lemma CS_finite2 x: finite_c x -> cardinalp x.
Lemma infinite_nz y: infinite_c y -> y <> \0c.
Lemma infinite_card_limit1 x: infinite_c x -> opred x = x.
Lemma infinite_card_limit2 x: infinite_c x -> limit_ordinal x.
```

We know that A is equipotent to B if and only if A^+ is equipotent to B^+. We deduce Proposition 8 [4, p. 162], two cardinals that have the same successor are equal. If B is the cardinal of A we get: if $a \not\in A$, the cardinal of $A \cup \{a\}$ is the successor of the cardinal of A. This is the same as: if $c \in C$, then the cardinal of C is the successor of the cardinal of $C - \{c\}$. Thus, if C is equipotent to $C - \{c\}$, it is an infinite set.

```
Lemma CS_succ a: cardinalp (csucc a).
Lemma succ_zero: csucc \0c = \1c.
Lemma succ_one: csucc \1c  = \2c.

Theorem succ_injective1: {when cardinalp &, injective csucc}.
Lemma csucc_pr a b: ~ (inc b a) ->
  cardinal (a +s1 b) = csucc (cardinal a).
Lemma csucc_pr1 a b:
  cardinal ((a -s1 b) +s1 b) = csucc (cardinal (a -s1 b)).
Lemma csucc_pr2 a b: inc b a ->
  cardinal a = csucc (cardinal (a -s1 b)).
Lemma infinite_set_pr a b: inc b a ->
  a \Eq (a -s1 b) ->
  infinite_set a.
Lemma infinite_set_pr1 a b: inc b a ->
  a \Eq (a -s1 b) -> infinite_set (a -s1 b).
Lemma infinite_set_pr2 x:
   infinite_o x -> ~(inc x x) -> infinite_set x.
```

The axiom of infinity. Bourbaki has an axiom that says that there is an infinite set. This axiom is not needed in COQ, because ℕ (the type `nat`) is infinite, since the successor function is a bijection $\mathbb{N} \to \mathbb{N} - \{0\}$. We can then consider the least infinite ordinal, and call it ω_0, or ω. If x is an ordinal, then x is infinite if and only if $\omega \subset x$. This can be restated as: x is a finite ordinal if and only if $x \in \omega$.

Since the successor of a finite ordinal is finite, ω is a limit ordinal. Since a limit ordinal is infinite, ω is the least limit ordinal. Since any infinite cardinal is a limit ordinal, ω is the least infinite cardinal. More properties of ordinal numbers will be given in Chapter 11.

```
Definition omega0 := least_ordinal infinite_o (cardinal nat).

Lemma nat_infinite_set: infinite_set nat.
Lemma omega0_pr:
   [/\ ordinalp omega0, infinite_o omega0 &
   (forall z, ordinalp z -> infinite_o z -> sub omega0 z)].

Lemma OS_omega: ordinalp omega0.
Lemma OIS_omega: infinite_o omega0.
Lemma omega_P1 x: ordinalp x ->
  (infinite_o x <-> sub omega0 x).
Lemma omega_P2 x: inc omega0 <-> finite_o x.
Lemma CS_omega: cardinalp omega0.
Lemma CIS_omega: infinite_c omega0.
Lemma omega_limit0: omega0 = opred omega0.
Lemma omega_limit: limit_ordinal omega0.
Lemma limit_ge_omega x: limit_ordinal x -> omega0 <=o x.
Lemma omega_limit3 x: infinite_c x -> sub omega0 x.
Lemma infinite_setP x: infinite_set x <-> infinite_c (cardinal x).
Lemma infinite_set_pr3 x: omega0 <=o x -> infinite_c (cardinal x).
```

Properties of equipotent sets. We state here some lemmas that will be useful when defining operations on cardinals. All singletons are equipotent, as well as all sets with exactly two elements (they are equipotent to the canonical doubleton 2). A set with cardinal one or two has one or two elements.

```
(* Lemma set2_equipotent x y: x <> y -> doubleton x y \Eq \2c.
   Lemma set1_equipotent x y: singleton x \Eq singleton y. *)
Lemma cardinal_set1 x: cardinal(singleton x) = \1c.
Lemma cardinal_set2 x x': x <> x' ->  cardinal (doubleton  x x') = \2c.
Lemma set_of_card_oneP x: cardinal x = \1c <-> singletonp x.
Lemma set_of_card_twoP x: cardinal x = \2c <-> doubletonp x.
```

Products of equipotent sets are equipotent (if E_ι is equipotent to F_ι, we choose a bijection f_ι via the axiom of choice, and consider the family of functions $(f_\iota)_\iota$). We also consider the case of the product of two sets (the bijection is `ext_to_prodC`).

```
Definition fgraphs_equipotent x y :=
  domain x = domain y
  /\ (forall i, inc i (domain x) ->  (Vg x i) =c (Vg y i)).
Definition equipotent_ex E F :=
   choose (fun z=>  bijection_prop z E F).

Lemma equipotent_ex_pr E F:
```

```
  E \Eq F -> bijection_prop (equipotent_ex E F) E F.
Lemma equipotent_ex_pr1 E F:
  E =c F -> bijection_prop (equipotent_ex E F) E F.
Lemma Eq_setXb x y:
  fgraphs_equipotent x y ->  (productb x) =c (productb y).
Lemma Eq_setX a b a' b':
  a =c a' ->  b =c b' ->  (a \times b) =c (a' \times b').
```

If A, B and C are sets, then $A \times B$ is equipotent to $B \times A$, and $A \times (B \times C)$ is equipotent to $(A \times B) \times C$ (the bijections are obvious). We have $(A \cup B) \times C = (A \times C) \cup (B \times C)$.

```
Lemma product2A a b c:
  (a \times(b \times c)) =c ((a \times b) \times c).
Lemma distrib_union_prod2: right_distributive product union2.
Lemma distrib_inter_prod2: right_distributive product intersection2.
```

Two sets $A \times B$ and $A' \times B'$ are disjoint if B and B' are disjoint; this is the case when B and B' are distinct singletons.

```
Lemma disjoint_pr a b:
  (forall u, inc u a -> inc u b -> False) -> disjoint a b.
Lemma disjointU2_pr0 a b x y:
  disjoint x y -> disjoint (a \times x) (b \times y).
Lemma disjointU2_pr1 x y:
  x <> y -> disjoint (singleton x) (singleton y).
Lemma disjointU2_pr a b x y:
  x <> y -> disjoint (a *s1 x) (b *s1 y).
```

Two unions $\bigcup X_\iota$ and $\bigcup Y_\iota$ with the same index set are equipotent if the sets are equipotent and if each family is mutually disjoint. As a particular case, we get conditions for $A \cup B$ and $A' \cup B'$ to be equipotent.

```
Lemma Eq_indexed_c a b: (a *s1 b) =c a.
Lemma Eq_disjointU X Y:
  fgraphs_equipotent X Y ->
  mutually_disjoint X ->  mutually_disjoint Y ->
  (unionb X) =c (unionb Y).
Lemma Eq_disjointU1 X Y:
  fgraphs_equipotent X Y ->
  (disjointU X) =c (disjointU Y).
Lemma union2Lv a b: union2 a b = unionb (variantLc a b).
Lemma mutually_disjoint_prop3 D f:
  mutually_disjoint (Lg D (fun i => f i *s1 i)).
Lemma disjointLv a b: disjoint a b ->
   mutually_disjoint (variantLc a b).
Lemma Eq_disjointU2 a b a' b':
  disjoint a b -> disjoint a' b' -> a =c a' -> b =c b' ->
  (a \cup b) =c (a' \cup b').
```

Given two sets A and B, we can consider a family f defined on a doubleton $\{x, y\}$ such that $f(x) = A$ and $f(y) = B$. If F is the canonical family associated to A and B (i.e., if $F(0) = A$ and $F(1) = B$), then there is a bijection g such that $f = F \circ g$.

```
Definition doubleton_fam f a b :=
```

```
  exists x y, [/\ x<>y, fgraph f,  domain f = doubleton x y,
    Vg f x = a &  Vg f y = b].

Lemma doubleton_fam_P f a b:
  doubleton_fam f a b <->
    [/\ fgraph f, doubletonp (domain f) & range f = doubleton a b].
Lemma doubleton_fam_canon f:
  doubleton_fam (Lg C2 f) (f C0) (f C1).
Lemma two_terms_bij a b f: doubleton_fam f a b ->
  let F := (variantLc a b) in
  exists g, (bijection g /\ target g = domain F /\
    f = F \cf (graph g)).
```

4.1 The cardinal of a set

The cardinal of a set, and some basic properties have been introduced earlier.

4.2 Order relation between cardinals

Let $\mathrm{X} \leq_c \mathrm{Y}$ be the the relation "X is equipotent to some subset of Y". This can be restated as: there is an injection $\mathrm{X} \to \mathrm{Y}$. The relation remains true if X and Y are replaced by equipotent sets. In particular, it is equivalent to $\mathrm{card}(\mathrm{X}) \leq_c \mathrm{card}(\mathrm{Y})$.

Bourbaki defines $\mathfrak{r} \leq_{\mathrm{Card}} \mathfrak{n}$ as "$\mathfrak{r}$ and $\mathfrak{n}$ are cardinals and $\mathfrak{r} \leq_c \mathfrak{n}$". This is an order relation (antisymmetry follows from the Cantor-Bernstein theorem).

```
Definition equipotent_to_subset x y:= exists2 z, sub z y & x \Eq z.

Lemma eq_subset_ex_injP x y:
  equipotent_to_subset x y <->
  (exists f, injection_prop f x y).
Lemma eq_subset_pr2 a b a' b':
   a =c a' -> b =c b' ->
   equipotent_to_subset a b -> equipotent_to_subset a' b'.
Lemma eq_subset_cardP x y:
  equipotent_to_subset x y <-> equipotent_to_subset (cardinal x) (cardinal y).
```

Basic properties. We define $a \leq_{\mathrm{card}} b$ to be "a and b are cardinals and $a \subset b$".

This definition is equivalent to the previous one. Assume that a and b are cardinals (in the von Neumann sense). If $a \subset b$, then there is an injection $a \to b$. Conversely, assume that there is an injection $a \to b$, and let's show $a \subset b$. Since a and b are ordinals, one of $a \subset b$ or $b \subset a$ holds. In the second case, there is an injection $b \to a$ and the Cantor Bernstein theorem says that a and b are equipotent; since they are cardinals they are equal, thus $a \subset b$. We deduce: if $f : \mathrm{A} \to \mathrm{B}$ is injective then $\mathrm{card}(\mathrm{A}) \leq \mathrm{card}(\mathrm{B})$. We also deduce: if $\mathrm{A} \subset \mathrm{B}$, then $\mathrm{card}(\mathrm{A}) \leq \mathrm{card}(\mathrm{B})$; if $\mathrm{card}(c) \leq \mathrm{card}(\mathrm{B}$ then there is a subset A of B with cardinal $\mathrm{card}(c)$.

```
Definition cardinal_le x y :=
  [/\ cardinalp x, cardinalp y & sub x y].
Definition cardinal_lt a b := cardinal_le a b /\ a <> b.
Notation "x <=c y" := (cardinal_le x y) (at level 60).
Notation "x <c y" := (cardinal_lt x y) (at level 60).
```

```
Lemma cardinal_le_aux1 x y:
  x <=c y -> equipotent_to_subset x y.
Lemma cardinal_le_aux2P x y: cardinalp x -> cardinalp y ->
  (equipotent_to_subset x y <-> x <=c y).
Lemma eq_subset_cardP1 x y:
  equipotent_to_subset x y <-> (cardinal x) <=c (cardinal y).
Lemma incr_fun_morph f: injection f ->
  (cardinal (source f)) <=c (cardinal (target f)).
Lemma sub_smaller a b:
  sub a b ->  (cardinal a) <=c (cardinal b).
Lemma sub_smaller_contra Z X: cardinal Z <=c cardinal X ->
   exists2 Y, sub Y X & cardinal Z = cardinal Y.
```

Note that $a \leq_{\text{card}} b$ implies $a \leq_{\text{ord}} b$ so that $\leq_{\text{card}}$ is trivially a well-ordering.

```
Lemma ocle x y:
  x <=c y -> x <=o y.
Lemma cleR x: cardinalp x -> x <=c x.
Lemma cleT b a c:
  a <=c b -> b <=c c -> a <=c c.
Lemma cleA x y:
  x <=c y -> y <=c x -> x = y.
Lemma cle_wor' E (x:= intersection E): cardinal_set E -> nonempty E ->
   inc x E /\ (forall y, inc y E -> x <=c y).
Theorem cle_wor: worder_r cardinal_le.
```

Some consequences. Note that $\leq_{\text{card}}$ is a total ordering since it is a well-ordering.

```
Lemma cle_eqVlt a b : a <=c b -> (a = b \/ a <c b).
Lemma cleNgt a b: a <=c b -> ~(b <c a).
Lemma cltNge a b: a <c b -> ~(b <=c a).
Lemma clt_leT b a c: a <c b -> b <=c c -> a <c c.
Lemma cle_ltT at b c: a <=c b -> b <c c -> a <c c.
Lemma clt_ltT b a c: a <c b -> b <c c -> a <c c.
Lemma wordering_cle_pr x:
  cardinal_set x ->
  worder_on (graph_on cardinal_le x) x.
Lemma cleT_ell a b:
  cardinalp a -> cardinalp b -> [\/ a = b, a <c b | b <c a].
Lemma cleT_el a b:
  cardinalp a -> cardinalp b -> a <=c b \/ b <c a.
Lemma cleT_ee a b:
  cardinalp a -> cardinalp b -> a <=c b \/ b <=c a.
```

Minimum and maximum of cardinals. These quantities exist as the ordering of cardinals is total. We show associativity and distributivity.

```
Definition cmax:= Gmax cardinal_le.
Definition cmin:= Gmin cardinal_le.

Lemma CS_cmax x y: cardinalp x -> cardinalp y -> cardinalp(cmax x y).
Lemma CS_cmin x y: cardinalp x -> cardinalp y -> cardinalp(cmin x y).
Lemma cmax_xy x y: x <=c y -> cmax x y = y.
```

```
Lemma cmax_yx x y: y <=c x -> cmax x y = x.
Lemma cmin_xy x y: x <=c y -> cmin x y = x.
Lemma cmin_yx x y: y <=c x -> cmin x y = y.
Lemma cmaxC x y: cardinalp x -> cardinalp y -> cmax x y = cmax y x.
Lemma cminC x y: cardinalp x -> cardinalp y -> cmin x y = cmin y x.

Lemma cmax_p1 x y:  cardinalp x -> cardinalp y ->
   x <=c (cmax x y) /\ y <=c (cmax x y).
Lemma cmin_p1 x y:  cardinalp x -> cardinalp y ->
   (cmin x y)  <=c x /\ (cmin x y) <=c y.
Lemma cmax_p0 x y z: x <=c z -> y <=c z -> (cmax x y) <=c z.
Lemma cmin_p0 x y z: z <=c x -> z <=c y -> z <=c (cmin x y).
Lemma cmaxA x y z: cardinalp x -> cardinalp y -> cardinalp z ->
    cmax x (cmax y z) = cmax (cmax x y) z.
Lemma cminA x y z: cardinalp x -> cardinalp y -> cardinalp z ->
    cmin x (cmin y z) = cmin (cmin x y) z.
Lemma cminmax x y z:
  cardinalp x -> cardinalp y -> cardinalp z ->
  cmin x (cmax y z) = cmax (cmin x y) (cmin x z).
Lemma cmaxmin x y z:
  cardinalp x -> cardinalp y -> cardinalp z ->
  cmax x (cmin y z) = cmin (cmax x y) (cmax x z).
```

Ordinal and cardinal comparison.

```
Lemma cardinal_pr3 a: ordinalp a -> cardinal a <=o a.
Lemma colt1 a x: cardinalp x -> inc a x ->
  cardinal a <c x.
Lemma ocle1 y x:
  x <=o y -> (cardinal x) <=c (cardinal y).
Lemma oclt  x y: x <c y -> x <o y.
Lemma ocle2P x y: cardinalp x ->  ordinalp y ->
   ((y <o x) <-> (cardinal y <c x)).
Lemma ordinals_card_ltP y: cardinalp y ->
  forall x, inc x y <-> (ordinalp x /\ (cardinal x) <c y).
Lemma ocle3 x y:
   cardinalp x -> cardinalp y -> x <=o y -> x <=c y.
Lemma oclt3 x y:
   cardinalp x -> cardinalp y -> x <o y -> x <c y.
```

Cardinal comparison and finiteness. We write here $x+1$ as short for: the cardinal of the ordinal successor of x. We shall see later on that it is the cardinal sum of x and 1. If x is a cardinal, $x \neq x+1$ is equivalent to x is a finite cardinal. As seen above, $x = x+1$ is equivalent to x is an infinite cardinal. Note that x is finite if and only if $x < \omega$; infinite if and only if $\omega \leq x$.

```
Lemma finite_cP x: finite_c x <->  (cardinalp x /\ x <> csucc x).
(* Lemma infinite_cP x: infinite_c x <-> (cardinalp x /\ x = csucc x). *)
Lemma finite_c_P2 x: finite_c x <-> (x <c omega0).
Lemma infinite_c_P2 x: infinite_c x <-> (omega0 <=c x).
Lemma finite_dichot x: cardinalp x -> finite_c x \/ infinite_c x.
Lemma finite_dichot1 x: finite_set x \/ infinite_set x.
```

If a is finite and b is infinite, then $a < b$. If $b \leq c$ then c is infinite. If $d \leq a$ then d is finite. The last lemma here says: if $f : \mathrm{A} \to \mathrm{B}$ is injective and B is finite, then A is finite.

```
Lemma cle_fin_inf a b: finite_c a -> infinite_c b -> a <=c b.
Lemma clt_fin_inf a b: finite_c a -> infinite_c b -> a <c b.
Lemma cle_inf_inf a b: infinite_c a -> a <=c b -> infinite_c b.
Theorem cle_fin_fin a b: finite_c b -> a <=c b -> finite_c a.
Lemma sub_image_finite_set A B f:
  finite_set B -> (forall x, inc x A -> inc (f x) B) ->
  {inc A &, injective f} ->
  finite_set A.
```

A subset of a finite set is finite. A cardinal x is infinite if and only if $\omega \leq x$.

```
Lemma succ_of_finite x: finite_o x -> csucc x = osucc x.
Lemma sub_finite_set x y: sub x y -> finite_set y -> finite_set x.
```

We show here that if x is an ordinal, it is a finite or infinite set if and only if it is a finite or infinite ordinal, and this implies $x < \omega$ or $\omega \leq x$.

```
Section OrdinalFinite.
Variable x:Set.
Hypothesis ox: ordinalp x.

Lemma ordinal_finite1: finite_set x -> finite_o x.
Lemma ordinal_finite2: infinite_set x -> infinite_o x.
Lemma ordinal_finite3: finite_set x -> x <o omega0.
Lemma ordinal_finite4: infinite_set x -> omega0 <=o x.
End OrdinalFinite.
```

Some properties of zero and one. We have $0 \leq \mathfrak{a}$ for every cardinal $\mathfrak{a}$, and $1 \leq \mathfrak{a}$ if moreover $\mathfrak{a} \neq 0$. We have card(E) ≥ 1 if and only if E is non-empty. We have card(E) ≥ 2 if and only if E has at least two elements.

```
Lemma olt1 x: x <o \1o -> x = \0o.
Lemma olt2 a: a <o \2c ->  a = \0o \/ a = \1o.
Lemma oge1P x: (\1o <=o x) <-> (\0o <o x).
Lemma oge1 x: ordinalp x -> x <> \0o -> \1o <=o x.
Lemma ozero_dichot x: ordinalp x -> x = \0o \/ \0o <o x.
Lemma oone_dichot x y: x <o y -> (y = \1o \/ \1o <o y).

Lemma cle0x x: cardinalp x -> \0c <=c x.
Lemma czero_dichot x: cardinalp x -> x = \0c \/ \0c <c x.
Lemma clt0 x: x <c \0c -> False.
Lemma cle0 a: a <=c \0c -> a = \0c.
Lemma card_ne0_pos x: cardinalp x -> x <> \0c -> \0c <c x.
Lemma card_gt_ne0 x y: x <c y -> y <> \0c.
Lemma succ_nz  n: csucc n <> \0c.
Lemma succ_positive a: \0c <c (csucc a).

Lemma clt_01: \0c <c \1c.
Lemma cle_01: \0c <=c \1c.
Lemma cle_12: \1c <=c \2c.
Lemma clt_02: \0c <c \2c.

Lemma olt_01: \0o <o \1o.
```

```
Lemma ole_01: \0o <=o \1o.

Lemma cge1P x: (\1c <=c x) <-> (\0c <c x).
Lemma clt1 x: x <c \1o -> x = \0c.
Lemma cge1 x: cardinalp x -> x <> \0c -> \1c <=c x.
Lemma clt2 a: a <c \2c ->  a = \0c \/ a = \1c.
Lemma cge2 x: cardinalp x ->  x <> \0c ->  x <> \1c  -> \2c <=c x.
Lemma cle1P E: \1c <=c (cardinal E) <-> nonempty E.
Lemma cle2P E:
  \2c <=c (cardinal E)  <-> exists a b, [/\ inc a E, inc b E & a <> b].
```

Supremum of a family of cardinals. Let's introduce the sets C_a and C'_a formed of all x such that $x \leq_{\mathrm{card}} a$ and $x <_{\mathrm{card}} a$ respectively. We assume that a is a cardinal, for otherwise the sets are empty. If $x \in \mathrm{C}'_a$ then $x <_{\mathrm{ord}} a$ so that $x \in a$; this shows that C'_a is a set. Similarly, if $x \in \mathrm{C}_a$, then $x \in a^+$. Note that C_a is also the set of all cardinals in a^+ as well as the set of all $\mathrm{card}(t)$, where $t \in \mathfrak{P}(a)$

```
Definition cardinals_le a:= Zo (osucc a) cardinalp.
Definition cardinals_lt a:= Zo a (fun b => b <c a).

Lemma cardinals_leP a : cardinalp a ->
  forall b, (inc b (cardinals_le a) <-> (b <=c a)).
Lemma cardinals_ltP a: cardinalp a ->
  (forall b, inc b (cardinals_lt a) <-> (b <c a)).
Lemma cardinals_le_alt a: cardinalp a ->
  (cardinals_le a) = fun_image (\Po a) cardinal.
```

Given a family of cardinals there is a least upper bound (an upper bound a such that, for all other upper bounds b we have $a \leq b$). It is called the supremum of the family. This is Proposition 2 in [4, p. 160]. The proof is as follows. Let a be the union of the family: This is the least upper bound (as an ordinal). It suffices to show that it is a cardinal. So assume that b is an ordinal equipotent to a and $b <_{\mathrm{ord}} a$. It is not an upper bound of the family, thus for some element c of the family we have $b <_{\mathrm{ord}} c \leq_{\mathrm{ord}} a$. Let A, B and C be the cardinals of these quantities. We get $\mathrm{B} \leq_{\mathrm{card}} \mathrm{C} \leq_{\mathrm{card}} \mathrm{A}$. Since $\mathrm{A} = \mathrm{B}$, these three cardinals are equal. Thus b and c are equipotent. Since c is a cardinal, we get $c \leq_{\mathrm{ord}} b$, absurd.

```
Lemma CS_sup E: cardinal_set E -> cardinalp (\csup E).
Lemma card_ub_sup E y: cardinalp y -> (forall i, inc i E -> i <=c y) ->
  \csup E <=c y.
Lemma card_sup_ub E: cardinal_set E ->
  forall i, inc i E -> i <=c \csup E.
Lemma card_sup_image E f g:
  (forall x, inc x E -> f x <=c g x) ->
  \csup (fun_image E f) <=c \csup (fun_image E g).

Lemma cardinal_supremum1 x:
  cardinal_set x ->
  exists! b, [/\ cardinalp b,
    (forall a, inc a x -> a <=c b) &
    (forall c, cardinalp c ->  (forall a, inc a x -> a <=c c) ->
      b <=c c)].

Theorem cardinal_supremum2 x:
  fgraph x -> cardinal_fam x ->
```

```
  exists!b, [/\ cardinalp b,
    (forall a, inc a (domain x) ->(Vg x a) <=c b) &
    (forall c, cardinalp c ->
      (forall a, inc a (domain x) -> (Vg x a) <=c c) ->
      b <=c c)].
```

Proposition 3 in [4, p. 160] says that $\mathrm{card}(y) \leq \mathrm{card}(x)$ if there is a surjection of x onto y (the right inverse of the surjection being injective). As a consequence, the range of a function is not bigger than the source.

```
Lemma surjective_cle x y:
  (exists z, surjection_prop z x y) ->
  cardinal y <=c cardinal x.
Lemma image_smaller f: function f ->
  cardinal (Imf f) <=c cardinal (source f).
Lemma range_smaller f:
  fgraph f -> cardinal (range f) <=c cardinal (domain f).
Lemma image_smaller1 f x: function f ->
  cardinal (Vfs f x)  <=c cardinal x.
Lemma fun_image_smaller a f: cardinal (fun_image a f) <=c cardinal a.
```

4.3 Operations on cardinals

The remainder of the chapter is independent of the definition of the cardinals.

Given a family of cardinals $(\mathfrak{a}_\iota)_{\iota\in I}$, the cardinal of the sum of these sets is called the *cardinal sum* and denoted by $\sum_{\iota\in I} \mathfrak{a}_\iota$; the cardinal of the product is called the *cardinal product* and denoted $\mathsf{P}_{\iota\in I} \mathfrak{a}_\iota$. The qualificative “cardinal” will be omitted if there is no risk of confusion. The associated operations are called *addition* and *multiplication*. The notation $\prod_{\iota\in I} \mathfrak{a}_\iota$ will later be used for both the cartesian product and the cardinal product.

```
Definition csum x := cardinal (disjointU x).
Definition cprod x := cardinal (productb x).
Definition csumb a f := csum (Lg a f).
Definition cprodb a f := cprod (Lg a f).
```

In general, we shall consider a functional graph X. However, for any X, we may consider $i \mapsto X_i$ on the domain of X. This is a functional graph, and it has the same sum and product as X.

```
Lemma csum_gr f: csumb (domain f) (Vg f) = csum f.
Lemma cprod_gr f: cprodb (domain f) (Vg f) = cprod f.
Lemma csumb_exten A f g : {inc A, f =1 g} -> csumb A f = csumb A g.
Lemma cprodb_exten A f g : {inc A, f =1 g} -> cprodb A f = cprodb A g.
```

Proposition 4 of [4, p. 160] says that the cardinal sum or cardinal product of the family $\mathrm{card}(E_\iota)$ is the cardinal of the sum or the product of the sets E_ι. In other terms, if $\mathfrak{a}_\iota = \mathrm{card}(E_\iota)$ then $\mathrm{card}(\prod E_\iota) = \mathsf{P}\mathfrak{a}_\iota$ and $\mathrm{card}(\mathsf{S} E_\iota) = \sum \mathfrak{a}_\iota$. One can notice that the cardinal of a union is at most the cardinal of the disjoint union[3].

[3] Bourbaki has no notation for the disjoint union; we use a big S by analogy with the big P

```
Theorem cprod_pr f: cprod f =  cprodb (domain f) (fun a => cardinal (Vg f a)).
Theorem csum_pr f: csum f = csumb (domain f) (fun a => cardinal (Vg f a)).
Lemma csum_pr3 f g:
  domain f = domain g ->
  (forall i, inc i (domain f) -> Vg f i =c Vg g i) ->
  csum f = csum g.
Lemma csum_pr4 f:
   mutually_disjoint f ->
   cardinal (unionb f) = csumb (domain f) (fun a => cardinal (Vg f a)).
Lemma csum_pr1 f:
  cardinal (unionb f)
  <=c csumb (domain f) (fun a => cardinal (Vg f a)).
Lemma csum_pr1_bis X f:
  cardinal (unionf X f) <=c csumb X (fun a => cardinal (f a)).
Lemma csum_pr4_bis X f:
  (forall i j, inc i X -> inc j X -> i = j \/ disjoint (f i) (f j)) ->
  cardinal (unionf X f) = csumb X (fun a => cardinal (f a)).
```

Proposition 5 [4, p. 161] says that if f is a bijection from K to I and if $\mathfrak{a}_\iota$ is a cardinal then ("commutativity" of the sum and product):

$$\sum_{\kappa\in K}\mathfrak{a}_{f(\kappa)}=\sum_{\iota\in I}\mathfrak{a}_\iota,\qquad \prod_{\kappa\in K}\mathfrak{a}_{f(\kappa)}=\prod_{\iota\in I}\mathfrak{a}_\iota. \tag{4.1}$$

If the family $(J_\lambda)_{\lambda\in L}$ is a partition of I, then ("associativity" of the sum and product):

$$\sum_{\iota\in I}\mathfrak{a}_\iota=\sum_{\lambda\in L}\Big(\sum_{\iota\in J_\lambda}\mathfrak{a}_\iota\Big),\qquad \prod_{\iota\in I}\mathfrak{a}_\iota=\prod_{\lambda\in L}\Big(\prod_{\iota\in J_\lambda}\mathfrak{a}_\iota\Big). \tag{4.2}$$

Let $((\mathfrak{a}_{\lambda,\iota})_{\iota\in J_\lambda})_{\lambda\in L}$ be a family of families of cardinals. Let $I=\prod J_\lambda$. Distributivity of product over sum is

$$\prod_{\lambda\in L}\Big(\sum_{\iota\in J_\lambda}\mathfrak{a}_{\lambda,\iota}\Big)=\sum_{f\in I}\Big(\prod_{\lambda\in L}\mathfrak{a}_{\lambda,f(\lambda)}\Big). \tag{4.3}$$

The relations are trivial for the product, and in the case of the union, we have to check that the families are disjoint. Formulas (4.1), (4.2) and (4.3) are numbered (4), (5) and (6) by Bourbaki.

```
Theorem csum_Cn X f:
  target f = domain X -> bijection f ->
  csum X = csum (X \cf (graph f)).
Lemma csum_Cn' X f: bijection f ->
  csumb (target f) X = csumb (source f) (fun i => (X (Vf f i))).
Theorem cprod_Cn X f:
  target f = domain X ->  bijection f ->
  cprod X = cprod (X \cf (graph f)).
Theorem csum_An f g:
  partition_w_fam g (domain f) ->
  csum f = csumb (domain g) (fun l => csumb (Vg g l) (Vg f)).
Theorem cprod_An f g:
  partition_w_fam g (domain f) ->
  cprod f = cprodb (domain g) (fun l => cprodb (Vg g l) (Vg f)).
Theorem cprodDn f:
  cprodb (domain f) (fun l => csum (Vg f l)) =
  csumb (productf (domain f) (fun l => (domain (Vg f l))))
    (fun g => (cprod (Lg (domain f) (fun l => Vg (Vg f l) (Vg g l))))).
```

The case of two arguments. Given two sets $\mathfrak{a}$ and $\mathfrak{b}$, we can consider a family F defined on a doubleton $\{x, y\}$ such that $F(x) = \mathfrak{a}$ and $F(y) = \mathfrak{b}$. By commutativity, the cardinal sum and cardinal product of the family depends only on $\mathfrak{a}$ and $\mathfrak{b}$. It is denoted by $\mathfrak{a}+\mathfrak{b}$ and $\mathfrak{a}.\mathfrak{b}$ respectively. Commutativity is obvious.

```
Definition csum2 a b :=  csum (variantLc a b).
Definition cprod2 a b := cprod (variantLc a b).
Notation "x +c y" := (csum2 x y) (at level 50).
Notation "x *c y" := (cprod2 x y) (at level 40).

Lemma CS_sum2 a b: cardinalp (a +c b).
Lemma CS_prod2 a b: cardinalp (a *c b).

Lemma csum2_pr a b f:
  doubleton_fam f a b -> a +c b = csum f.
Lemma cprod2_pr a b f:
  doubleton_fam f a b -> a *c b = cprod f.

Fact card_commutative_aux a b:
  doubleton_fam (Lvariantc b a) a b.
Lemma csumC a b: a +c b = b +c a.
Lemma cprodC a b: a *c b = b *c a.
```

More properties.

```
Lemma disjointU2_pr3 a b x y: y <> x ->
  (a +c b) =c ((a *s1 x) \cup (b *s1 y)).
Lemma csum2_pr2 a b a' b':
  a =c a' -> b =c b' ->
  a +c b = a' +c b'.
Lemma csum2cl x y: (cardinal x) +c y = x +c y.
Lemma csum2cr x y: x +c (cardinal y) = x +c y.
Lemma csum2_pr5 a b: disjoint a b ->
   cardinal (a \cup b) = a +c b.
Lemma cardinal_setC2 a b: sub a b ->
   cardinal b = a +c (b -s a).
Lemma cardinal_setC3 a b:
  (a -s b) =c (b -s a) -> a =c b.

Lemma cprod2_pr1 a b:
  a *c b = cardinal (a \times b).
Lemma cprod2_pr2 a b: (a \times cardinal b) =c (a \times b).
Lemma cprod2cl x y: (cardinal x) *c y = x *c y.
Lemma cprod2cr x y: x *c (cardinal y) = x *c y.
```

Some commutativity properties. We have $\sum_{i\in I} f_i + \sum_{i\in I} g_i = \sum_{i\in I}(f_i + g_i)$. The proof is a bit tricky. Let $K = I \times \{\alpha, \beta\}$, where α and β are two distinct elements. We define a function h on K that associates f_i to (i, α) and g_i to (i, β). We have $\sum f_i = \sum_\alpha h_i$ where the index α denotes the restriction of h to the first part of K (this is the commutativity theorem, we use some auxiliary lemmas in order to ease the proof). We have $\sum f_i + \sum g_i = \sum h_k$ by associativity. If we consider K as the disjoint union where the first component is fixed, we can reapply the associativity theorem.

```
Definition quasi_bij f I J :=
```

```
  [/\ (forall x, inc x I -> inc (f x) J),
  (forall x y, inc x I -> inc y I -> f x = f y -> x = y) &
  (forall y, inc y J -> exists2 x, inc x I &  y = f x)].
Lemma cardinal_commutativity_aux f I J g (F := (Lf f I J)) (G := Lg J g) :
  quasi_bij f I J ->
  [/\ target F = domain G, bijection F &
    G \cf (graph F) = Lg I (fun z => Vg G (f z))].
Lemma csum_Cn2 J g I f : quasi_bij f I J ->
  csumb J g = csumb I (fun z => (g (f z))).
Lemma cprod_Cn2 J g I f : quasi_bij f I J ->
  cprodb J g = cprodb I (fun z => (g (f z))).

Lemma sum_of_sums_aux I f g: (* 82 *)
  (csumb I f) +c (csumb I g) =  csumb I (fun i => (f i) +c (g i)) /\
  (cprodb I f) *c (cprodb I g) = cprodb I (fun i => (f i) *c (g i)).
Lemma sum_of_sums f g I:
  (csumb I f) +c (csumb I g) = csumb I (fun i => (f i) +c (g i)).
Lemma prod_of_prods f g I:
  (cprodb I f) *c (cprodb I g) = cprodb I (fun i => (f i) *c (g i)).
```

As a corollary, if $\mathfrak{a}$, $\mathfrak{b}$ and $\mathfrak{c}$ are cardinals we have

$$\mathfrak{a}+\mathfrak{b}=\mathfrak{b}+\mathfrak{a} \qquad \text{and} \qquad \mathfrak{a}\mathfrak{b}=\mathfrak{b}\mathfrak{a}, \tag{4.4}$$

$$\mathfrak{a}+(\mathfrak{b}+\mathfrak{c})=(\mathfrak{a}+\mathfrak{b})+\mathfrak{c} \qquad \text{and} \qquad \mathfrak{a}(\mathfrak{b}\mathfrak{c})=(\mathfrak{a}\mathfrak{b})\mathfrak{c}, \tag{4.5}$$

$$\mathfrak{a}(\mathfrak{b}+\mathfrak{c})=\mathfrak{a}\mathfrak{b}+\mathfrak{a}\mathfrak{c}. \tag{4.6}$$

Associativity of the product is a consequence of equipotency of $\mathrm{A}\times(\mathrm{B}\times\mathrm{C})$ and $(\mathrm{A}\times\mathrm{B})\times\mathrm{C}$. Associativity of the sum is a consequence of associativity of $\cup$, commutativity of $+$ and $\cup$, and the property that $\mathfrak{a}+(\mathfrak{b}+\mathfrak{c})$ is equipotent $\mathfrak{a}_i\cup(\mathfrak{b}_j\cup\mathfrak{c}_k)$, if the indices are distinct; the current proof is four times shorter than the original one. The same idea can be used for (4.6). The quantity $\mathfrak{a}(\mathfrak{b}+\mathfrak{c})$ is equipotent to $\mathfrak{a}\times(\mathfrak{b}_i\cup\mathfrak{c}_j)$ hence to $(\mathfrak{a}\times\mathfrak{b}_i)\cup(\mathfrak{a}\times\mathfrak{c}_j)$, and $(\mathfrak{a}\times\mathfrak{b})_i\cup(\mathfrak{a}\times\mathfrak{c})_j$, by associativity of the product. We show in fact $(\mathfrak{b}+\mathfrak{c})\mathfrak{a}=\mathfrak{b}\mathfrak{a}+\mathfrak{c}\mathfrak{a}$, because, basically we use equipotency of $(\mathrm{A}\cup\mathrm{B})\times\mathrm{C}$ and $(\mathrm{A}\times\mathrm{C})\cup(\mathrm{B}\times\mathrm{C})$. The same techniques as above show

$$\mathfrak{a}\sum_{\iota\in\mathrm{I}}\mathfrak{b}_\iota=\sum_{\iota\in\mathrm{I}}\mathfrak{a}\mathfrak{b}_\iota. \tag{4.7}$$

```
Lemma cprodA a b c:
  a *c (b *c c) = (a *c b) *c c.
Lemma csumA a b c:
   a +c (b +c c) = (a +c b) +c c.
Lemma csumCA a b c: a +c (b +c c) = b +c (a+c c).
Lemma csprodACA: interchange cprod2 cprod2.
Lemma csumACA: interchange csum2 csum2.
Lemma cprodDr a b c:
  a *c (b +c c) = (a *c b) +c (a *c c).
Lemma cprodDl  a b c:
  (b +c c) *c a = (b *c a) +c (c *c a).
Lemma cprod2Dn a f:
  a *c (csum f) = csumb (domain f) (fun i => a *c (Vg f i)).
```

4.4 Properties of the cardinals 0 and 1

If a family is empty, the sum is zero and the product is one. If a family has a single element that is a cardinal, this element is the sum or the product.

```
Lemma csum_trivial f: domain f = emptyset -> csum f = \0c.
Lemma csum_trivial0 f: csumb emptyset f = \0c.
Lemma csum_trivial1 x f: domain f = singleton x -> csum f = cardinal (Vg f x).
Lemma csum_trivial2 x f: domain f = singleton x -> cardinalp (Vg f x) ->
  csum f =  Vg f x.
Lemma csum_trivial3 x f: csumb (singleton x) f = cardinal (f x).
Lemma csum_trivial4 f a:
  csum (restr f (singleton a))  = cardinal (Vg f a).

Lemma cprod_trivial f: domain f = emptyset -> cprod f = \1c.
Lemma cprod_trivial0 f: cprodb emptyset f = \1c.
Lemma cprod_trivial1 x f: domain f = singleton x -> cprod f = cardinal(Vg f x).
Lemma cprod_trivial1 x f: domain f = singleton x -> cardinalp (Vg f x) ->
  cprod f =  Vg f x.
Lemma cprod_trivial3 x f: cprodb (singleton x) f = cardinal (f x).
Lemma cprod_trivial4 f a:
  cprod (restr f (singleton a)) = cardinal (Vg f a).
```

We have $\sum_{x\in A\cup B} f(x) = \sum_{x\in A} f(x) + \sum_{x\in B} f(x)$ whenever A and B are disjoint. A special case is when B is a singleton. Assume $g : X \to I$ is a mapping. Let X_i the set of all x such that $g(x) = i$. This is a partition of X, so that the cardinal of X is the sum of the cardinals of the X_i.

```
Lemma csumA_setU1 A b f: ~ (inc b A) ->
  csumb (A +s1 b) f = csumb A f +c (f b).
Lemma csumA_setU2 A B f: disjoint A B ->
  csumb (A \cup B) f = csumb A f +c csumb B f.
Lemma cprodA_setU2 A B f: disjoint A B ->
  cprodb (A \cup B) f = cprodb A f *c cprodb B f.
Lemma cprodA_setU1 A b f: ~ (inc b A) ->
  cprodb (A +s1 b) f = cprodb A f *c (f b).
Lemma card_partition_induced X f F:
  (forall x, inc x X -> inc (f x) F) ->
  cardinal X = csumb F (fun k => cardinal (Zo X (fun z => f z = k))).
Lemma card_partition_induced1 X f F g:
  (forall x, inc x X -> inc (f x) F) ->
  csumb X g = csumb F (fun i => csumb (Zo X (fun z => f z = i)) g).
```

One can remove 0 in a sum and 1 in a product. This is Proposition 6 [4, p. 162]. The result is clear for the sum, because $0_i = \emptyset$ (where 0_i means $0 \times \{i\}$). Conversely, is a sum is zero, the union of the x_i is empty, so eaxh x_i is empty, and each term of the sum is zero. In the case of a product, it is a trivial consequence of `bijective_prj`. If the family has two elements, this gives nice results. If a factor of a product is zero, so is the product itself.

```
Theorem csum_zero_unit f j:
  sub j (domain f) ->
  (forall i, inc i ((domain f) -s j) -> (Vg f i) = \0c) ->
  csum f = csumb j (Vg f).
Theorem cprod_one_unit f j:
  sub j (domain f) ->
  (forall i, inc i ((domain f) -s j) -> (Vg f i) = \1c) ->
```

```
  cprod f = cprodb j (Vg f).
Lemma csum_zero_unit_bis f:
  (allf f (fun z => z = \0c)) <-> csum f = \0c.

Lemma csum0r a: cardinalp a ->  a +c \0c = a.
Lemma csum0l a: cardinalp a -> \0c +c a = a.
Lemma csum_0l a: \0c +c a = cardinal a.
Lemma csum_nz a b: a +c b = \0c -> (a = \0c /\ b = \0c).
Lemma cprod1r a: cardinalp a -> a *c \1c = a.
Lemma cprod1l a: cardinalp a ->\1c *c a = a.
Lemma cprod0r a: a *c \0c = \0c.
Lemma cprod0l a: \0c *c a = \0c.
Lemma cprod_eq0 f:
  (exists2 i, inc i (domain f) & cardinal (Vg f i) = \0c) ->
  cprod f = \0c.
```

Let $\mathfrak{a}$ and $\mathfrak{b}$ be two cardinals; consider a set I equipotent to $\mathfrak{b}$ and the two families $\mathfrak{a}_\iota = \mathfrak{a}$ and $\mathfrak{c}_\iota = 1$. Then

$$\mathfrak{a}\mathfrak{b} = \sum_{\iota\in I} \mathfrak{a}_\iota; \qquad \mathfrak{b} = \sum_{\iota\in I} \mathfrak{c}_\iota. \tag{4.8}$$

The first formula is obtained from the second after multiplication by $\mathfrak{a}$, and using distributivity. We prove a similar result for $I = \mathfrak{b}$, and when $\mathfrak{a}$ and $\mathfrak{b}$ are any sets. The cardinal successor of x is $x+1$ (since this is $\mathrm{Card}(x^+)$ and x^+ is the disjoint union of x and a singleton). Since 2 is the successor of 1, it follows $1+1=2$ and $x+x=2x$.

```
Lemma csum_of_ones b: csum (cst_graph b \1c) = cardinal b.
Lemma csum_of_same a b: csum (cst_graph b a) = a *c b.
Lemma csum_of_ones1 b j: cardinalp b -> b \Eq j ->
  csumb j (fun _ => \1c) = b.
Lemma csum_of_same1 a b j:
  cardinalp a -> cardinalp b -> b \Eq j ->
  csum (cst_graph j a) = a *c b.
Lemma csucc_pr3 x: csucc (cardinal x) = x +c \1c.
Lemma csucc_pr4 x: cardinalp x ->  csucc x = x +c \1c.
Lemma csucc_pr5 a: cardinal (csucc a) = csucc a.
Lemma card_two_pr: \1c +c \1c = \2c.
Lemma two_times_n n: \2c *c n = n +c n.
```

Proposition 7 [4, p. 162] says that a cardinal product is non-zero if and only if each factor is non-zero (because a product is non-empty if and only if no factor is empty). Proposition 8 [4, p. 162] asserts injectivity of the successor function, namely that if $\mathfrak{a}$ and $\mathfrak{b}$ are two cardinals such that $\mathfrak{a}+1=\mathfrak{b}+1$ then $\mathfrak{a}=\mathfrak{b}$. In effect, there exists X equipotent to $\mathfrak{a}$, Y equipotent to $\mathfrak{b}$, and $u\notin X$, $v\notin Y$ such that $X\cup\{u\} = Y\cup\{v\}$. If $u=v$, then $X=Y$; otherwise, if $Z=Y\cap Y$, we have $X=Z\cup\{v\}$ and $Y=Z\cup\{u\}$, so that if $\mathfrak{c}=\mathrm{Card}(Z)$ we have $\mathfrak{a}=\mathfrak{b}=\mathfrak{c}+1$. The proof is rather long. However this is a trivial consequence of `succ_injective1`.

```
Definition card_nz_fam f := allf f (fun z => z <> \0c).
Theorem cprod_nzP f: card_nz_fam f <-> (cprod f <> \0c).
Lemma cprod2_nz a b: a <> \0c -> b <> \0c ->  a *c b <> \0c.
Lemma cprod2_eq0 a b: a *c b = \0c -> a = \0c \/ b = \0c.
Theorem succ_injective a b: cardinalp a ->  cardinalp b ->
  a +c \1c = b +c \1c -> a = b.
```

4.5 Exponentiation of cardinals

If $\mathfrak{a}$ and $\mathfrak{b}$ are two cardinals, the cardinal of the set of functions from $\mathfrak{b}$ to $\mathfrak{a}$ is denoted $\mathfrak{a}^{\mathfrak{b}}$, by abuse of notations[4]. Proposition 9 [4, p. 163] says that we can replace $\mathfrak{a}$ and $\mathfrak{b}$ by equipotent sets.

```
Definition cpow a b := cardinal (functions b a).
Notation "x ^c  y" := (cpow x y) (at level 30).

Lemma CS_pow a b: cardinalp (a ^c b).
Lemma cpow_pr0 a b: a ^c b = cardinal (gfunctions b a).
Lemma cpow_pr a b a' b':
  a =c a' -> b =c b' -> a ^c b = a' ^c b'.
Lemma cpow_prr a b b': b =c b' -> a ^c b = a ^c b'.
Lemma cpow_prl a a' b:  a =c a' -> a ^c b = a' ^c b.
Lemma cpowcl a b: (cardinal a) ^c b = a ^c b.
Lemma cpowcr a b: a ^c (cardinal b) = a ^c b.
Theorem cpow_pr1 x y:
  cardinal (functions y x) = (cardinal x) ^c (cardinal y).
```

Proposition 10 [4, p. 163] says that if $\mathfrak{a}$ and $\mathfrak{b}$ are two cardinals, I is a set with cardinal $\mathfrak{b}$ and $\mathfrak{a}_\iota$ is the constant family $\mathfrak{a}$, then $\mathfrak{a}^{\mathfrak{b}} = \prod_{\iota\in I} \mathfrak{a}_\iota$. This is a trivial consequence of the fact that the set of functions and the set of graphs of functions are equipotent. Note that Card(I) = $\mathfrak{b}$ implies that $\mathfrak{b}$ is a cardinal, hence, we give also another version of the theorem.

A consequence is that, if $\mathfrak{a}$ and $\mathfrak{b}$ are cardinals, $(\mathfrak{a}_\iota)_{\iota\in I}$ and $(\mathfrak{b}_\iota)_{\iota\in I}$ are families of cardinals we have

$$\mathfrak{a}^{\sum_{\iota\in I}\mathfrak{b}_\iota} = \prod_{\iota\in I}\mathfrak{a}^{\mathfrak{b}_\iota}, \qquad \Big(\prod_{\iota\in I}\mathfrak{a}_\iota\Big)^{\mathfrak{b}} = \prod_{\iota\in I}\mathfrak{a}_\iota^{\mathfrak{b}}. \tag{4.9}$$

The proof of the first formula is as follows. Let $\mathfrak{a}_\iota = \mathfrak{a}$. We have $\mathfrak{a}^{\sum_{\iota\in I}\mathfrak{b}_\iota} = \prod_J \mathfrak{a}_\iota$, where J is any set whose cardinal is $\sum_{\iota\in I}\mathfrak{b}_\iota$; we choose the disjoint union of the sets $\mathfrak{b}_\iota$. We have a natural partition of J and we can apply the associativity of the product. For the second formula, let $\mathfrak{a}_{\iota\beta} = \mathfrak{a}_\iota$ for $\iota \in I$ and $\beta \in \mathfrak{b}$. This is a family defined on the product $I \times \mathfrak{b}$, for which we have two natural partitions (all elements with the same ι, or all elements with the same β); we can apply associativity of the product twice. We also have

$$\mathfrak{a}^{\mathfrak{b}+\mathfrak{c}} = \mathfrak{a}^{\mathfrak{b}}\mathfrak{a}^{\mathfrak{c}}, \qquad (\mathfrak{a}\mathfrak{b})^{\mathfrak{c}} = \mathfrak{a}^{\mathfrak{c}}\mathfrak{b}^{\mathfrak{c}}, \qquad \mathfrak{a}^{\mathfrak{b}\mathfrak{c}} = (\mathfrak{a}^{\mathfrak{b}})^{\mathfrak{c}}. \tag{4.10}$$

```
Lemma cpow_pr2 a b: cprod (cst_graph b a) = a ^c b.
Theorem cpow_pr3 a b j: cardinal j = b ->
  cprod (cst_graph j a) = a ^c b.
Lemma cpow_sum a f:
  a ^c (csum f) =  cprodb (domain f) (fun i => a ^c (Vg f i)).
Lemma cpow_prod b f: (* 60 *)
  (cprod f) ^c b = cprodb (domain f) (fun i => (Vg f i) ^c b).
Lemma cpow_sum2 a b c:  a ^c (b +c c) = (a ^c b) *c (a ^c c).
Lemma cpow_prod2 a b c: (a *c b) ^c c =  (a ^c c) *c (b ^c c).
Lemma cpow_pow a b c: a ^c (b *c c) = (a ^c b) ^c c.
```

[4] The trouble seems to be that 4^2 and 2^4 denote the set of graphs of mappings from 2 to 4 or from 4 to 2; these sets are obviously distinct, but have the same number of elements; hence $4^2 = 2^4$ is true with these new notations, see page 101. Some authors write ${}^{E}F$ for the set of functions $E \to F$.

Proposition 11 [4, p. 164] states that

$$\mathfrak{a}^0 = 1, \qquad \mathfrak{a}^1 = \mathfrak{a}, \qquad 1^{\mathfrak{a}} = 1, \qquad 0^{\mathfrak{b}} = 0 \quad (\mathfrak{b} \neq 0). \tag{4.11}$$

The Bourbaki proof is the following. We want to compute the number of functions from F to E is some cases. If F is empty, there is only the empty function; if F is a singleton then E^F and E are equipotent (the bijection is `product1_canon`), if E has a single element, there is only one function, a constant; finally if the source is non-empty and the target is empty, there is no function. We use different properties. In the first two cases, we replace the power by a product whose index set has 0 or 1 element, and simplify the result. In the third case we rewrite 1 as a product whose index set is empty, and use distributivity (4.9).

Note that $\mathfrak{a}^2 = \mathfrak{a}.\mathfrak{a}$.

```
Lemma cpowx0 a: a ^c \0c =  \1c.
Lemma cpow00: \0c ^c \0c =  \1c.
Lemma cpowx1c a: a ^c \1c = cardinal a.
Lemma cpowx1 a: cardinalp a -> a ^c \1c = a.
Lemma cpow1x a: \1c ^c  a = \1c.
Lemma cpow0x a: a <> \0c ->\0c ^c a = \0c.
Lemma cpowx2 a: a ^c \2c = a *c a.
```

Characteristic Function. For any set X, and any subset A of X, the function ϕ_A defined on X by 1 if $x \in A$ and 0 otherwise, is called the characteristic function of A. We state some trivial results. We then show that a subset of X is uniquely characterized by its characteristic function. Any function f defined on X with values 0 or 1 is a characteristic function (of the set of all x such that $f(x) = 1$).

We deduce Proposition 12 [4, p. 164]: the cardinal of the power set of X is 2^X.

```
Definition char_fun A B :=  Lf (varianti A \1c \0c)  B (doubleton \1c \0c).

Lemma char_fun_axioms A B:
  lf_axiom (varianti A \1c \0c) B (doubleton \1c \0c).
Lemma char_fun_f A B: function (char_fun A B).
Lemma char_fun_V A B x:
  inc x B -> Vf (char_fun A B) x = varianti A \1c \0c x.
Lemma char_fun_V_cardinal A B x:
  inc x B -> cardinalp (Vf (char_fun A B) x).
Lemma char_fun_V_a  A B x: sub A B -> inc x A ->
  Vf (char_fun A B) x = \1c.
Lemma char_fun_V_b A B x: sub A B -> inc x (B -s A) ->
  Vf (char_fun A B) x = \0c.
Lemma char_fun_injectiveP A A' B: sub A B -> sub A' B ->
  ((A=A') <-> (char_fun A B = char_fun A' B)).
Lemma char_fun_surjective X f: function f -> source f = X ->
  target f = (doubleton \1c \0c) ->
  exists2 A, sub A X & char_fun A X = f.

Theorem card_setP X: cardinal (\Po X) = \2c ^c  X.
```

4.6 Order relation and operations on cardinals

We shall write $a - b$ for the cardinal of the complement of b in a. This operation will be studied later on.

```
Definition cdiff a b := cardinal (a -s b).
Notation "x -c y" := (cdiff x y) (at level 50).
```

Proposition 13 [4, p. 164] states that $\mathfrak{a} \geq \mathfrak{b}$ if and only if there exists $\mathfrak{c}$ such that $\mathfrak{a} = \mathfrak{b} + \mathfrak{c}$ (for simplicity, we assume all three quantities to be cardinals, although $\mathfrak{c}$ could be any set, and both relations $\mathfrak{a} \geq \mathfrak{b}$ and $\mathfrak{a} = \mathfrak{b} + \mathfrak{c}$ imply that $\mathfrak{a}$ is a cardinal).

Proof. Assume B equipotent to a subset X of A and let C be the complement. Then $\mathrm{B}+\mathrm{C}$ is equipotent to $\mathrm{B}_1 \cup \mathrm{C}_2$ (where $\mathrm{B}_1 = \mathrm{B} \times \{1\}$). We can replace B by X, then omit the indices, so that $\mathrm{B}+\mathrm{C}$ is equipotent to A. Conversely, if A is equipotent $\mathrm{B}_1 \cup \mathrm{C}_2$, there is a bijection $\mathrm{B}_1 \cup \mathrm{C}_2 \to \mathrm{A}$, and by restriction, an injection $\mathrm{B}_1 \to \mathrm{A}$, and by composition, an injection $\mathrm{B} \to \mathrm{A}$.

Using von Neumann cardinals simplifies the proof: let c be the cardinal of the complement of b in a. If a and b are cardinals and $b \leq a$ then $b \subset a$ and $a = b + c$. This is also $a = b + (a - b)$.

```
Lemma CS_diff a b: cardinalp (a -c b).
Lemma cardinal_setC A E: sub A E ->
  (cardinal A) +c (E -c A)  = cardinal E.
Lemma cdiff_pr a b: b <=c a -> b +c (a -c b) = a.
Theorem cardinal_le_setCP a b:
  cardinalp a -> cardinalp b ->
  ((b <=c a) <-> (exists2 c, cardinalp c & b +c c = a)).
```

Proposition 14 [4, p. 165] says that if $\mathfrak{a}_\iota \geq \mathfrak{b}_\iota$ (for two families of cardinals) we have

$$\sum_{\iota\in\mathrm{I}} \mathfrak{a}_\iota \geq \sum_{\iota\in\mathrm{I}} \mathfrak{b}_\iota, \qquad \mathrm{P}_{\iota\in\mathrm{I}} \mathfrak{a}_\iota \geq \mathrm{P}_{\iota\in\mathrm{I}} \mathfrak{b}_\iota. \tag{4.12}$$

The first formula is shown as follows. We have a bijection from $\mathfrak{b}_\iota$ into a subset E_ι of $\mathfrak{a}_\iota$, hence a bijection from $\mathfrak{b}_\iota \times \{\iota\}$ into a subset $\mathrm{E}_\iota \times \{\iota\}$ of $\mathfrak{a}_\iota \times \{\iota\}$. This gives a bijection from the disjoint union $\bigcup \mathfrak{b}_\iota \times \{\iota\}$ into a subset $\bigcup \mathrm{E}_\iota \times \{\iota\}$ of $\bigcup \mathfrak{a}_\iota \times \{\iota\}$. The proof of the second formula is similar: we get a bijection from $\prod \mathfrak{b}_\iota$ into a subset $\prod \mathrm{E}_\iota$ of $\prod \mathfrak{a}_\iota$.

As a corollary, we obtain a smaller result if we restrict the domain of the sum or the product; in the case of a product, we assume all factors nonzero (proof: missing terms are replaced by zero, or one). The power is increasing with respect to both arguments. [Note: using von Neumann cardinals simplifies the proof, since in the first three cases, we must show $\mathrm{card}(\mathrm{A}) \leq \mathrm{card}(\mathrm{B})$ and $\mathrm{A} \subset \mathrm{B}$ is trivial]. If we have a family of sets, each of which has cardinal $\leq n$, then the cardinal of the union is at most n times the number of elements of the family.

```
Theorem csum_increasing f g:
  domain f = domain g ->
  (forall x, inc x (domain f) -> (Vg f x) <=c (Vg g x)) ->
  (csum f) <=c (csum g).
Theorem cprod_increasing f g:
  (forall x, inc x (domain f) -> (Vg f x) <=c (Vg g x)) ->
  (cprod f) <=c (cprod g).

Lemma cardinal_uniona X n:  (forall x, inc x X -> cardinal x <=c n) ->
  cardinal (union X) <=c n *c cardinal X.
Lemma csum_increasing1 f j:
  sub j (domain f) -> (csum (restr f j)) <=c (csum f).
Lemma cprod_increasing1 f j: card_nz_fam f ->
  sub j (domain f) -> (cprod (restr f j)) <=c (cprod f).
```

```
Lemma csum_increasing6 f j:  cardinalp (Vg f j) ->
  inc j (domain f) -> (Vg f j) <=c (csum f).
Lemma cprod_increasing6 f j:  cardinalp (Vg f j) -> card_nz_fam f ->
  inc j (domain f) -> (Vg f j) <=c (cprod f).
```

Comparison with two argument functions.

```
Lemma csum_Mlele a b a' b':
  a <=c a' -> b <=c b' -> (a +c b) <=c (a' +c b').
Lemma csum_Mleeq a b b': b <=c b' -> (b +c a) <=c (b' +c a).
Lemma csum_Meqle a b b': b <=c b' -> (a +c b) <=c (a +c b').
Lemma csum2_pr6 a b: cardinal (a \cup b) <=c a +c b.
Lemma cprod_Mlele a b a' b':
  a <=c a' -> b <=c b' -> (a *c b) <=c (a' *c b').
Lemma cprod_Meqle a b b': b <=c b' -> (a *c b) <=c (a *c b').
Lemma cprod_Mleeq a b b': b <=c b' -> (b *c a) <=c (b' *c a).
Lemma csum_M0le a b: cardinalp a -> a <=c (a +c b).
Lemma cprod_M1le a b: cardinalp a -> b <> \0c -> a <=c (a *c b).
Lemma csum_eq1 a b: cardinalp a -> cardinalp b -> a +c b = \1c ->
    (a = \0c \/ b = \0c).
Lemma cprod_eq1 a b: cardinalp a -> cardinalp b -> a *c b = \1c ->
    (a = \1c /\ b = \1c).
Lemma cpow_Mleeq x y z: x <=c y  ->  x <> \0c -> x ^c  z <=c y ^c z.
Lemma cpow_Meqle x a b: x <> \0c -> a <=c b -> x ^c a <=c x ^c b.
Lemma cpow_Mlele a b a' b':
  a <> \0c -> a <=c a' -> b <=c b' ->  (a ^c b) <=c (a' ^c b').
Lemma cpow_M2le x y: x <=c y -> \2c ^c x <=c \2c ^c y.
Lemma cpow_Mle1 a b:
  cardinalp a -> b <> \0c  -> a <=c (a ^c b).
```

Given a set E, there is an injection $f : \mathrm{E} \to \mathfrak{P}(\mathrm{E})$ but no surjection. One injection is $x \mapsto \{x\}$. Assume that there is a surjection, and consider y such that $f(y)$ is the set F of all $x \in \mathrm{E}$ such that $x \not\in f(x)$. Both claims $y \in f(y)$ and $y \not\in f(y)$ hold, contradiction.

We deduce Cantor's theorem (Theorem 2, [4, p. 165]) stating that $2^{\mathfrak{a}} > \mathfrak{a}$ for every cardinal $\mathfrak{a}$, so that there is no set containing all cardinals (for otherwise this set would have a greatest element).

```
Theorem cantor a: cardinalp a ->
  a <c (\2c ^c  a).
Lemma cantor_bis: non_coll cardinalp.
Lemma infinite_pow2 x: infinite_c x -> infinite_c (\2c ^c x).
```

Let c be a cardinal, and E the set of all elements of 2^c such that $\mathrm{card}(x) \le c$. If $x \in \mathrm{E}$, then x is an ordinal (since 2^c is an ordinal) and $\mathrm{card}(x) \le c$. Conversely, if x is an ordinal such that $\mathrm{card}(x) \le c$, then $\mathrm{card}(x) < 2^c$. This says that the ordinal x is less than 2^c, thus $x \in \mathrm{E}$.

If $x \in \mathrm{E}$ and $y \le x$, then y is a subset of x, $\mathrm{card}(y) \le \mathrm{card}(x)$ so that $y \in \mathrm{E}$. We deduce that E is an ordinal. Note that card(E) cannot be $\le c$ since E is irreflexive, so that $c < \mathrm{card}(\mathrm{E})$. Let d be any cardinal such that $c < d$. If $x \in \mathrm{E}$, we have $\mathrm{card}(x) < d$, this implies that the ordinal x is less than d, thus $x \in d$. In other terms, $\mathrm{E} \subset d$. This relation says that E is a cardinal. We can restate $\mathrm{E} \subset d$ as $\mathrm{E} \le d$, and $c < \mathrm{card}(\mathrm{E})$ as $c < \mathrm{E}$. This means that E is the least cardinal greater than c. We call this the *cardinal successor* of c.

```
Definition cnext c := Zo (\2c ^c c) (fun z => cardinal z <=c c).
```

```
Lemma cnextP c: cardinalp c -> forall x,
  (inc x (cnext c) <-> (ordinalp x /\ cardinal x <=c c)).
Lemma cnext_pr1 c (a:= cnext c): cardinalp c ->
 [/\ cardinalp a, c <c a & forall c', c <c c' -> a <=c c'].

Lemma CS_cnext x: cardinalp x -> cardinalp (cnext x).
Lemma cnext_pr4 x y: cardinalp y -> x <c cnext y -> x <=c y.
Lemma cnext_pr2 x: cardinalp x -> x <c cnext x.
Lemma cnext_pr3 x: cardinalp x ->  cnext x <=c \2c ^c x.
Lemma cnext_pr5P x : cardinalp x ->
   forall y, (x <c y <-> cnext x <=c y).
Lemma cnext_pr4P y: cardinalp y ->
   forall x, (x <c cnext y <-> x <=c y).
Lemma cnext_pr6 x y: x <=c y -> cnext x <=c cnext y.
```

Chapter 5

Natural integers. Finite sets

Bourbaki makes a distinction between finite and infinite cardinals. Finite cardinals are identified with *natural integers*, which are entities satisfying some arithmetic properties (addition, multiplication, subtraction and division are studied in the next chapter) derived from an induction principle and a successor function. There is a set **N** containing all finite cardinals, so that we have statements of the form: if $n \in \mathbf{N}$ then $n \neq n+1$, instead of: if $\mathfrak{a}$ is an infinite cardinal then $\mathfrak{a} = \mathfrak{a}+1$. In the Bourbaki theory, a cardinal is a set: one could try to prove $1+1=2$ by showing that $x \in 1+1$ is equivalent to $x \in 2$. However since 2 is constructed via the axiom of choice, the statement $1 \in 2$ is unprovable (all we we know is that 2 is a set with two distinct elements). For this reason, natural integers are sometimes considered as urelements (objects that may appear at the left-hand side of $\in$, but never the right-hand-side). In Version 4 of this document, a natural integer will be a finite von Neumann ordinal. This gives an explicit form for integers (for instance 2 is $\{\emptyset, \{\emptyset\}\}$) and the set of integers (the least limit ordinal). This form is however not adapted to computations (there is no explicit form of the sum of two ordinals, and the ordinal sum of two cardinals is not always a cardinal).

Integers are presented in [13] as follows. There is a symbol O and a symbol S, and two operations $a+b$ and $a \cdot b$ (sum and product), defined on integers, which are a finite (maybe empty) sequences of letters S followed by a single O. The five axioms are

Axiom 1 $\forall a, \mathrm{S}a \neq \mathrm{O}$.
Axiom 2 $\forall a, a+\mathrm{O} = a$.
Axiom 3 $\forall a \forall b, a+\mathrm{S}b = \mathrm{S}(a+b)$.
Axiom 4 $\forall a, a \cdot \mathrm{O} = \mathrm{O}$.
Axiom 5 $\forall a \forall b, a \cdot \mathrm{S}b = (a \cdot b)+a$.

The first axiom has an unusual form, since most axioms are of the form $a \implies b = c$. This axiom is built-in in Coq: an object of a type with n constructors is defined by a single constructor: an integer is either O or Sa, but not both. This means that if c an integer, one and only one of axioms 2 and 3 apply to $a+c$.

The axiom implies injectivity of S (by induction on the size of the arguments). Note that Coq defines addition an multiplication by induction on the first argument.

In the system presented above, it is impossible to prove $\forall a, a = \mathrm{O}+a$, although the result is obvious for any a. Thus a new principle is needed. It says something like: "If all the strings in a pyramidal family are theorems, then so is the universally quantified string which summarizes them". (We get a pyramid if we center the statements $a = \mathrm{O}+a$, for consecutive values of a). The whole pyramid has an infinite number of statements, and proving it requires an infinite proof. Assume that each line can be shown from the previous one, using exactly the same argument. Then the proof has the form P and Q and Q and Q, etc. It is infinite, but not too much, hence is accepted. The induction principle is: "Suppose u is a variable, and $\mathrm{X}\{u\}$ is a well-formed formula in which u occurs free. If both $\forall u : \langle \mathrm{X}\{u\} \supset \mathrm{X}\{\mathrm{S}u/u\}\rangle$ and $\mathrm{X}\{0/u\}$ are theorems, then $\forall u : \mathrm{X}\{u\}$ is also a theorem." This is built-in in COQ, under the form

```
nat_ind =
[eta nat_rect]
     : forall P : nat -> Prop,
       P 0 -> (forall n : nat, P n -> P n.+1) -> forall n : nat, P n
```

In this chapter we shall prove that the Bourbaki integers satisfy the induction principle, under the form

```
Nat_induction
     : forall r : property,
       r \0c ->
       (forall n : Set, inc n Nat -> r n -> r (csucc n)) ->
       forall n : Set, inc n Nat -> r n
```

and as a consequence, that all these definitions are essentially the same. The proof of the principle is as follows: the least element of the set (assumed non-empty) of elements not satisfying a property is either O or Sa. This is a consequence of the fact that **N** is well-ordered. Note that the property shown by induction (X, P, r, in the examples) is quantified in COQ, but neither in [13] nor in Bourbaki.

An important property of integers is the possibility of defining a function by induction. This is a COQ example

```
Fixpoint add (n m:nat) {struct n} : nat :=
   match n with
   | O => m
   | S p => S (add p m)
   end.
```

It defines (by induction on the first argument) the unique function $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ satisfying Axioms 2 and 3. Another example is the following

```
Fixpoint nat_to_B (n:nat) :=
  match n with O => \0c | S m => csucc (nat_to_B m) end.
```

One can define **N** as the range of this function which becomes a bijection $\mathbb{N} \to \mathbf{N}$, and **N** is thus isomorphic to the Bourbaki set of integers. In a version 2 of this document, this isomorphism was used to convert theorems proved in the standard library of COQ into theorems about finite cardinals.

In the Bourbaki framework, one can define functions by induction (i.e., by transfinite induction on the well-ordered set **N**). So, one could define, for each m, a function $f_m : \mathbf{N} \to \mathrm{E}_m$, which is the unique surjective function satisfying the two axioms, show that $\mathrm{E}_m \subset \mathbf{N}$, extend the function $f'_m : \mathbf{N} \to \mathbf{N}$, then merge all these functions to get $f : \mathbf{N} \times \mathbf{N} \to \mathbf{N}$ and show that the axioms of the sum are satisfied (for an example of a function defined by induction using a second argument, see the definition of the binomial coefficient below). On the other hand, one can define sum, product and exponentiation of ordinals (see exercises 2.17 and 2.18) by transfinite induction, these operations are not functions (there is no set containing all ordinals), but produce a finite ordinal when the arguments are finite ordinals, thus produce a finite cardinal when the arguments are finite cardinals. It is hence possible to define addition on finite cardinals without defining the set of integers.

5.1 Definition of integers

Recall that x is *finite* if it is an ordinal not equipotent to its successor x^+. This is equivalent to $x \in \omega$, and implies that x is a cardinal. A finite cardinal will be called an *integer*. The set of integers will be denoted by **N**, or `Nat` in the Coq code, instead of ω. This is an infinite set (Theorem 1, [4, p. 184]). Proposition 1 ([4, p. 166]) says: a cardinal x is finite if and only if its successor is finite.

```
Definition Nat := omega0.
Definition natp x := inc x Nat.

Lemma infinite_Nat: infinite_set Nat.
Lemma NatP a: natp a <-> finite_c a.

Theorem finite_succP x: cardinalp x ->
  (finite_c (csucc x) <-> finite_c x).
```

The following lemmas are trivial. Note that, if a is an integer, $x \in a$ says that x is also an integer, so x and a are cardinals, $x \in a$ is equivalant to $x <_o a$ thus to $x <_c a$. Thus $x \in a^+$ is equivalent to $x \leq_c a$ (where a^+ is the ordinal or cardinal successor of a).

```
Lemma NS_succ x: natp x -> natp (csucc x).
Lemma NS_nsucc x: cardinalp x -> natp (csucc x) -> natp x.
Lemma CS_nat x: natp x -> cardinalp x.
Lemma card_nat n: natp n -> cardinal n = n.
Lemma finite_set_nat n: natp n -> finite_set n.
Lemma NsumOr x: natp x -> x +c \0c = x.
Lemma Nsum0l x: natp x  -> \0c +c x = x.
Lemma Nprod1l x: natp x -> \1c *c x = x.
Lemma Nprod1r x: natp x -> x *c \1c = x.
Lemma NS_le_nat a b: a <=c b -> natp b -> natp a.
Lemma NS_lt_nat a b: a <c b -> natp b -> natp a.
Lemma Nsucc_rw x: natp x ->  csucc x = x +c \1c.
Lemma succ_of_nat n: natp n -> csucc n = osucc n.
Lemma Nat_dichot x: cardinalp x -> natp x \/ infinite_c x.
Lemma Nat_le_infinite a b: natp a -> infinite_c b -> a <=c b.
Lemma OS_nat x: natp x -> ordinalp x.
Lemma NS_inc_nat a: natp a -> forall b, inc b a -> natp b.
Lemma Nmax_p1 x y: natp x -> natp y ->
  [/\ natp (cmax x y), x <=c (cmax x y) & y <=c (cmax x y)].
Lemma NltP a: natp a -> forall x, x <c a <-> inc x a.
Lemma NleP a: natp a -> forall x,
  (x <=c a <-> inc x (csucc a)).
Lemma Nsucc_i a: natp a -> inc a (csucc a).
Lemma nat_irreflexive n: natp n -> ~(inc n n).
```

5.2 Inequalities between integers

The successors of zero are one, two, three and four. We list some trivial properties. We show here that $2n = n + n$, hence $2 + 2 = 2.2 = 4$ and $2^4 = 4^2$.

```
Definition card_three := csucc card_two.
```

```
Definition card_four := csucc card_three.
Notation "\3c" := card_three.
Notation "\4c" := card_four.

Lemma NS0: natp \0c.
Lemma NS1: natp \1c.
Lemma NS2: natp \2c.
Lemma NS3: natp \3c.
Lemma NS4: natp \4c.

Lemma two_plus_two: \2c +c \2c = \4c.
Lemma two_times_two: \2c *c \2c = \4c.
Lemma power_2_4: \2c ^c \4c = \4c ^c \2c.

Lemma cardinal_tripleton x y z: x <> y -> x <> z -> y <> z ->
  cardinal (tripleton x y z) = \3c.
Lemma cardinal_ge3 E:  \3c <=c cardinal E ->
  exists a b c,
  [/\ inc a E, inc b E, inc c E & [/\  a <> b, a <> c & b <> c]].
Lemma cardinal_doubleton x y: cardinal (doubleton x y) <=c \2c.
```

Proposition 2 [4, p. 166] says that if $\mathfrak{a}$ is a cardinal and n an integer, if $\mathfrak{a} \le n$ then $\mathfrak{a}$ is an integer. If n is an integer and $n \neq 0$, then there is a unique integer m such that $n = m+1$. In this case $a < n$ is equivalent to $a \le m$.

We consider here the first statement. If $\mathfrak{a} = \mathfrak{a}+1$ then $(\mathfrak{a}+\mathfrak{b})+1 = \mathfrak{a}+\mathfrak{b}$ (by associativity and commutativity). Thus, if $\mathfrak{a}+\mathfrak{b}$ is finite so is $\mathfrak{a}$. Assume now $\mathfrak{b}$ non-zero, so that $\mathrm{Card}(\mathfrak{b}) = \mathfrak{c}+1$ for some $\mathfrak{c}$. Then $\mathfrak{a}\mathfrak{b} = \mathfrak{a}\mathfrak{c}+\mathfrak{a}$. Thus, if $\mathfrak{a}\mathfrak{b}$ is finite, so is $\mathfrak{a}$.

Now, if $\mathfrak{a} \le n$ there is $\mathfrak{b}$ such that $\mathfrak{a}+\mathfrak{b} = n$. (Later on, Bourbaki shows that this quantity $\mathfrak{b}$ is an integer, is unique and is called the difference). It follows that if n is finite, then $\mathfrak{a}$ is finite also. By symmetry, $\mathfrak{b}$ is also finite, so that the difference is finite; we also have $\mathfrak{b} \le n$. We have already seen that quantities less than an integer are integers (since if $a \le b$ and $b < \omega$ then $a < \omega$). We show here: if $b < a$, then $a - b$ is non-zero, since $a = b + (a - b)$.

```
Lemma Nat_in_sumr a b: cardinalp b -> natp (a +c b) -> natp b.
Lemma Nat_in_suml a b: cardinalp a -> natp (a +c b) -> natp a.
Lemma Nat_in_product a b: cardinalp a ->
  b <> \0c -> natp (a *c b) -> natp a.

Lemma cdiff_le1 a b: cardinalp a -> a -c b <=c a.
Lemma cdiff_nz a b:  b <c a -> (a -c b) <> \0c.
Lemma NS_diff a b: natp a -> natp (a -c b).
Lemma NS_nsucc x: cardinalp x -> natp (csucc x) -> natp x.
```

We consider now the *predecessor* of x. This is the ordinal predecessor x^-, with a different name, `cpred` instead of `opred`. We know $x^- = x$ when x is zero or an infinite cardinal.

same holds when x is zero. Assume now that x is a non-zero finite cardinal, so that $x = y^+$ for some ordinal y and $x^- = y$. Clearly, y is finite, so that y^+ is also the cardinal successor of y.

We state: if x is a cardinal (resp. an integer) so is x^-, and the predecessor of the successor of x is x.

```
Lemma cpred0: cpred \0c = \0c.
```

```
Lemma cpred1: cpred \1c = \0c.
Lemma cpred2: cpred \2c = \1c.
Lemma cpred_inf a: infinite_c a -> cpred a = a.
Lemma cpred_pr n: natp n -> n <> \0c ->
  (natp (cpred n) /\  n = csucc (cpred n)).
Lemma NS_pred a: natp a -> natp (cpred a).
Lemma CS_pred a: cardinalp a -> cardinalp (cpred a).

Lemma cpred_pr1 n: cardinalp n -> cpred (csucc n) = n.
Lemma cpred_pr2 n: natp n -> cpred (csucc n) = n.
Lemma cpred_pr3 n: natp n ->
  n = \0c \/ exists2 m, natp m & n = csucc m.
Lemma cpred_nz n: cardinalp n ->  cpred n <> \0c -> \0c <c cpred n.
```

Let's show

$$n \in \mathbf{N} \implies (a \le n \iff a < n^+) \tag{5.1}$$

$$(a \le b \iff a^+ \le b^+) \tag{5.2}$$

$$a \in \mathbf{N} \vee b \in \mathbf{N} \implies (a^+ \le b \iff a < b) \tag{5.3}$$

$$(a < b \iff a^+ < b^+) \tag{5.4}$$

The first implication of (5.1) follows from $n < n^+$. Assume $a < n^+$. There is c such that $a + c = n + 1$. This cardinal is non-zero, so that $c = d + 1$ for some d. Thus $a + d + 1 = n + 1$. We get $a + d = n$, thus $a \le n$. [Alternate proof: $a \le n$ and $a < n^+$ are equivalent to $a \in n^+$.] Consider now (5.2). If $a \le b$, we know that $a + 1 \le b + 1$. Assume $a^+ \le b^+$, and let's show $a \le b$; the result is easy if b is infinite (as $b = b^+$, and a is finite or not). If b is finite, then $a \le a^{\le} b^+$; we cannot have equality (since otherwise a is finite and $a^+ \le a$). So $a < b^+$, and the result follows from (5.1).

Consider now (5.3). In case $b = 0$, the result is trivial. In the case where b is infinite and a finite, the result is clear. So assume b finite, $b = c^+$. Now $a < c^+$ is equivalent to $a \le c$ and to $a^+ \le c^+$. [Alternate proof when b is finite: we assume that a is a cardinal. Then $a^+ \le b$ says that a^+ is finite, so that a is finite; note that $a < b$ also implies a finite. The property becomes $a^+ \in b^= \iff a \in b$, where a^+ means cardinal successor, but can be replaced by ordinal successor, so it becomes $a^+ <_o b^= \iff a <_o b$.] The proof of (5.4) is similar. We distinguish between the finite and infinite case.

```
Lemma cleS0 a: cardinalp a -> a <=c (csucc a).
Lemma cleS a: natp a -> a <=c (csucc a).
Lemma cltS a: natp a -> a <c (csucc a).
Lemma cle_24: \2c <=c \4c.
Lemma clt_24: \2c <c \4c.
Lemma cpred_le a: cardinalp a -> cpred a <=c a.
Theorem cltSleP n: natp n -> forall a,
  (a <c (csucc n) <-> a <=c n).
Lemma cleSSP a b: cardinalp a -> cardinalp b ->
  (csucc a <=c csucc b <-> a <=c b).
Lemma ordinal_incSS a b: ordinalp b ->
  (inc a b <-> inc (osucc a) (osucc b)).
Lemma cleSlt0P a b: cardinalp a -> natp b ->
  (csucc a <=c b <-> a <c b).
Lemma cleSltP a: natp a -> forall b,
  (csucc a <=c b <-> a <c b).
Lemma cltSSP n m: cardinalp n -> cardinalp m ->
```

```
  ((csucc n <c csucc m) <-> (n <c m)).
Lemma cleSS a b : a <=c b -> csucc a <=c csucc b.
Lemma cltSS a b : a <c b -> csucc a <c csucc b.
Lemma cpred_lt n: natp n -> n <> \0c -> cpred n <c n.
Lemma cpred_pr6 k i: natp k -> \1c <=c i -> i <=c csucc k ->
  [/\ natp (cpred i), i = csucc (cpred i) & cpred i <=c k].
```

We also state: if E is a subset of **N**, its intersection is the least element of E (if E is empty, the intersection is not in E, but is nevertheless an integer). If p is a property, E the set of integers satisfying the property, we haqve a variant: yje least ordinal satisfying yje property is in fact the least integer. xs

```
Lemma inf_Nat E (x:= intersection E): sub E Nat -> nonempty E ->
   inc x E /\ (forall y, inc y E -> x <=c y).
Lemma NS_inf_Nat E (x:= intersection E): sub E Nat -> natp x.
Lemma least_int_prop n (p: property) (y:= least_ordinal p n):
   natp  n -> p n ->
   [/\ natp y, p y& forall z, natp z -> p z -> y <=c z].
```

If $X \subset Y$, $X \neq Y$ and Y is finite, then $\mathrm{card}(X) < \mathrm{card}(Y)$. Bourbaki says that the converse is true by definition. In fact, if Y is empty, it is finite, otherwise there is $x \in Y$, and if X is the complement of $\{x\}$ in Y, then $\mathrm{card}(Y) = \mathrm{card}(X) + 1$. The relation $\mathrm{card}(X) < \mathrm{card}(Y)$ implies that $\mathrm{card}(X)$ is finite, hence $\mathrm{card}(X) + 1$ is also finite.

```
Lemma card_finite_setP x: finite_set x <-> natp (cardinal x).
Lemma emptyset_finite: finite_set (emptyset).
Lemma strict_sub_smaller x y: ssub x y -> finite_set y ->
   (cardinal x) <c (cardinal y).
Lemma strict_sub_smaller1 y:
  (forall x, ssub x y -> (cardinal x) <c (cardinal y)) <->
  finite_set y.
```

The image of a finite set by a function is finite. We give some variants of this property. Consider two sets E and F with the same cardinal, and a function $f : E \to F$. Assume E finite. If f is injective and not surjective, then $f(E)$ is a strict subset of F equipotent to F, thus cannot be finite. Hence we claim: if f injective, then f is bijective. Assume f surjective. It has a right inverse, which is injective, so that f is also bijective.

```
Lemma finite_image f: function f -> finite_set (source f) ->
  finite_set (Imf f).
Lemma finite_image_by f A: function f ->
  finite_set A -> finite_set (Vfs f A).
Lemma finite_fun_image a f: finite_set a ->
  finite_set  (fun_image a f).
Lemma equipotent_domain f: fgraph f -> domain f \Eq f.
Lemma finite_graph_domain f: fgraph f ->
  (finite_set f <-> finite_set (domain f)).
Lemma finite_range f: fgraph f -> finite_set(domain f) ->
  finite_set(range f).

Lemma bijective_if_same_finite_c_inj f:
  cardinal (source f) = cardinal (target f) -> finite_set (source f) ->
  injection f -> bijection f.
```

```
Lemma bijective_if_same_finite_c_surj f:
  cardinal (source f) = cardinal (target f) -> finite_set (source f) ->
  surjection f -> bijection f.
```

5.3 The set of natural integers

As explained as the start of the chapter, we introduced the set of integers as early as possible (Bourbaki introduced it very lately). This short section contain only one simple result, namely that **N** can be well-ordered by the relation "$x \in \mathbf{N}$ and $y \in \mathbf{N}$ and $x \leq_{\mathrm{Card}} y$" denoted $x \leq_{\mathbf{N}} y$. Thus every non-empty subset of **N** has a least element. Every property p satisfied by some $n \in \mathbf{N}$ is satisfied by a least n. In fact, we say: either $p(0)$ holds, or there is n such that $p(n)$ is false but $p(n+1)$ is true. If we replace p by its opposite we get: assume p is true for zero, false for some integer. then there is n such that $p(n)$ is true, and $p(n+1)$ is false.

```
Definition Nat_order := graph_on cardinal_le Nat.
Definition Nat_le x y := [/\ natp x, natp y & x <=c y].
Definition Nat_lt x y := Nat_le x y /\ x<>y.
Notation "x <=N y" := (Nat_le x y) (at level 60).
Notation "x <N y" := (Nat_lt x y) (at level 60).

Lemma Nat_order_wor: worder_on Nat_order Nat.
Lemma Nat_order_leP x y:
  gle Nat_order x y <-> x <=N y.

Lemma Nat_wordered X: sub X Nat -> nonempty X ->
  inc \0c X \/
  (exists a, [/\ natp a, inc (csucc a) X & ~ (inc a X)]).

Lemma sub_natI_prop X (x := intersection X): sub X Nat ->
  [/\ natp x, forall t, t <c x -> ~ (inc t X) &
      nonempty X -> inc x X].

Lemma wleast_int_prop (prop:property):
  (exists2 x, natp x & prop x) ->
  prop \0c \/ (exists x, [/\ natp x, prop (csucc x) & ~ prop x]).
```

We consider some properties of intervals (there will be more of them in the next chapter).

```
Section NatIinterval.
Variables (a b: Set).
Hypotheses (aN: inc a Nat) (bN: inc b Nat).

Lemma Nint_ccP x:
  (inc x (interval_cc Nat_order a b) <-> (a <=N x /\ x <=N b)).
Lemma Nint_coP c:
  (inc x (interval_co Nat_order a b) <-> (a <=N x /\ x <N b)).
Lemma Nint_ccP1 x:
  (inc x (interval_cc Nat_order a b) <-> (a <=c x /\ x <=c b)).
Lemma Nint_coP1 x:
  (inc x (interval_co Nat_order a b) <-> (a <=c x /\ x <c b)).
End NatIinterval.
```

5.4 The principle of induction

Bourbaki states the principle of induction, the Criterion C61, in the following form: *Let* $R\{n\}$ *be a relation in a theory* $\mathcal{T}$ *(where n is not a constant of* $\mathcal{T}$*). Suppose that the relation*

$$R\{0\} \text{ and } (\forall n)((n \text{ is an integer and } R\{n\}) \implies R\{n+1\})$$

is a theorem in $\mathcal{T}$*. Under these conditions, the relation*

$$(\forall n)((n \text{ is an integer }) \implies R\{n\})$$

is a theorem in $\mathcal{T}$*.*

The proof is by contradiction. Assume the result false for some n, and consider the least element m of the set of all integers $\le n$ that do not satisfy R, (it exists, since the set is non-empty and is well-ordered). Our proof is similar (with "the set of all integers $\le n$ that do not satisfy R" replaced by "the set of integers that do not satisfy R").

We give four variants of the principle. Let $S(n)$ be the relation: "n is an integer and $R(p)$ is true for all integers $p < n$." If $S(n)$ implies $R(n)$, then R is true for all integers.

If $R(a)$ is true, and if $R(n)$ together with $a \le n$ (respectively $a \le n < b$) implies $R(n+1)$, then for all n such that $a \le n$ (respectively $a \le n \le b$), the relation $R(n)$ is true. In both cases n must be an integer (in the second case, we assume b integer, so that $n < b$ or $n \le b$ implies that n is an integer). Bourbaki uses induction on $a \le n < b \implies R(n)$, but it is simpler to use $a \le n \le b \implies R(n)$.

The last variant is: if $a \le n < b$ and $R(n+1)$ implies $R(n)$, if moreover $R(b)$ is true, then $a \le n \le b$ implies $R(n)$. The proof is by induction on $P = \neg R$. If R is false for some n with $a \le n \le b$ then P is true for some c with $a \le c < b$. On the other hand, if $a \le n < b$ then $P(n)$ implies $P(n+1)$.

```
Lemma Nat_induction (r:property):
  (r \0c) -> (forall n, natp n -> r n -> r (csucc n)) ->
  (forall n, natp n -> r n).
Lemma Nat_induction1 (r:property):
  let s:= fun n => forall p, p <c n -> r p in
  (forall n, natp n -> s n -> r n) ->
  (forall n, natp n -> r n).
Lemma Nat_induction2 (r:property) k:
  natp k -> r k ->
  (forall n, natp n -> k <=c n -> r n -> r (csucc n)) ->
  (forall n, natp n -> k <=c n -> r n).
Lemma Nat_induction3 (r:property) a b:
  natp a -> natp b -> r a ->
  (forall n, a <=c n -> n <c b -> r n -> r (csucc n)) ->
  (forall n, a <=c n -> n <=c b -> r n).
Lemma Nat_induction4: (r:property) a b:
  natp a -> natp b ->  r b ->
  (forall n, a <=c n -> n <c b -> r (csucc n) -> r n) ->
  (forall n, a <=c n -> n <=c b -> r n).
Lemma Nat_induction6 (P: property):
   P \0c ->
   (forall n, natp n -> (forall k, k<=c n -> P k) -> P (csucc n)) ->
   (forall n, natp n -> P n).
```

We rewrite our induction principle variant 3, replacing $a \le n \le b$ by $n \in [a, b]$.

```
Lemma Nat_induction3_v (r:property) a b:
  natp a -> natp b ->  r a ->
  (forall n, inc n (interval_co Nat_order a b) -> r n -> r (csucc n)) ->
  (forall n, inc n (interval_cc Nat_order a b) -> r n).
```

The empty set is finite, and if X is finite then $X \cup \{x\}$ is finite (in particular, a singleton, or a doubleton, are finite). We then show a partial converse, that will be useful for induction on finite sets. If X has cardinal zero, it is the empty set, and if X has cardinal $n+1$, it is of the form $X' \cup \{x\}$, where X' has cardinal n.

```
Lemma setU1_finite X x:
  finite_set X -> finite_set( X+ s1 x).
Lemma set1_finite x: finite_set(singleton x).
Lemma set2_finite x y: finite_set(doubleton x y).
Lemma finite_set_scdo:  finite_set (substrate canonical_doubleton_order).
Lemma setU1_succ_card x n: cardinalp n -> cardinal x = csucc n ->
  exists u v, [/\ x = u +s1 v, ~(inc v u) & cardinal u = n].
```

The induction principle on finite sets is now: If a property P is true for the empty set, if $P(a)$ implies $P(a \cup \{b\})$, then P is true for every finite set. In general P has the form: if A then B. Note: if $b \in a$, then $a \cup \{b\} = a$, and we have a version where we add the condition $b \not\in a$.

```
Lemma finite_set_induction0 (s:property):
  s emptyset -> (forall a b, s a ->  ~(inc b a) -> s (a +s1 b)) ->
  forall x, finite_set x -> s x.
Lemma finite_set_induction (s:property):
  s emptyset -> (forall a b, s a -> s (a +s1 b)) ->
  forall x, finite_set x -> s x.
Lemma finite_set_induction1 (A B:property):
  (A emptyset -> B emptyset) ->
  (forall a b, (A a -> B a) -> A(a +s1 b) -> B(a +s1 b)) ->
  forall x, finite_set x -> A x -> B x.
```

In some cases P is false for the empty set. If P is true for all singletons, then P is true for every non-empty finite set.

```
Lemma finite_set_induction2 (A B:property):
  (forall a, A (singleton a) -> B (singleton a)) ->
  (forall a b, (nonempty a -> A a -> B a) ->
   nonempty a -> A(a +s1 b) -> B(a +s1 b)) ->
  forall x, finite_set x -> nonempty x -> A x -> B x.
```

If $s(x)$ is the successor of x, and b is a cardinal, we have $a + s(b) = s(a+b)$ and $a \cdot s(b) = a \cdot b + a$, $a^{s(b)} = a^b.a$. We deduce by induction that **N** is stable by addition, multiplication and power.

```
Lemma csum_via_succ a b: cardinalp b -> a +c (csucc b) = csucc (a +c b).
Lemma cprod_via_sum a b: cardinalp b -> a *c (csucc b) = (a *c b) +c a.

Lemma csum_nS a b: natp b ->
  a +c (csucc b) = scucc (a +c b).
Lemma csum_Sn a b: natp a ->
   (csucc a) +c b = csucc (a +c b).
```

```
Lemma cprod_nS a b: natp b ->
  a *c (csucc b) = (a *c b) +c a.
Lemma cprod_Sn m n: natp m -> (csucc m) *c n = n +c (m *c n).
Lemma cpow_succ' a b: cardinalp b ->
  a ^c (csucc b) = (a ^c b) *c a.
Lemma cpow_succ a b: natp b ->
  a ^c (csucc b) = (a ^c b) *c a.

Lemma NS_sum a b: natp a -> natp b -> natp (a +c b).
Lemma NS_prod a b: natp a -> natp b -> natp (a *c b).
Lemma NS_pow a b: natp a -> natp b -> natp (a ^c b).
Lemma NS_pow2 n: natp n -> natp (\2c ^c n).
Lemma setU2_finite x y:
  finite_set x -> finite_set y -> finite_set (x \cup y).
Lemma finite_prod2 u v:
   finite_set u -> finite_set v -> finite_set (u \times v).
```

We state now some properties of subtraction: $x-0=x$, and $a-b=0$ when $a \leq b$ (these are obvious). We have $a+b=b$ when a is finite and b infinite (by induction on a); thus $b-a=b$ when b is infinite and a is finite (note that the difference cannot be finite). Next, if a and b are ordinals, $a<b$ then $a^+ - b^+$ and $a-b$ are equipotent (here the minus sign is set difference; if E is the set of ordinals x such that $b<x<a$, the first set is $\mathrm{E} \cup \{a\}$; the seconfd second is $\mathrm{E} \cup \{b\}$). One deduces $a^+ - b^+ = a-b$ (here the minus sign is cardinal difference; the relation holds whether or not a and b are finite or infinite. In the finite case, a^+ is also the ordinal successor). Finally, by induction $(a+b)-b=a$.

```
Lemma cdiff_n0 a: natp a -> a -c \0c = a.
Lemma cdiff_wrong a b: a <=c b -> a -c b = \0c.
Lemma csum_fin_infin a b: finite_c a -> infinite_c b -> a +c b = b.
Lemma cdiff_fin_infin a b: finite_c a -> infinite_c b ->  b -c a = b.
Lemma cdiff_succ_aux a b:
  b <o a -> cardinal ((a +s1 a) -s (b +s1 b)) = cardinal (a -s b).
Lemma cdiff_succ a b:
  cardinalp a -> cardinalp b -> (csucc a) -c (csucc b) = a -c b.
Lemma cdiff_pr1' a b: cardinalp a -> natp b -> (a +c b) -c b  = a.
Lemma cdiff_pr1 a b: natp a -> natp b -> (a +c b) -c b  = a.
```

5.5 Finite subsets of ordered sets

Let $\leq$ be an order relation on a set E that makes it a directed set, a lattice, or a totally ordered set; and let X be a finite non-empty subset of E. Then X has an upper bound, or has a least upper bound and a greatest lower bound, or has a least and greatest element respectively (Proposition 3, [4, p. 170]). We have to show that there is an x such that $\mathrm{P}(x,\mathrm{X})$. By assumption, this is true if X is a doubleton (therefore, if X is a singleton). If $\mathrm{X} = \mathrm{Y} \cup \{b\}$ and $\mathrm{P}(a,\mathrm{X})$ we have to show the property for the doubleton $\{a,b\}$.

```
Lemma finite_set_induction3 (p:Set -> Set -> Prop) E:
  (forall a b, inc a E -> inc b E -> exists y, p (doubleton a b) y) ->
  (forall a b x y, sub a E -> inc b E -> p a x -> p (doubleton x b) y->
    p (a +s1 b) y) ->
  (forall X x, sub X E -> nonempty X -> p X x -> inc x E) ->
  forall X,  finite_set X -> nonempty X -> sub X E -> exists x, p X x.
```

```
Lemma finite_subset_directed_bounded r X:
  right_directed r -> finite_set X -> nonempty X -> sub X (substrate r) ->
  bounded_above r X.
Lemma finite_subset_lattice_inf r X:
  lattice r -> finite_set X -> nonempty X ->  sub X (substrate r) ->
  exists x, greatest_lower_bound r X x.
Lemma finite_subset_lattice_sup r X:
  lattice r -> finite_set X -> nonempty X ->  sub X (substrate r) ->
  exists x, least_upper_bound r X x.
Lemma finite_subset_torder_greatest r X:
  total_order r -> finite_set X -> nonempty X ->  sub X (substrate r) ->
  has_greatest (induced_order r X).
Lemma finite_subset_torder_least r X:
  total_order r - >finite_set X -> nonempty X ->  sub X (substrate r) ->
  has_least (induced_order r X).
```

Some consequences. A nonempty finite set[1] has a maximal element, and if totally ordered, has a greatest element. A finite totally ordered set is well-ordered. We consider the special case where the set is a subset of the integers.

```
Lemma finite_set_torder_greatest r:
  total_order r -> finite_set (substrate r) -> nonempty (substrate r)
  -> has_greatest r.
Lemma finite_set_torder_least r:
  total_order r -> finite_set (substrate r) -> nonempty (substrate r)
  -> has_least r.
Lemma finite_set_torder_wor r:
  total_order r ->finite_set (substrate r) -> worder r.
Lemma finite_set_maximal r:
  order r ->finite_set (substrate r) -> nonempty (substrate r) ->
  exists x, maximal r x.
Lemma finite_set_minimal r:
  order r -> finite_set (substrate r) -> nonempty (substrate r) ->
  exists x, minimal r x.
Lemma finite_subset_Nat X: sub X Nat -> finite_set X -> nonempty X ->
   exists2 n, inc n X & forall m, inc m X -> m <=c n.
Lemma Nat_sup_pr T (s:= \csup T) k:
  natp k -> (forall i, inc i T -> i <=c k) ->
  [/\ natp s, s <=c k,
    (forall i, inc i T -> i <=c s) &
    (T = emptyset \/ inc s T)].
```

In a lattice, any non-empty finite set has a sup and an inf. We state lemmas of the form $\sup(\sup(X), x) = \sup(X \cup \{x\})$ in the case where X has a supremum, and also if X is a finite non-empty set.

```
Section LatticeProps.

Variable (r: Set).
Hypothesis lr: lattice r.
Let E := substrate r.

Lemma lattice_finite_sup2 x:
  finite_set x -> nonempty x -> sub x E -> has_supremum r x.
```

[1]The word "nonempty" is missing in Bourbaki

```
Lemma lattice_finite_inf2 x:
  finite_set x -> nonempty x -> sub x E -> has_infimum r x.
Lemma lattice_finite_sup3P x y:
  finite_set x -> nonempty x -> sub x E ->
  (gle r (supremum r x) y <-> (forall z, inc z x -> gle r z y)).
Lemma lattice_finite_inf3P  x y:
  finite_set x -> nonempty x -> sub x E ->
  (gle r y (infimum r x) <-> (forall z, inc z x -> gle r y z)).
Lemma supremum_setU1 a b:
  sub a E -> has_supremum r a -> inc b E ->
  supremum r (a +s1 b) =  sup r (supremum r a) b.
Lemma infimum_setU1 a b:
  sub a E -> has_infimum r a -> inc b E ->
  infimum r (a +s1 b) =  inf r (infimum r a) b.
End LatticeProps.
```

5.6 Properties of finite character

If E is a set, a property P⦃X⦄ (where X is a subset of E) is said to be of *finite character* if the set $\mathfrak{S}$ of all X satisfying P is of finite character; this means $X \in \mathfrak{S}$ if and only if every finite subset Y of X satisfies $Y \in \mathfrak{S}$. Example: the set of totally ordered subsets of an ordered set. Theorem 1 [4, p. 171] states: Every nonempty[2] set $\mathfrak{S}$ of subsets of a set E which is of finite character has a maximal element (when ordered by inclusion).

```
Definition finite_character s:=
  forall x, (inc x s) <-> (forall y, (sub y x /\ finite_set y) -> inc y s).

Lemma finite_character_example r: order r ->
  finite_character(Zo (\Po (substrate r)) (fun z =>
    total_order (induced_order r z))).
Lemma maximal_inclusion s: finite_character s -> nonempty s ->
  exists x, maximal (sub_order s) x.
Lemma maximal_inclusion_aux: let s := emptyset in
  finite_character s /\
  ~ (exists x, maximal (sub_order s) x).
```

Study of limit and predecessor of orderings. On page 75 we introduced the notion of a limit ordinal or ordinal predecessor. We explained that these notions could be extended to any well-ordering. Here is the code.

```
Lemma succ_study (r: relation) x
  (r' := fun a b => r a b /\ a <> b)
  (Ap := fun y A => forall t, inc t A <->  [/\ r x t, r t y & t <> x])
  (Ax:= fun y => exists A, Ap y A)
  (Ay := fun y => choose (Ap y))
  (ly := fun y => the_least (graph_on r (Ay y))):
  worder_r r ->
  (forall y, r' x y -> Ax y) ->
  [/\ (r x x -> ~ (exists y, r' x y) -> forall t, r t t -> r t x),
      (forall y, r' x y -> r' x (ly y)) ,
      (forall y, r' x y -> forall t, r' x t -> r (ly y) t) &
```

[2] The word “nonempty” is missing in Bourbaki

```
     (forall y y', r' x y -> r' x y' ->  ly y = ly y')].

Lemma limit_study (r: relation) y B
  (C := B +s1 y)
  (R := (graph_on r C))
  (x := supremum R B):
  (forall a, inc a B <-> r a y /\ a <> y) ->
  worder_r r -> r y y ->
     [/\ (least_upper_bound R B x /\ r x y),
         (B = emptyset -> forall t, r t t -> r y t),
         ( x <> y ->
          [/\ inc x B, x = the_greatest (graph_on r B),
           (forall z, r x z -> r z y -> z = x \/ z = y) &
           (forall z, r z z -> r z x \/ r y z)]) &
         (x = y -> forall B1, nonempty B1 -> finite_set B1 -> sub B1 B ->
           inc (the_greatest (graph_on r B1)) B)].
```

Isomorphism $\mathbb{N} \to \mathbf{N}$. Define by induction on the type `nat` a function $n \to \bar{n}$ by $\bar{0} = 0$ and $\overline{n+1} = \bar{n}+1$ (the first $n+1$ is the successor on `nat`, the second is the cardinal successor. Then $\bar{n}$ is a natural number, and each natural number can be uniquely written in the form $\bar{n}$. This function respects operations (addition, multiplication, exponentiation). It is also compatible with subtraction and ordering.

```
Fixpoint nat_to_B (n:nat) :=
  if n is m.+1 then csucc (nat_to_B m) else  \0c.

Lemma nat_to_B_succ n:
  csucc (nat_to_B n) = (nat_to_B n.+1).
Lemma nat_to_B_Nat n:natp (nat_to_B n).
Lemma nat_to_B_injective: injective nat_to_B.
Lemma nat_to_B_surjective x: natp x -> exists n, x = nat_to_B n.
Lemma nat_to_B_sum x y: nat_to_B (x + y) = nat_to_B x +c nat_to_B y.
Lemma nat_to_B_prod x y: nat_to_B (x * y) = nat_to_B x *c nat_to_B y.
Lemma nat_to_B_pow x y: nat_to_B (x ^ y) = nat_to_B x ^c nat_to_B y.
Lemma nat_to_B_le x y: x <=y <-> nat_to_B x <=c nat_to_B y.
Lemma nat_to_B_lt x y: x < y <-> nat_to_B x <c nat_to_B y.
Lemma nat_to_B_diff x y: nat_to_B (x - y) = nat_to_B x -c nat_to_B y.
Lemma nat_to_B_max a b: nat_to_B (maxn a b) = cmax (nat_to_B a) (nat_to_B b).
Lemma nat_to_B_min a b: nat_to_B (minn a b) = cmin (nat_to_B a) (nat_to_B b).
Lemma nat_to_B_pos n: 0 <n <-> \0c <c nat_to_B n.
Lemma nat_to_B_gt1 n: 1 <n <-> \1c <c nat_to_B n.
Lemma nat_to_B_ifeq a b u v: let N := nat_to_B in
case: ifP => ab; first by rewrite(eqP ab) /=(Y_true (erefl _)).
```

Let's say that X is z-infinite when $\emptyset \in \mathrm{X}$ and $x \in \mathrm{X}$ implies $x^+ \in \mathrm{X}$. The image of `nat_to_B` is z-infinite; **N** is z-infinite. The intersection of a family of z-infinite sets is z-infinite. **N** is a subset every z-infinite set. So: if X is z-infinite, then the intersection of all z-infinite subsets of X is **N**.

```
Definition  z_infinite X :=
  inc emptyset X /\ forall x, inc x X -> inc (osucc x) X.

Lemma z_infinite_nat: z_infinite (IM nat_to_B).
Lemma z_infinite_nat2: z_infinite Nat.
```

```
Lemma z_infinite_omega X: z_infinite X -> sub Nat X.
Lemma z_infinite_I X:
  (forall t, inc t X -> z_infinite t) -> nonempty X ->
  z_infinite (intersection X).
Lemma z_infinite_I2 X (Y := Zo (\Po X) z_infinite) :
  z_infinite X -> (intersection Y) = Nat.
```

Infinite squares. Let's show the following property: if x is an infinite cardinal, then $x.x = x$. We shall prove later on that $X \times X$ is equipotent to X, whenever X is an infinite set. These two properties are equivalent if the Axiom of Choice holds; the proof given here does not depend on the axiom of choice.

Given a pair of ordinals $x = (a, b)$, we define $\bar{x}$ to be the greatest of a and b. Let $x' = (\bar{x}, a, b)$. Define $x \le^* y$ to be $x' \le_{\text{lex}} y'$, where $\le_{\text{lex}}$ compares the three components in lexicographic order. This is a well-ordering (the lexicographic product of three well-ordered sets is well-ordered). We call it the *canonical ordering of pairs of ordinals.* We give here a direct proof: Let X be a non-empty set of pairs of ordinals. The set of all $\bar{x}$ (for $x \in X$) has a least element γ; the set of all $\text{pr}_1 x$ with $\bar{x} = \gamma$, has a least element, say α; the set of all $\text{pr}_2 x$ with $\bar{x} = \gamma$ and $\text{pr}_1 x = \alpha$ has a least element, say β. Then (α, β) is the least element of X for $\le^*$.

Let $p(x)$ be the property $\text{card}(x \times x) = x$. We show that p holds for any infinite cardinal by transfinite induction: we consider an infinite cardinal κ, assume $p(x)$ true for every infinite cardinal x such that $x < \kappa$, and deduce $p(\kappa)$. Note that $x \le \text{card}(x \times x)$ is obvious.

Since $\le^*$ is a well-ordering, it induces a well-ordering on $\kappa \times \kappa$, and we can consider its ordinal λ. This means that we have a bijection $f : \kappa \times \kappa \to \lambda$, such that $x \le^* y$ if and only if $f(x) \le_{\text{ord}} f(y)$. Notice that, if $y \in \kappa$, $f(x) \in f(y)$ is equivalent to $f(x) <_{\text{ord}} f(y)$, thus $x <^* y$. Let z be the greatest of the two components of y. Then $x <^* y$ implies that the two components of x are $\le_{\text{ord}} z$, thus $<_{\text{ord}} z^+$, so that $x \in z^+ \times z^+$. This gives a bound on the cardinal of $f(y)$: let t be the cardinal of z^+; we have $\text{card}(f(y)) \le t.t$. This is obviously $< \kappa$ if z is finite. On the other hand, κ is a limit ordinal, so that $z <_{\text{ord}} \kappa$ implies $z^+ <_{\text{ord}} \kappa$, so that $t < \kappa$, and $t^2 = t$. It follows: $\text{card}(f(y)) < \kappa$. Since $f(y)$ is an ordinal, it follows $f(y) <_{\text{ord}} \kappa$. Thus $f(y) \in \kappa$; it follows $\lambda \subset \kappa$, and $\text{card}(\lambda) \le \kappa$. Since λ is equipotent to $\kappa \times \kappa$, the conclusion follows.

As a byproduct, we have: for any infinite cardinal X, there is a bijection $f : X \times X \to X$ so such that $x \le^* y$ is equivalent to $f(x) \le f(y)$.

```
Definition ordinal_pair x :=
  [/\ pairp x, ordinalp (P x) & ordinalp (Q x)].
Definition ord_pair_le  x y:=
   [/\ ordinal_pair x, ordinal_pair y &
   ((P x) \cup (Q x) <o (P y) \cup (Q y)
    \/ ((P x) \cup (Q x) = (P y) \cup (Q y)
        /\ ((P x) <o (P y)
           \/ (P x = P y /\ Q x <=o Q y))))].

Lemma ordering_pair1 x: ordinal_pair x ->
   ((P x <=o Q x) /\  ((P x) \cup (Q x) = Q x))
   \/ ((Q x <=o P x) /\  ((P x) \cup (Q x) = P x)).
Lemma ordering_pair2 x: ordinal_pair x -> ordinalp  ((P x) \cup (Q x)).
Lemma ordering_pair3 x y : ord_pair_le x y ->
  inc x (coarse (osucc (((P y) \cup (Q y))))).
Lemma ordering_pair4 x: ordinal_pair x -> ord_pair_le x x.

Lemma well_ordering_pair: worder_r ord_pair_le. (* 80 *)
```

```
Lemma infinite_product_aux k: infinite_c k ->  (* 80 *)
  (forall z,  infinite_c z -> z <o k -> z *c z = z) ->
  let lo:= graph_on ord_pair_le (coarse k) in
  k *c k = k /\
  exists2 f, bijection_prop f (coarse k) k &
     (forall x y, inc x (source f) -> inc y (source f) ->
       (glt lo x y <-> (Vf x f) <o (Vf y f))).
Lemma infinite_product_alt x : infinite_c x -> x *c x = x.
Lemma infinite_product_prop2 k: infinite_c k ->
  let lo:= graph_on ord_pair_le (coarse k) in
  exists2 f, bijection_prop f (coarse k) k &
     (forall x y, inc x (source f) -> inc y (source f) ->
       (glt lo x y <-> (Vf f x) <o (Vf f y))).
```

Chapter 6

Properties of integers

This chapter studies some properties of integers; for instance, from $a+b = a'+b$ or $a+b \le a'+b$ one deduces $a = a'$ or $a \le a'$; moreover $a < b$ is equivalent to $a+1 \le b$ (these properties are false when some arguments are infinite). One can perform Euclidean division, from which expansion to base b can be deduced; this means that every number n can uniquely be written as $\sum c_i b^i$, when the base b is at least two, the coefficients satisfy $c_i < b$, the index i belongs to an interval $[0,k[$ and k is non-zero, then c_{k-1} is non-zero. It is easy to compare integers given their expansion (but formalizing this is not trivial). We prove that, when $b = 10$, then n is equal to $\sum c_i$ modulo 3 or modulo 9. (we prove some properties of the modulo function, although it is not part of Bourbaki's theory of sets; we also study even and odd integers, the base two logarithm, and the Fibonacci sequence). The remainder of the chapter studies combinatorial analysis: after introducing the factor and binomial functions, we can answer questions of the type: given two totally ordered finite sets E and F, what is the number of increasing mappings (or injections, surjections, bijections) $\mathrm{E} \to \mathrm{F}$?

6.1 Operations on integers and finite sets

By *operation* on a set E, one means a function $g : \mathrm{E} \times \mathrm{E} \to \mathrm{E}$, often denoted by an infix symbol such as $a+b$. There are some unary operations $\mathrm{E} \to \mathrm{E}$ such as x^+, the successor of x. The sum of two cardinals may be considered as an operation (but there is no set of cardinals). Binary operations may be generalized to more than two arguments. Given a list x_1, $x_2, \dots, x_n$ of $n \ge 2$ terms, one can define a function F as follows

$$\mathrm{F}(x_1, x_2, \dots, x_n) = g(x_1, \mathrm{F}(x_2, \dots, x_n)), \tag{6.1}$$

and $\mathrm{F}(a) = a$, so that $\mathrm{F}(x_1, x_2) = g(x_1, x_2)$. Note that if g maps $\mathrm{E} \times \mathrm{G}$ to G, and g_0 is some function $\mathrm{E} \to \mathrm{G}$, we can define $\mathrm{F}(x_1) = g_0(x_1)$, so that F maps a non-empty sequence of elements of E onto an element of G. We say that e is a unit of g if $g(e,x) = x$ whatever x. In this case, we may define $\mathrm{F}() = e$, $\mathrm{F}(x) = g(e,x)$, so that F is defined on L(E), the set of lists of E, otherwise, it is defined only on $\mathrm{L}_e(\mathrm{E})$, the set of non-empty lists of E. Here, we may identify a list with a function $[1,n] \to \mathrm{E}$, and a non-empty list corresponds to the case $n \ne 0$. If X is the function associated to the list $x_1, \dots, x_n$ and X' the function associated to $x_2, \dots, x_n$, we have $\mathrm{X}'(i) = \mathrm{X}(i+1)$, and (a) says $\mathrm{F}(\mathrm{X}) = g(\mathrm{X}(1), \mathrm{F}(\mathrm{X}'))$. Every finite totally ordered set is uniquely order-isomorphic to an interval $[1,n]$. Thus, given a mapping $x : \mathrm{I} \to \mathrm{E}$, where I is finite and totally ordered, we may consider x as a list; if l is the least element of I, and if x' is the restriction of x to $\mathrm{I} - \{l\}$, with the induced order, relation (6.1) states $\mathrm{F}(x) = g(x_l, \mathrm{F}(x'))$.

We say that g is associative if $g(a,g(b,c)) = g(g(a,b),c)$. This implies

$$F(x_1,x_2,\dots,x_n) = g(F(x_1,\dots,x_{n-1}),x_n) \tag{6.2}$$

and in particular that, if x is as above, if g is the greatest element of I, x'' is the restriction of x to $I-\{g\}$, with the induced order, we get $F(x) = g(F(x''),x_g)$. More generally, given any partition of the interval $[1,n]$ as $[1,n] = \bigcup X_k$, where each X_k is non-empty, if $k < l$ implies $u < v$ whenever $u \in X_k$ and $v \in X_l$, if x'_k denotes the list of x_i with $i \in X_k$ (with the induced ordering), and $x''_k = F(x_k)$ then $F(x) = F(x'')$.

We say that g is commutative if $g(a,b) = g(b,a)$. If g is commutative and associative we have for instance $F(a,b,c) = F(b,a,c)$. More generally, $F(x)$ becomes independent of the ordering of the elements of the list. The associativity theorem can be simplified: the condition "$k < l$ implies $u < v$ whenever $u \in X_k$ and $v \in X_l$" becomes unnecessary. If moreover e is a unit, the condition "X_k is non-empty" can be dropped as well.

In a previous version of our software, we introduced the notion of non-empty list, non-empty sequence, etc, so as to handle the case where there is no unit element (for instance, intersection of sets, infimum function on **N**, etc), see 14.5. The SSREFLECT library proposes a module *bigops*, that implements this kind of operations. In the example that follows, I is a finite type (a finite totally ordered set) and the operation is applied to all x_i where $i \in I$ satisfies a predicate P. There is a function $p : I \to J$ that defines a partition $X_k = p^{-1}(\{k\})$ of I. Let's compare the associativity theorem of bigops and the associativity of cardinals in Gaia:

```
(*
Lemma partition_big : forall (I J : finType) (P : pred I) p (Q : pred J) F,
    (forall i, P i -> Q (p i)) ->
      \big[*%M/1]_(i | P i) F i =
         \big[*%M/1]_(j | Q j) \big[*%M/1]_(i | P i && (p i == j)) F i.
Theorem csum_An f g:
  partition_w_fam g (domain f) ->
  csum f = csumb (domain g) (fun l => csumb (Vg g l) (Vg f)).
*)
```

Similarly, we may compare the commutativity theorems:

```
(*
Definition perm_eq (s1 s2 : seq T) := all (same_count1 s1 s2) (s1 ++ s2).
Lemma eq_big_perm : forall (I : eqType) r1 r2 (P : pred I) F,
  perm_eq r1 r2 ->
  \big[*%M/1]_(i <- r1 | P i) F i = \big[*%M/1]_(i <- r2 | P i) F i.

Theorem csum_Cn X f:
  target f = domain X -> bijection f ->
  csum X = csum (X \cf (graph f)).
*)
```

There is a fundamental difference between the SSREFLECT theory, and the Bourbaki theorems proved so far. In one case, we start with a binary operation and extent it to finite lists of arguments, and in the other case, we start with an operation defined for many arguments, and study the case of two arguments. Note that a finite sum of integers is finite, but an infinite sum of integers is not always an integer. One may define the sum and product of a finite sequence of ordinals; one can also define an infinite sum, but not an infinite product.

```
Lemma Nsum_M0le a b: natp a -> a <=c (a +c b).
```

```
Lemma Nprod_M1le a b: natp a -> b <> \0c -> a <=c (a *c b).
Lemma NleT_ell a b: natp a -> natp b ->
   [\/ a = b, a <c b | b<c a].
Lemma NleT_el a b: natp a -> natp b ->
  a <=c b \/ b <c a.
Lemma NleT_ee a b: natp a -> natp b ->
  a <=c b \/ b <=c a.
```

Induction formulas for sum and product. We show here

$$x_j + \sum_{i\in J} x_i = \sum_{i\in J\cup\{j\}} x_i, \tag{6.3}$$

$$x_j \cdot \prod_{i\in J} x_i = \prod_{i\in J\cup\{j\}} x_i, \tag{6.4}$$

whenever $j \notin J$. This is a trivial consequence of the associativity theorem in the case of a partition formed of two sets, one of them being a singleton. If the domain of the family (x_i) is $J \cup \{j\}$, then the right hand side of the first equation is also $\sum x_i$.

```
Lemma induction_sum0 f a b: (~ inc b a) ->
  csum (restr f (a +s1 b)) =
  csum (restr f a) +c (Vg f b).
Lemma induction_prod0 f a b: (~ inc b a) ->
  cprod (restr f (a +s1 b)) =
  (cprod (restr f a)) *c (Vg f b).
Lemma induction_sum1 f a b:
  domain f = a +s1 b -> (~ inc b a) ->
  csum f = csum (restr f a) +c  (Vg f b).
Lemma induction_prod1 f a b:
  domain f = a +s1 b -> (~ inc b a) ->
  cprod f = cprod (restr f a) *c (Vg f b).
Lemma csum_fs f n: natp n -> csumb (csucc n) f = csumb n f +c (f n).
Lemma csumb0 (f: fterm) : csumb \0c f = \0c.
Lemma csumb1 (f: fterm): cardinalp (f \0c) ->   csumb \1c f = f \0c.
```

Finite sums and products. A *finite family of integers* is a functional graph $i \mapsto x_i$ where the index set I is finite and each x_i is an integer.

```
Definition finite_int_fam f:=
  (allf f natp) /\ finite_set (domain f).
```

Proposition 1 [4, p. 171] says that if $(a_i)_{i\in I}$ is a finite family of integers, then $\sum_{i\in I} a_i$ and $\prod_{i\in I} a_i$ are integers. As a consequence, if $J \subset I$ then $\sum_{i\in J} a_i$ and $\prod_{i\in J} a_i$ are integers. The proof is by induction on the finite set J via formulas (6.3) and (6.4).

```
Section FiniteIntFam.
Variable f: Set.
Hypothesis fif: finite_int_fam f.

Lemma finite_sum_finite_aux x:
  sub x (domain f) -> natp (csum (restr f x)).
```

```
Lemma finite_product_finite_aux  x:
  sub x (domain f) -> natp (cprod (restr f x)).
Theorem finite_sum_finite: natp (csum f).
Theorem finite_product_finite: natp (cprod f).

End FiniteIntFam.
```

We have obvious consequences. For instance, a finite union of finite sets is finite. As explained above, this is easy by induction. Bourbaki says that, if E is the union and S is the sum, then S is finite, and, since there is a surjection from S onto E, we have card(E) ≤ S, so that card(E) is finite. However card(E) ≤ S has already been stated (as corollary to Proposition 4). A finite product of finite sets is a finite set (since the cardinal of the product is the product of the cardinals). Since a^b is a product, it is finite if a and b are finite. Thus, the power set of a finite set is finite (these results were proved in the previous chapter).

```
Lemma finite_union_finite f:
  (allf f finite_set) -> finite_set (domain f) -> finite_set(unionb f).
Lemma finite_product_finite_set f:
  (allf f finite_set) -> finite_set (domain f) -> finite_set(productb f).
```

6.2 Strict inequalities between integers

Proposition 2 [4, p. 173] says that $a < b$ if and only if there is c such that $0 < c$ and $b = c + a$ (a, b and c being integers). Assume $a < b$. We know that there exists a cardinal c such that $b = c + a$; obviously c is a non-zero integer. Conversely, since $a < a + 1$, and $1 \le c$, we get $a + 1 \le a + c$ and we conclude by transitivity.

```
Lemma strict_pos_P a: natp a -> (\0c <> a <-> \0c <c a).
Lemma strict_pos_P1 a: natp a -> (a <> \0c <-> \0c <c a).
Lemma card_ltP1 a b: natp b -> a <c b ->
  exists c, [/\ natp c, c <> \0c & a +c c = b].
Theorem card_ltP a b: natp a -> natp b ->
  (a <c b <->  exists c, [/\ natp c, c <> \0c & a +c c = b]).
```

We deduce

$$a \le a', b < b' \implies a + b < a' + b', \quad a \cdot b < a' \cdot b' \text{ (when } a' \neq 0).$$

(write $b' = b + c$, where $c > 0$, so that $a' + b' = (a' + b) + c$ and $a' \cdot b' = (a' \cdot b) + a' \cdot c$). Note; in `csum_Meqlt` only b needs to be an integer (the proof is by induction, and holds because $a < a'$ implies $a + 1 < a' + 1$). Note: $ab < ab'$ holds when b or b' are infinite, but we cannot prove it now; so we state it in the case b and b' finite. In the case $b = 1$, the proof is by induction: assume $1 < b'$ and $a < ab'$. Then $a + 1 < ab' + 1$ and $ab' + 1 \le ab' + b$, hence $a + 1 < (a + 1)b'$.

```
Lemma csum_M0lt a b: natp a -> b <> \0c -> a <c a +c b.
Lemma csum_Mlelt a b a' b': natp a' ->
  a <=c a' -> b <c b' -> (a +c b) <c (a' +c b').
Lemma csum_Mlteq a a' b:  natp b ->
  a <c a' -> (a +c b) <c (a'+c b).
Lemma csum_Meqlt a a' b:  natp b ->
  a <c a' -> (b +c a) <c (b +c a').
```

```
Lemma cprod_Meqlt a b b':
  natp a -> natp b'  -> b <c b' -> a <> \0c ->
  (a *c b) <c (a *c b').
Lemma cprod_M1elt a b a' b': natp a' ->
  a <=c a' -> b <c b' -> a' <> \0c ->
  (a *c b) <c (a' *c b').
Lemma cprod_M1lt a b: natp a ->
  a <> \0c -> \1c <c b -> a <c (a *c b).
```

Proposition 3 [4, p. 173] says that $\sum a_i < \sum b_i$ and $\prod a_i < \prod b_i$ for two families of integers with the same index set I, if $a_i \le b_i$ for each i and $a_j < b_j$ for some j. In the case of a product, $b_i > 0$ is required. Consider the partition $\mathrm{J} \cup \{j\}$ of I. We have $\sum_{i\in\mathrm{I}} a_i = \mathrm{A} + a_j$, where $\mathrm{A} = \sum_{i\in\mathrm{J}} a_i$. In the same way, $\sum_{i\in\mathrm{I}} b_i = \mathrm{B} + b_j$, and $\mathrm{A} \le \mathrm{B}$. Moreover A and B are integers, so that we can apply the previous formulas.

```
Theorem finite_sum_lt f g:
  finite_int_fam f -> finite_int_fam g -> domain f = domain g ->
  (forall i, inc i (domain f) -> (Vg f i) <=c  (Vg g i)) ->
  (exists2 i, inc i (domain f) &  (Vg f i) <c (Vg g i)) ->
  (csum f) <c (csum g).
Theorem finite_product_lt f g:
  finite_int_fam f -> finite_int_fam g -> domain f = domain g ->
  (forall i, inc i (domain f) -> (Vg f i) <=c (Vg g i)) ->
  (exists2 i, inc i (domain f) &  (Vg f i) <c (Vg g i)) ->
  card_nz_fam g ->
  (cprod f) <c (cprod g).
```

We have $a^b < a'^b$ if $a < a'$ and $b \ne 0$. We have $a^b < a^{b'}$ if $a > 1$ and $b < b'$ (the case $a = 0$ is special, if $a = 1$, both terms are 1). We have $a < a^b$ if $b \ge 2$.

```
Lemma cpow_nz a b: a <> \0c -> (a ^c b) <> \0c.
Lemma cpow2_nz x: \2c ^c x <> \0c.
Lemma cpow2_pos x: \0c <c \2c ^c x.
Lemma cpow_M1tle lt1 a a' b:
  natp a' -> natp b ->
  a <c a' -> b <> \0c ->  (a ^c b) <c (a' ^c b).
Lemma cpow_Meqlt a b b':
  natp a -> natp b' ->
  b <c b' -> \1c <c a ->  (a ^c b) <c (a ^c b').
Lemma cpow2_MeqltP n m: natp n -> natp m ->
    (\2c ^c n <c \2c ^c m <-> n <c m).
Lemma cpow_M1lt a b: natp a -> natp b ->
  \1c <c b ->  a <c (b ^c a).
```

Simplifications. If $a + b = a + b'$ or if $ab = ab'$ then $b = b'$ (all arguments are integers; $a \ne 0$ in the case of a product).

```
Section Simplifications.
Variables (a b b' :Set).
Hypotheses (aN: natp a) (bN: natp b) (b'N: natp b').

Lemma csum_eq2l:  a +c b = a +c b' -> b = b'.
Lemma csum_eq2r: b +c a = b' +c a -> b = b'.
Lemma cprod_eq2l: a <> \0c -> a *c b = a *c b' -> b = b'.
```

```
Lemma cprod_eq2r: a <> \0c -> b *c a = b' *c a -> b = b'.
End Simplifications.
```

Subtraction. Given two cardinals a and b such that $a \le b$, there exists a cardinal c such that $b = a + c$; it is called the *difference*, and denoted by $b - a$. The operation is called *subtraction*. If a and b are integers, then c is an integer[1] and is uniquely defined by this relation.

```
Lemma cdiff_wrong a b: (a <=c b) -> a -c b = \0c.
Lemma cdiff_rpr a b: b <=c a -> (a -c b) +c b = a.
```

We have $(a + b) - b = a$ for all integers.[2] Uniqueness means that if $a + b = c$ then $a = c - b$ and $b = c - a$.

```
Lemma cdiff_pr1 a b: natp a -> natp b ->
  (a +c b) -c b  = a.
Lemma cdiff_pr2 a b c: natp a -> natp b ->
  a +c b = c -> c -c b = a.
Lemma cdiff_pr3 a b n:
   natp n -> a <=c b -> b <=c n -> (n -c b) <=c (n -c a).
Lemma cdiff_pr7 a b c:
  a <=c b -> b <c c -> natp c -> (b -c a) <c (c -c a).
Lemma cdiff_pr8 n p q: q <=c p -> p <=c n -> natp n ->
  (n -c p) +c q = n -c (p -c q).
Lemma card_setC4 E A: sub A E ->
  finite_set E -> cardinal (E -s A) = (cardinal E) -c (cardinal A).
Lemma cardinal_setC5 A B: finite_set B -> sub A B -> A =c B -> A = B.
Lemma cdiffA2 a b c: natp a -> natp b ->
 c <=c a -> (a +c b) -c c  = (a -c c) +c b.
Lemma cdiffSn a b: natp a -> b <=c a ->
   (csucc a) -c b = csucc (a -c b).
```

For any cardinal a, the predecessor a^- of a is $a - 1$. Proof. If a is a non-zero finite ordinal, then $a = b^+$, where $b = a^-$. Now b is a finite ordinal, thus an integer, and $b^+ = b + 1$ so that $a - 1 = b^+ - 1 = (b + 1) - 1 = b = a^-$. If a is zero, then both $a - 1$ and a^- are zero. Assume now that a is an infinite cardinal. In particular, a is a limit ordinal, so that $a^- = a$. On the other hand $a - 1$ is the cardinal of $a' = a \setminus \{0\}$. Let $b = a \cup \{a\}$ and $b' = a' \cup \{0\}$. These sets are equipotent (since $b = a^+$, $b' = a$ and a is an infinite cardinal). Thus $b - \{a\}$ and $b' - \{0\}$ are equipotent (since $a \in b$ and $0 \in a'$). Thus a and a' are equipotent, qed.

We deduce: if b is a non-zero integer, then $a < b$ is equivalent to $a \le b - 1$.

We have $(b - a) + (b' - a') = (b + b') - (a + a')$ if the first two differences are defined. We have the trivial formulas: $a - a = 0$ and $a - 0 = a$.

Note that $a - 1$ is the predecessor a^- of a: if a is infinite, we know $a - 1 = a^- = a$; the same is true if $a = 0$; otherwise $a = a^- + 1$;

We have $(a - b) - c = a - (b + c)$ if $a \ge b + c$. Taking $c = 1$, and denoting by P and S the predecessor and successor, we have $\mathrm{P}(a - b) = a - \mathrm{S}b$. We have $a - b \le a$; and if $b < a$, then $(a - b) - 1 < a$. If $b + 1 \le a$ then $b < a$.

```
Lemma cdiff_nn a: a -c a = \0c.
```

[1] If $a \le b$ and b is an integer, then a is an integer.

[2] Note: assume one argument infinite; if $a \le b$, the LHS is zero, otherwise a.

```
Lemma cdiff_0n n : \0c -c n = \0c.

Lemma cdiff_pr4 a b a' b': natp a -> natp b ->
  natp a' -> natp b' ->
  a <=c b -> a' <=c b' ->
  (b +c b') -c (a +c a') = (b -c a) +c (b' -c a').
Lemma cdiffA a b c:
  natp a -> natp b -> natp c ->
  (b +c c) <=c a -> (a -c b) -c c = a -c (b +c c).
Lemma cpred_pr4 a: cardinalp a ->
  cpred a = a -c \1c.
Lemma cdiff_lt_pred a b: natp b -> b <> \0c ->
   (a <c b <-> a <=c (b -c \1c)).
Lemma cdiff_nz1 a b: natp a -> natp b ->
  (csucc b) <=c a -> a -c b <> \0c.
Lemma cdiff_A1 a b: natp a -> natp b ->
  (csucc b) <=c a -> cpred (a -c b) = a -c (csucc b).
Lemma cdiff_ab_le_a a b: (a -c b) <= a.
Lemma cdiff_ab_lt_a a b: natp a -> b <=c a -> b <> \0c ->
   a -c b <c a.
Lemma cdiff_lt_symmetry' n p: natp p -> p <> \0c ->
  cpred (p -c n) <c p.
Lemma cdiff_lt_symmetry n p: natp p ->
  n <c p -> cpred (p -c n) <c p.
Lemma double_diff n p: natp n ->
  p <=c n -> n -c (n -c p) = p.
Lemma csucc_diff a b: natp a -> natp b  ->
  (csucc b) <=c a -> a -c b = csucc (a -c (csucc b)).
Lemma cdiff_pr5 a b c: cardinalp a -> cardinalp b -> natp c ->
  (a +c c) -c (b +c c) = a -c b.
Lemma cdiff_pr6 a b: natp a -> natp b ->
  (csucc a) -c (csucc b) = a -c b.
Lemma cprodBl a b c: natp a -> natp b -> natp c ->
   a *c (b -c c) = (a *c b) -c (a *c c).
```

Consider an injective function f from A into B, which are sets with cardinals a and b. Let c be the cardinal of the complement of the image, we have $a+c=b$. We deduce $b-a=c$ when the target is finite.

```
Lemma cardinal_complement_image1 f (S := source f) (T := target f) :
  injection f ->
  (cardinal (T -s (Imf f))) +c (cardinal S) = cardinal T.
Lemma cardinal_complement_image f (S := source f) (T := target f) :
  injection f -> finite_set T ->
  cardinal (T -s (Imf f)) = (cardinal T) -c (cardinal S).
```

Ordering on N. We give here some trivial properties.

```
Lemma NleR a: inc a Nat -> a <=N a.
Lemma NleT a b c: a <=N b -> b <=N c -> a <=N c.
Lemma NleA a b: a <=N b -> b <=N a -> a = b.
```

Simplification in inequalities. We show here that if $a+b \le a+b'$, $a+b < a+b'$, $ab \le ab'$ or $ab < ab'$ then $b \le b'$ or $b < b'$ if inequality is strict in the assumption; in the case of a product,

a must be non-zero. This is because $\leq$ is a total ordering, and the opposite relation yields a contradiction. We deduce that $(a+c)-(b+c) = a-b$, even when b is greater than a.

If $c \leq a+b$, then $c-b \leq a$ (this holds even for infinite cardinals). If the first inequality is strict, so is the second (note: if $b \leq c$ is false, then $c-b=0$, and the first result is trivial; the second holds only if $a \neq 0$, so we give two variants; note also that for the first varint b is an integer as $b \leq c$ and in the second variant c is an integer since $c < a+b$)).

```
Section Simplification.
Variables a b c: Set.
Hypothesis (aN: natp a) (bN: natp b) (cN: natp c).

Lemma csum_le2l: (a +c b) <=c (a +c c) ->  b <=c c.
Lemma csum_le2r: (b +c a) <=c (c +c a) -> b <=c c.
Lemma csum_lt2l: (a +c b) <c (a +c c) -> b <c c.
Lemma csum_lt2r: (b +c a) <c (c +c a) -> b <c c.
Lemma cprod_le2l: a <> \0c ->  (a *c b) <=c  (a *c c) -> b <=c c.
Lemma cprod_le2r: a <> \0c ->  (b *c a) <=c  (c *c a) -> b <=c c.
Lemma cprod_lt2l: (a *c b) <c  (a *c c) -> b <c c.
Lemma cprod_lt2r: (b *c a) <c  (c *c a) -> b <c c.
End Simplification.

Lemma csum_lt2l a b c:
  natp a -> natp b -> natp c ->
  (a +c b) <c (a +c c) -> b <c c.
Lemma cprod_le2l a b c:
  natp a -> natp b -> natp c -> a <> \0c ->
  (a *c b) <=c  (a *c c) -> b <=c c.
Lemma cprod_lt2l a b c:
  natp a -> natp b -> natp c -> a <> \0c ->
  (a *c b) <c (a *c c) -> b <c c.
Lemma cdiff_pr9 n p q: natp n -> natp p -> natp q -> q <=c p ->
  (n <=c p -c q <-> n +c q <=c p).
Lemma cdiff_Mle a b c: natp a -> natp b ->
  c <=c (a +c b) ->  (c -c b) <=c a.
Lemma cdiff_Mlt a b c: natp a -> natp c ->
  b <=c c -> c <c (a +c b) ->  (c -c b) <c a.
Lemma cdiff_Mlt' a b c: natp a -> natp b ->
  a <> \0c -> c <c (a +c b) ->  (c -c b) <c a.
```

6.3 Intervals in sets of integers

Bourbaki says that there is a set formed of all x such that "x is a cardinal and $x \leq_{\text{card}} a$" (note that, by definition, $x \leq_{\text{card}} a$ implies that x is a cardinal) . He denotes this set by $[0,a]$; this is a subset of the sets of integers, and, as such, is well-ordered by $\leq_{\text{card}}$. The notation $[a,b]$ could be interpreted as the set of all x such that "$a \leq_{\text{card}} x$ and $x \leq_{\text{card}} b$".

In what follows, we shall define $[a,b]$ as the closed interval with end-points a and b for $\leq_{\mathbf{N}}$, ordered by $\leq_{\text{card}}$. We have already shown that, if a and b are integers, $x \in [a,b]$ is equivalent to $a \leq_{\text{card}} x$ and $x \leq_{\text{card}} b$. The intervals $[0,a[$, $[0,a]$, and $[1,a]$ will be used a lot.

```
Definition Nintcc a b := interval_cc Nat_order a b.
Definition Nint a:= interval_co Nat_order \0c a.
Definition Nintc a:= Nintcc \0c a.
```

```
Definition Nint1c a:= Nintcc \1c a.
Definition Nint_cco a b := graph_on cardinal_le (Nintcc a b).
```

We give here some basic properties of intervals. Note that $[0, n[= n$ as this is the set of all integers $< n$.

Note that $[a, b]$ is a subset of **N** even when a and b are not integers. We have $[0, a+1[= [0, a] = [0, a[\cup \{a\}$. We have $[0, 1[= \{0\}$ and $[0, 0[= \emptyset$.

```
Lemma Nint_S a b: sub (Nintcc a b) Nat.
Lemma Nint_S1 a: sub (Nint a) Nat.
Lemma Nintc_i b x: inc x (Nintc b) -> x <=c b.
Lemma NintcP b: natp b -> forall x, inc x (Nintc b) <-> x <=c b.
Lemma Nint1cP b: natp b -> forall x,
  inc x (Nint1c b) <->  (x <> \0c /\ x <=c b).
Lemma Nint1cPb b: natp b -> forall x,
  inc x (Nint1c b) <->  (\1c <=c x /\ x <=c b).
Lemma NintE n: natp n -> Nint n = n.
Lemma NintP a: natp a -> forall x,
  (inc x (Nint a) <-> x <c a).
Lemma Nint_co_cc p: natp p -> Nintc p = Nint (csucc p).
Lemma NintcE n: natp n -> Nintc n = csucc n.
Lemma NintsP a: natp a -> forall x,
  (inc x (Nint (csucc a)) <-> x <=c a).

Lemma Nint_co00: Nint \0c = emptyset.
Lemma Nint_co01: (inc \0c (Nint \1c) /\ Nint \1c = singleton \0c).
Lemma Nint_cc00: Nintc \0c = singleton \0c.
Lemma Nint_si a: natp a -> inc a (Nint (csucc a)).
Lemma Nint_M a: natp a -> sub (Nint a) (Nint (csucc a)).
Lemma Nint_M1 a b: natp b ->   a <=c b -> sub (Nint a) (Nint b).
Lemma Nint_pr4 n: natp n ->
  ( ((Nint n) +s1 n = (Nint (csucc n))) /\ ~(inc n (Nint n))).
Lemma Nint_pr5 n (si := Nintcc \1c n): natp n ->
  ( (si +s1 \0c = Nintc n)  /\ ~(inc \0c si)).
Lemma inc0_int01: inc \0c (Nint \1c).
Lemma inc0_int02: inc \0c (Nint \2c).
Lemma incsx_intsn x n: natp n ->
  inc x (Nint n) -> inc (csucc x) (Nint (csucc n)).

Lemma Nat_induction5 (r:property) a:
  natp a -> r \0c ->
  (forall n, n <c a -> r n -> r (csucc n)) ->
  (forall n, n <=c a -> r n).
```

Let $\leq_{[a,b]}$ be the ordering of $[a, b]$. This is a well-ordering. If a and b are integers, then $x \leq_{[a,b]} y$ is equivalent to $a \leq x \leq y \leq b$.

```
Section IntervalNatwo.
Variables (a b: Set).
Hypotheses (aN: natp a)(bN: natp b).

Lemma Ninto_wor: worder_on (Nint_cco a b) (Nintcc a b).
Lemma Ninto_gleP x y:
  gle (Nint_cco a b) x y <->
    [/\ inc x (Nintcc a b), inc y (Nintcc a b) & x <=c y].
```

```
Lemma Ninto_gleP2 x y:
  gle (Nint_cco a b) x y <->  [/\ a <=c x, y <=c b & x <=c y].
End IntervalNatwo.
```

Let $\leq_{[0,a[}$ be the ordering of $[0,a[$. This is a well-ordering. If a is an integer, then $x \leq_{[0,a[} y$ is equivalent to $x \leq y < a$.

```
Definition Nint_co a :=
  graph_on cardinal_le (Nint a).

Section IntervalNatwo1.
Variable (a: Set).
Hypothesis (aN: natp a).

Lemma Nintco_wor:worder_on (Nint_co a) (Nint a).
Lemma Nintco_gleP x y:
  gle (Nint_co a) x y <-> (x <=c y  /\ y <c a).
End IntervalNatwo1.
```

Let n be an integer; let I be the segment with end-point n for the ordering of $\mathbf{N}$. This is the set of all k such that $k < n$, thus is the interval $[0,n[$; if we use von Neumann cardinals, we see that this is n.

```
Lemma segment_Nat_order n: natp n ->  segment Nat_order n = Nintc n.
Lemma segment_Nat_order1 n: natp n -> segment Nat_order n = n.
```

Notations. Assume that $(X_i)_i$ is a family of cardinals, indexed by an interval $[a,b]$. Instead of $\sum_{i\in[a,b]} X_i$ one may write $\sum_{a\leq i\leq b} X_i$ or $\sum_{i=a}^{b} X_i$. Instead of $\sum_{i\in[a,b[} X_i$ one may write $\sum_{i=a}^{b-1} X_i$.

The cardinal of an interval. We consider now the function $z \mapsto z+b$, $(z \in [0,a], z+b \in [a,a+b])$, that has $z \mapsto z-b$ as inverse, and hence is a bijection. Proposition 4 [4, p. 174] says that these functions are order isomorphisms (Bourbaki specifies "strictly increasing").

```
Definition rest_plus_interval a b :=
  Lf(fun z => z +c b)(Nintcc \0c a)(Nintcc b (a +c b)).

Definition rest_minus_interval a b :=
  Lf(fun z => z -c b) (Nintcc b (a +c b)) (Nintcc \0c a)

Theorem restr_plus_interval_is a b
        (f := (rest_plus_interval a b))
        (g := (rest_minus_interval a b)):
  natp a -> natp b ->
  [/\ bijection f,  bijection g, g = inverse_fun f &
   order_isomorphism f (Nint_cco \0c a) (Nint_cco b (a +c b))].
```

We have $[0,b+1] = [0,b] \cup \{b+1\}$ and the union is disjoint. By induction $[0,b]$ has $b+1$ elements, and by application of the isomorphism shown above, $[a,b]$ has $(b-a)+1$ elements. [Note: $[0,b] = b+1$ This is Proposition 5 [4, p. 174]. As a consequence, the set of integers is infinite (since $[0,n]$ is a subset of $\mathbf{N}$ we have $n+1 \leq \text{card}(\mathbf{N})$, so that $\text{card}(\mathbf{N})$ cannot be of the form n). This argument is used at the start of Chapter 6: the axiom that asserts the existence of an infinite set is equivalent to the assertion that there exists a set containing all finite cardinals. Note that $[a,b]$ is finite whatever a, b (since in the bad case, the interval is empty).

```
Lemma card_Nintc a: natp a -> cardinal (Nintc a) = csucc a.
Lemma card_Nintcp a: natp a -> a <> \0c -> cardinal (Nintc (cpred a)) = a.
Lemma card_Nint a: natp a -> cardinal (Nint a) = a.
Theorem card_Nintcc a b: a <=N b-> cardinal (Nintcc a b) = csucc (b -c a).
Lemma card_Nint1c a: natp a -> cardinal (Nint1c a) = a.
Lemma finite_Nintcc a b: finite_set (Nintcc a b).
Lemma finite_Nint a: finite_set (Nint a).
Lemma infinite_Nat: ~(finite_set Nat).
```

Isomorphism of finite totally ordered sets. Proposition 6 [4, p. 175] asserts that every finite totally ordered set is isomorphic to a unique interval $[1,n]$, where $n \geq 1$ is the number of elements. It is sometimes easier to use the interval $[0,n[$. Note the result holds also for $n = 0$ (in this case, the set is empty, as well as the interval). Proof. Let E and F be two finite equipotent sets; assume that they are totally ordered. The ordering is then a well-ordering. So one set is uniquely isomorphic to segment of the other. Assume for instance that E is isomorphic to a segment I of F. Now I, E and F have the same cardinal. Since F is finite, $\mathrm{I} \subset \mathrm{F}$ says $\mathrm{I} = \mathrm{F}$. **Note:** Bourbaki uses Corollary 2 to Proposition 2 of § 2, no. 2 (it says: if X is a subset of a finite set E, and $\mathrm{X} \neq \mathrm{E}$, then $\mathrm{Card(X)} < \mathrm{Card(E)}$. Maybe Corollary 4 was intended, since it says that an injection is bijective).

We also show: if $f : \mathrm{E} \to \mathrm{E}'$ is a strictly decreasing function between two well ordered sets, then E is finite. Proof: as E is well ordered, there is an order morphism $\mathrm{E} \to \mathbf{N}$ or $\mathbf{N} \to \mathrm{E}$; in the first case, we use the fact that the image of the morphism is a segment of $\mathbf{N}$, so is an interval I_n, and E is finite. In the second case, we just have a strictlty decreasing function $g : \mathbf{N} \to \mathrm{E}'$ by composing the morphism with f. Now $g(n+1) < g(n)$ says that the image of g has no least elementt. Thos contracts the well order property of E'.

```
Lemma isomorphism_worder_finite r r':
  total_order r -> total_order r' ->
  finite_set (substrate r) -> (substrate r) \Eq (substrate r') ->
  exists! f, order_isomorphism f r r'.
Theorem finite_ordered_interval r: total_order r ->
  finite_set (substrate r) ->
  exists! f, order_isomorphism f r
    (Nint_cco \1c (cardinal (substrate r))).
Theorem finite_ordered_interval1 r: total_order r ->
  finite_set (substrate r) ->
  exists! f, order_isomorphism f r
    (Nint_co (cardinal (substrate r))).
Lemma worder_decreasing_finite r r' (f:fterm):
  worder r -> worder r' ->
  (forall i, inc i (substrate r) -> inc (f i) (substrate r')) ->
  (forall i j, glt r i j -> glt r' (f j) (f i)) ->
  finite_set (substrate r).
```

Induction properties of sums and products on intervals. We consider here a variant of (6.3). We shall write $f(n)$ instead of x_i and $\mathrm{F}(n)$ for $\sum_{i<n} f(i)$. Note: in the code, all sums are of the form $\sum_{i\in\mathrm{A}}$ where A is some set. Forst instance $i < n$ is replaced by $i \in \mathrm{I}_n$ where I_n is trhe set of integers $< n$, but more generally by $i \in n$. In this case we have

$$F(0) = 0, \qquad F(n+1) = f(n) + F(n).$$

We shall see later on that the unique function F satisfying this equation (for every integer n) can be defined by induction. In the case of ordinals, the inductive definition is a bit easier to manipulate.

```
Lemma induction_on_sum n f (sum := fun n => csumb n f):
  natp n -> sum (csucc n) = (sum n) +c (f n).
Lemma induction_on_prod n f (prod := fun n=> cprodb n f):
  natp n -> prod (csucc n) = (prod n) *c (f n).
```

An immediate consequence, using commutativity, is

$$\sum_{0\le i\le n} f(i) = f(0) + \sum_{1\le i\le n} f(i) = f(0) + \sum_{0\le i\le n-1} f(i+1) = \sum_{0\le i\le n} f(n-i). \tag{6.5}$$

```
Lemma fct_sum_rec0 f n: natp n ->
  csumb (Nintc n) f = (csumb (Nint1c n) f) +c (f \0c).
Lemma fct_sum_rec1 f n: natp n ->
  csumb (csucc n) f = (csumb n (fun i=> f (csucc i))) +c (f \0c).
Lemma fct_sum_rev f n (I := (csucc n)):
  natp n -> csumb I f = csumb I (fun i=> f (n -c i)).
```

6.4 Finite sequences

A *finite sequence* is a family $(x_i)_{i\in I}$ whose index set is a finite subset of **N** (Bourbaki says: a finite set of integers). Let f be the unique isomorphism f of the interval $[1, n]$ onto I (with the natural ordering on I). Then $x_{f(k)}$ is defined for $k \in [1, n]$. It is called the *kth term of the sequence*. If $k = 1$ or $k = n$, it is called the first or last term.

6.5 Characteristic functions on sets

The characteristic function has been introduced above. All lemmas that follow are trivial and could have been proved earlier (except for the complement since subtraction was defined later on).

The function ϕ_A (for $A \subset E$) is constant if and only if $A = E$ or $A = \emptyset$. Proposition 7 [4, p. 176] lists additional properties.

$$\phi_{E-A}(x) = 1 - \phi_A(x),$$

$$\phi_{A\cap B}(x) = \phi_A(x)\phi_B(x),$$

$$\phi_{A\cap B}(x) + \phi_{A\cup B}(x) = \phi_A(x) + \phi_B(x).$$

```
Lemma char_fun_V_aa A x: inc x A ->
  Vf (char_fun A A) x = \1c.
Lemma char_fun_V_bb A x: inc x A ->
  Vf (char_fun emptyset A) x = \0c.
Lemma char_fun_constant A B:
   sub A B -> (cstfp  (char_fun A B) B)  -> (A=B \/ A = emptyset).
```

```
Lemma char_fun_setC A B x: sub A B -> inc x B ->
  Vf (char_fun (B -s A) B) x = \1c -c (Vf (char_fun A B) x).
Lemma char_fun_setI A A' B x: sub A B -> sub A' B -> inc x B ->
  Vf (char_fun (A \cap A') B) x
  =  (Vf (char_fun A B) x)  *c (Vf (char_fun A' B) x).
Lemma char_fun_setU A A' B x: sub A B -> sub A' B -> inc x B ->
  (Vf (char_fun (A \cap A') B) x)
  +c (Vf (char_fun (A \cup A') B) x)
  =  (Vf (char_fun A B) x) +c (Vf (char_fun A' B) x).
```

6.6 Euclidean Division

Theorem 1 [4, p. 176] says that, if $b > 0$, a and b are integers, there exist unique integers q (called *quotient*) and r (called *remainder*) such that $a = bq + r$ and $r < b$. We show that the conditions are equivalent to $bq \le a < b(q+1)$ and $r = a - bq$. Thus q is the least integer such that $a < b(q+1)$. This inequality is satisfied for $q = a$, this shows existence and uniqueness of q.

```
Definition cdivision_prop a b q r :=
  a = (b *c q) +c r /\ r <c b.
Lemma cdivision_prop_alt a b q r: natp a -> natp b ->
  natp q -> natp r -> b <> \0c ->
  (cdivision_prop a b q r <->
    [/\ (b *c q) <=c a, a <c (b *c csucc q) & r =  a -c (b *c q)]).

Lemma cdivision_unique a b q r q' r': natp a -> natp b ->
  natp q -> natp r -> natp q' -> natp r' -> b <> \0c ->
  cdivision_prop a b q r -> cdivision_prop a b q' r' ->
  (q = q' /\ r =r').

Lemma cdivision_exists a b: natp a -> natp b -> b <> \0c ->
  exists q r,
    [/\ natp q, natp r & cdivision_prop a b q r].
```

Originally q was the least ordinal satisfying $a < b(q+1)$. From $q \le a$ we deduce $q \in \mathbf{N}$. In the current definition, we consider the least integer. This gives an explicit form for the quotient; the remainder is defined by $a - bq$.

If the remainder of the division of a by b is zero we say that b *divides* a. We use here a strict definition: a and b are assumed to be integers.

```
(* old def:
    Definition cquo a b := least_ordinal (fun q => a <c b *c (csucc q)) a.
*)
Definition cquo a b := intersection (Zo Nat (fun q => a <c b *c (csucc q))).
Definition crem a b := a -c (b *c (cquo a b)).
Definition cdivides b a :=
  [/\ natp a, natp b & crem a b = \0c].

Notation "x %/c y" := (cquo x y) (at level 40).
Notation "x %%c y" := (crem x y) (at level 40).
Notation "x %|c y" := (cdivides x y) (at level 40).

Lemma cdivision a b (q := a %/c b) (r :=  (a %%c b)):
```

```
  natp a -> natp b -> b <> \0c ->
  [/\ natp q, natp r & cdivision_prop a b q r].
```

We state some properties of division.

```
Lemma cquo_zero a: a %/c \0c = \0c.
Lemma crem_zero a: natp a -> a %%c \0c = a.
Lemma NS_quo a b: natp (a %/c b).
Lemma NS_rem a b: natp a -> natp b -> natp (a %%c b).
Lemma cdiv_pr a b: natp a -> natp b ->
  a = (b *c (a %/c b)) +c (a %%c b).
Lemma crem_pr a b: natp a -> natp b -> b <> \0c ->
  (a %%c b) <c b.
Lemma cquorem_pr a b q r:
  natp a -> natp b -> natp q -> natp r ->
  cdivision_prop a b q r -> (q = a %/c b /\ r = a %%c b).
Lemma cquorem_pr0 a b q:
  natp a -> natp b -> natp q -> b <> \0c ->
  a = (b *c q) -> (q = a %/c b /\ \0c = a %%c b).
Lemma crem_small a b: natp b -> a <c b -> a = a %%c b.
Lemma cquo_small a b: natp b -> a <c b -> a %/c b = \0c.
```

Note. Bourbaki says: the number q introduced above is *the integral part of the quotient of a by b*, since in the set of rational numbers $\mathbf{Q}$, there exists c such that $a = bc$, and q is the integral part of c. He reserves the term *quotient* only to the case where the remainder is zero. Moreover he says "writing a/b or $\frac{a}{b}$ will imply that b divides a". This is an abuse of notations. (It is all right to say that $a < b$ implies that a and b are integers, because we can consider as a short-hand for "$a < b$ and $a \in \mathbf{N}$ and $b \in \mathbf{N}$", but, if $d(a,b)$ is the property that b divides a, and $q(a,b)$ is the quotient, then a/b cannot be a short-hand for "$q(a,b)$ and $d(a,b)$" because this expression makes no sense (it is neither a term nor a relation). Moreover, the convention is not always respected since Bourbaki proves

$$\sum_{i=1}^{n} i = \frac{1}{2} n(n+1). \tag{6.6}$$

Now some consequences when division is exact. Bourbaki says: every multiple a' of a multiple a of b is a multiple of b. One can restate this as: if b divides a, then b divides ac.

```
Lemma cdivides_pr a b: b %|c a -> a =  b *c (a %/c b).
Lemma cdivides_pr1 a b: natp a -> natp b ->
   b %|c (b *c a).
Lemma cdivides_pr2 a b q:
  natp a -> natp b -> natp q -> b <> \0c ->
  a = b *c q -> q = a %/c b.
Lemma cdivides_one a: natp a -> \1c %|c  a.
Lemma cquo_one a: natp a -> a %/c \1c = a.
Lemma cdivides_pr3 a b q:
  b %|c a -> q = a %/c b ->  a = b *c q.
Lemma cdivides_pr4 b q: natp b -> natp q -> b <> \0c ->
  (b *c q) %/c b = q.
Lemma cdivision_of_zero n: natp n ->
  (n %|c \0c /\ \0c %/c n = \0c).
Lemma cdivides_zero n: natp n -> n %|c \0c.
Lemma crem_of_zero n: natp n -> \0c %%c n = \0c.
```

```
Lemma cdivision_itself a: natp a -> a <> \0c ->
  (a %|c a /\ a %/c a = \1c).
Lemma cdivides_itself n: natp n -> n %|c n.
Lemma cquo_itself a: natp a -> a <> \0c ->
   a %/c a = \1c.
Lemma cdivides_trans a b a':
  a  %|c a'-> b  %|c a -> b  %|c a'.
Lemma cdivides_trans1 a b a':
  a  %|c a' -> b %|c a -> a' %/c b = (a' %/c a) *c (a  %/c b).
Lemma cdivides_trans2 a b c: natp c ->
  b  %|c a -> b %|c (a *c c).
Lemma divides_smaller a b: b %|c a -> a <> \0c -> b <=c a.
```

The first lemma says $(ac)/(bc) = a/b$ even when division is not exact. If b divides a and a', it divides the sum and the difference.

```
Lemma cquo_simplify a b c:
  natp a -> natp b -> natp c -> b <> \0c -> c <> \0c ->
  (a *c c) %/c (b *c c) =  a %/c b.
Lemma cdivides_and_sum a a' b: b %|c a ->  b %|c a' ->
   (b %|c  (a +c a') /\
    (a +c a') %/c b = (a %/c b) +c (a' %/c b)).
Lemma cdivides_and_difference a a' b:
  a' <=c a ->   b %|c a ->  b %|c a' ->
  [/\ b %|c  (a -c a'), (a' %/c b) <=c (a %/c b) &
    (a -c a') %/c b = (a %/c b) -c (a' %/c b)].
Lemma cdivides_diff1 x a b: natp b -> x %|c a ->  x %|c (a +c b) -> x %|c b.
```

Definition by induction. We know how to define a function by transfinite induction on a well-ordered set, and we know that **N** is well-ordered. The definitions here are a bit technical, details can be found on page 194.

Let a be an object, $\mathrm{P}(x)$ a function of one variable, $\mathrm{Q}(x, y)$ a function of two variables. There is a unique surjective function f (resp. g) defined on **N** such that $f(0) = a$ and $f(n+1) = \mathrm{P}(f(n))$, (resp. $g(0) = a$ and $g(n+1) = \mathrm{Q}(n, g(n))$). Uniqueness is obvious by induction.

```
Definition induction_defined0 (h: fterm2)  (a: Set) :=
  transfinite_defined Nat_order
  (fun u => Yo(source u = \0c) a
       (h (cpred (source u))(Vf u (cpred (source u)) ))).

Definition induction_defined (s: fterm)  (a: Set) :=
  transfinite_defined Nat_order
  (fun u => Yo(source u = \0c) a (s (Vf u (cpred (source u))))).

Lemma induction_defined_pr0 h a (f :=  induction_defined0 h a):
    [/\ source f = Nat,  surjection f, Vf \f 0c = a &
     forall n, natp n -> Vf f (csucc n) = h n (Vf f n)].
Lemma induction_defined_pr s a (f :=  induction_defined s a):
    [/\ source f = Nat,  surjection f, Vf f \0c = a &
     forall n, natp n -> Vf f (csucc n) = s (Vf f n)].
Lemma integer_induction0 h a: exists! f,
  [/\ source f = Nat, surjection f,
    Vf \f 0c = a &
    forall n, natp n -> Vf f (csucc n) = h n (Vf f n)].
```

```
Lemma integer_induction s a: exists! f,
  [/\ source f = Nat,  surjection f, Vf f \0c = a &
  forall n, natp n -> Vf f (csucc n) = s (Vf f n)].
```

If g is as above, we can consider $n \mapsto g(n)$.

```
Definition induction_term s a := Vf (induction_defined s a).
Lemma induction_term0 s a:
  induction_term s a \0c = a.
Lemma induction_terms s a n:
  natp n ->
  induction_term s a (csucc n) = s n (induction_term s a n).
```

6.7 Expansion to base b

Proposition 8 [4, p. 177] is *Let b be an integer* > 1. *For each integer $k > 0$ let* E_k *be the lexicographic product of the family* $(\mathrm{J}_h)_{0\le h\le k-1}$ *of intervals all identical with* $[0, b-1]$*; For each* $r = (r_0, r_1, \dots, r_{k-1}) \in \mathrm{E}_k$*, let* $f_k(r) = \sum_{h=0}^{k-1} r_h b^{k-h-1}$*; then the mapping f_k is an isomorphism of the ordered set* E_k *onto the interval* $[0, b^k - 1]$. Bourbaki notes that (if $a > 0$) there is a least integer k such that $a < b^k$ hence a unique sequence r_h such that

$$a = \sum_{h=0}^{k-1} r_h b^{k-h-1} \tag{6.7}$$

subject to the conditions $0 \le r_h \le b-1$ for $0 \le h \le k-1$ and $r_0 > 0$.

Discussion. We say that (6.7) is a BE-expansion, and it is a normalized expansion if either $k = 0$ (the sum is empty) or $r_0 > 0$. The quantity r_h is the digit of index k, and r_0 is the leading digit. We can restate the theorem as: every integer has an expansion to base b for some k, and a unique normalized expansion. Two numbers expressed in base b with k digits can be compared using only the value of the digits, starting with the leading digits. We can complete the theorem as follows: one can add or remove zero leading digits in the expansion, hence given two numbers with k and k' digits, one can add leading zeroes to the smallest sequence, then apply the theorem, or else remove leading zeroes in order to get normalized expansions, and then the number that has the smallest number of digits is the smallest number.

Note that $0 \le r_h$ holds trivially, so that $0 \le r_h \le b-1$ is equivalent to $r_h \le b-1$, thus to $r_h < b$. The interval $[0, b-1]$ is the interval $[0, b[$, and $[0, b^k - 1]$ is $[0, b^k[$. The sum $\sum_{h=0}^{k-1} \mathrm{X}_h$ is also the sum $\sum_{h\in[0,k[} \mathrm{X}_h$ or $\sum_{h<k} \mathrm{X}_h$. The condition $0 \le h \le k-1$ is equivalent to $0 \le k-h-1 \le k-1$, and the sum can be rewritten as $\sum_{h=0}^{k-1} r_{k-h-1} b^h$. If $s_k = r_{k-h-1}$, we get

$$a = \sum_{h=0}^{k-1} s_h b^h \tag{6.8}$$

subject to the conditions $0 \le s_h \le b-1$ for $0 \le h \le k-1$ and $s_{k-1} \ne 0$ is the normalization condition. We call this is LE-expansion.[3] Associated to $f_k(r)$ is the function $g_k(s)$.

If $\psi(s)$ is the sequence $(s_1, \dots, s_k)$, then

$$g_{k+1} = s_0 + b.g_k(\psi(s)) \tag{6.9}$$

[3]According to Wikipedia, little-endian storage means: increasing numeric significance with increasing memory addresses; big-endian is its opposite, most-significant byte first.

(Bourbaki has a similar formula with f and ϕ). This is a recursive definition; one can convert it into an iterative one: as long as there are digits, multiply by b and add the next digit. We start with s_{k-1} and terminate with s_0. This means that we consider digits from left to right, r_0 then r_1, then r_2, etc. This is called the Horner scheme for the formula (6.7), and is used by every computer program to read numbers. If s is represented as a list, then s_0 is the head of the list and $\psi(s)$ is its tail, and (6.9) is the natural way to associate a value to the list.

A consequence of (6.9) is that s_0 and $g_k(\psi(s))$ are the remainder and quotient of the Euclidean division of a by b, and this shows uniqueness by induction. Computers use this method for printing numbers, i.e., finding the sequence s_h given a; the number k is not known a priori. In practice, one has either fixed-size numbers, say $< 2^{32}$, case where an apriori bound can be found; or else $a = \sum_0^{K-1} S_h B^h$, for some B, case where $k \le nK$ for some n, which is the size of the expansion of B in base b. The digits are computed one after the other, stored in a buffer. After that, the number is normalized (useless zeroes are removed).

Assume $s_0 < b$, $s'_0 < b$; then $s_0 + bg \le s_0 + bg'$ if and only if either $g = g'$ and $s_0 \le s'_0$ or $g < g'$. This condition is equivalent to $(g, s_0) \le (g', s'_0)$, where (g, s_0) is in the substrate of some lexicographic product $F_k \times J$ of two sets. By induction, we can identify F_k with the set of sequences $(s_1, \dots, s_k)$. Because s_0 comes after g, this set is the lexicographic product *in reverse order* of the sets J_h. Thus, the ordering of the sequence r_h is the lexicographic product of the sets J_h.

Consider now the sequence $\Psi(s) = (s_0, \dots, s_{k-1})$, then

$$g_{k+1} = g_k(\Psi(s)) + s_k.b^k. \tag{6.10}$$

It happens that s_k and $g_k(\Psi(s))$ are the quotient and remainder of the Euclidean division of g_{k+1} by b^k. This is an alternate way to show uniqueness of the expansion (one could use it to print a number in a computer program, the drawback being that one has to compute all b^k in decreasing order). Note that $s_k b^k \le g_{k+1} < (s_k + 1)b^k$. This shows that two numbers of the same size with distinct leading digits compare as their leading digits, and shows the theorem. We shall use this approach since Ψ is just the restriction.

We define an *expansion* to be a family of k terms, all less than b, where k and b are integers, $b \ge 2$. The domain of the family is the interval $[0, k[$, it is the set of integers i with $i < k$. The associated *value* is $\sum f_i b^i$. It is an integer. If we have an expansion of length $k+1$, the restriction to $[0, k[$ is an expansion. The values are the same, up to the quantity $f_k b^k$. Similarly, we can extend an expansion from size k to size $k+1$.

```
Lemma b_power_k_large a b: natp a -> natp b ->
  \1c <c b -> a <> \0c -> exists k,
    [/\ natp k, (b ^c k) <=c a & a <c (b ^c (csucc k))].

Definition expansion f b k :=
  [/\ natp b, natp k, \1c <c b &
  [/\ fgraph f, domain f = k &
  forall i, inc i (domain f) -> (Vg f i) <c b]].

Definition expansion_value f b :=
  csumb (domain f) (fun i=> (Vg f i) *c (b ^c i)).
```

We assume locally that (f, k, b) and $(g, k'+1, b)$ are expansions.

```
Section Base_b_expansion.
```

```
Variables f b k k': Set.
Hypothesis Exp: expansion f b k.
Hypothesis Expg: expansion g b (csucc k').
Hypothesis ck' : cardinalp k'.

Lemma expansion_prop0P i:
   (inc i (domain f)) <-> i <c k.
Lemma expansion_prop1 i:
  i <c k -> natp (Vg f i).
Lemma expansion_prop2:
  finite_int_fam  (Lg (domain f) (fun i=> (Vg f i) *c  (b ^c i))).
Lemma expansion_prop3: natp (expansion_value f b).
Lemma expansion_prop4: natp k'.
Lemma expansion_prop5:
  expansion (restr g k') b k'.
Lemma expansion_prop6: natp (Vg g k').
Lemma expansion_prop7:
  (expansion_value g b) =
   (expansion_value (restr g k') b) +c (Vg g k *c (b ^c k')).
End  Base_b_expansion.
```

Denote by $s(f)$ or by $s_k(f)$ the sum $\sum_{i<k} f_i$. We have $s_k(f) < b^k$, and $s_{k+1}(f) = s_k(f) + f_k b^k$. (we also have $s_{k+1}(f) = s_k(f')b + f_0$, where f' is f shifted by one position). As a consequence the quotient and remainder of the division of $s_{k+1}(f)$ by b^k are f_k and $s_k(f)$. This shows uniqueness of the expansion, namely that $s_k(f) = s_k(g)$ implies $f_i = g_i$ for all $i < k$.

```
Lemma expansion_prop8 f b k x
    (h:=  Lg (csucc k) (fun i=> Yo (i=k) x (Vg f i))) :
   expansion f b k -> natp x -> x <c b ->
    (expansion h b (csucc k) /\
      expansion_value h  b =
      (expansion_value f b) +c ((b ^c k)  *c x)).
Lemma expansion_prop8_rev f b k x
  (h := Lg (csucc k) (fun i => Yo (i = \0c) x (Vg f (cpred i)))):
  expansion f b k -> natp x -> x <c b ->
  (expansion h b (csucc k) /\
      expansion_value h b =  (expansion_value f b) *c b  +c x).
Lemma expansion_prop9 f b k: expansion f b k ->
    (expansion_value f b) <c (b ^c k).
Lemma expansion_prop10 f b k: cardinalp k ->
  expansion f b (csucc k) ->
  cdivision_prop (expansion_value f b) (b ^c k) (Vg f k)
  (expansion_value (restr f k) b).
Lemma expansion_unique f g b k:
  expansion f b k -> expansion g b k ->
  expansion_value f b = expansion_value g b -> f = g.
Lemma expansion_prop11 f g b k: cardinalp k ->
  expansion f b (csucc k) -> expansion g b (csucc k) ->
  (Vg f k) <c (Vg g k) ->
  (expansion_value f b) <c (expansion_value g b).
```

Consider the two following properties. $\mathrm{P}(f,g)$ says that there exists an index i in the range of g, not in the range of f such that g_i is not zero. It obviously implies $s(f) < s(g)$. Condition $\mathrm{Q}(f,g)$ first says that there is an index n such that $f_i = 0$ and $g_i = 0$ for $i \geq n$, provided that these expressions are defined (and f_i and g_i are defined for $i < n$). Such an index exists if

P(f, g) and P(g, f) are false. We have then $s(f) = s_n(f)$ and $s(g) = s_n(g)$. The Bourbaki claim is then that $s(f)$ and $s(g)$ can be compared lexicographically (replacing i by $n - i$). Condition Q says moreover that there exists k such that $f_i = g_i$ for $k < i < n$, so that $s(f)$ and $s(g)$ compare the same as $s_k(f)$ and $s_k(g)$. The case $k = -1$ is special, since it says that $s(f) = s(g)$. Thus, assuming $s(f) \neq s(g)$, there is a least such k [in other terms: if $s(f) \neq s(g)$ there is a greatest index k such that $f_k \neq g_k$]. Now condition Q says $f_k < g_k$. We have shown above that this implies $s_k(f) < s_k(g)$. The theorem is now $s(f) < s(g)$ if, and only if, one of P or Q is true. Notice that the five cases P(f, g), P(g, f), Q(f, g), Q(g, f), R(f, g) are mutually exclusive (here R is the condition that there is n, such that $f_i = g_i$ for $i < n$, and for all indices $i \geq n$ for which f_i and g_i are defined, the value is zero; it implies $s(f) = s(g)$).

```
Lemma expansion_restr1 f b k l:
  expansion f b k -> l <=c  k ->
  expansion (restr f l) b l.
Lemma expansion_restr2 f b k l:
  expansion f b k -> l <=c k ->
  (forall i, l <=c i -> i <c k -> Vg f i = \0c) ->
  expansion_value (restr f l) b = expansion_value f b.

Lemma expansion_prop12 f g b kf kg l n:
  n <=c kf -> n <=c kg ->  l <c n ->
  (forall i, n <=c i -> i <c kf -> Vg f i = \0c) ->
  (forall i, n <=c i -> i <c kg -> Vg g i = \0c) ->
  (forall i, l <c i ->  i <c n -> Vg f i = Vg g i) ->
  expansion f b kf -> expansion g b kg ->
  (Vg f l) <c  (Vg g l) ->
  (expansion_value f b)  <c (expansion_value g b).  (* 70 *)
Lemma expansion_prop13 f g b kf kg l:
  kf <=c l -> l <c kg ->
  expansion f b kf -> expansion g b kg ->
  Vg l g <> \0c ->
  (expansion_value f b) <c (expansion_value g b).
Lemma expansion_prop14 f g b kf kg:
  expansion f b kf -> expansion g b kg ->
  (expansion_value f b) <c  (expansion_value g b) ->
  (exists l, [/\ kf <=c l, l <c kg & Vg g l <> \0c])
 \/ (
  exists l n,
  [/\ n <=c kf, n <=c kg, l <c n &
  [/\ (forall i, n <=c i -> i <c kf -> Vg f i = \0c),
  (forall i, n <=c i -> i <c kg -> Vg g i = \0c) ,
  (forall i, l <c i ->  i <c n -> Vg f i = Vg g i) &
   (Vg f l)<c  (Vg g l)]]). (* 70 *)
Lemma expansion_prop15 f g b n:
  expansion f b n -> expansion g b n ->
  ( (expansion_value f b) <c  (expansion_value g b)
    <->  exists k,
         [/\ k <c n, (Vg f k) <c  (Vg g k) &
          (forall i, k <c i ->  i <c n -> Vg f i = Vg g i)]).
```

We consider the set S(A, B) of all functional graphs f, whose domain is a subset of A, and whose range is a subset of B. An expansion to base b is in S($\mathbf{N}$, b).

```
Definition sub_fgraphs A B :=  unionf (\Po A) (gfunctions ^~ B).
```

```
Lemma sub_fgraphsP A B f:
   inc f (sub_fgraphs A B) <-> exists2 C, sub C A & inc f (gfunctions C B).
Lemma expansion_bounded1 f k b : expansion f b k ->
  inc f (sub_fgraphs Nat b).
```

If $a < b^k$ there is an expansion of length k (proof by induction, the highest term is the quotient of the division by b^{k-1}). Since $a < b^a$, there is at least one expansion. We can remove leading initial coefficients, thus assume that either the expansion is empty, or that the leading coefficient is non-zero. This form is unique.

```
Definition exp_boundary f k :=
  (k = \0c \/ (k <> \0c /\  Vg f (cpred k) <> \0c)).
Definition expansion_of f b k a :=
   expansion f b k /\ expansion_value f b = a.
Definition expansion_normal_of f b k a :=
  expansion_of f b k a /\ (exp_boundary f k).

Section TheExpansion.
Variable b: Set.
Hypothesis bN: natp b.
Hypothesis bp: \1c <c b.

Lemma expansion_exists1 a k:
   natp k -> natp a -> a <c (b ^c k) ->
  exists f, expansion_of f b k a.
Lemma expansion_exists2 a: natp a ->
  exists k f, expansion_of f b k a.
Lemma expansion_exists3 a: natp a
  exists k f, expansion_normal_of f b k a.
Lemma expansion_unique1 a f k f' k':
  expansion_normal_of f b k a -> expansion_normal_of f' b k' a ->
  f = f' /\ k = k'.
```

We can now define the expansion in base b of a. It is the functional graph f (where k is the cardinal of the domain of f) such that $a = \sum b^i f_i$. The expansion of zero is the empty graph; the expansion of a digit (a number a such that $0 < a < b$) is the functional graph that maps zero to a.

For reasons explained below, we shall consider $\mathrm{R}(x) = \sum f_i$. If x is zero or a digit, then $\mathrm{R}(x) = x$, otherwise $\mathrm{R}(x) < x$ (if $k+1$ is the number of terms, then $f_k < b^k f_k$, as f_k and k are non-zero).

```
Definition the_expansion a :=
  select (fun z => expansion_normal_of z b (cardinal (domain z)) a)
     (sub_fgraphs Nat b).

Lemma the_expansion_pr a (z := the_expansion a):
   natp a ->
   expansion_normal_of z b (cardinal (domain z))  a.

Lemma the_expansion_zero: the_expansion \0c = emptyset.
Lemma the_expansion_digit a:
  a <> \0c -> a <c b -> the_expansion a = singleton (J \0c a).

Definition the_contraction a :=  csum (the_expansion a).
```

```
Lemma the_contraction_zero: the_contraction b \0c = \0c.
Lemma the_contraction_digit a: a <c b -> the_contraction a = a.
Lemma the_contraction_non_digit a: b <=c a -> natp a ->
   the_contraction a <c a.
Lemma the_contraction_non_zero a: natp a -> a <> \0c ->
   the_contraction a <> \0c.
```

We define $\mathrm{R}_n(a)$ by induction as follows: $\mathrm{R}_0(a) = a$ and $\mathrm{R}_{n+1}(a) = \mathrm{R}(\mathrm{R}_n(a))$. We have $\mathrm{R}_{n+1}(a) = \mathrm{R}_n(\mathrm{R}(a))$ and $\mathrm{R}_{n+m}(a) = \mathrm{R}_n(\mathrm{R}_m(a))$ (these relations hold whenever R_n is defined by induction). Let $\mathrm{T}(a) = \mathrm{R}_a(a)$. We have $\mathrm{T}(a) < b$. Proof. We show by induction on a that $c \le a$ implies $\mathrm{T}(c) < b$; the result follows, as $a \le a$. Assume the result true for a, and let's show it for $a+1$. So assume $c \le a+1$. If $c \le a$ we can use the induction hypothesis, so it suffices to consider $c = a+1$. Note that, if $c < b$, then $\mathrm{R}(c) = c$ and $\mathrm{R}_n(c) = c$, thus $\mathrm{T}(c) = c$, and the result holds. Otherwise $\mathrm{R}(c) < c$. We have $\mathrm{T}(a+1) = \mathrm{R}_a(\mathrm{R}(a+1))$. Since $c = a+1$, $\mathrm{R}(c) < c$ says $\mathrm{R}(a+1) \le a$. Let $x = \mathrm{R}(a+1)$. We have $\mathrm{T}(a+1) = \mathrm{R}_a(x) = \mathrm{R}_{a-x}(\mathrm{R}_x(x)) = \mathrm{R}_{a-x}(\mathrm{T}(x))$. By induction $\mathrm{T}(x) < b$ so that $\mathrm{R}_{a-x}(\mathrm{T}(x)) = \mathrm{T}(x)$ and the conclusion holds.

```
Definition contraction_rec a :=
  induction_defined (the_contraction) a.
Definition contraction_rep a := Vf (contraction_rec a) a.

Lemma contraction_rec0 a: Vf (contraction_rec a) \0c = a.
Lemma contraction_rec_succ a n: natp n ->
  Vf (contraction_rec a)(csucc n) = the_contraction (Vf (contraction_rec a) n).
Lemma contraction_rec_succ' a n: natp n ->
  Vf (contraction_rec a)(csucc n) = Vf (contraction_rec (the_contraction a)) n.
Lemma contraction_rec_succ'' a n m: natp n -> natp m ->
  Vf (contraction_rec a) (n +c m) =
     Vf (contraction_rec (Vf (contraction_rec a) n)) m.
Lemma NS_contraction_rec a n: natp a -> natp n ->
   natp (Vf (contraction_rec a) n).
Lemma contraction_rec_non_zero a n: natp n -> natp a -> a <> \0c ->
   (Vf (contraction_rec a) n) <> \0c.
Lemma contraction_rep_dig a : natp a -> contraction_rep a <c b.
Lemma contraction_rep_non_zero a: natp a -> a <> \0c ->
   (contraction_rep a) <> \0c.
End TheExpansion.
```

Computing modulo. We say that a and b are equal *modulo* n (where n is a non-zero integer), if a and b have the same remainder in the division by n. This is obviously an equivalence relation on $\mathbf{N}$; it is compatible with addition and multiplication. In what follows, we fix an integer B, and compute modulo B.

Note that $a\mathrm{B}+b$ and b are equal modulo B. We then show that if $a = 1$ mod B, then $a^n = 1$ mod B.

```
Notation "m = n %c[mod d ]" := (m %%c d = n %%c d)
   (at level 70, n at next level,
   format "'[hv ' m '/'  =  n '/'  %c[mod  d ] ']'").
Definition eqmod B a b:=  a = b %c[mod B].

Section ModuloProps.
Variable B: Set.
```

```
Hypothesis BN: natp B.
Hypothesis Bnz: B <> \0c.

Lemma eqmod_equivalence (R:= graph_on (eqmod B) Nat):
    equivalence R /\ substrate R = Nat.
Lemma crem_prop a b: natp a -> natp b ->
  B *c a +c b = b %c[mod B].
Lemma crem_sum a b: natp a -> natp b ->
   a +c b = a %%c B +c b %%c B %c[mod B].
Lemma crem_prod a b: natp a -> natp b ->
  a *c b = (a %%c B) *c (b %%c B) %c[mod B].
Lemma eqmod_sum a b a' b': natp a -> natp b ->
  natp a' -> natp b' ->
  a = a' %c[mod B] -> b = b' %c[mod B] -> a +c b = a' +c b' %c[mod B].
Lemma eqmod_prod a b a' b': natp a -> natp b ->
  natp a' -> natp b' ->
  a = a' %c[mod B] -> b = b' %c[mod B] -> a *c b = a' *c b' %c[mod B].
Lemma eqmod_rem a: natp a -> a =  a %%c B %c[mod B].
Lemma eqmod_succ  a a': natp a -> natp a' ->
   a = a' %c[mod B] -> csucc a = csucc a' %c[mod B].
Lemma eqmod_pow1  a n: natp a -> natp n ->
   a = \1c %c[mod B] -> a ^c n = \1c %c[mod B].
Lemma eqmod_pow2 a b n: natp a -> natp b -> natp n ->
   a = \1c %c[mod B] -> b *c a ^c n = b %c[mod B].
```

Assume $b = 1$ mod B. Then $\sum a_i b^i = \sum a_i$ modulo B.

```
Lemma eqmod_pow3  f b k: expansion f b k ->
 b = \1c %c[mod B] -> expansion_value f b = csum f %c[mod B].
End ModuloProps.
```

Define $5 = 4 + 1$, so that $2 + 3 = 5$. Define $9 = 3 \times 3$. Define $10 = 5 + 5$ so that $10 = 9 + 1$. We deduce that 10 is 1 mod 3 and 1 mod 9.

```
Definition card_five := csucc card_four.
Definition card_nine := \3c *c \3c.
Definition card_ten :=  card_five +c card_five.
Notation "\9c" := card_nine.
Notation "\10c" := card_ten.
Notation "\5c" := card_five.

Lemma NS5 : natp \5c.
Lemma NS9 : natp \9c.
Lemma NS10 : natp \10c.

Lemma card3_nz: \3c <> \0c.
Lemma card9_nz: \9c <> \0c.
Lemma card_sum_3_2: \3c +c \2c = \5c.
Lemma card_prod_3_3: \10c = csucc \9c.
Lemma card_mod_10_9: \10c = \1c %c[mod \9c].
Lemma card_mod_10_3: \10c = \1c %c[mod \3c].
Lemma cgt10_1: \1c <c \10c.
```

We can now state that two quantities $\sum a_i 10^i$ and $\sum a_i$ have the same remainder modulo 3.

```
Definition expansion_ten f k :=
  [/\ natp k, fgraph f, domain f = k &
  forall i, inc i (domain f) -> (Vg f i) <c \10c].
Lemma divisibiliy_by_three  f k: is_base_ten_expansion f k ->
  let g:= (Lg (domain f) (fun i=> (Vg f i) *c (\10c ^c i))) in
  csum g = csum f %c[mod \3c].
```

Let x be an integer, $\mathrm{B} = b+1$ and write $x = \sum \mathrm{B}^i x_i$. Let $\mathrm{R}(x) = \sum x_i$, $\mathrm{R}_n(x)$ the n-th iteration of R, and $\mathrm{T}(x)$ the fix-point of this iteration. Then x and $\mathrm{R}(x)$ are equal modulo b (and modulo any divisor of b). If $\mathrm{B} = 10$ these quantities are equal modulo 9 and 3. By induction $\mathrm{R}_n(x)$ is equal to x modulo b, so that $\mathrm{T}(x)$ is also x modulo b; since $\mathrm{T}(x) \le b$ we deduce: If $x = 0$, then $\mathrm{T}(x) = x$; if x is a multiple of b, then $\mathrm{T}(x) = b$, otherwise, $\mathrm{T}(x)$ is the remainder of the division of x by b.

```
Lemma crem_succ a B: natp a -> natp B -> \1c <c B ->
  csucc a = csucc (a %%c B) %c[mod B].
Lemma divisibiliy_by_nine f k: expansion_ten f k ->
  let g:= (Lg (domain f) (fun i=> (Vg f i) *c (\10c ^c i))) in
  csum g = csum f %c[mod \9c].
Lemma eqmod_contraction b a:
   natp a -> natp b -> b <> \0c ->
   a = the_contraction (csucc b) a %c[mod b].
Lemma eqmod_contraction_rep b a
   (x := contraction_rep (csucc b) a) (y := a %%c b) :
   natp a -> natp b -> b <> \0c ->
   [/\ a = x %c[mod b],
    (a = \0c -> x = \0c) &
    (a <> \0c -> (y = \0c -> x = b) /\ (y <> \0c -> x = y))].
   ( (a = \0c -> x = \0c)
   /\ (a <> \0c ->
       (y = \0c -> x = b) /\ (y <> \0c -> x = y))).
Lemma eqmod_contraction_rep9 a
   (x := contraction_rep \10c a) (y := a %%c \9c) :
   natp a ->
  [/\ a = x %c[mod \9c],
    (a = \0c -> x = \0c) &
    (a <> \0c -> (y = \0c -> x = \9c) /\ (y <> \0c -> x = y))].
```

Even and odd numbers. We say that x is even if its remainder in the division by two is zero, it is odd otherwise (the remainder is then one). The successor of an even number is odd, the sum of two odd numbers is even, etc.

```
Definition evenp n := natp n  /\ n %%c \2c = \0c.
Definition oddp n := natp n /\ ~ (evenp n).

Lemma crem_02: \0c %%c \2c = \0c.
Lemma crem_12: \1c %%c \2c = \1c.
Lemma crem_22: \2c %%c \2c = \0c.
Lemma oddp_alt n:  (oddp n <-> natp n /\ n %%c \2c = \1c).
Lemma evenp_mod2 n: natp n -> (evenp n <-> n = \0c %c[mod \2c]).
Lemma oddp_mod2 n: natp n -> (oddp n <-> n = \1c %c[mod \2c]).

Lemma succ_of_even n: evenp n -> oddp (csucc n).
Lemma succ_of_evenP n: natp n -> (evenp n <-> oddp (csucc n)).
```

```
Lemma succ_of_odd n: oddp n -> evenp (csucc n).
Lemma succ_of_oddP n: natp n -> (oddp n <-> evenp (csucc n)).

Lemma even_zero: evenp \0c.
Lemma odd_one: oddp \1c.
Lemma even_two: evenp \2c.

Lemma csum_of_even a b: evenp a -> evenp b -> evenp (a +c b).
Lemma csum_of_even_odd a b: evenp a -> oddp b -> oddp (a +c b).
Lemma csum_of_odd a b: oddp a -> oddp b -> evenp (a +c b).
Lemma csum_of_evenP n m: evenp n -> natp m  ->
  (evenp (n +c m) <-> evenp m).
Lemma csum_of_oddP n m: oddp n -> natp m  ->
  (evenp (n +c m) <-> oddp m).
```

We study here double and half.

```
Definition cdouble n := \2c *c n.
Definition chalf n := n  %/c \2c.

Lemma NS_double n: natp n -> natp (cdouble n).
Lemma NS_half n: natp (chalf n).

Lemma cdouble0: cdouble \0c = \0c.
Lemma even_double n: natp n -> evenp (cdouble n).
Lemma odd_succ_double n: natp n -> oddp (csucc (cdouble n)).

Lemma half_even n: evenp n -> n = cdouble (chalf n).
Lemma half_odd n: oddp n -> n = csucc (cdouble (chalf n)).
Lemma even_half n: natp n -> chalf (cdouble n) = n.
Lemma odd_half n: natp n -> chalf (csucc (cdouble n)) = n.
Lemma half0: chalf \0c = \0c.
Lemma half1: chalf \1c = \0c.
Lemma half2: chalf \2c = \1c.
Lemma cdouble_halfV n: natp n ->
   n = cdouble (chalf n) \/ n = csucc (cdouble (chalf n)).

Lemma double_sum a b: cdouble  a +c cdouble b =  cdouble (a +c b).
Lemma double_prod a b:  a *c cdouble b =  cdouble (a *c b).
Lemma double_succ a: natp a -> cdouble (csucc a) = csucc (csucc (cdouble a)).
Lemma half_succ n : natp n -> chalf (csucc (csucc n)) = csucc (chalf n).
Lemma cdouble_pow2 n: natp n -> cdouble(\2c ^c n) = \2c ^c (csucc n).
```

We study monotonicity of double and half.

```
Lemma double_inj a b: natp a -> natp b ->  cdouble a = cdouble b -> a = b.
Lemma double_monotone a b: natp a -> natp b ->
   (cdouble a <=c cdouble b <-> a <=c b).
Lemma double_monotone2 a b: natp a -> natp b ->
  (cdouble a <c cdouble b <-> a <c b).
Lemma double_monotone3 a b: natp a -> natp b ->
   (csucc (cdouble a) <c cdouble b <-> a <c b).
Lemma half_monotone n m: natp n -> natp m -> n <=c m ->
  chalf n <=c chalf m.
Lemma half_monotone2 n m: natp n -> natp m ->
  n <=c (chalf m) -> cdouble n <=c m.
```

```
Lemma double_le_odd1 p k: natp k -> natp p ->
  cdouble p <=c csucc (cdouble k) ->  p <=c k.
Lemma double_le_odd2 p k: natp k -> natp p ->
  csucc (cdouble k)  <=c cdouble p ->  csucc k <=c p.

Lemma cle_n_doublen n: natp n -> n <=c cdouble n.
Lemma cle_Sn_doublen n: natp n -> n <> \0c ->  csucc n <=c cdouble n.
Lemma clt_n_doublen n: natp n -> n <> \0c -> n <c cdouble n.
Lemma cle_halfn_n n: natp n -> chalf n <=c n.
Lemma cle_halfSn_n n: natp n ->  chalf (csucc n) <=c n.
Lemma double_nz n: natp n -> n <> \0c ->
    (cdouble n <> \0c /\ cdouble n <> \1c).
Lemma doubleS_nz n: natp n -> n <> \0c ->
    (csucc (cdouble n) <> \0c /\ csucc (cdouble n) <> \1c).
Lemma cltn_n2 n: natp n -> n <> \0c -> n <c cdouble n.
```

We deduce the following induction principle: in order for P to be true for all integers, it suffices that P(0) and P(1) are true, that P(h) implies P($2h$), and P(h),P($h+1$) imply P($2h+1$). We also show

$$\sum_{i=0}^{n} F_i = \sum_{i=0}^{n/2} F_{2i} + \sum_{i=0}^{(n-1)/2} F_{2i+1}$$

```
Lemma even_odd_dichot n (m := csucc n): natp n ->
  [\/ m = \1c,
      (m = cdouble (chalf m) /\ chalf m <=c n) |
      [/\  m = csucc (cdouble (chalf m)), (chalf m) <=c n &
          csucc (chalf m) <=c n]].
Lemma fusc_induction (p: property):
   p \0c -> p \1c -> (forall k, natp k -> p k -> p (cdouble k)) ->
   (forall k, natp k -> p k -> p (csucc k) -> p (csucc (cdouble k))) ->
  forall n, natp n -> p n.
Lemma split_sum_even_odd (F: fterm) n:
  natp n -> n <> \0c ->
  csumb (Nintc n) F =
  csumb (Nintc (chalf n)) (fun k => F (cdouble k))
  +c csumb (Nintc (chalf (cpred n))) (fun k => F (csucc (cdouble k))).
Lemma split_sum_even_odd_alt (F: fterm) n:
  natp n ->
  csumb (Nint n) F =
  csumb (Nint (chalf (csucc n))) (fun k => F (cdouble k))
  +c csumb (Nint (chalf n)) (fun k => F (csucc (cdouble k))).
```

Base two logarithm. We state here some properties of base two logarithm; this is the least integer $\ln(n)$ such that $n < 2^{\ln(n)}$. If n is non-zero, this integer is non-zero and characterized by

$$2^{\ln(n)-1} \leq n < 2^{\ln(n)}.$$

If k is the logarithm of n, then $k+1$ is the logarithm of $2n$ and $2n+1$, and 2^k is even (the case $n = 0$ is exceptional).

```
Definition clog2 n := least_ordinal (fun z => n <c \2c ^c z) n.

Lemma clog0 : clog2 \0c = \0c.
```

```
Lemma clog_nz n (m := clog2 n): natp n -> n <> \0c ->
   [/\ natp m, m <> \0c, \2c ^c (cpred m) <=c n & n <c \2c ^c m].
Lemma clog_pr n m: natp n -> natp m ->
    \2c ^c m <=c n -> n <c \2c ^c (csucc m) -> clog2 n = csucc m.
Lemma clog_double n: natp n -> n <> \0c -> clog2 (cdouble n) = csucc (clog2 n).
Lemma clog_succ_double n: natp n ->
    clog2 (csucc (cdouble n)) = csucc (clog2 n).
Lemma power2_log_even n: natp n ->  n <> \0c -> evenp (\2c ^c  (clog2 n)).
Lemma log2_pow n: natp n -> clog2 (\2c ^c n) = (csucc n).
Lemma NS_log n: natp n -> natp (clog2 n).
Lemma clog1 : clog2 \1c = \1c.
```

We consider here the operation that reverts the digits in base two of a number n. Assume $n = \sum_k a_k 2^k$. We consider $\bar{n} = \sum_k a_{p-k} 2^k$. Here $p+1$ is the number of terms in the expansion (so that $k \le p$). We assume the expansion normalized so that $a_p \ne 0$. As a result, $\bar{n}$ is odd (unless n is zero).

If $n = 2k$, then $\bar{n} = \bar{k}$, and if $n = 2k+1$ then $\bar{n} = 2^p + \bar{k}$; note that p is the base two logarithm of n. If we reverse twice the digits of n, we get n again (provided that n is odd). Thus: reverting three times is the same as reverting once (and this holds also for zero).

```
Definition base_two_reverse n :=
   let F := the_expansion \2c n in
   let p := cardinal (domain F) in
   expansion_value (Lg p (fun z => (Vg F (p -c (csucc z))))) \2c.

Lemma base2_expansion_prop n (F:= the_expansion \2c n)
    (p := cardinal (domain F)) : natp n -> n <> \0c ->
    [/\ expansion_of F \2c p n, p <> \0c &  Vg F (cpred p) = \1c].
Lemma log2_pr1 n (k := \2c ^c (cpred (clog2 n))) : natp n -> n <> \0c ->
  n = k +c (n %%c k).
Lemma log2_pr2 n : natp n ->
   clog2 n = cardinal (domain (the_expansion \2c n)).
Lemma base2r_zero:  base_two_reverse \0c = \0c.
Lemma base2r_one:  base_two_reverse \1c = \1c.
Lemma base2r_even n: natp n ->
  base_two_reverse (cdouble n) = base_two_reverse n.
Lemma base2r_odd n: natp n ->
  base_two_reverse(csucc (cdouble n)) = \2c ^c (clog2 n) +c base_two_reverse n.
Lemma NS_reverse n: natp n -> natp (base_two_reverse n).

Lemma base2r_oddp n: natp n -> n <> \0c -> oddp (base_two_reverse n).
Lemma base2r_oddK n (r := base_two_reverse) :
  oddp n -> r (r n) = n.
Lemma base2r_oddK_bis n (r := base_two_reverse) :
  natp n -> r (r (r n))  = r n.
```

Division by three. We state here some properties of division by three.

```
Lemma div3_props: [/\ \0c %%c \3c = \0c, \1c %%c \3c = \1c& \2c %%c \3c = \2c].
Lemma div3_props2: \3c %%c \3c = \0c /\ \4c %%c \3c = \1c.
Lemma div3_vals n (m := n %%c \3c): natp n ->
   [\/ m = \0c, m = \1c | m = \2c].
Lemma cmodmod n p: natp n -> natp p -> p <> \0c ->  n %%c p = n %c[mod p].
Lemma cmodmod2 n: natp n  -> n %%c \2c = n %c[mod \2c].
```

```
Lemma cmodmod3 n: natp n  ->  n %%c \3c = n %c[mod \3c].
Lemma double_mod3 n: natp n ->
  (n %%c \3c = \0c <-> (cdouble n) %%c \3c = \0c).
```

Fibonacci numbers. We first define by induction a function that returns a pair, denote by F_n the first term, and show that this function satisfies

$$F_0 = 0, F_1 = 1, \quad F_{n+2} = F_n + F_{n+1}, \tag{6.11}$$

We have $F_2 = 1$, and F_n is strictly increasing for $n \geq 2$.

```
Definition Fib2_rec :=
 induction_term  (fun _ v => (J (Q v) (P v +c Q v))) (J \0c \1c).
Definition Fib n := P (Fib2_rec n).

Lemma Fib_rec n : natp n -> Fib (csucc (csucc n)) = Fib n +c Fib (csucc n).
Lemma Fib0: Fib \0c = \0c.
Lemma Fib1: Fib \1c = \1c.
Lemma Fib2: Fib \2c = \1c.
Lemma Fib3: Fib \3c = \2c.
Lemma NS_Fib n: natp n -> natp (Fib n).
Lemma Fib_gt0 n: natp n -> n <> \0c ->  (Fib n) <> \0c.
Lemma Fib_smonotone n m: natp n -> natp m -> n <> \0c -> n <> \1c ->
  n <c m -> Fib n <c Fib m.
Lemma Fib_monotone n m: natp n -> natp m -> n <=c m -> Fib n <=c Fib m.
Lemma Fib_gt1 n: natp n -> \2c <c n -> \1c <c Fib n.
Lemma Fib_eq1 n: natp n -> (Fib n = \1c <-> (n = \1c \/ n = \2c)).
Lemma Fib_eq n m: natp n -> natp m ->
  (Fib n = Fib m <-> [\/ n = m, (n = \1c /\ m = \2c) |(n = \2c /\ m = \1c) ]).
```

We have

$$F_{n+m+1} = F_n F_m + F_{n+1} F_{m+1}$$

thus

$$F_{2n+1} = F_n^2 + F_{n+1}^2, \qquad F_{2n+2} = F_{n+1}(2F_n + F_{n+1}).$$

The second formula gives an efficient way of computing the Fibonacci numbers. From the first one we deduce: if F_{m+1} divides F_n, it divides F_{n+m+1}. By induction, if p divides n, then F_p divides F_n. So, if n and m are two integers, p is the greatest common divisor of n and m, then F_p divides F_n and F_m. We shall show later on that F_p is the greatest common divisor of F_n and F_m. Note that $F_n F_{m+1} - F_{n+1} F_m = (-1)^n F_{n-m}$. If $n = m+1$ or $n = m+2$ this is a Bezout relation between F_m and F_{m+1}.

In the special case $m = 2$, we get $F_{n+3} = F_n + 2F_{n+1}$, so that F_n and F_{n+3} are the same modulo two. So F_n is even if and only if n is a multiple of three.

```
Lemma Fib_add n m: natp n -> natp m ->
  Fib (csucc (n +c m)) =
  (Fib n) *c (Fib m) +c (Fib (csucc n)) *c (Fib (csucc m)).
Lemma Fib_add3  n: natp n ->
    Fib (n +c \3c) =  Fib n +c cdouble (Fib (csucc n)).
Lemma Fib_sub n m: natp n -> natp m -> m <=c n->
  Fib (n -c m) = Yo (evenp m)
    (Fib n *c Fib (csucc m) -c Fib (csucc n) *c Fib m)
    (Fib (csucc n) *c Fib m -c Fib n *c Fib (csucc m)).
```

```
Lemma Fib_odd n (square := fun a => a *c a): natp n ->
  Fib (csucc (cdouble n)) = square (Fib n) +c square (Fib (csucc n)).
Lemma Fib_even n: natp n ->
  Fib(cdouble (csucc n)) = Fib (csucc n) *c (cdouble (Fib n) +c Fib (csucc n)).
Lemma Fib_div n m: n %|c m -> Fib n %|c Fib m.
Lemma Fib_is_even_mod3 n: natp n ->
   (evenp (Fib n) <-> \3c %|c n).
```

We say that n is *composite* if it has a non-trivial divisor a (i.e., $1 < a < n$). This is the same as: there exists a and b such that $n = ab$, $1 < a$ and $1 < b$.

We have: if n is composite, $n \neq 4$, then F_n is composite. If a is a non-trivial divisor of n, then F_a divides F_n. Note that $F_a < F_n$, so that if $a \neq 2$, this is a non-trivial divisor. The formula for F_{2k} gives a factorisation in the case $a = 2$.

```
Definition composite n :=
  exists a, [/\ natp a, \1c <c a, a <c n & a %|c n].

Lemma composite_prod a b: natp a -> natp b -> \1c <c a -> \1c <c b ->
   composite (a *c b).
Lemma composite_prod_rev n: natp n -> composite n ->
   exists a b, [/\  natp a, natp b, \1c <c a, \1c <c b & n = a *c b].

Lemma composite_even_fib n: natp n -> \2c <c n ->
  composite (Fib (cdouble n)).
Lemma composite_fib n: natp n -> n  <> \4c->
  composite n -> composite (Fib n).
```

Coprimality. We say that x and y are coprime when only 1 divides x and y. We write this $x \perp y$. Note that $x \perp y$ is equivalent to $x \perp x+y$. We have $ux = 1+vy$ (unless $x = 0$, case where $y = 1$). This is called the Bezout relation. (proof by induction on $x+y$; a Bezout relation for x and y gives a Bezout relation for x and $x+y$).

```
Definition coprime a b :=
  [/\ natp a, natp b & forall x, x %|c a -> x %|c  b -> x = \1c].

Lemma coprime_S a b: coprime a b -> coprime b a.
Lemma cdivides_oner x: x %|c \1c -> x = \1c.
Lemma coprime_sump a b k: natp k ->  coprime a b ->
   coprime a (b +c (a *c k)).
Lemma coprime_sum a b: coprime a b ->  coprime a (b +c a).
Lemma coprime_diff a b: natp b -> coprime a (b +c a) -> coprime a b.

Lemma coprime_bezout1 a b: natp a -> natp b ->
  (exists u v, [/\ natp u, natp v & a *c u = \1c +c b *c v]) ->
  coprime a b.

Lemma coprime_bezout2 a b: coprime a b ->
  (a = \0c /\ b = \1c) \/
  exists u v, [/\ natp u, natp v & a *c u = \1c +c b *c v].
Lemma coprime3 a b c: coprime a b -> natp c -> a %|c (c *c b) -> a %|c c.
```

Application: if $p \perp q$ then

$$\frac{(p-1)(q-1)}{2} = \sum_{k=1}^{q-1} \lfloor \frac{kp}{q} \rfloor$$

Let f_k be the generic term of the sum. Since $f_0 = 0$, the sum is $\sum_{k<q} f_k$. We pretent that twice this quantity is $(p-1)(q-1)$. If $q = 0$, then $p = 1$, the result is obvious. If $q = 1$, the result is obvious as well. So, assume $q \geq 2$. The sum is $\sum_{i<q-1} f_{i+1}$, as well as $\sum_{i<q_1} f_{(q-2-i)+1}$. So twice the sum is $\sum_{i<q-1} f_{i+1} + f_{(q-2-i)+1}$. It suffices to show that eaacj term is $p-1$. Let $k = i+1$. Write $kp = \mathrm{A}q + r$ and $(q-k)p = \mathrm{B}q + s$ by euclidean division. The objective is $\mathrm{A}+\mathrm{B} = p-1$. but $qp = q(\mathrm{A}+\mathrm{B}) + (r+s)$ so $\mathrm{A}+\mathrm{B} = p-(r+s)/q$. Note that $(r+s)/q < 2$ since $r < q$ and $s < q$. Now $(r+s)/q = 0$ says $r = s = 0$ so $kp = \mathrm{A}q$; this relation says q divides k but $0 < k < q$ absurd. So $(r+s)/q = 1$, $\mathrm{A}+\mathrm{B}-1$.

```
Lemma coprime_example p q: coprime p q ->
   (p -c \1c) *c (q -c \1c) = cdouble(csumb q (fun k => (k *c p) %/c q)).
```

6.8 Combinatorial analysis

The shepherd principle. Let E and F be two sets, $f : \mathrm{E} \to \mathrm{F}$ a function. For each i, we consider E_i, the inverse image of i by f. Let c_i be the cardinal of E_i, and a the cardinal of E. Since the E_i are mutually disjoint, we get $a = \sum c_i$. Assume $c_i = c$ for all i, and let b be the cardinal of F (the number of terms in the sum). It follows $a = bc$.

This is known in French as the shepherd's principle, and Bourbaki states it in Proposition 9 [4, p. 179] as: If f is a function from a set with cardinal $\mathfrak{a}$ onto a set with cardinal $\mathfrak{b}$, and if all sets $f^{-1}\langle\{x\}\rangle$ have the same cardinal $\mathfrak{c}$, then $\mathfrak{a} = \mathfrak{b}\mathfrak{c}$. Surjectivity of f is superfluous.

```
Theorem shepherd_principle f c: function f ->
  (forall x, inc x (target f) -> cardinal (Vfi1 f x) = c) ->
  cardinal (source f) = (cardinal (target f)) *c c.
```

Factorial. Bourbaki defines the *factorial* of n, denoted by $n!$, as $\prod_{i<n}(i+1)$. It satisfies $0! = 1$ and $(n+1)! = n!(n+1)$. There is a unique function satisfying this property.

```
Definition factorial n := cprodb n csucc.

Lemma factorial_succ n: natp n ->
  factorial (csucc n) = (factorial n) *c (csucc n).
Lemma CS_factorial n: cardinalp (factorial n).
Lemma factorial0: factorial \0c = \1c.
Lemma factorial1: factorial \1c = \1c.
Lemma factorial2: factorial \2c = \2c.
Lemma factorial_nz n: natp n -> factorial n <> \0c.
Lemma NS_factorial n: natp n -> natp (factorial n).
Lemma factorial_prop f: f \0c = \1c ->
  (forall n, natp n -> f (csucc n) =  (f n) *c (csucc n)) ->
  forall x, natp x -> f x = factorial x.
```

We show how the factorial function could have been defined by induction. By induction if $a \leq b$, $a!$ divides $b!$; so that $(b-a)!$ divides $b!$.

```
Lemma factorial_induction n: natp n ->
 factorial n = induction_term  (fun a b=> b *c (csucc a)) \1c n.
Lemma quotient_of_factorials a b:
```

```
  natp a -> natp b -> b <=c a ->
  (factorial b) %|c (factorial a).
Lemma quotient_of_factorials1 a b:
  natp a -> natp b -> b <=c a ->
  (factorial (a -c b))b %|c (factorial a).
Lemma factorial_monotone a b: natp b -> a <=c b ->
     factorial a <=c factorial b.
```

Number of injections. Proposition 10 [4, p. 170] says that the number of injections from a set with m elements to a set with n elements is $\mathrm{A}_{nm} = n!/(n-m)!$. Note that, if such an injection exists, we have $m \le n$; otherwise the number is zero.

We first show that if $m \le n$, then $\mathrm{A}_{nm}(n-m)! = n!$, so that $\mathrm{A}_{nm}(n-m) = \mathrm{A}_{n,m+1}$.

```
Definition number_of_injections b a :=
 (factorial a) %/c (factorial (a -c b)).

Lemma number_of_injections_pr a b:
  natp a -> natp b -> b <=c a ->
  (number_of_injections b a) *c (factorial (a -c b)) = factorial a.
Lemma number_of_injections_base a: natp a ->
  number_of_injections \0c a = \1c.
Lemma number_of_injections_rec a b:
  natp a -> natp b -> b <c a ->
  (number_of_injections b a) *c (a -c b) =
  number_of_injections (csucc b) a.
Lemma NS_number_of_injections a b:
  natp (number_of_injections b a).
```

The proof of the proposition is by induction on m. The case $m = 0$ is obvious. Let A be a set with m elements, $a \not\in \mathrm{A}$, so that $\mathrm{A}' = \mathrm{A} \cup \{a\}$ has $m+1$ elements. Let F a set with n elements. Let G and G′ be the sets of injections $\mathrm{A} \to \mathrm{F}$ and $\mathrm{A}' \to \mathrm{F}$. If $f' \in \mathrm{G}'$, its restriction to A is injective, thus in G. Conversely, $f \in \mathrm{G}$ can be extended into a element of G′ by assigning to a any value not in the range. There are $n-m$ possibilities. The shepherd principle, applied to the restriction, considered as function $\mathrm{G}' \to \mathrm{G}$, says that the cardinal of G′ is the cardinal of G times $n-m$. It suffice to apply the induction formula for A_{nm}.

```
Definition injections  E F :=
  Zo (set_of_functions E F)(injection).

Lemma number_of_injections_prop E F:
  finite_set F ->
  cardinal E <=c cardinal F ->
  cardinal (injections E F)  =
    number_of_injections (cardinal E) (cardinal F). (* 98 *)
```

We consider here some properties of permutations (the set of permutations of E is a group).

```
Lemma inverse_bij_bp g E E':
  (bijection_prop g E E') ->  bijection_prop (inverse_fun g) E' E.
Lemma permutation_Si E x:
   inc x (permutations E) -> inc (inverse_fun x) (permutations E).
```

```
Lemma permutation_id E: inc (identity E) (permutations E).
Lemma permutations_set0:
  (permutations emptyset) = singleton (identity emptyset).
Lemma permutation_Si E x:
   inc x (permutations E) -> inc (inverse_fun x) (permutations E).
Lemma permutation_Sc E x y:
   inc x (permutations E) -> inc y (permutations E) ->
   inc (x \co y) (permutations E).
Lemma permutation_coP E x y:
   inc x (permutations E) -> inc y (permutations E) ->
   (x \coP y).
Lemma permutation_A E x y z:
  inc x (permutations E) -> inc y (permutations E) -> inc z (permutations E) ->
  x \co (y \co z) = (x \co y) \co z.
Lemma permutation_lK E x y:
  inc x (permutations E) -> inc y (permutations E) ->
  (inverse_fun x) \co (x \co y) = y.
Lemma permutation_lK' E x y:
  inc x (permutations E) -> inc y (permutations E) ->
  x \co ( (inverse_fun x) \co y) = y.
Lemma permutation_rK E x y:
  inc x (permutations E) -> inc y (permutations E) ->
  (x \co y) \co (inverse_fun y) = x.
Lemma permutation_if_inj E f: finite_set E -> function_prop f E E ->
  injection f -> inc f (permutations E).
Lemma permutations_finite E: finite_set E ->
   permutations E = injections E E.
```

An injection from E into itself is a bijection when E is finite. Since $A_{nn} = n!$, we deduce that $n!$ is the number of permutations of E.

```
Lemma number_of_permutations E: finite_set E ->
  cardinal (permutations E)  = (factorial (cardinal E)).
```

Permutations of I_n**.** We consider some properties of $\mathfrak{S}_n$, the set of permutations of the intervall$_n$, where n is a natural number. Recall that $I_n = n$.

```
Lemma perm_int_inj n f: natp n -> inc f (permutations n) ->
    (forall x y, x <c n -> y <c n -> Vf f x = Vf f y -> x = y).
Lemma perm_int_surj n f: natp n -> inc f (permutations n) ->
   forall y, y <c n -> exists2 x, x <c n & Vf f x = y.
```

A special permutation is a the transposition (i, j): it is defined by $f(i) = j$, $f(j) = i$ and otherwise $f(k) = k$. In any case $f(f(k)) = k$. If we extend a permutation of I_n by defining $f(n) = n$, we get a permutation of I_{n+1}. The two functions $i \mapsto i+1$ and $i \mapsto i-1$ are permutations of I_n; one being the inverse of the other; provide that we compute module n/

If E is a subset of I_n with k elements, there is $f \in \mathfrak{S}_n$ such that E is the image of I_k. Proof by induction on n or k.

```
Lemma transposition_prop n i j
   (f:=Lf (fun z => Yo (z = i) j (Yo (z = j) i z)) n n):
    natp n -> inc i n -> inc j n ->
   [/\ inc f (permutations n), Vf f i = j, Vf f j = i,
```

```
     forall k, inc k n -> k <> i -> k <> j -> (Vf f k)  = k &
     forall k, inc k n -> Vf f (Vf f k) = k].
Lemma extension_perm n s (es := extension s n n):
  natp n -> inc s (permutations n) ->
  [/\ inc es (permutations (csucc n)), Vf es n = n &
    forall i, inc i n -> Vf es i = Vf s i].
Lemma rotation_prop n (m := csucc n)
   (f := Lf (fun i => Yo (i = \0c) n (cpred i)) m m)
   (g := Lf (fun i => Yo (i = n) \0c (csucc i)) m m):
  natp n ->
  [/\ inc f (permutations m), inc g (permutations m),  inverse_fun f = g &
     [/\ Vf f \0c = n, forall i, i <c n -> Vf f (csucc i) = i,
         forall i, i <=c n -> i <> \0c -> Vf f i = (cpred i),
      Vf g n = \0c &  forall i, i <c n -> Vf g i = (csucc i)  ] ].
Lemma partial_rotation n f (k := Vf (inverse_fun f) n)
      (g:= fun i => Yo (i <c k) (Vf f i) (Vf f (csucc i)))
      (G := Lf g n n):
  natp n -> inc f (permutations (csucc n)) ->
  [/\ k <=c n, Vf f k = n, lf_axiom g n n & inc G (permutations n)].

Lemma permutation_exists1 n i:
    natp n -> i <c n ->
    exists2 f, inc f (permutations n) & Vf f \0c = i.
Lemma permutation_exists2 E n: natp n -> sub E n -> (* 78 *)
   exists2 f,  inc f (permutations n) & E = Vfs f (cardinal E).
```

Enumeration of a finite set. Let E be a totally ordered, finite set, of cardinal n. Then E is order isomorphic to I_n. The proof is by induction on n. Let x be the greatest element of E and $E' = E - \{x\}$. By induction E' is order isomorphic to I_{n-1}. It suffices to extent this isomorphic by assigning to $n-1$ the value x. The isomorphism is obviously unique as I_n is well-ordered . **Note:** This was already proved, since E is well-ordered.

```
(*
  Lemma finset_enum_exists r: (* 66 *)
    total_order r -> finite_set (substrate r) ->
    exists f, order_isomorphism f (Nint_co (cardinal (substrate r))) r.
  Lemma finset_enum_unique r f f':
    order_isomorphism f (Nint_co (cardinal (substrate r))) r ->
    order_isomorphism f' (Nint_co (cardinal (substrate r))) r ->
    f = f'.
*)
```

We give here an alternate proof in the case where E is a subset of the integers, with the natural order.

Let K be a finite subset of **N** of cardinal q. There is a unique strictly increasing function that maps I_q onto K. We first define

$$E_K(n+1) = E_K(n) \cup \{\bigcap(K - E_K(n))\}, \qquad E_K(0) = \emptyset.$$

Since $K - E_K(n)$ is a set of integers, the intersection is the least element of this set (or zero, when the set is empty). It follows (when $n \le q$) that $S = E_K(n)$ is an initial segment of K (elements of S are smaller than other elements), of cardinal n. In particular, $E_K(q) = K$.

```
Definition nth_more K S :=  S +s1 intersection (K -s S).
```

```
Definition nth_elts K := induction_term (fun _ S => nth_more K S) emptyset.
Definition segment_nat K  S:=
  sub S K /\  (forall i j, inc i S -> inc j (K -s S) -> i <c j).

Lemma nth_set0 x (y := intersection x) : x = emptyset -> y = \0c.
Lemma nth_set2 K S: sub K S ->  nth_more K S = S +s1 \0c.
Lemma nth_set3 K: nth_more K K = K +s1 \0c.
Lemma nth_set4 K S (S':= nth_more K S) (x:= intersection (K -s S)):
   sub K Nat -> segment_nat K S -> S <> K ->
   [/\ segment_nat K S', inc x (S' -s S) & cardinal S' = csucc (cardinal S)].
Lemma nth_set5 K n (S:= nth_elts K n):
   natp n -> sub K Nat -> n <=c cardinal K ->
   (segment_nat K S /\ cardinal S = n).
Lemma nth_set6 K (n:= cardinal K):
  natp n -> sub K Nat -> (nth_elts K n) = K.
Lemma nth_set_M K n m:
  natp n -> m <=c n -> sub (nth_elts K m) (nth_elts K n).
```

We define

$$e_K(n) = \bigcup (E_K(n+1) - E_K(n)).$$

Assume $n < q$. Then $E_K(n+1)$ is the union of $E_K(n)$ and a singleton $\{x\}$. We have $e_K(n) = x$, so that $e_K(n)$ is the least element of $K - E_K(n)$, and greater than all elements of $E_K(n)$. Thus, e_K is a strictly increasing bijection $I_q \to K$. Note: let f be a strictly increasing function, define on I_q with values in $\mathbf{N}$, and K its target;

```
Definition nth_elt K n := union (nth_elts K (csucc n) -s nth_elts K n).

Lemma nth_set7 K n (S:= (nth_elts K n)) (x:= nth_elt K n) :
   natp n -> sub K Nat -> n <c cardinal K ->
   [/\ inc x (K -s S), inc x (nth_elts K (csucc n)),
       forall y, inc y (K -s S) -> x <=c y
     & forall y, inc y S -> y <c x].
Lemma nth_elt_inK K n:
   natp n -> sub K Nat -> n <c cardinal K ->
   inc (nth_elt K n) K.
Lemma nth_elt_ax K: sub K Nat -> natp (cardinal K) ->
   lf_axiom (nth_elt K) (cardinal K) K.
Lemma nth_elt_monotone K n m:
   natp n -> sub K Nat -> n <c cardinal K ->
   m <c n -> (nth_elt K m) <c (nth_elt K n).
Lemma nth_elt_bf K (f := Lf (nth_elt K) (cardinal K) K):
  sub K Nat -> natp (cardinal K) ->
  (bijection (nth_set_fct K) /\
  forall i j,  inc i (source f) -> inc j (source f) -> i <c j ->
       Vf f i <c Vf f j).
Lemma nth_elt_surj K a:  sub K Nat -> natp (cardinal K) -> inc a K ->
  exists2 n, n <c (cardinal K) & a = (nth_elt K n).
Lemma nth_elt_exten k f: natp k ->
   (forall i, i <c k  -> natp (f i)) ->
   (forall i j, i <c j -> j <c k  -> f i <c f j) ->
   (forall i, i<c k -> (nth_elt (fun_image k f) i = f i)). (* 61 *)
```

We consider here more properties of e_K. Assume $K \subset n$, and let K' be the complement. Consider the function that maps i to $e_k(i)$ for $i < k$ and $e_{K'}(i-k)$ otherwise, where $k =$

card($'$K). This is a permutation of I_n It agrees with e_K on I_K. If we add n as a greatest element to K$*$ we get a permutation σ of I_{n+1}, such that $\sigma(k) = n$, σ restricted to I_k is strictly increasing, and its range is K.

Let's state some properties of e_K: if n is an integer, $K \subset I_n$ has q elements, if K is non-empty and σ is a permutation of I_q, then $e_K(\sigma(0)) \in K$; conversely, if $a \in K$ there is a permutation σ such that $a = e_K(\sigma(0))$.

```
Definition nth_elt_dbl K n :=
  fun i => Yo (i <c (cardinal K)) (nth_elt K i)
              (nth_elt (n -s K) (i -c (cardinal K))).

Lemma nth_elt_dbl_prop K n: natp n -> inc K (\Po n) ->
  lf_axiom (nth_elt_dbl K n) n n
  /\ inc (Lf (nth_elt_dbl K n) n n) (permutations n).

Lemma nth_elt_dbl_prop_bis n K (k := cardinal K)
  (s:= (Lf (nth_elt_dbl (K +s1 n) (csucc n)) (csucc n) (csucc  n))):
 natp n -> inc K (\Po n) ->
     [/\ inc s (permutations (csucc n)),
    (forall x, inc x K -> exists2 i, i <c k & x = (Vf s i)),
    (forall i, i <c k -> inc (Vf s i) K),
    Vf s k = n &
    (forall i j, i<c j -> j <c k -> (Vf s i) <c (Vf s j))].

Lemma nth_elt_prop7 K n s: natp n -> inc K (\Po n) ->
   inc s (permutations (cardinal K)) -> nonempty K ->
   inc (nth_elt K (Vf s \0c)) K.

Lemma nth_elt_prop8 K n a: natp n -> inc K (\Po n) -> inc a K ->
  exists2 f, inc f (permutations (cardinal K)) & a = nth_elt K (Vf f \0c).
```

Consider now a subset K of I_n with k elements and its enumeration r_K. If x is a strict upper bound of K, $K' = K \cup \{x\}$, then e_K and $e_{K'}$ agree on I_k. One deduces, by induction, that there exists a permutation of I_n that agrees on I_k with e_K. We can express this as: there is a permutation σ of I_n satisfying (h), namely, that σ is strictly increasing on I_k and the image of I_k is K. We deduce that there is σ satisfying (H), namely, it is a permutation of I_{n+1}, satisfies (h) and moreover $\sigma(k) = n$.

Number of partitions of a set. Proposition 11 [4, p. 180] says "let E be a finite set with n elements, and let $(p_i)_{1\le i\le h}$ be a finite sequence of integers such that $\sum_{i=1}^{h} p_i = n$. Then the number of coverings $(X_i)_{1\le i\le h}$ of E by mutually disjoints sets X_i such that $\mathrm{card}(X_i) = p_i$ for $1 \le i \le h$ is equal to $n!/(\prod_{i=1}^{h} p_i!)$."

Comments. The expression "number of coverings by mutually disjoint sets" is a bit too long, and we simplify it as "partition". Normally, a partition consists in non-empty sets, i.e., $p_i \neq 0$. Here $p_i = 0$ is allowed. In what follows, we shall consider a set I, a sequence of integers $(p_i)_{i\in I}$ such that $\sum p_i = n$, families $(X_i)_{i\in I}$ such that $\mathrm{card}(X_i) = p_i$ for $i \in I$. We shall prove that the number of such families is $n!/(\prod_i p_i!)$.

We shall denote by p the family $(p_i)_{i\in I}$, by I its domain, and by C_{pE} the set of all functional graphs $I \to \mathfrak{P}(E)$, satisfying the cardinality condition. The first result is that this set is non-empty if the cardinal of E is $\sum p_i$.

```
Definition partition_with_pi_elements p E f :=
```

```
  [/\ domain f = domain p,
  (forall i, inc i (domain p) -> cardinal (Vg f i) = Vg p i) &
    partition_w_fam f E].
Definition partitions_pi p E :=
  Zo (gfunctions (domain p) (\Po E)) (partition_with_pi_elements p E).
Lemma partitions_piP p E f:
  inc f (partitions_pi p E) <-> partition_with_pi_elements p E f.
Lemma fif_cardinal i p:
  finite_int_fam p -> inc i (domain p) -> cardinalp (Vg p i).
Lemma pip_prop0 p E f: partition_with_pi_elements p E f ->
  forall i, inc i (domain f) ->  sub (Vg f i) E.

Lemma number_of_partitions1 p E:
  finite_int_fam p -> csum p = cardinal E ->
  nonempty (partitions_pi p E).
```

The proof of the proposition is by application of the shepherd principle to a function $\Phi : \mathrm{Q(E)} \to \mathrm{C}_{p\mathrm{E}}$, where Q(E) denotes the set of permutations of E (we know that its cardinal is $n!$). Fix $f \in \mathrm{C}_{p\mathrm{E}}$ (we know that it exists). Let $g \in \mathrm{Q(E)}$. We can consider the sets h_i of all $g(x)$ for $x \in f_i$. Since g is injective, f_i and g_i have the same cardinal. The family of these h_i is in $\mathrm{C}_{p\mathrm{E}}$, and will be denoted by $\Phi(g)$. Note that Φ is surjective.

```
Definition partitions_aux f g:=
  Lg (domain f) (fun i => Vfs g (Vg f i)).

Lemma number_of_partitions3 p E f g:
  partition_with_pi_elements p E f -> inc g  (permutations E) ->
  inc (partitions_aux f g) (partitions_pi p E).

Lemma number_of_partitions4 p E f:
  finite_int_fam p -> csum p = cardinal E ->
  partition_with_pi_elements p E f ->
  surjection (Lf (partitions_aux f)
    (permutations E) (partitions_pi p E)). (* 58 *)
```

This function Φ is not injective: $\Phi(g) = \Phi(h)$ if and only if $h^{-1} \circ g$ is a bijection that leaves each f_i invariant.

```
Lemma number_of_partitions5P p E f g h:
  finite_int_fam p -> csum p = cardinal E ->
  partition_with_pi_elements p E f ->
  inc h (permutations E) -> inc g (permutations E) ->
  ((partitions_aux p E f g = partitions_aux p E f h) <->
  (forall i, inc i (domain p) ->
      Vfs ((inverse_fun h) \co g) (Vf f i) = (Vf f i))).
```

Assume that $h^{-1} \circ g$ is a bijection that leaves each f_i invariant. Let w_i be the restriction of this function to f_i. It is a permutation of f_i, so is in $\mathrm{Q}(f_i)$. Let $\Psi(h)$ be the family of all w_i. It is an element of $\prod \mathrm{Q}(f_i)$. Since f_i is a covering, this function is injective. Since the f_i are mutually disjoint, this function is surjective (the proof is long, but the arguments are trivial).

```
Lemma number_of_partitions6 p E f h:
  finite_int_fam p -> csum p = cardinal E ->
```

```
  partition_with_pi_elements p E f ->
  inc h (permutations E) ->
  lf_axiom (fun g=> Lg (domain p)(fun i=>  (restriction2
    ((inverse_fun h) \co g)
    (Vg f i) (Vg f i))))
  (Zo  (permutations E)
    (fun g => (partitions_aux f g = partitions_aux f h)))
  (productb (Lg (domain p)(fun i=> (permutations (Vg f i))))).
Lemma number_of_partitions7 p E f h: (* 124 *)
  finite_int_fam p -> csum p = cardinal E ->
  partition_with_pi_elements p E f ->
  inc h (permutations E) ->
  bijection(Lf (fun g=> Lg (domain p)(fun i=> (restriction2
    ((inverse_fun h) \co g)
    (Vg f i) (Vg f i))))
  (Zo  (permutations E)
    (fun g => (partitions_aux f g = partitions_aux f h)))
  (productb (Lg (domain p)(fun i=> (permutations (Vg f i)))))).
```

The number of functions h that take the same value as g under Φ is the cardinal of the image of Ψ. This depends only on f, and in fact is $b = \prod_{i=1}^{h} p_i!$. If $a = n!$ is the cardinal of the source of Φ, the shepherd principle says that the cardinal of C_{pE} times b is a. This can be weakened to $\mathrm{card}(C_{pE}) = a/b$.

```
Theorem number_of_partitions p E
   (num:= factorial (cardinal E))
   (den := cprodb (domain p) (fun z => factorial (Vg p z))):
  finite_int_fam p -> csum p = cardinal E ->
        [/\ num = cardinal (partitions_pi p E) *c den,
        natp num, natp den, den <> \0c &
        finite_set (partitions_pi p E)].
Theorem number_of_partitions_bis p E:
 finite_int_fam p -> csum p = cardinal E ->
  cardinal (partitions_pi p E) =
  (factorial (cardinal E)) %/c
          (cprodb (domain p) (fun z => factorial (Vg p z))).
```

We consider here the special case where the family has two elements m and p, so that $n = m + p$.

```
Lemma number_of_partitions_p2 E m p
   (num := factorial (m +c p))
   (den := (factorial m) *c (factorial p))
   (x := cardinal (partitions_pi (variantLc m p) E)):
  natp m -> natp p -> cardinal E = (m +c p) ->
   [/\ natp x, num = x *c den,natp num, natp den & den <> \0c].
```

The binomial coefficient. Bourbaki defines the *binomial coefficient* $b_{np} = \binom{n}{p}$ as the number of subsets of p elements in a set of n elements, after showing that

$$b_{n,p} = \frac{n!}{p!(n-p)!} \text{ if } p \le n, \qquad b_{n,p} = 0 \text{ otherwise.} \tag{6.12}$$

He shows in Proposition 13 [4, p. 181]) that it satisfies the following relation.

$$\binom{n}{0} = 1, \quad \binom{0}{p+1} = 0, \quad \binom{n+1}{p+1} = \binom{n}{p+1} + \binom{n}{p}. \tag{6.13}$$

The Coq standard library defines the binomial coefficient via (6.12), then shows (6.13). Our initial implementation was by induction in `nat`, and we used equivalence between `nat` and the Bourbaki integers. The SSREFLECT library defines the binomial coefficient via (6.13) as follows.

```
Fixpoint binomial_rec n m :=
  match n, m with
  | n'.+1, m'.+1 => binomial_rec n' m + binomial_rec n' m'
  | _, 0 => 1
  | 0, _.+1 => 0
  end.

Definition binomial := nosimpl binomial_rec.
Notation "''C' ( n , m )" := (binomial n m)
  (at level 8, format "''C' ( n ,  m )") : nat_scope.
```

We show here the power of the SSREFLECT library by giving the proof of the following relation

$$\sum_{i=0}^{n} \binom{n}{i} a^{n-i} b^{i} = (a+b)^{n}, \tag{6.14}$$

first on `nat`, then in any ring (under the assumption $ab = ba$).

```
Theorem Pascal a b n :
  (a + b) ^ n = \sum_(i < n.+1) 'C(n, i) * (a ^ (n - i) * b ^ i).
Proof.
elim: n => [|n IHn]; rewrite big_ord_recl muln1 ?big_ord0 //.
rewrite expnS {}IHn /= mulnDl !big_distrr /= big_ord_recl muln1 subn0.
rewrite !big_ord_recr /= !binn !subnn bin0 !subn0 !mul1n -!expnS -addnA.
congr (_ + _); rewrite addnA -big_split /=; congr (_ + _).
apply: eq_bigr => i _; rewrite mulnCA (mulnA a) -expnS subnSK //.
by rewrite (mulnC b) -2!mulnA -expnSr -mulnDl.
Qed.


Lemma exprDn_comm x y n (cxy : comm x y) :
  (x + y) ^+ n = \sum_(i < n.+1) (x ^+ (n - i) * y ^+ i) *+ 'C(n, i).
Proof.
elim: n => [|n IHn]; rewrite big_ord_recl mulr1 ?big_ord0 ?addr0 //=.
rewrite exprS {}IHn /= mulrDl !big_distrr /= big_ord_recl mulr1 subn0.
rewrite !big_ord_recr /= !binn !subnn !mul1r !subn0 bin0 !exprS -addrA.
congr (_ + _); rewrite addrA -big_split /=; congr (_ + _).
apply: eq_bigr => i _; rewrite !mulrnAr !mulrA -exprS -subSn ?(valP i) //.
by rewrite  subSS (commrX _ (commr_sym cxy)) -mulrA -exprS -mulrnDr.
Qed.
```

We show here the proof of the Coq standard library; here x and y are real numbers, as well as the binomial coefficient. `pascal` is relation (6.13), `tech5` is the equivalent of `big_ord_recr`, `decomp_sum` is the equivalent of `big_ord_recl`, `plus_sum` is the equivalent of `big_split`, `scal_sum` is the equivalent of `big_distrr`. Instead of using the lemmas `bin0` or `binn`, that say $b_{nn} = b_{n0} = 1$, the proof uses relation (6.12) as well as $n! \neq 0$.

```
Lemma binomial:
  forall (x y:R) (n:nat),
    (x + y) ^ n = sum_f_R0 (fun i:nat => C n i * x ^ i * y ^ (n - i)) n.
```

```
Proof.
  intros; induction  n as [| n Hrecn].
  unfold C; simpl; unfold Rdiv;
    repeat rewrite Rmult_1_r; rewrite Rinv_1; ring.
  pattern (S n) at 1; replace (S n) with (n + 1)%nat; [ idtac | ring ].
  rewrite pow_add; rewrite Hrecn.
  replace ((x + y) ^ 1) with (x + y); [ idtac | simpl; ring ].
  rewrite tech5.
  cut (forall p:nat, C p p = 1).
  cut (forall p:nat, C p 0 = 1).
  intros; rewrite H0; rewrite <- minus_n_n; rewrite Rmult_1_l.
  replace (y ^ 0) with 1; [ rewrite Rmult_1_r | simpl; reflexivity ].
  induction  n as [| n Hrecn0].
  simpl; do 2 rewrite H; ring.

  (* N >= 1 *)
  set (N := S n).
  rewrite Rmult_plus_distr_l.
  replace (sum_f_R0 (fun i:nat => C N i * x ^ i * y ^ (N - i)) N * x) with
    (sum_f_R0 (fun i:nat => C N i * x ^ S i * y ^ (N - i)) N).
  replace (sum_f_R0 (fun i:nat => C N i * x ^ i * y ^ (N - i)) N * y) with
    (sum_f_R0 (fun i:nat => C N i * x ^ i * y ^ (S N - i)) N).
  rewrite (decomp_sum (fun i:nat => C (S N) i * x ^ i * y ^ (S N - i)) N).
  rewrite H; replace (x ^ 0) with 1; [ idtac | reflexivity ].
  do 2 rewrite Rmult_1_l.
  replace (S N - 0)%nat with (S N); [ idtac | reflexivity ].
  set (An := fun i:nat => C N i * x ^ S i * y ^ (N - i)).
  set (Bn := fun i:nat => C N (S i) * x ^ S i * y ^ (N - i)).
  replace (pred N) with n.
  replace (sum_f_R0 (fun i:nat => C (S N) (S i) * x ^ S i * y ^ (S N - S i)) n)
    with (sum_f_R0 (fun i:nat => An i + Bn i) n).
  rewrite plus_sum.
  replace (x ^ S N) with (An (S n)).
  rewrite (Rplus_comm (sum_f_R0 An n)).
  repeat rewrite Rplus_assoc.
  rewrite <- tech5.
  fold N.
  set (Cn := fun i:nat => C N i * x ^ i * y ^ (S N - i)).
  cut (forall i:nat, (i < N)%nat -> Cn (S i) = Bn i).
  intro; replace (sum_f_R0 Bn n) with (sum_f_R0 (fun i:nat => Cn (S i)) n).
  replace (y ^ S N) with (Cn 0%nat).
  rewrite <- Rplus_assoc; rewrite (decomp_sum Cn N).
  replace (pred N) with n.
  ring.
  unfold N; simpl; reflexivity.
  unfold N; apply lt_O_Sn.
  unfold Cn; rewrite H; simpl; ring.
  apply sum_eq.
  intros; apply H1.
  unfold N; apply le_lt_trans with n; [ assumption | apply lt_n_Sn ].
  intros; unfold Bn, Cn.
  replace (S N - S i)%nat with (N - i)%nat; reflexivity.
  unfold An; fold N; rewrite <- minus_n_n; rewrite H0;
    simpl; ring.
  apply sum_eq.
  intros; unfold An, Bn; replace (S N - S i)%nat with (N - i)%nat;
    [ idtac | reflexivity ].
```

```
  rewrite <- pascal;
    [ ring
      | apply le_lt_trans with n; [ assumption | unfold N; apply lt_n_Sn ] ].
  unfold N; reflexivity.
 unfold N; apply lt_O_Sn.
  rewrite <- (Rmult_comm y); rewrite scal_sum; apply sum_eq.
  intros; replace (S N - i)%nat with (S (N - i)).
  replace (S (N - i)) with (N - i + 1)%nat; [ idtac | ring ].
  rewrite pow_add; replace (y ^ 1) with y; [ idtac | simpl; ring ];
    ring.
  apply minus_Sn_m; assumption.
  rewrite <- (Rmult_comm x); rewrite scal_sum; apply sum_eq.
  intros; replace (S i) with (i + 1)%nat; [ idtac | ring ]; rewrite pow_add;
    replace (x ^ 1) with x; [ idtac | simpl; ring ];
      ring.
  intro; unfold C.
  replace (INR (fact 0)) with 1; [ idtac | reflexivity ].
  replace (p - 0)%nat with p; [ idtac | apply minus_n_O ].
  rewrite Rmult_1_l; unfold Rdiv; rewrite <- Rinv_r_sym;
    [ reflexivity | apply INR_fact_neq_0 ].
  intro; unfold C.
  replace (p - p)%nat with 0%nat; [ idtac | apply minus_n_n ].
  replace (INR (fact 0)) with 1; [ idtac | reflexivity ].
  rewrite Rmult_1_r; unfold Rdiv; rewrite <- Rinv_r_sym;
    [ reflexivity | apply INR_fact_neq_0 ].
```

We define here a function c_{np} by induction on n; the definition is a bit obscure since we define by induction an auxiliary term f_n, where f_n is a functional graph on **N**, then say $c_{np} = f_n(p)$.

```
Definition binom n m :=
  Vg (induction_term
      (fun _ T: Set => Lg Nat (fun z => variant \0c \1c
          (Vg T z +c Vg T (cpred z)) z))
       (Lg Nat (variant \0c \1c \0c))
       n) m.
Lemma binom00: binom \0c \0c = \1c.
Lemma binom0Sm m: natp m -> binom \0c (csucc m) = \0c.
Lemma binomSn0 n: natp n -> binom (csucc n) \0c = \1c.
Lemma binomSnSm n m: natp n -> natp m  ->
  binom (csucc n) (csucc m) = (binom n (csucc m)) +c (binom n m).
Lemma NS_binom n m: natp n -> natp m  -> natp (binom n m).
```

Let $f(n,p)$ be the product of c_{np} by $p!(n-p)!$. Note that if $p > n$, then $n-p$ is zero by convention and $(n-p)!$ is one. We have then $(n-p)! = \text{if}(p < n, n-p, 1) \cdot (n-(p+1))!$. This gives us an induction property for f, from which we deduce $f(n,p) = \text{if}(p \le n, n!, 0)$. We restate this as: if $p > n$ the binomial coefficient is zero, else $f(n,p) = n!$. This formula shows that $p!(n-p)!$ divides $n!$.

```
Lemma binom_alt_pr n m: natp n -> natp m  ->  (* 84 *)
  (binom n m) *c  (factorial m) *c (factorial (n -c m)) =
  Yo (m <=c n) (factorial n) \0c.
Lemma binom_bad n m: natp n -> natp m ->
  n <c m -> binom n m = \0c.
Lemma binom_good n m: natp n -> natp m ->
```

```
  m <=c n ->
  (binom n m) *c  (factorial m) *c (factorial (n -c m)) = (factorial n).
```

We then have

$$\binom{n}{p} = \binom{n}{n-p} = \frac{n!}{p!(n-p)!} \quad \text{when } p \le n. \tag{6.15}$$

```
Lemma binom_pr0 n p
   (num := factorial n)
   (den:=  (factorial p) *c (factorial (n -c p))):
   natp n -> natp p -> p <=c n ->
   den %|c num /\ binom n p = num %/c den.

Lemma binom_pr1 n p: natp n -> natp p ->
  p <=c n ->
  binom n p = (factorial n) %/c ((factorial p) *c (factorial (n -c p))).

Lemma binom_symmetric n p: natp n ->
    p <=c n -> binom n p = binom n (n -c p).
Lemma binom_symmetric2 n m: natp n -> natp m ->
    binom (n +c m) m = binom (n +c m) n.
```

We show here

$$\binom{n}{0} = 1, \qquad \binom{n}{1} = n, \qquad \binom{n+1}{2} = \frac{n(n+1)}{2}. \tag{6.16}$$

If $p \le n$, the binomial coefficient is non-zero; if $p = n$ it is one, and if $p \le n+1$ it is a strictly increasing function of n.

```
Lemma binom0 n: natp n -> binom n \0c = \1c.
Lemma binom1 n: natp n -> binom n \1c = n.
Lemma binom2a  n: natp n ->
  \2c *c (binom (csucc n) \2c) = n *c (csucc n).
Lemma binom2 n: natp n ->
 binom (csucc n) \2c = (n *c (csucc n)) %/c \2c.
Lemma binom_nn n: natp n -> binom n n = \1c.
Lemma binom_pr3 n p: natp n -> natp p ->
  p <=c n -> binom n p <> \0c.

Lemma binom_monotone1 k n m:
  natp k -> natp n -> natp m ->
  k <> \0c -> k <=c (csucc n) -> n <c m ->
  (binom n k) <c (binom m k).
Lemma binom_monotone2 k n m:
  natp k -> natp n -> natp m ->
  k <> \0c -> k <=c (csucc n) ->  k <=c (csucc m) ->
  (n <c m <->  (binom n k) <c (binom m k)).
```

We show here that $\binom{n}{k}$ has a maximom at $k = n/2$. We start with

$$(m+1)\binom{m}{n} = (n+1)\binom{m+1}{n+1} \qquad n\binom{n+p}{p} = (p+1)\binom{n+cp}{p+1}$$

Fix n and lrt $C_k = \binom{n}{k}$. It follows $(n-k)C_k = (k+1)C_{k+1}$ so that $C_k \le C_{k+1}$ if and only if $2k+1 \le n$. Let $q = n/2$. If n is even the previous relation is $k < q$, otherwise it is $k \le q$, case where $C_q = C_{q+1}$ and the maximum is reached twice. In the code that follows we write: C_k is maxima if $k = q$ or $n - k = q$.

```
Lemma mul_Sm_binom m n : natp n -> natp m ->
  csucc m *c  binom m n = csucc n *c binom (csucc m) (csucc n).
Lemma mul_Sm_binom_1 n p : natp n -> natp p ->
  n *c (binom (n +c p) p) = (csucc p) *c binom (n+c p) (csucc p).
Lemma binom_rec1 n k: natp n -> k <=c n ->
  (n -c k) *c (binom n k) = (csucc k) *c binom n (csucc k).
Lemma binom_monotone3 n k: natp n -> k <=c n ->
  ( (binom n k) <=c binom n (csucc k)  <-> csucc (cdouble k) <=c n).
Lemma binom_monotone4 n k: natp n -> k <=c n ->
  ( (binom n k) <c binom n (csucc k)  <-> cdouble (csucc k) <=c n).
Lemma binom_half_aux n: natp n -> oddp n ->
    binom n (chalf n) = binom n (csucc (chalf n)).
Lemma binom_max n k: natp n -> k <=c n ->
  (binom n k) <=c binom n (chalf n).
Lemma binom_monotone_max_arg n k (h := chalf n): natp n -> k <=c n ->
  ( (binom n k) = (binom n h) <-> (k = h \/ n -c k = h) ).
```

The last lemma of the previous section says that that $c_{m+p,p}$ is the number of partitions of a set E with $n = m + p$ elements into two sets with m and p elements. For completeness, we show that $c_{n,p}$ is the number of partitions of E into two sets with $n - p$ and p elements (if $p > n$, this number is zero, there is no partition of E into two subsets with p and k elements, whatever k). Let Q_p be the set of all subsets A of E that have p elements. If $A \in Q_p$ then $E - A \in Q_{n-p}$ and $A \mapsto E - A$ is a bijection. Moreover $(A, E - A)$ is a partition with $(p, n - p)$ elements. The cardinal of Q_p is b_{np}, the Bourbaki definition of the binomial coefficient. Thus $b_{np} = c_{np}$ whenever $p \le n$. The relation also holds if $p > n$ since Q_p is empty and $c_{np} = 0$.

```
Lemma number_of_partitions_p3 E m p: natp m -> natp p ->
  cardinal E = m +c p ->
  cardinal (partitions_pi (variantLc m p) E) =
  binom (m +c p) m.
Lemma number_of_partitions_p4 E n m: natp n -> natp m ->
  cardinal E = n ->
  cardinal (partitions_pi (variantLc m (n -c m)) E) =
  binom n m.

Definition subsets_with_p_elements p E:=
  Zo (\Po E)(fun z=> cardinal z =p).

Lemma subsets_with_p_elements_pr n p E: natp n -> natp p ->
  cardinal E = n ->
  binom n p = cardinal (subsets_with_p_elements p E).
Lemma subsets_with_p_elements_pr0 n p: natp n -> natp p ->
  binom n p = cardinal (subsets_with_p_elements p n).
Lemma bijective_complement n p E: natp n -> natp p ->
  p <=c n -> cardinal E = n ->
  bijection (Lf (complement E)
    (subsets_with_p_elements p E)(subsets_with_p_elements (n -c p) E)).
```

The following formula is true whenever a_i are permutable elements of a ring

$$\sum_{\sum p_i = n} \frac{n!}{p_1! \cdots p_k!} a_1^{p_1} ... a_k^{p_k} = (a_1 + a_2 + \cdots + a_k)^n. \tag{6.17}$$

Here the sum is over all ordered tuples $p_1, p_2, \dots, p_k$ such that $p_1 + p_2 + \cdots + p_k = n$. If m is the sum of the first $k-1$ terms, so that $p_k = n - m$, if we factor out powers of a_k and use associativity, the previous expression becomes

$$\sum_m \left(\sum_{\sum p_j = m} \frac{m!}{p_1! \cdots p_{k-1}!} a_1^{p_1} ... a_{k-1}^{p_{k-1}} \right) \binom{n}{m} a_k^{n-m}$$

We can proceed by induction on k, starting with $k = 2$, for which we use induction on n. We show here an alternate method, valid only when the quantities a_i are integers. In the special case $a_i = 1$ the formula reduces to

$$\sum_{\sum p_i = n} \frac{n!}{p_1! \cdots p_k!} = k^n, \qquad \sum_p \binom{n}{p} = 2^n. \tag{6.18}$$

Let E be a finite set of cardinal n, I an index set, and $\mathrm{A}_{n\mathrm{I}}$ be the set of all mappings p with $\sum_{i \in \mathrm{I}} p(i) = n$. This relation implies that $p(i) \in [0, n]$. Thus we complete the definition of $\mathrm{A}_{n\mathrm{I}}$ by requiring that the target of p is the interval $[0, n]$. We shall compute the cardinal of this set later on.

Let $\mathrm{B}_{n\mathrm{I}}$ be the set of graphs of elements of $\mathrm{A}_{n\mathrm{I}}$. We have that $p \in \mathrm{B}_{n\mathrm{I}}$ if and only if p is a functional graph defined on I and $\sum p_i = n$. There is a similar property for graphs with $p \in \mathrm{B}_{n\mathrm{I}}$ instead of $\sum p_i = n$. Thus we can rewrite (6.17) and (6.18) with $\sum p_i \le n$.

```
Definition functions_sum_eq F n:=
  Zo (functions F (Nintc n)) (fun z=> csum (P z) = n).
Definition functions_sum_le E n:=
  Zo (functions E (Nintc n)) (fun z=> csum (P z) <=c n).
Definition graphs_sum_eq F n:=
  fun_image (functions_sum_eq F n) graph.
Definition graphs_sum_le F n:=
  fun_image (functions_sum_le F n) graph.

Lemma setof_suml_auxP F n: natp n -> forall f,
  inc f (graphs_sum_le F n) <->
  [/\ domain f = F, (csum f) <=c n,  fgraph f & cardinal_fam f].
Lemma setof_sume_auxP F n: natp n -> forall f,
  inc f (graphs_sum_eq F n) <->
  [/\ domain f = F, csum f = n, fgraph f& cardinal_fam f].
```

Consider now a finite sequence of integers a_i (We shall see in the next chapter that if a_i is infinite, both terms in (6.17) are equal to the greatest element element of the family, the case where the number of terms is infinite can also be handled, but the result has no great interest). We let k be the cardinal of the index set I, and consider a set F whose cardinal is $\sum a_i$. We know that there is a partition F_i of F with $\mathrm{card}(\mathrm{F}_i) = a_i$. If f is a function $\mathrm{E} \to \mathrm{F}$, we consider the set $f_i = f^{-1}\langle \mathrm{F}_i \rangle$ of all elements of E such that $f(x) \in \mathrm{F}_i$. Denote by $\phi(f)$ the mapping $i \mapsto \mathrm{card}\, f_i$, and by $\Phi(f)$ the mapping $i \mapsto f_i$. We have $\phi(f) \in \mathrm{B}_{n\mathrm{I}}$ and $\Phi(f) \in \mathrm{C}_{\phi(f),\mathrm{E}}$, where $\mathrm{C}_{p\mathrm{E}}$ denotes the set of all partitions with p_i elements of E.

If P is in $\mathrm{C}_{p\mathrm{E}}$ we consider a family of bijections $h_i : [1, p_i] \to \mathrm{P}_i$. If we take $\mathrm{P} = \Phi(f)$, the function $f \circ h_i$ maps $[1, p_i]$ to F_i. Its restriction to F_i is a element of $\mathscr{F}([1, p_i], \mathrm{F}_i)$. Denote it by

$\Psi(f)(i)$. Now $\Psi(f) \in \prod \mathscr{F}([1, p_i], \mathrm{F}_i)$. The mapping $f \to (\Phi(f), \Psi(f))$ is a bijection (for fixed $\phi(f)$), as can be easily shown (the proof is a bit technical, thus long).

The case $a_i = 1$ is a bit easier. Here each F_i has one element, and we can identity Φ and Ψ (given a partition E_i of E, if $\mathrm{F}_i = \{y_i\}$, f maps E_i into F_i if and only if $f(x) = y_i$ for $x \in \mathrm{E}_i$). This makes the proof much shorter.

```
Lemma sum_of_gen_binom E F n: natp n -> cardinal E = n ->
  csumb  (graphs_sum_eq F n) (fun p => cardinal (partitions_pi p E))
  = (cardinal F) ^c n. (* 94 *)
Lemma sum_of_gen_binom0 E n a:  (* 242 *)
  natp n ->  cardinal E = n -> finite_int_fam a ->
  (csum a) ^c n =
     csumb (graphs_sum_eq (domain a) n)
       (fun p =>
         (cardinal (partitions_pi p E)) *c
            (cprodb (domain a) (fun i=> ((Vg a i) ^c (Vg p i))))).
```

We consider now a special case where F is the canonical doubleton. Its cardinal is 2. To each $m \in [0, n]$ we can associate the function defined on F that maps the first element to m and the second to $n - m$. This is a bijection onto $\mathrm{B}_{n\mathrm{F}}$, and we can use it to perform a change of variables in the sum, and we can apply `number_of_partitions_p4`, and we get the binomial coefficient (second part of formula (6.18)). We give an alternate proof: we count the number of subset of E, according to their cardinal p.

We prove the Pascal formula (6.14) by induction. If we replace n by $n+1$, isolate the term $p = 0$, write $p = k+1$ and use the binomial relation we get

$$\sum_{i=0}^{n} \binom{n}{k} a^{k+1} b^{n+1-k-1} + \sum_{i=0}^{n} \binom{n}{k+1} a^{k+1} b^{n+1-k-1} + b^{n+1}.$$

We re-introduce p in the second sum, factor out a and b and we get

$$a\left[\sum_{i=0}^{n} \binom{n}{k} a^k b^{n-k}\right] + \left[\sum_{i=0}^{n} \binom{n}{p} a^p b^{n-p}\right] b.$$

It suffices to apply the induction hypothesis. The proof is similar to that given above, just much longer.

```
Lemma sum_of_gen_binom2 n: natp n ->
  cumb (Nintc n) (binom n) = \2c ^c n.
Lemma sum_of_binomial n: natp n ->
  csumb  (Nintc n) (binom n) = \2c ^c n.

Lemma sum_of_binomial2 a b n:
  natp n ->
  csumb (Nintc n) (fun p => (binom n p) *c (a ^c p) *c (b ^c (n -c p)))
  = (a +c b) ^c n.    (* 55 *)
```

Number of increasing functions. Consider two finite totally ordered sets E and F. Denote by $\mathscr{S}(\mathrm{E}, \mathrm{F})$ the set of strictly increasing mappings from E into F, and by $\mathscr{A}(\mathrm{E}, \mathrm{F})$ the set of increasing mappings from E into F. We pretend that

$$\mathrm{card}(\mathscr{S}(\mathrm{E}, \mathrm{F})) = \binom{\mathrm{card}(\mathrm{F})}{\mathrm{card}(\mathrm{E})}.$$

$$\text{card}(\mathscr{A}(\mathrm{E},\mathrm{F})) = \binom{n+p-1}{p} \text{ if card(E)} = p \text{ and card(F)} = n.$$

The first relation is a consequence of the the fact that the mapping $f \mapsto \mathrm{R}(f)$ is a bijection from $\mathscr{S}(\mathrm{E},\mathrm{F})$ onto the set of subsets with p elements of F, where $\mathrm{R}(f)$ denotes the range of f. It requires Theorem 3 (§ 2, no. 5) (uniqueness of isomorphisms between well-ordered sets). [note: if u and v are two strictly increasing functions with the same source range, target, they must be equal. For otherwise there would exists a least element x such that $u(x) \neq v(x)$ since the source is finite and totally ordered. We have $u(x) = v(a)$ for some a. If $a < x$ we get $v(a) = u(a) = u(x)$, hence $a = x$ by injectivitity of u. The case $a = x$ is also excluded, so that $x < a$ and $v(x) < v(a)$, thus $v(x) < u(x)$. By symmetry $u(x) < v(x)$, absurd.]

```
Definition functions_incr r r' :=
  (Zo (functions (substrate r) (substrate r'))
    (fun z => increasing_fun z r r')).
Definition functions_sincr r r' :=
  (Zo (functions (substrate r) (substrate r'))
    (fun z => strict_increasing_fun z r r')).

Lemma cardinal_set_of_increasing_functions1 r r' f:
  total_order r -> strict_increasing_fun f r r' ->
  (order_morphism f r r' /\ cardinal (substrate r) = cardinal (Imf f)).

Lemma cardinal_set_of_increasing_functions2 r r':
  total_order r -> total_order r' ->
  finite_set (substrate r)  ->  finite_set (substrate r')  ->
  bijection (Lf (fun z  => Imf z)
    (functions_sincr r r')
    (subsets_with_p_elements (cardinal (substrate r)) (substrate r'))).

Lemma cardinal_set_of_increasing_functions r r':
  total_order r -> total_order r' ->
  finite_set (substrate r)  ->  finite_set (substrate r')  ->
  cardinal (functions_sincr r r')
  = binom (cardinal (substrate r')) (cardinal (substrate r)).
```

We reduce the study of $\mathscr{A}$ to the study of $\mathscr{S}$. We start with the case where E is an interval of **N**, say $[0,p]$. For f to be increasing (resp., strictly increasing), it suffices that $f(x) \leq f(x+1)$, (resp. $f(x) < f(x+1)$). Note that a strictly increasing function is injective.

```
Lemma increasing_prop0 p f r: natp p -> order r ->
  (forall i, i <=c p -> inc (f i) (substrate r)) ->
  (forall n,  n <c p -> gle r (f n) (f (csucc n))) ->
  (forall i j, i <=c j -> j <=c p ->
    gle r (f i) (f j)).
Lemma increasing_prop1 p f: natp p ->
  (forall i, i <=c p -> inc (f i) Nat) ->
  (forall n, n <c p -> (f n) <=c (f (csucc n))) ->
  (forall i j, i <=c j -> j <=c p -> (f i) <=c (f j)).
Lemma strict_increasing_prop0 p f r: natp p -> order r ->
  (forall n,  n <c p -> glt r (f n) (f (csucc n))) ->
  (forall i j, i <c j -> j <=c p ->  glt r (f i) (f j)).

Lemma increasing_prop p f r: natp p -> function f ->
  source f = (csucc p) -> order r -> substrate r = target f ->
```

```
  (forall n, n <c p -> gle r (Vf f n) (Vf f (csucc n))) ->
  increasing_fun f (Nint_cco \0c p) r.

Lemma strict_increasing_prop p f r: natp p -> function f ->
  source f = (csucc p) -> order r -> substrate r = target f ->
  (forall n, n <c p -> glt r (Vf f n) (Vf f (csucc n))) ->
  (injection f /\
    strict_increasing_fun f (Nint_cco \0c p) r).
```

We assume now that F is also a subset of **N**, so that $f(x) < f(x+1)$ can be replaced by $f(x)+1 \le f(x+1)$. In fact, if f is strictly increasing, we get $x \le f(x)$, and we can subtract $x+1$ from the previous inequality, so that $f(x)-x \le f(x+1)-(x+1)$.

Denote by I_p the interval $[0,p[$. We get: if f is strictly increasing $\mathrm{I}_p \to \mathrm{I}_{n+p}$, then the function $i \mapsto f(i)-i$ is decreasing, with values in I_{n+1}.

```
Lemma strict_increasing_prop1 f p:
  natp p ->  (forall i, i <c p -> natp (f i))
  -> (forall i j, i <c j -> j <c p -> (f i) <c (f j)) ->
  (forall i, i <c p -> i <=c (f i)).
Lemma strict_increasing_prop2 f p:
  natp p ->  (forall i, i <c p -> natp (f i))
  -> (forall i j, i <c j -> j <c p -> (f i) <c (f j)) ->
  (forall i j, i <=c j -> j <c p -> ((f i) -c i) <=c ((f j) -c j)).
Lemma strict_increasing_prop3 f p n:
  natp p -> natp n -> (forall i, i <c p -> natp (f i))
  -> (forall i j, i <c j -> j <c p -> (f i) <c (f j)) ->
  (forall i, i <c p -> (f i) <c (n +c p)) ->
  (forall i, i <c p -> ((f i) -c i) <=c n).
```

If f is a mapping, denote by $s(f)$ the mapping $i \mapsto f(i)-i$, and by $a(f): i \mapsto f(i)+i$. We have shown that s is a mapping from $\mathscr{S}(\mathrm{I}_p,\mathrm{I}_{n+p})$ into $\mathscr{A}(\mathrm{I}_p,\mathrm{I}_{n+1})$. It is a bijection with inverse a. Thus we get

$$\mathrm{card}(\mathscr{A}(\mathrm{I}_p,\mathrm{I}_{n+1})) = \mathrm{card}(\mathscr{S}(\mathrm{I}_p,\mathrm{I}_{n+p})) = \binom{n+p}{p}.$$

```
Lemma cardinal_set_of_increasing_functions3 n p:
  natp n -> natp p ->
  cardinal (functions_incr (Nint_co p)  (Nint_cco \0c n))
   = binom (n +c p) p. (* 166 *)
```

We compute the cardinal of $\mathscr{A}(\mathrm{E},\mathrm{F})$ in the general case. If F is empty, there is a unique function $\mathrm{E} \to \mathrm{F}$ if E is empty, and none if E is non-empty. Assume that E has p elements and F has $n > 0$ elements. There is a unique order isomorphism f between between E and $[0,p[$, and g between F and $[0,n-1]$. Assume $h: \mathrm{E} \to \mathrm{F}$ increasing. Then $k = g \circ h \circ f^{-1} : [0,p[\to [0,n[$. Since $h = g^{-1} \circ k \circ f$, we have that h is increasing if and only if k is increasing, thanks to the additional lemmas. This gives an isomorphism between $\mathscr{A}(\mathrm{E},\mathrm{F})$ and $\mathscr{A}(\mathrm{I}_p,\mathrm{I}_n)$, and these two sets have the same number of elements.

```
Lemma cardinal_set_of_increasing_functions4 r r'
   (n := cardinal (substrate r'))
   (p := cardinal (substrate r)):
      total_order r -> total_order r' ->
```

```
finite_set (substrate r) -> finite_set (substrate r') ->
cardinal (functions_incr r r')
= binom ((n +c p) -c \1c) p. (* 89 *)
```

Number of ordered pairs. We compute now the number a_n of pairs (i,j) such that $1 \le i \le j \le n$, and the number b_n of pairs satisfying $1 \le i < j \le n$. This is the number of increasing (resp. strictly increasing) mappings of a set with two elements into the interval $[1,n]$, which has n elements. Our previous results show

$$a_n = \frac{n(n+1)}{2} = \binom{n+1}{2}, \qquad b_n = \frac{n(n-1)}{2} = \binom{n}{2}.$$

The Bourbaki proof of these relations (Proposition 14 [4, p. 181]) is different. He notices that $a_n = b_n + n$ since $i \le j$ is equivalent to $i < j$ or $i \le j$. A subset of $[1,n]$ is of cardinal two if and only if it is a doubleton $\{i,j\}$ with $i \ne j$, and we may assume $i < j$; hence b_n is the number of subsets of cardinal two of $[1,n]$. The link between a_n and b_n is given by the following trivial relation:

$$\binom{n+1}{2} = \frac{n(n+1)}{2} = \binom{n}{2} + n.$$

```
Lemma binom_2plus n: natp n ->
   binom (csucc n) \2c =  (n *c (csucc n)) %/c \2c.
Lemma binom_2plus0 n: natp n ->
  binom (csucc n) \2c = (binom n \2c) +c n.

Lemma cardinal_pairs_lt n: natp n ->
  cardinal (Zo (coarse Nat)
   (fun z => [/\ \1c <=c (P z), (P z) <c (Q z) &  (Q z) <=c n])) =
  (binom n \2c).

Lemma cardinal_pairs_le n: natp n ->
  cardinal(Zo (coarse Nat)
    (fun z=> [/\ \1c <=c  (P z), (P z) <=c (Q z) & (Q z) <=c n])) =
     (binom (csucc n) \2c)
```

A corollary is the following formula

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} = \binom{n+1}{2}.$$

This formula is obvious by induction on the type `nat`; the prof by induction on $\mathbf{N}$ is a bit longer. Let s_n be the sum and s'_n be the sum $\sum_{i \le n}(n-i)$. By re-ordering indices, we have $s'_n = s_n$; moreover $s_n + s'_n = \sum_{i \le n} n$, which is $n(n+1)$, and we get the result by division by two. We could follow Bourbaki, showing that s_n is the cardinal of the set E of all pairs (i,j) with $1 \le i \le j \le n$, since E is the union of the sets $[1,j] \times \{j\}$, i.e., the disjoint union of the intervals $[1,j]$, which are of cardinal j. In fact, we say $[1,n] = [0,n+1[-\{0\}$.

```
Lemma fct_sum_const1 f n m:
  natp n ->  (forall i, i <c n -> f i = m) ->
  csumb (Nint n) f = n *c m.
Lemma sum_of_i n: natp n ->
  csumb n id = binom n \2c.
Lemma sum_of_i3 n: natp n ->
  csumb n id = binom n \2c.
Lemma sum_of_i2 n: natp n ->
  csumb (Nintcc \1c n) id = (binom (csucc n) \2c).
```

Number of monomials. Consider a set E, a law of composition of E, and elements x, y, z, etc, of E. Consider a combination of these variables, where x appears 3 times, z appears twice and y appears once. If the law is associative and commutative, the combination is equal to $x\cdot(x\cdot(x\cdot(y\cdot(z\cdot z))))$. This is called a monomial, and denoted by x^3yz^2. The total number of factors (here six) is called the degree. Assume that we have a second law of composition $a+b$, and that the usual rules apply. This means that $(a+b)^n$ can be expanded as a sum of monomials: $(a+b)^n = \sum \gamma_{ij} a^i b^j$. It happens that $\gamma_{ij} = \binom{n}{i}$ if $i+j=n$ (this is the explanation of the term "binomial coefficient"). More generally, $(\sum x_i)^n = \sum_{\mathrm{I}\in \mathrm{S}_n} \Gamma_\mathrm{I} x^\mathrm{I}$, where I is a mapping $i\mapsto n_i$, x^I denotes the monomial $x_1^{n_1}x_2^{n_2}\cdots x_p^{n_p}$. The total degree of the monomial is $\sum n_i = n$. Taking $a=b=1$ gives $\sum_p \binom{n}{p} = 2^n$. Taking $a=-1$ and $b=1$ gives $\sum_p (-1)^p\binom{n}{p} = 0$ (The first result has already been proved, the second is the object of Exercice 5.2). We have also $\sum_{\mathrm{I}\in\mathrm{S}_n} \Gamma_\mathrm{I} = p^n$ (it can be shown, by induction on the number of variables, that Γ_I is the number of coverings of a set with n elements by subsets with n_i elements). The cardinal of the set S_n is the object of the next theorem. We compute it by induction on both n and p.

Let E be a set with h elements, $\bar{\mathrm{A}}_n$ and $\bar{\mathrm{B}}_n$ be the sets of functions u with $\sum_{i\in\mathrm{E}} u(i) \le n$ and $\sum_{i\in\mathrm{E}} u(i) = n$ respectively. Let A_{nh} and B_{nh} the cardinals of these sets. Proposition 15 [4, p. 182]) says

$$\mathrm{A}_{nh} = \binom{n+h}{h} \qquad \mathrm{B}_{nh} = \binom{n+h-1}{h-1}.$$

We have $\mathrm{A}_{nh} = \mathrm{B}_{nh} + \mathrm{A}_{n-1,h}$ since $\bar{\mathrm{A}}_n$ is the disjoint union of $\bar{\mathrm{B}}_n$ and $\bar{\mathrm{A}}_{n-1}$. If $x\not\in \mathrm{E}$, every function u such that $\sum u(i) \le n$ can be uniquely extended to $\mathrm{E}\cup\{x\}$ in such a way as $\sum_{i\in\mathrm{E}\cup\{x\}} u(i) = n$. This gives $\mathrm{B}_{n,h+1} = \mathrm{A}_{nh}$. The formulas follow by induction (they are trivial for $h=0$ and $n=0$). One difficulty of the Bourbaki's proof is that he has not yet defined the set of integers; as a consequence, he adds the condition that the target of u is the interval $[0,n]$, so that $\bar{\mathrm{A}}_{n-1}$ is not a subset of $\bar{\mathrm{A}}_n$; it is nevertheless isomorphic to the complement of $\bar{\mathrm{B}}_n$ in $\bar{\mathrm{A}}_n$. There are two other solutions: we may consider functions with target $\mathbf{N}$, or graphs of functions. Since we already introduced the set of graphs of functions such that $\sum u_i = n$, we use graphs. We first show that the sets have the same number of elements.

```
Lemma sof_sum_eq_equi F n: natp n ->
  (functions_sum_eq F n) \Eq (graphs_sum_eq F n).
Lemma sof_sum_le_equi F n: natp n ->
  (functions_sum_le F n) \Eq (graphs_sum_le F n).

Lemma set_of_functions_sum0 f:
  (forall a, natp a -> f \0c a = \1c) ->
  (forall a, natp a -> f a \0c = \1c) ->
  (forall a b, natp a -> natp b ->
     f (csucc a) (csucc b) = (f (csucc a) b) +c (f a (csucc b))) ->
  forall a b, natp a -> natp b -> f a b = (binom (a +c b) a).
Lemma set_of_functions_sum1 E x n:
  inc n Nat ->  ~ (inc x E) ->
  (graphs_sum_le E n)  \Eq   (graphs_sum_eq (E +s1 x) n).

Lemma set_of_functions_sum2 E n: natp n ->
  cardinal(graphs_sum_le E (csucc n))
  = (cardinal (graphs_sum_eq E (csucc n)))
    +c (cardinal (graphs_sum_le E n)).

Lemma set_of_functions_sum3 E:
  cardinal (graphs_sum_le E \0c) = \1c.
Lemma set_of_functions_sum4 n: natp n ->
```

```
    cardinal (graphs_sum_le emptyset n) = \1c.

Lemma set_of_functions_sum_pr n h
  (intv:= fun h => (Nint h))
  (sle:= fun n h => graphs_sum_le (intv h) n)
  (seq := fun n h =>  graphs_sum_eq (intv h) n)
  (A:= fun n h => cardinal (sle n h))
  (B:=  fun n h => cardinal (seq n h)):
  natp n  -> natp h  ->
  (A n h =  B n (csucc h) /\ A n h = (binom (n +c h) n)).
```

We give now a variant of the theorem[4]. Let C_{pn} be the set of functions $y : [0, p] \to [0, n]$ such that $\sum y_i \le n$. We pretend that the cardinal of C_{pn} is $A_{n,p+1} = \binom{n+p+1}{p+1}$. This is the previous result for $h = p+1$ (if $h = 0$, there is a unique function defined on a set with h elements, the empty function, and the sum is zero).

Consider the function x, defined by induction via $x_0 = y_0$ and $x_{i+1} = y_{i+1} + x_i + 1$. This is a strictly increasing function $[0, p] \to [0, n+p]$ and $y \mapsto x$ is a bijection. So C_{pn} has the same cardinal as $\mathscr{S}(I_{p+1}, I_{n+p+1})$. As noticed above, the function x is uniquely defined by its range, which is any subset of $p+1$ elements chosen among $n+p+1$, whence the result.

In our proof, we shall use the function z defined by $z_i = x_i - i$. We have $y_0 = z_0$, and $y_{i+1} = z_{i+1} - z_i$; this defines the mapping $z \mapsto y$. On the other hand, z_i is the sum of the restriction of y to the interval $[0, i]$ (there is no need to define it by induction) and is obviously increasing. All we have to do is show that $y \mapsto z$ is a bijection $C_{pn} \to C'_{pn}$ where $C'_{pn} = \mathscr{A}([0, p], [0, n])$. We first show that $A_{n,p+1}$ is the cardinal of C'_{pn}.

```
Definition graphs_sum_le_int p n :=
  graphs_sum_le (Nintc p) n.

Definition functions_incr_int p n :=
  (Zo (functions (Nintc p)(Nintc n))
    (fun z => increasing_fun z (Nint_cco \0c p) (Nint_cco \0c n))).

Lemma card_set_of_increasing_functions_int p n:
  natp p -> natp n ->
  cardinal (functions_incr_int p n) =
  binom (csucc (n +c p)) (csucc p).
```

If $R(f, i)$ denotes the restriction of the function f to the interval $[0, i]$ we have $R(R(f, i), j) = R(f, j)$ if $j \le i$. If $S(f, i)$ is the sum of the restriction of $R(f, i)$ we have $S(f, 0) = f(0)$ and $S(f, i+1) = f(i+1) + S(f, i)$.

```
Lemma induction_on_sum3 f m:
  fgraph f -> natp m ->
  domain f = Nintc m ->
  (forall a, inc a (domain f) -> cardinalp (Vg f a)) ->
  (csum (restr f (Nintc \0c)) = (Vg f \0c)
    /\ (forall n, n <=c m ->
      (csum (restr f (Nint n))) +c (Vg f n)
      = csum (restr f (Nint (csucc n))))).
```

Given a function y, we consider z such that $z_i = S(y, i)$. We first show that z maps $[0, p]$ into $[0, n]$, and then that $z \in C'_{pn}$ if $y \in C_{pn}$ (note that $i \mapsto S(y, i)$ is increasing). We then show

[4] Suggested by Jean-Baptiste Pomet

that $y \mapsto z$ is injective and surjective. The key relation is $z_0 = y_0$ and $z_{i+1} = y_{i+1} + z_i$ (trivial consequence of `induction_on_sum3`); it says that y is uniquely defined from z. Moreover, given an increasing function z', if $y_0 = z'_0$ and $y_{i+1} = z'_{i+1} - z'_i$, the same formula is satisfied by z', hence $z = z'$, thus proving surjectivity.

```
Definition csum_to_increasing_fun y :=
  fun i => csum (restr y (Nintc i)).

Definition csum_to_increasing_fct y n p :=
  Lf (csum_to_increasing_fun y)
  (Nintc p) (Nintc n).

Lemma csum_to_increasing1 y n p:
  natp n -> natp p ->
  inc y (graphs_sum_le_int p n) ->
  lf_axiom (csum_to_increasing_fun y) (Nintc p) (Nintc n).

Lemma csum_to_increasing2 n p:
  natp n -> natp p ->
  lf_axiom (fun y=> (csum_to_increasing_fct y n p))
  (graphs_sum_le_int p n)
  (functions_incr_int p n).

Lemma csum_to_increasing4 n p:
  natp n -> natp p ->
  injection (Lf (fun y=> (csum_to_increasing_fct y n p))
  (graphs_sum_le_int p n)
  (functions_incr_int p n)).

Lemma csum_to_increasing5 n p:
  natp n -> natp p ->
  surjection (Lf (fun y=> (csum_to_increasing_fct y n p))
  (graphs_sum_le_int p n)
  (functions_incr_int p n)). (* 83 *)

Lemma csum_to_increasing6 n p:
  natp p -> natp n ->
  cardinal (graphs_sum_le_int p n) =
  binom (csucc (n +c p)) (csucc p).
```

We show here the equivalence between our definitions of the binomial coefficient and that of COQ.

```
Lemma nat_to_B_fact n: nat_to_B (n`!) = factorial ( nat_to_B n).
Lemma nat_to_B_binom m n:
   nat_to_B 'C(m,n) = binom (nat_to_B m) (nat_to_B n).
Lemma nat_to_B_quorem a b :
  nat_to_B(a %/ b) = (nat_to_B a) %/c (nat_to_B b) /\
  nat_to_B(a %% b) = (nat_to_B a) %%c (nat_to_B b).
```

6.9 More combinatorial analysis

This section is a complement of the previous one. We study properties of families of numbers (mainly integers), defined by induction, using the SSREFLECT library. This section is completely independent of the remainder of the work on Bourbaki.

There are several variants of (6.19); one is in the SSREFLECT library. If we iterate p times the second formula of (6.19) we get (6.20) where $(n)_p = n!/(n-p)!$ denotes the falling factorial.

$$\binom{m+n}{m} = \binom{m+n}{n}, \qquad 2\binom{n+1}{2} = n(n+1).$$

$$n\binom{n+p}{p} = (p+1)\binom{n+p}{p+1}, \qquad (n-k)\binom{n}{k} = (k+1)\binom{n}{k+1}. \tag{6.19}$$

$$(n-k)_p\binom{n}{k} = (k+p)_p\binom{n}{k+p}. \tag{6.20}$$

```
Lemma binom_mn_n m n : 'C(m + n, m) = 'C(m + n, n).
Lemma bin2' n: 'C(n.+1,2) * 2 = n * n.+1.
Lemma mul_Sm_binm_1 n p: n * 'C(n+p,p) = p.+1 * 'C(n+p,p.+1).
Lemma mul_Sm_binm_r n k p:
   (n-k) ^_ p * 'C(n,k) = (k +p) ^_ p * 'C(n,k+p).
Lemma mul_Sm_binm_2 n k: (n-k) * 'C(n,k) = k.+1 * 'C(n,k.+1).
Lemma bin_fact1 n m:  'C(n+m,m) * (m`! * n`!) = (n+m)`!.
```

There are many variants of (6.21); for instance we get (6.22) for $n = k+q$; the products are zero if $k > n$.

$$\binom{j+k+q}{j+k}\binom{j+k}{j} = \binom{k+q}{k}\binom{j+k+q}{j}. \tag{6.21}$$

$$\binom{j+n}{j+k}\binom{j+k}{j} = \binom{n}{k}\binom{j+n}{j}. \tag{6.22}$$

$$\sum_{i\le n}\binom{n+1}{i+1}f(i) = \sum_{i<n}\binom{n}{i+1}f(i) + \sum_{i\le n}\binom{n}{i}f(i). \tag{6.23}$$

```
Lemma binom_exchange j k q:
  'C(j+k+q,j+k) * 'C(j+k,j) = 'C(k+q,k) * 'C(j+k+q,j).
Lemma binom_exchange1 j k n:
  'C(j+n,j+k) * 'C(j+k,j) = 'C(n,k) * 'C(j+n,j).
Lemma sum_bin_rec (n :nat) (f: nat -> nat):
  \sum_(i<n.+1) 'C(n.+1,i.+1) * (f i) =
  \sum_(i<n) 'C(n,i.+1) * (f i) +  \sum_(i<n.+1) 'C(n,i) * (f i).
```

The following is nice.

$$\binom{n}{2} + 6\binom{n+1}{4} = \binom{n}{2}^2. \tag{6.24}$$

```
Lemma F7a n: 'C(n, 2) + 6 * 'C(n.+1, 4) = 'C(n, 2) ^ 2.
```

6.9.1 Number of derangements

This corresponds to Exercise 5.8. The problem was originallly studied by Euler. We consider p_n defined by

$$p_{n+1} = n(p_n + p_{n-1}), \qquad p_0 = 1, \quad p_1 = 0.$$

It satisfies the relation $p_{n+1} = (n+1)p_n - (-1)^n$ (note that $p_n > 0$ for $n > 1$). We have

$$\sum_{i=0}^{n} \binom{n}{i} p_i = n!, \qquad \sum_{i=0}^{n} \binom{n}{i} p_{i+1} = n.n!.$$

```
Fixpoint der_rec n :=
   if n is n'.+1 then if n' is  n''.+1 then n' * (der_rec n'' + der_rec n')
   else 0 else 1.
Definition derange n := nosimpl der_rec n.

Lemma derange0: derange 0 = 1.
Lemma derange1: derange 1 = 0.
Lemma derangeS n: derange n.+2 = (n.+1) * (derange n + derange n.+1).
Lemma derangeS1 n (p := n.+1 * derange n):
   derange n.+1 = if (odd n) then p.+1 else p.-1.
Lemma derange_sum n:
  \sum_(i<n.+1) 'C(n,i) * (derange i) =  n`! /\
  \sum_(i<n.+1) 'C(n,i) * (derange i.+1) =  n * n`!.
```

6.9.2 Number of increasing mappings

We show here that the number of integer sequences $(x_i)_i$ of length m such that $\sum x_i \le n$ is $\binom{n+m}{n}$. This was proved above as `csum_to_increasing6`.

The expression `#|[set f:T | P f]|` denotes the cardinal of the set of all f of type T that satisfy P. Here T must be a finite type. In the previous section we considered the graphs of functions $\mathrm{E} \to \mathrm{F}$ satisfying some conditions. As E and F are finite, the set of functions $\mathrm{E} \to \mathrm{F}$ is finite. One deduces that the set of graphs is finite as well. This gives a type T. We shall consider later on the number of sequences x such that $n = \sum_{i \le k} x_i 2^i$, where x_i is zero, one or two, x_k is non-zero. The relation $k \le n$ says that this number is finite, but constructing the type T is uneasy.

Instead of a sequence, we consider a function f, defined on I_m, the set of integers $< m$, and the condition that $\sum_{i<m} f(i)$ is $= n$, $< n$ or $\le n$. In any case $f(i) \le n$ so that we may assume that f is a function $\mathrm{I}_m \to \mathrm{I}_{n+1}$. This gives our finite type T, and three sets $\mathrm{T}_=(m,n)$, $\mathrm{T}_<(m,n)$ and $\mathrm{T}_\le(m,n)$.

```
Definition Ftype m n := {ffun 'I_m -> 'I_(n.+1)}.

Definition monomial_lt m n (f:Ftype m n) :=  \sum_(i<m) (f i) < n.
Definition monomial_le m n (f:Ftype m n) :=  \sum_(i<m) (f i) <= n.
Definition monomial_eq m n (f:Ftype m n) :=  \sum_(i<m) (f i) == n.
```

Let T' be the set of sequences. Then $\mathrm{T}'_<(m,n+1) = \mathrm{T}'_\le(m,n)$; this says that $\mathrm{T}_<(m,n+1)$ is equipotent to $\mathrm{T}_\le(m,n)$. Note that $\mathrm{T}_\le(m,n)$ is the disjoint union of $\mathrm{T}_=(m,n)$ and $\mathrm{T}_<(m,n)$; if $m = 0$ or $n = 0$, then $\mathrm{T}_\le(m,n)$ has a single element (the constant function with value zero).

```
Lemma card_set_pred: forall (T:finType) (P:T -> bool),
  #|[set f:T | P f]| =  #|[pred f:T | P f]|.

Lemma G3_a (m n: nat):
  #|[set f:Ftype m n  | monomial_le  f ]|
  =  #|[set f:Ftype m n  | monomial_eq f ]|
  +  #|[set f:Ftype m n  | monomial_lt f ]|.
Lemma G3_b (m n: nat):
  #|[set f:Ftype m n.+1  | monomial_lt f ]| =
  #|[set f:Ftype m n  | monomial_le  f ]|.
Lemma G3_c (n: nat):  #|[set f:Ftype 0 n  | monomial_le  f ]| = 1.
Lemma G2_d (m: nat):  #|[set f:Ftype m 0  | monomial_le  f ]| = 1.
```

Our result follows from the fact that $T_=(m+1,n)$ and $T_\le(m,n)$ are equipotent (if $x \in T_=(m+1,n)$, then $x_m = n - \sum_{i<n} x_i$). (total size: 131 lines)

```
Lemma G3_e (m n: nat):
  #|[set f:Ftype m.+1 n  | monomial_eq  f ]|
  =  #|[set f:Ftype m n  | monomial_le f ]|.
Lemma G3_f m n:
 #|[set f:Ftype m n  | monomial_le  f ]|  = 'C(n+m,m)
```

As noted above, if $y_k = \sum_{i<k} x_i$, then $k \mapsto y_k$ is increasing, $k \mapsto y_k + k$ is strictly increasing and the problem becomes: compute the number of (strictly) increasing functions. The solution is given by the following lemmas of the SSREFLECT library (total size: 114 lines)

```
Lemma card_ltn_sorted_tuples m n :
  #|[set t : m.-tuple 'I_n | sorted ltn (map val t)]| = 'C(n, m).
Lemma card_sorted_tuples m n :
  #|[set t : m.-tuple 'I_n.+1 | sorted leq (map val t)]| = 'C(m + n, m).
Lemma card_partial_ord_partitions m n :
  #|[set t : m.-tuple 'I_n.+1 | \sum_(i <- t) i <= n]| = 'C(m + n, m).
Lemma card_ord_partitions m n :
  #|[set t : m.+1.-tuple 'I_n.+1 | \sum_(i <- t) i == n]| = 'C(m + n, m).
```

These lemmas compute the number of sequences of length m with value in I_n satisfying some properties. We give an alternate proof of the result by using the first lemma; hence we consider the sequence of all $y_k + k$ (modulo $n+m$). Note that these values are $< n+m$, so that the modulo is an injective coercion into a finite type (total size: 148 lines).

```
Lemma subseq_iota a n b m :
  b <=a -> a + n <= b + m ->
  subseq (iota a n) (iota b m).
Lemma subseq_iota1 i m: i<m -> subseq ([:: i;  i.+1]) (iota 0 m.+1).
Lemma sorted_prop  f m:
   sorted ltn (mkseq f m.+1) <-> (forall i, i<m -> f i < f (i.+1)).
Definition bin_to_seq m n (f:Ftype m.+1 n) :=
  map_tuple [ffun z:'I_(m.+1) =>
             @inord (n+m) (\sum_(i<(m.+1) | i<=z) (f i) + z)]
  (ord_tuple (m.+1)).

Lemma G3_f' (m n: nat):
  #|[set f:Ftype m n  | monomial_le  f ]| = 'C(n+m,m).
```

6.9.3 Stirling numbers

Stirling numbers of the second kind, denoted $\left\{ n \atop p \right\}$, are the number of ways to partition a set of n things into p nonempty subsets. Multiplied by $p!$, this is the number of surjections from a set with n elements onto a set of p elements. We have

$$\left\{ n+1 \atop p+1 \right\} = \left\{ n \atop p \right\} + (p+1)\left\{ n \atop p+1 \right\}$$

(completed by $\left\{ 0 \atop 0 \right\} = 1$; other values are zero).

```
Fixpoint stirling2_rec n m :=
  match n, m with
  | n'.+1, m'.+1 => m *stirling2_rec n' m + stirling2_rec n' m'
  | 0, 0 => 1
  | 0, _.+1 => 0
  | _ .+1, 0 => 0
  end.

Definition stirling2 := nosimpl stirling2_rec.
Definition nbsurj n m := (stirling2 n m) *  m`!.

Notation "''St' ( n , m )" := (stirling2 n m)
  (at level 8, format "''St' ( n ,  m )") : nat_scope.
Notation "''Sj' ( n , m )" := (nbsurj n m)
  (at level 8, format "''Sj' ( n ,  m )") : nat_scope.

Lemma stirE : stirling2 = stirling2_rec.
Lemma stir00 : 'St(0, 0) = 1.
Lemma nbsurj00 : 'Sj(0, 0) = 1.
Lemma stirn0 n : 'St(n.+1, 0) = 0.
Lemma nbsurjn0 n : 'Sj(n.+1, 0) = 0.
Lemma stir0n m : 'St(0, m.+1)  = 0.
Lemma nbsurj0n m : 'Sj(0, m.+1)  = 0.
Lemma stirS n m :  'St(n.+1, m.+1) = (m.+1) * 'St(n, m.+1) + 'St(n, m).
Lemma nbsurjS n m :  'Sj(n.+1, m.+1) = (m.+1) * ('Sj(n, m.+1) + 'Sj(n, m)).
```

We list some properties

$$\left\{ n \atop n \right\} = 1, \quad \left\{ n+1 \atop 2 \right\} = 2^n - 1, \quad \left\{ n+1 \atop n \right\} = \binom{n+1}{2}, \quad \left\{ n+2 \atop n \right\} = \binom{n+3}{4} + 2\binom{n+2}{4}.$$

$$\mathrm{S}_{n,n} = n!, \quad \mathrm{S}_{n+1,2} = 2^{n+1} - 2, \quad \mathrm{S}_{n+1,n} = \binom{n+1}{2} n!, \quad \mathrm{S}_{n+2,n} = (\binom{n+3}{4} + 2\binom{n+2}{4}) n!.$$

```
Lemma stir_n1  n: 'St(n.+1, 1) = 1.
Lemma nbsurj_n1  n: 'Sj(n.+1, 1) = 1.
Lemma stir_n2  n: 'St(n.+1, 2) = (2 ^n - 1).
Lemma nbsurj_n2  n: 'Sj(n.+1, 2) = (2 ^n.+1 - 2).
Lemma stir_small n p: 'St(n, (n+p).+1) = 0.
Lemma stir_small1 n p: n < p -> 'St(n, p) = 0.
Lemma nbsurj_small n p: 'Sj(n, (n+p).+1) = 0.
Lemma stir_nn n: 'St(n, n) = 1.
Lemma nbsurj_nn n: 'Sj(n, n) = n`!.
```

```
Lemma stir_Snn n: 'St(n.+1, n) = 'C(n.+1,2).
Lemma nbsurj_Snn n: 'Sj(n.+1, n) = 'C(n.+1,2) * n`!.
Lemma stir_SSnn n: 'St(n.+2, n) = 'C(n.+3,4) + 2 * 'C(n.+2,4).
Lemma nbsurj_SSnn n: nbsurj n.+2 n = ('C(n.+3,4) + 2 * 'C(n.+2,4)) * n`!.
```

Stirling numbers of the first kind, denoted by $\left[{n \atop p}\right]$ count the number of ways to arrange n objects into k cycles. They satisfy

$$\begin{bmatrix} n+1 \\ p+1 \end{bmatrix} = \begin{bmatrix} n \\ p \end{bmatrix} + n \begin{bmatrix} n \\ p+1 \end{bmatrix}$$

(completed by $\left[{0 \atop 0}\right] = 1$; other values are zero).

```
Fixpoint stirling1_rec n m :=
  match n, m with
  | n'.+1, m'.+1 => n' *stirling1_rec n' m + stirling1_rec n' m'
  | 0, 0 => 1
  | 0, _.+1 => 0
  | _ .+1, 0 => 0
  end.

Definition stirling1 := nosimpl stirling1_rec.

Notation "''So' ( n , m )" := (stirling1 n m)
  (at level 8, format "''So' ( n ,  m )") : nat_scope.


Lemma stir1_E : stirling1 = stirling1_rec.
Lemma stir1_00 : 'So(0, 0) = 1.
Lemma stir1_n0 n : 'So(n.+1, 0) = 0.
Lemma stir1_0n m : 'So(0, m.+1)  = 0.
Lemma stir1_S n m :  'So(n.+1, m.+1) = n * 'So(n, m.+1) + 'So(n, m).
```

We list some properties

$$\begin{bmatrix} n+1 \\ n \end{bmatrix} = n!, \qquad \begin{Bmatrix} n+1 \\ n \end{Bmatrix} = 1, \qquad \begin{bmatrix} n \\ n \end{bmatrix} = 1, \qquad \begin{bmatrix} n+1 \\ n \end{bmatrix} = \begin{pmatrix} n+1 \\ 2 \end{pmatrix}.$$

```
Lemma stir1_Sn1 n :  'So(n.+1,1) = n `!.
Lemma stir_Sn1 n :  'St(n.+1,1) = 1.
Lemma stir1_small n p: 'So(n, (n+p).+1) = 0.
Lemma stir1_small1 n p: n < p -> 'So(n,p) = 0.
Lemma stir1_nn n :  'So(n,n) = 1.
Lemma stir1_Snn n: 'So(n.+1, n) = 'C(n.+1,2).
```

6.9.4 Euler numbers

Let A_{nm} or $\left\langle {n \atop m} \right\rangle$ be the number of permutations σ of $[1, n]$ with m ascents ($\sigma(x) < \sigma(x+1)$ occurs m times). They satisfy

$$\left\langle {n+1 \atop m+1} \right\rangle = (n-m) \left\langle {n \atop m} \right\rangle + (m+2) \left\langle {n \atop m+1} \right\rangle. \tag{6.25}$$

There is no such partition when $n \le m$ (in particular if $n = 0$). If $m = 0$, σ has to be strictly decreasing and there is only one such function: $\sigma(x) = n+1-x$. If we reverse a permutation, an ascent becomes a descent, so that there a symmetry (formula (6.26) below).

```
Fixpoint euler_rec n m :=
  match n, m with
    | n'.+1, m'.+1 => m.+1 *euler_rec n' m + (n'-m') * euler_rec n' m'
    | 0, _ => 0
    |  _.+1, 0 => 1
  end.

Definition euler := nosimpl euler_rec.

Notation "''Eu' ( n , m )" := (euler n m)
  (at level 8, format "''Eu' ( n ,  m )") : nat_scope.

Lemma eulerE : euler = euler_rec.
Lemma euler0m m : 'Eu(0, m) = 0.
Lemma eulern0 n : 'Eu(n.+1, 0) = 1.
Lemma eulerS n m : 'Eu(n.+1, m.+1) = m.+2 * 'Eu(n, m.+1) + (n-m) * 'Eu(n,m).
```

We have the following formulas. Formulas (6.27) and (6.28) can be generalized (the result is then trivial by inversion on (6.31)). We avoid here using negative numbers by rewriting the formulas as: two sums of natural numbers are equal. Formula (6.30) holds by induction on n. Formula (6.31) (Worpitzky) holds also when the exponent is zero, provided that $k \neq 0$. One deduces (6.32). We prove this formula only in the case $k > 0$, case where the sum can be extendend to all indices $i < n$. We explicit the formula for $k = 5$.

(6.26)
$$\left\langle {n \atop m} \right\rangle = \left\langle {n \atop n-m-1} \right\rangle$$

(6.27)
$$\left\langle {n \atop 1} \right\rangle = 2^n - \binom{n+1}{1}, \quad \left\langle {n \atop 2} \right\rangle = 3^n - 2^n\binom{n+1}{1} + \binom{n+1}{2},$$

(6.28)
$$\left\langle {n \atop 4} \right\rangle = 4^n - 3^n\binom{n+1}{1} + 2^n\binom{n+1}{2} - \binom{n+1}{3}$$

(6.29)
$$\sum_{i=0}^{n} \left\langle {n+1 \atop i} \right\rangle = (n+1)!$$

(6.30)
$$\sum_{i=0}^{n} \left\langle {n+1 \atop i} \right\rangle \binom{i}{p} = \mathrm{S}_{n+1,n+1-p}.$$

(6.31)
$$k^{n+1} = \sum_{i=0}^{n} \left\langle {n+1 \atop i} \right\rangle \binom{k+i}{n+1}$$

$$\sum_{0<i<n} i^k = \sum_{i<k} \left\langle {k \atop i} \right\rangle \binom{n+i}{k+1}. \tag{6.32}$$

$$\sum_{i<n} i^5 = \binom{n}{6} + 26\binom{n+1}{6} + 66\binom{n+2}{6} + 26\binom{n+3}{6} + \binom{n+4}{6} \tag{6.33}$$

```
Lemma euler_small n p: 'Eu(n, n+p) = 0.
Lemma euler_small1 n p: n <= p -> 'Eu(n,p) = 0.
Lemma euler_nn n: 'Eu(n.+1,n) = 1.
Lemma eulern1 n: 'Eu(n,1) + 'C(n.+1, 1) = 2 ^ n.
Lemma eulern2 n: 'Eu(n,2) + 2^n * 'C(n.+1,1)  = 3 ^n + 'C(n.+1, 2).
Lemma eulern3 n:
  'Eu(n,3) +  3^n * (n.+1) + 'C(n.+1, 3) = 4 ^n + 2^n * 'C(n.+1, 2).

Lemma euler_sub n m: m <=n -> 'Eu(n.+1,m) = 'Eu(n.+1, n-m).
Lemma euler_sum n : \sum_(i<n.+1) 'Eu(n.+1,i) = (n.+1)`!.
Lemma euler_sum_aux n p:
    \sum_(i<n.+1) 'Eu(n.+1,i) * 'C(i,p) = 'Sj(n.+1, n.+1 - p).
Lemma euler_sum_pow k n :
   k ^n.+1 = \sum_(i<n.+1) 'Eu(n.+1,i) * 'C(k+i,n.+1).
Lemma sum_pow_euler n k:
   \sum_(i<n) i ^k.+1 =  \sum_(i<k.+1) 'Eu(k.+1,i) * 'C(n+i,k.+2).
Lemma F9aux2 n:  \sum_(i<n) i ^5 =
   'C(n, 6) + 26* 'C(n + 1, 6) +  66 * 'C(n + 2, 6) +
   26 * 'C(n + 3, 6) + 'C(n + 4, 6).
```

The sum of all Stirling numbers (for given n) is called the Bell number B_n. According to Exercise 7d, this is the number of partitions of a set with n elements. It satisfies the following relation

$$\mathrm{B}_{n+1} = \sum_{k=0}^{n} \binom{n}{k} \mathrm{B}_k \tag{6.34}$$

that follows from

$$\sum_k \binom{n}{k} \left\{ {k \atop p} \right\} = \left\{ {n+1 \atop p+1} \right\}. \tag{6.35}$$

```
Definition Bell n := \sum_(i<n.+1) 'St(n, i).
Lemma stir_Snn' n: 'St(n.+1, n) = \sum_(i<n.+1) i.
Lemma Bell_rec n: Bell n.+1 = \sum_(k<n.+1) 'C(n,k) * Bell k.
```

Formula (6.36) corresponds to Exercise 7a. Note that the generic term is also $(n)_i \mathrm{P}_{k,i}$ where $(n)_i = i!\binom{n}{i}$ is the falling factorial. Proof is by induction on. Formula (6.37) is an easy consequence.

$$\sum_{i=0}^{k} \binom{n}{i} \mathrm{S}_{k,i} = n^k, \tag{6.36}$$

$$\sum_{i<n} i^k = \sum_{i=0}^{k} \binom{n}{i+1} \mathrm{S}_{k,i}. \tag{6.37}$$

```
Lemma sum_nbsurj n k: \sum_(i<k.+1) 'Sj(k,i) * 'C(n,i) = n ^k.
Lemma sum_pow n k:
   \sum_(i<n) i ^k =  \sum_(i<k.+1) 'Sj(k,i) * 'C(n,i.+1).
```

We express n^k (for odd k) is as a linear combination of quantities of the form $\binom{n+i}{2i+1}$.

$$n^1 = \binom{n}{1}, \qquad n^3 = 6\binom{n+1}{3} + \binom{n}{1} \qquad n^5 = 120\binom{n+2}{5} + 30\binom{n+1}{3} + \binom{n}{1}.$$

```
Lemma n1bin n: n ^1 = 'C(n,1).
Lemma n2bin n: n ^2 = n * 'C(n,1).
Lemma n3bin n: n ^3 = 'C(n,1) + 6 * 'C(n.+1,3).
Lemma n4bin n: n ^4 =  n* 'C(n,1) +  6 * n*'C(n.+1,3) .
Lemma n5bin n: n ^5 =  'C(n,1) + 30 * 'C(n.+1,3) +  120 * 'C(n.+2,5).
Lemma n6bin n: n ^6 =  n*'C(n,1) + 30 * n* 'C(n.+1,3) +  120 *  n*'C(n.+2,5).
```

One can express n^i using $\binom{n}{j}$ for $j \le i$. Examples.

```
Lemma F6_aux n: n ^2 = 2 * 'C(n,2)  + 'C(n,1).
Lemma F7_aux n:  n ^ 3 = 6 * 'C(n, 3) +  6 * 'C(n, 2) + 'C(n, 1).
Lemma F8_aux n:
    n ^ 4 = 24 * 'C(n, 4) + 36 * 'C(n, 3) +  14 * 'C(n, 2) + 'C(n, 1).
Lemma F9_aux n:
    n ^ 5 = 120 * 'C(n, 5) +  240 * 'C(n, 4) + 150 * 'C(n, 3) +
     30 * 'C(n, 2) + 'C(n, 1).
```

We give some examples. Comments: The numbers in (6.43) are too big for Coq, so that we replace 5040 by 7!, and 15120 by 21.6!.

$$\sum_{i<n} i^2 = \binom{n}{2} + 2\binom{n}{3}. \tag{6.38}$$

$$\sum_{i<n} i^3 = \binom{n}{2} + 6\binom{n}{3} + 6\binom{n}{4}. \tag{6.39}$$

$$\sum_{i<n} i^4 = \binom{n}{2} + 14\binom{n}{3} + 36\binom{n}{4} + 24\binom{n}{5}. \tag{6.40}$$

$$\sum_{i<n} i^5 = \binom{n}{2} + 30\binom{n}{3} + 150\binom{n}{4} + 240\binom{n}{5} + 120\binom{n}{6}. \tag{6.41}$$

$$\sum_{i<n} i^6 = \binom{n}{2} + 62\binom{n}{3} + 540\binom{n}{4} + 1560\binom{n}{5} + 1800\binom{n}{6} + 720\binom{n}{7}. \tag{6.42}$$

$$\sum_{i<n} i^7 = \binom{n}{2} + 126\binom{n}{3} + 1806\binom{n}{4} + 8400\binom{n}{5} + 16800\binom{n}{6} + 15120\binom{n}{7} + 5040\binom{n}{8}. \tag{6.43}$$

```
Lemma F6aux n: \sum_(i<n) i ^2 =  'C(n,2) + 2 * 'C(n,3).
Lemma F7aux n:
   \sum_(i<n) i ^3 = 'C(n, 2) +  6 * 'C(n, 3) +  6 * 'C(n, 4).
Lemma F8aux n: \sum_(i<n) i ^4 =
   = 'C(n, 2) + 14 * 'C(n, 3) + 36 * 'C(n, 4)  + 24 * 'C(n, 5).
Lemma F9aux n:  \sum_(i<n) i ^5
  = 'C(n, 2) + 30 * 'C(n, 3) + 150 * 'C(n, 4)  + 240 * 'C(n, 5)
   + 120 * 'C(n, 6).
Lemma F10aux n:  \sum_(i<n) i ^6 =
  'C(n, 2) + (31 *2) * 'C(n, 3) + (90 * 6) * 'C(n, 4)  + (65 * 24) * 'C(n, 5)
   + (15 * 120) * 'C(n, 6) + 720 * 'C(n, 7).
Lemma F11aux n:  \sum_(i<n) i ^7 =
  'C(n, 2) + (63 * 2) * 'C(n, 3) + (301 * 3`!) * 'C(n, 4)
   + (350 * 4`!) * 'C(n, 5)
   + (140 * 5`!) * 'C(n, 6) + (21 * 6`!) * 'C(n, 7)  + 7`! * 'C(n, 8).
```

We express n^k and $\sum n^k$ for even k as a linear combination of quantities of the form T_k or U_k, defined by

$$\mathrm{U}_k(n) = \frac{n}{k+1}\binom{n+k}{2k+1} = \binom{n+k+1}{2k+2} + \binom{n+k}{2k+2},$$

$$\mathrm{T}_k(n) = \frac{2n+1}{2k+1}\binom{n+k}{2k} = \binom{n+k+1}{2k+1} + \binom{n+k}{2k+1}.$$

We have $\mathrm{T}_{k+1}(n) + \mathrm{U}_k(n) = \mathrm{T}_{k+1}(n+1)$, so that if x^n is a linear combination of $\mathrm{U}_i(n)$ with some coefficients, then $\sum x^i$ is is a linear combination of $\mathrm{T}_i(n)$ with the same coefficients.

$$n^2 = \mathrm{U}_0(n), \qquad n^4 = 12\mathrm{U}_1(n) + \mathrm{U}_0(n), \qquad n^6 = 360\mathrm{U}_3(n) + 60\mathrm{U}_1(n) + \mathrm{U}_0(n)$$

$$\sum n^2 = \mathrm{T}_1(n), \qquad \sum n^4 = 12\mathrm{T}_2(n) + \mathrm{T}_1(n), \qquad \sum n^6 = 360\mathrm{T}_3(n) + 60\mathrm{T}_2(n) + \mathrm{T}_1(n)$$

```
Definition U_nkbin n k := 'C(n+k,k.*2.+2) + 'C((n+k).+1,k.*2.+2).
Lemma UT_nkbin n k:  T_nkbin n k.+1 + U_nkbin n.+1 k =  T_nkbin n.+1 k.+1.
Definition T_nkbin n k := 'C(n+k,k.*2.+1) + 'C((n+k).+1,k.*2.+1).

Lemma U_nkbin_pr n k : n * 'C(n+k,k.*2.+1) =  (k.+1) * (U_nkbin n k).
Lemma T_nkbin_pr n k : n.*2.+1 * 'C(n+k,k.*2) =  (k.*2.+1) * (T_nkbin n k).
Lemma n2bin' n: n ^2 = U_nkbin n 0.
Lemma n4bin' n: n ^4 = (U_nkbin n 0) +  12 * (U_nkbin n 1).
Lemma n6bin' n:
     n ^6 = (U_nkbin n 0) + 60* (U_nkbin n 1)+ 360 * (U_nkbin n 2).

Lemma sn2bin n: \sum_(i<n.+1) i ^2 = T_nkbin n 1.
Lemma sn4bin' n:
    \sum_(i<n.+1) i ^4 = (T_nkbin n 1) +  12 * (T_nkbin n 2).
Lemma sn6bin' n:
 \sum_(i<n.+1) i ^6 = (T_nkbin n 1) + 60 * (T_nkbin n 2) + 360 * (T_nkbin n 3).
```

6.9.5 Sum of powers.

Let $s_k(n) = \sum_{i<n} i^k$. This is a polynomial of degree $k+1$ in n. Write as $\sum_{i\le k+1} p_{ki} n^i$. One has $p_{k0} = 0$ and $(i+1)p_{k,i+1} = \binom{k}{i}\mathrm{B}_{k-i}$, where B_j is the Bernoulli number. From $\mathrm{B}_0 = 1$, one deduces that the leading coefficient is $p_{k,k+1} = 1/(k+1)$. From $\mathrm{B}_1 = 1/2$, one deduces

$p_{k,k} = 1/2$. For $q > 0$, one has $p_{k,k-2q} = 0$. Moreover, $p_{k,k+1-2q}$ is negative for q even, positive otherwise.

Note that $s_k(n)$ is a polynomial in $a = \binom{n+1}{n}$ when k is odd. In the case $k = 1$, we get a, in the case $k = 3$, we get a^2; this is known as the Nicomachus formula.

$$\sum_{i<n} 1 = n \tag{6.44}$$

$$\sum_{i=0}^{n} i = \frac{n(n+1)}{2} = a, \quad \sum_{i<n} i = \binom{n}{2}. \tag{6.45}$$

$$\sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}, \qquad \sum_{i<n} i^2 = 2\binom{n}{3} + \binom{n}{2}. \tag{6.46}$$

$$\sum_{i=0}^{n} i^3 = \frac{[n(n+1)]^2}{4} = a^2. \tag{6.47}$$

$$\sum_{i=0}^{n} i^4 = \frac{n(n+1)(2n+1)(3n^2+3n-1)}{30}. \tag{6.48}$$

$$\sum_{i=0}^{n} i^5 = \frac{n^2(n+1)^2(2n^2+2n-1)}{12} = \frac{a^2(4a-1)}{3}. \tag{6.49}$$

```
Lemma F1a n: \sum_(i<n) 1 = n.
Lemma F1_aux n: \sum_(i<n) 1 = 'C(n,1).
Lemma F5aux n: \sum_(i<n) i = 'C(n,2).
Lemma F5 n: (\sum_(i<n.+1) i) * 2 = n * (n.+1).
Lemma F6 n: (\sum_(i<n.+1) i ^ 2 ) * 6 = n * (n.+1) * (2*n +1).
Lemma F7 n: (\sum_(i<n.+1) i ^ 3 ) * 4 = (n * (n.+1)) ^2.
Lemma F7b n: (\sum_(i<n.+1) i ^ 3 )  = 'C(n.+1,2) ^2.
Lemma F8 n: (\sum_(i<n.+1) i ^ 4 ) * 30 =
   n * (n.+1) * (n.*2.+1) * (3*n*n + 3*n -1).
Lemma F9 n: (\sum_(i<n.+1) i ^ 5 ) * 12 =
   n * n * (n.+1) * (n.+1)  * (2*n*n + 2*n -1).
Lemma F9' n (a := 'C(n.+1,2)):
    (\sum_(i<n.+1) i ^ 5 ) * 3 = a * a * (4*a -1).
```

Let $s'_k(n) = \sum_{i<n}(2i+1)^k$. We have $s'_k(n) = \mathrm{S}_k(2n) - 2^k\mathrm{S}_k(n)$. In some cases, the result has a nice expression.

$$\sum_{i<n} 2i+1 = n^2. \tag{6.50}$$

$$\sum_{i<n} (2i+1)^2 = \frac{n(4n^2-1)}{4} = 8\binom{n+1}{3} + n. \tag{6.51}$$

$$\sum_{i<n} (2i+1)^3 = n^2(2n^2-1). \tag{6.52}$$

$$\sum_{k<n} k!k = n! - 1. \tag{6.53}$$

```
Lemma F12 n: \sum_(i<n)  ((i*2).+1) = n ^2.
Lemma F13 n: (\sum_(i<n)  ((i*2).+1)^2) * 3 =  n * (n ^2 * 4 - 1).
Lemma F13' n: \sum_(i<n) (i.*2.+1)^2 = 8 * 'C(n.+1,3) + 'C(n,1).
Lemma F13 n: (\sum_(i<n)  (i.*2.+1)^2) * 3 =  n * (n ^2 * 4 - 1).
Lemma F13' n:
   \sum_(i<n) (i.*2.+1)^2 = 8 * 'C(n,3) + 8 * 'C(n,2) + 'C(n,1).
Lemma exp3_addn a b:  (a+b) ^3 =  a^3 + a^2 *b *3 + a*b^2 * 3 + b^3.
Lemma exp3_addn1 a:  (a.+1) ^3 =  a^3 + a^2 *3 + a * 3 + 1.
Lemma F14 n: (\sum_(i<n)  ((i*2).+1)^3) =  n^2 * (n ^2 * 2 - 1).
Lemma F22 n: \sum_(i<n) i `! * i = n `! .-1.
```

6.9.6 Other formulas

We show here the following formulas. For (6.55) and (6.56) we proceed as follows. We replace $i \le n$, i odd by $i = 2k+1$, $k \le n$. This gives more terms, by the additional terms are zero. Similarly, we replace i even by $i = 2k+2$. Using the binomial relation shows that (6.55) and (6.56) are equal. If we sum up these two quantities, we see that they are equal to the half of 2^n.

(6.54)
$$\sum_{i<m} \binom{n+i}{n} = \binom{n+m}{n+1}.$$

(6.55)
$$\sum_{i\le n,\, i \text{ odd}} \binom{n}{i} = 2^{n-1}.$$

(6.56)
$$\sum_{i\le n,\, i \text{ even}} \binom{n}{i} = 2^{n-1}.$$

```
Lemma pascal11 n: \sum_(i < n.+1) 'C(n, i) = 2 ^ n.
Lemma F23 n m: \sum_(i<m) 'C(n+i, n) = 'C(n+m, n.+1).
Lemma F24_a n: n > 0 ->
  \sum_(i<n) ( 'C(n, i.*2)) = \sum_(i<n) ('C(n, i.*2.+1)).
Lemma F24_b (n: nat): n > 0 ->
  \sum_(i<n.+1 | ~~ odd i) ( 'C(n, i)) = \sum_(i<n.+1 | odd i) ('C(n, i)).
Lemma F24 n: n > 0 ->
  \sum_(i<n.+1 | odd i) ( 'C(n, i)) = 2 ^ (n.-1).
Lemma F25 n: n > 0 ->
  \sum_(i<n.+1 | ~~ odd i) 'C(n, i) = 2 ^ (n.-1).
```

We show here

(6.57)
$$\sum_{i=0}^{r} \binom{t+r-i}{t}\binom{q+i}{q} = \binom{q+t+r+1}{t},$$

(6.58)
$$\sum_{k=q+1}^{n-p+q+1} \binom{n-k}{p-q-1}\binom{k-1}{q} = \binom{n}{p}.$$

The second formula is a consequence of the first, valid if $p \le n$ and $q < p$.

```
Lemma G5_a r t q:
  \sum_(i<r.+1) ( 'C(t+r - i, t) * 'C(q + i, q) ) = 'C(q+t+r+1,r).
Lemma G5_b n p q: p <= n -> q <p ->
 \sum_(q+1 <= k < (n - p + q + 1).+1) ( 'C(n-k, p-q-1) * 'C(k-1, q)) = 'C(n,p).
```

The relation (6.59) can be shown by induction, the other two relations follow.

$$\sum_{j=0}^{i}\binom{k}{j}\binom{l}{i-j}=\binom{k+l}{i}. \tag{6.59}$$

$$\sum_{k=0}^{n-p}\binom{n}{k}\binom{n}{p+k}=\binom{2n}{n-p}. \tag{6.60}$$

$$\sum_{i=n}^{n}\binom{n}{i}^2=\binom{2n}{n} \tag{6.61}$$

```
Lemma F36 k l i:
  (\sum_(j < i.+1) ('C(k, j)  *'C(l, (i - j)))) = 'C(k+l , i).
Lemma F37 n p: p<= n ->
  (\sum_(k < (n-p).+1) ('C(n, k)  *'C(n, p+k))) = 'C(2*n,n-p).
Lemma F38 n: \sum_(i<n.+1) 'C(n,i) ^2 = 'C(n.*2, n).
```

Catalan numbers. We first show

$$\frac{1}{n+1}\binom{2n}{n}=\binom{2n}{n}-\binom{2n}{n-1}. \tag{6.62}$$

```
Lemma bin_fact2n_a n : 'C(n.*2, n) * (n`!)^2 = (n.*2)`!.
Lemma bin_fact2n_b n : 0<n -> 'C(n.*2, n.-1) * (n`! * n.+1`!) = (n.*2)`!*n.
Lemma bin_fact2n_c n :'C(n.*2, n) * (n`! * n.+1`!) = (n.*2)`!*n.+1.
Lemma bin_fact2n_d n : 0<n ->  'C(n.*2, n.-1) <= 'C(n.*2, n).
Lemma bin_fact2n_e n : 0<n ->
   ('C(n.*2, n) - 'C(n.*2, n.-1)) * n.+1  = 'C(n.*2, n).
Lemma bin_fact2n_f n: 0 <n ->
  'C(n.*2.+2, n.+1) - 'C(n.*2.+2, n) = 'C(n.*2.+1, n.+1) - 'C(n.*2.+1, n.-1).
Lemma  bin_fact2n_g i: 0 < i ->
  (i.+2) * 'C(i.*2, i) + (3*i.+1) * 'C(i.*2, i.-1)
  = (i.+1) * 'C((i.+1).*2, i.+1).
```

The quantity introduced above is called the *Catalan* number. It satisfies a recurrence relation.

$$\mathrm{C}_n=\frac{1}{n+1}\binom{2n}{n},\qquad \mathrm{C}_0=1,\quad \mathrm{C}_{n+1}=\frac{4n+2}{n+2}\mathrm{C}_n.$$

From the recurrence we get $n!\mathrm{C}_n=2^n\prod_{i<n}2i+1$. We also have $\mathrm{C}_{n+1}=\binom{2n+1}{n+1}-\binom{2n+1}{n-1}$.

```
Definition catalan n := 'C(n.*2,n) %/(n.+1).

Lemma catalan0: catalan 0 = 1.
Lemma catalan_pos n: 0 < n -> catalan n = 'C(n.*2, n) - 'C(n.*2, n.-1).
Lemma catalan_prop n: n.+1 * catalan n = 'C(n.*2, n).
Lemma catalan_fact n: ((n`!)^2 * n.+1) * catalan n = (n.*2)`!.
Lemma catalan_rec n: (n.+2) * catalan n.+1 = (n.*2.+1.*2) * catalan n.
Lemma catalan_SSn n: 0 < n ->
  catalan n.+1 = 'C(n.*2.+1, n.+1) - 'C(n.*2.+1, n.-1).
Lemma catalan_prod n: (n.+1)`! * catalan n = \prod_(i<n) i.*2.+1.*2.
Lemma catalan_prod2 n: (n.+1)`! * catalan n = 2^n * \prod_(i<n) i.*2.+1.
```

A generalization is

$$\binom{n}{m+2}-\binom{n}{m}=\frac{2k+1}{m+2k+2}\binom{n+1}{m+2} \qquad (n=2m+2k+3).$$

```
Lemma catalan_spec m k (n := (m + k.+1).*2.+1):
   ( 'C(n, m) < 'C(n, m.+2)) &&
   ( (m+ k.*2.+3) * ('C(n, m.+2)-'C(n, m)) == 'C(n.+1, m.+2)* k.*2.+1 ).
```

Conider now the following definition

$$\mathrm{T}_{nk}=\binom{2n-1}{mn-k}-\binom{2n-1}{n-k-2}=\frac{2k+1}{n+k+1}\binom{2n}{n-k}.$$

```
Definition Tnk1 n k := 'C(n.*2.-1, n-k) - 'C(n.*2.-1, (n-k).-2).
Definition Tnk2 n k := ('C(n.*2, n-k) * (k.*2.+1)) %/ (n+k+1).

Lemma Tnk2_alt n k: k.+1 <= n ->
  (k.*2.+1)* 'C(n.*2, n-k) = (n+k).+1 * ('C(n.*2, (n-k)) - 'C(n.*2, n-k-1)).
Lemma Tnk2_div n k: k <= n ->
  (n + k + 1) * Tnk2 n k = 'C(n.*2, n-k) * (k.*2.+1).
Lemma Tnk2_div2 n k: k <= n ->
   (n-k)`! * (n + k + 1)`! * Tnk2 n k = n.*2 `!* (k.*2.+1).

Lemma Tnk1_eq_Tnk2 n k: k+2 <=n ->  Tnk1 n k =  Tnk2 n k.
Lemma Tnk2_n0 n: Tnk2 n 0 = catalan n.
Lemma Tnk2_nn n: Tnk2 n n = 1.
Lemma Tnk2_Snn n: Tnk2 n.+1 n = n.*2.+1.
Lemma Tnk2_sum n k: k < n ->
  (n-k)`!*(n+k+2)`! * ((Tnk2 n k) + (Tnk2 n k.+1)) = (n.*2.+1)`! * k.*2.+2.
```

Consider

$$\mathrm{T}_{nk}=\frac{n+1-k}{n+1}\binom{n+k}{n}.$$

This is called the Catalan Table http://oeis.org/A009766 This satisfies a recurrence relation. We first multiply by $n+1$, prove an equation, deduce that division is exact, and get $\mathrm{T}_{n+1,k+1}=\mathrm{T}_{n,k+1}+\mathrm{T}_{n+1,k}$. There is a trick: division is exact when $k \le n$, but $k=n+1$ is specil as the result is zero. This means that $\mathrm{T}_{n,n}$ and $\mathrm{T}_{n,n-1}$ are both equal to the Catalan number of index n.

```
Definition Cat' n k:= 'C(n+k,n) * (n.+1-k).
Definition Cat n k:= ('C(n+k,n) * (n.+1-k) ) %/ n.+1.

Lemma Cat_rec_aux n k: k <= n ->
  n.+1 * Cat' n.+1 k.+1 = n.+2 * Cat' n k.+1 + n.+1* Cat' n.+1 k.

Lemma Cat_dvd n k: k <= n -> Cat' n k = n.+1 * Cat n k.
Lemma Cat_rec n k: k <= n -> Cat n.+1 k.+1 = Cat n k.+1 + Cat n.+1 k.
Lemma CatnSn n:  Cat n n.+1 = 0.
Lemma Catnn n: Cat n n =  catalan n.
Lemma CatSnn n: Cat n.+1 n =  catalan n.+1.
Lemma bin_subnn k n: k <= n -> 'C(n.*2, n + k) =  'C(n.*2, n - k).
Lemma Cat_alt n k: k<=n -> Cat (n+k) (n-k) = Tnk2 n k.
```

We give here a third definition. It agrees with the previous definitions when $j < i$. Note that $M_{ij} = 0$ when $i < j$, $M_{ii} = 1$, $M_{i+1,i} = 2i + 1$.

```
Fixpoint Mi i j:=
  if i is i.+1 then
    if j is j.+1 then Mi i j + (Mi i j.+1).*2 + Mi i j.+2
    else Mi i 0 + Mi i 1
  else (j==0):nat.

Lemma Mi_rec i j: Mi i.+1 j.+1 = Mi i j + (Mi i j.+1).*2 + Mi i j.+2.
Lemma Mi_zero i j : i < j -> Mi i j = 0.
Lemma Mi_diag i : Mi i i = 1.
Lemma Mi_subdiag i : Mi i.+1 i = i.*2+1.
Lemma Mi_subsubdiag i : Mi i.+2 i = i.+2* i.*2.+1.
Lemma TMi_eq_Tnk2 i j: j <= i -> Mi i j = Tnk2 i j.
Lemma Mij_n0 n: Mi n 0 = catalan n.
```

6.9.7 Dyck paths

The aim of this section is to prove the the Catalan number satisfy the following recurrence

$$C_{n+1} = \sum_{k\le n} C_k C_{n-k}. \tag{6.63}$$

The idea is to compute, via different methods, the number of Dyck paths of a given length. We start with some ancillary lemmas.

```
Lemma take_rev n (T: Type) (s: seq T) : n <= size s ->
  take n (rev s) = rev (drop (size s - n) s).
Lemma eqseq_catr (T:eqType) (s1 s2 s3: seq T) :
  (s1 ++ s3 == s2 ++ s3) = (s1 == s2).
Lemma eqseq_cat_simp (T:eqType) (s1 s3: seq T) :
  (s1 ++ s3 == s3) = (nilp s1).
Lemma card_set_pred: forall (T:finType) (P:T -> bool),
  #|[set f:T | P f]| =  #|[pred f:T | P f]|.
```

Let's consider a sequence $(p_0, p_1, p_2, \dots)$ of points in the plane and the path formed of all line segments $[p_{i-1}, p_i]$. We assume that the first point is the origin and one goes from one point to the other by moving one unit of length to the right and one unit of length up or down. This means that the x-coordinate of p_i is i, while the y-coordinate y_i satisfies $y_0 = 0$, $y_{i+1} = y_i + a_i$, where a_i is $+1$ or -1. Note that y_i has the same parity as i, so that $y_i = 0$ says i even. A *Dyck path* of order n is a sequence $p_0, p_1, \dots, p_{2n}$ such that $y_i \ge 0$ for every i and $y_{2n} = 0$. It is uniquely characterized by the integers a_i, as $y_i = \sum_{j<i} a_j$.

The condition $\sum_{j<i} a_j \ge 0$ can be expressed as: the number of elements a_j with $j < i$ that are negative does not exceed the number of such elements that are positive. In what follows, we shall replace negative by false and positive by true. Moreover, if l is the sequence of the a_i up to j and l' the sequence up to $j+1$, then l' will be obtained by adjoining a_{j+1} in front of l. This simplifies proofs by induction, and we will never access an element of a list via its index.

Let $n_T(l)$ and $n_F(l)$ be the number of T or F (true or false) in l. We say that the list is balanced when $n_T(l) = n_F(l)$. A list is positive if either it is empty, or of the form a followed by s, where s is positive, and moreover, $n_T(l) \ge n_F(l)$. Note that the empty path is Dyck, and the shortest non-empty Dyck path is FT.

```
Definition DP_Tcount (l:seq bool) := count_mem true l.
Definition DP_Fcount (l:seq bool) := count_mem false l.
Definition DP_balanced l := (DP_Tcount l == DP_Fcount l).
Definition DyckFT := [:: false; true].

Fixpoint DP_pos l :=
  if l is a::l' then DP_pos l' && (DP_Tcount l >= DP_Fcount l) else true.

Definition Dyck_path l:= DP_balanced l && DP_pos l.
```

Consider a Dyck path l, and split it as $l = l_1 + +l_2$, where l_1 is formed of the k first elements of l. We have $n_{\mathrm{T}}(l_2) \geq n_{\mathrm{F}}(l_2)$. One deduces $n_{\mathrm{T}}(l_1) \leq n_{\mathrm{F}}(l_1)$. The converse holds, in particular we get: if we reverse the list, and swap true and false, we get a new Dyck path (This is geometrically obvious).

If we consider the sequence a_i we must revert it, and swap true and false. The concatenation of two positive (resp. Dyck) paths is positive (resp. Dyck). If the concatenation is Dyck, then one piece is Dyck if and only if the other part is also Dyck.

```
Lemma DP_posW l: DP_pos l -> DP_Fcount l <= DP_Tcount l.
Lemma DP_count l: DP_Tcount l + DP_Fcount l = size l.
Lemma Dyck_path_size l: Dyck_path l -> size l = (DP_Tcount l).*2.
Lemma Dyck_size_even s: Dyck_path s -> size s = (size s)./2.*2.
Lemma DP_count' m n l: size l = m + n ->
   (DP_Tcount l == m) = (DP_Fcount l == n).
Lemma DP_count_sym l1 l2:
  DP_balanced (l1 ++ l2) ->
  (DP_Fcount l2 <= DP_Tcount l2) = (DP_Tcount l1 <= DP_Fcount l1).
Lemma DP_prop2 l k (l1 := take k l) (l2 := drop k l) :  Dyck_path l ->
  (DP_Tcount l2 >= DP_Fcount l2) && (DP_Tcount l1 <= DP_Fcount l1).
Lemma DP_symmetry l: Dyck_path l -> Dyck_path (rev [seq ~~ x | x <- l]).

Lemma DP_pos_cat l1 l2: DP_pos l1 -> DP_pos l2 -> DP_pos (l1 ++ l2).
Lemma Dyck_path_cat l1 l2: Dyck_path l1 -> Dyck_path l2 ->
  Dyck_path (l1 ++ l2).
Lemma Dyck_sub_path l1 l2: Dyck_path l1 -> Dyck_path (l1++l2) -> Dyck_path l2.
Lemma Dyck_sub_path' l1 l2: Dyck_path l2 -> Dyck_path (l1++l2) -> Dyck_path l1.
Lemma DyckFT_Dyck: Dyck_path DyckFT.
```

We now count the number of lists l of size m with $n_{\mathrm{T}}(l) = n$. To such a list we associate a set E_l formed of those integers i such that the i-th element of l is T. If E is a set, its characteristic list χ_{E} is such that the i-th element of l is the boolean $i \in \mathrm{E}$. Note that the `nth` function takes as argument the value to be used when i is out of bounds, we use T here; this is not needed when l has the correct size. The main result is $n_{\mathrm{T}}(\chi_{\mathrm{E}}) = \mathrm{card}(\mathrm{E})$. So, the number of sequences with $n_{\mathrm{T}}(l) = n$ is the number of subsets of I_m with cardinal n, hence is $\binom{m}{n}$.

```
Definition char_seq m (X: {set 'I_m}) := [seq x \in X | x <- (enum 'I_m)].
Definition list_to_set m l := [set x:'I_m | nth false l x].

Lemma size_char_seq m (X: {set 'I_m}): size (char_seq X) = m.
Lemma char_seq_prop  m (X: {set 'I_m}) k (km: k < m):
  nth false (char_seq X) k = ((Ordinal km) \in X).
Lemma list_to_setK m l: size l = m -> char_seq (list_to_set m l) = l.
Lemma set_to_listK m  (X: {set 'I_m}) :  (list_to_set m (char_seq X)) = X.
```

```
Lemma list_to_set_inj n: injective (fun s: n.-tuple bool => list_to_set n s).
Lemma set_to_list_cardinal m (X: {set 'I_m}) :
  #|X| = DP_Tcount (char_seq X).
Lemma cardinal_tuple_nm n m:
  #|[set s: m.-tuple bool | DP_Tcount s == n]| = 'C(m,n).
```

We need two functions that split a list when either the count becomes negative, or when it becomes zero. This will gives two different ways of computing the number of Dyck paths.

The first function gives $a = \mathrm{S}_1(l)$ and $b = \mathrm{S}_2(l)$ with $l = a{+}{+}b$. We stop putting elements in b when the count becomes negative. It is obvious that $\mathrm{S}_2(l)$ is positive, but this is not needed. It is easy to see that l is positive if and only if $\mathrm{S}_1(l)$ is empty; and when $\mathrm{S}_1(l)$ is non-empty, then it ends with F and $\mathrm{S}_2(l)$ is a Dyck path.

```
Definition ends_withF l := (last true l == false).
Definition ends_withT l := (last false l == true).
Fixpoint DP_splita l :=
  if l is a::l' then let u:= (DP_splita l').1 in
     if(nilp u) then
       if  (DP_Tcount l >=  DP_Fcount l) then ([::],l) else ([:: a], l')
     else (a::u,(DP_splita l').2)
  else ([::],[::]).


Lemma DP_splita_recover l: (DP_splita l).1 ++ (DP_splita l).2 = l.
Lemma DP_splita_pos1 l: DP_pos l = nilp ((DP_splita l).1).
Lemma DP_splita_pos2 l: DP_pos (DP_splita l).2.
Lemma DP_splita_correct l (a:= (DP_splita l).1 ) :
  (nilp a) || ((ends_withF a) && (Dyck_path ((DP_splita l).2))).
```

We consider now an operation $m(s)$: we reverse s, swap the first digit, then reverse the result. This is the same as swapping the last digit of s. We define $\mathrm{M}(s)$ as the concatenation of $m(\mathrm{S}_1(l))$ and $\mathrm{S}_2(l)$. We have $\mathrm{S}_1(\mathrm{M}(l)) = m(\mathrm{S}_1(l))$ and $\mathrm{S}_2(\mathrm{M}(l)) = \mathrm{S}_2(l)$ (if $\mathrm{S}_1(l)$ is empty, then $\mathrm{M}(l) = l$), so that $\mathrm{M}(\mathrm{M}(l)) = l$.

We have $n_\mathrm{T}(m(l)) = n_\mathrm{F}(l) - 1$ if the last element of l is F. So assume l balanced, $\mathrm{S}_1(l)$ non-empty. Then $n_\mathrm{T}(\mathrm{M}(l)) = n - 1$ and $n_\mathrm{F}(\mathrm{M}(l)) = n + 1$ (where $n = n_\mathrm{T}(l) = n_\mathrm{F}(l)$). Conversely, if $n_\mathrm{T}(l) = n - 1$ and $n_\mathrm{F}(l) = n + 1$ then $\mathrm{M}(l)$ is balanced and $n_\mathrm{T}(\mathrm{M}(l)) = n$.

```
Definition swap_but_first l :=
   if l is a ::s then a:: [seq ~~ x | x <- s] else  nil.
Definition swap_but_last l := rev (swap_but_first (rev l)).
Definition Dyck_modify l :=
  (swap_but_last (DP_splita l).1) ++ (DP_splita l).2.

Lemma Dyck_modify_split l (s :=  (Dyck_modify l)) :
  ((DP_splita s).1 == swap_but_last (DP_splita l).1) &&
  ((DP_splita s).2 == (DP_splita l).2).
Lemma Dyck_modify_inv l: (Dyck_modify (Dyck_modify l)) = l.

Lemma swap_but_last_size l: size (swap_but_last l) = size l.
Lemma swap_but_last_nil l: nilp (swap_but_last l) = nilp l.
Lemma Dyck_modify_size l: size (Dyck_modify l) = size l.

Lemma swap_but_last_count l:
```

```
  ends_withF l -> (DP_Tcount(swap_but_last l)).+1 = DP_Fcount l.
Lemma Dyck_modify_size l: size (Dyck_modify l) = size l.
Lemma Dyck_modify_tcount l (s:=  (Dyck_modify l)) :
  DP_balanced l ->
  ((DP_splita l).1 == [::]) = false  ->
  ((DP_Tcount s).+1 == DP_Tcount l) && (DP_Fcount s == (DP_Tcount l).+1).
Lemma Dyck_modify_tcount_bis l (s:=  (Dyck_modify l)) :
  DP_Fcount l = (DP_Tcount l).+2 ->
  ((DP_Tcount s == (DP_Tcount l).+1) && (DP_balanced s)).
```

If we express the last two results in terms of sets, we get: the number of subsets of I_{2n+2} of cardinal n is equal to the number of subsets of cardinal $n+1$ whose characteristic list is not a Dyck path. Hence, the number of lists of size $2n+2$ with $n_\mathrm{T}(l) = n+1$ that are not Dyck paths has cardinal $\binom{2n+2}{n}$. So, the number of Dyck paths is the difference between this number and another binomial number. It is hence the Catalan number.

```
Lemma cardinal_swap_but_last n:
  #|[set X:{set 'I_(n.*2.+2) } | #|X| ==n ]| =
  #|[set X:{set 'I_(n.*2.+2)} | (#|X| == n.+1) && ~~Dyck_path (char_seq X) ]|.
Lemma cardinal_tuple_nSSn n:
  #|[set s: (n.*2.+2).-tuple bool | (DP_Tcount s == n.+1) && ~~Dyck_path s ]|
   = 'C(n.*2.+2,n).
Lemma cardinal_Dyck_path n:
  #|[set s: (n.*2).-tuple bool | Dyck_path s ]| = catalan n.
```

We now split out list l according to $\mathrm{S}_3(l) = u$ and $\mathrm{S}_4(l) = v$. We stop putting elements in v when the result becomes balanced. Note (this is east to prove) that, if the last element of l is F, then we put everything in u. On the other hand, if l ends with T, then $\mathrm{S}_4(l)$ end also with T. If u is non-empty, then v has to be a Dick path. In particular, if l is a Dyck path, so are u and v.

```
Fixpoint DP_splitb l :=
  if l is a::l' then let u:= (DP_splitb l').1 in let v:= (DP_splitb l').2 in
     if(nilp u) then
       if (nilp v) then if a then ([::], [::a]) else ([::a], [::])
       else if (DP_balanced v) then ([::a],v) else ([::],a::v)
     else (a::u,v)
  else ([::],[::]).

Lemma DP_splitb_recover l: (DP_splitb l).1 ++  (DP_splitb l).2 = l.
Lemma DP_splitn_lastT l:  ends_withT (DP_splitb (rcons l true)).2.
Lemma DP_splitb_pos2 l: nilp (DP_splitb l).1 || Dyck_path (DP_splitb l).2.
Lemma DP_splitb_Dyck12 l: Dyck_path l ->
   Dyck_path (DP_splitb l).1 && Dyck_path (DP_splitb l).2.
```

If v is a list, we denote by $\bar{v}$ the quantity FvT; conversely, we denote by $\underline{v}$ the list obtained from v by removing the first and last element. The height of a path is the maximum of the y-coordinate. These two operations increase or decrease the height, so we call them "raise" and "lower". If v is a Dyck path, then $\bar{v}$ is called *irreducible*, for reasons explained below.

```
Definition DP_lower l := (behead (behead (belast true l))).
Definition DP_raise v :=  false :: rcons v true.
Definition Dyck_irred l :=
```

```
  (l == DP_raise (DP_lower l))  && Dyck_path (DP_lower l).

Lemma DP_lowerK v:  (head true v == false) && ends_withT v ->
  DP_raise (DP_lower v) = v.
Lemma DP_raiseK v: DP_lower (DP_raise v) = v.
Lemma DP_raise_inj v1 v2: (DP_raise v1 == DP_raise v2) = (v1 == v2).
Lemma DP_count_catTF v :  (DP_balanced (DP_raise v)) = (DP_balanced v).
Lemma Dyck_path_augment l: Dyck_path l -> Dyck_path (DP_raise l).
Lemma Dyck_irred_Dyck l: Dyck_irred l -> Dyck_path l.
Lemma DP_lower_DyckK s:
  Dyck_path s -> ~~ nilp s -> DP_raise (DP_lower s) = s.
```

The quantity $\underline{S_4(l)}$ will be denoted by $S'_4(l)$. This is a Dyck path; the proof is a bit contrieved (we assume that l, as well as some other lists are non-empty). We first note that is l is a Dyck path, we have $S_4(u++l) = S_4(l)$, whatever u. In particular, $S_4(S_4(l)) = S_4(l))$. Let $v' = S_4(l)$, $a = S_1(v')$ and $b = S_2(v')$. If a is empty, then $b = v'$ is a Dyck path. Otherwise a ends with F. So v is a part of a, followed b', which is by F (the last element of a), followed by b, followed by T. So $S_4(v) = S_4(b')$; but $S_4(v) = v$, and $S_4(b')$ is a part of b', wich is a non-trivial part of v (remember that a is non-empty); aburd.

The converse is easier: if l is the concatenation of a Dyck path u, followed by F, followed by a Dyck path v, followed by T, then $u = S_3(l)$ and $u = S'_4(l)$. We denote this operation by $N(u, v)$. Note that, if the sizes are given, then N is injective.

It trivially follows: an irreducible Dyck path is a non-empty Dyck path that is never the concatenation of two non-trivial Dyck paths.

```
Lemma DP_splitb_prop1 l: Dyck_path l -> ~~nilp l ->
   ~~(nilp ((DP_splitb l).2)).

Lemma DP_splitb_prop2a l (v:= (DP_splitb l).2) : Dyck_path l -> ~~nilp l ->
  DP_raise (DP_lower v) = v.
Lemma DP_splitb_race1 s1 s2: Dyck_path s2 -> ~~ nilp s2 ->
    (DP_splitb (s1 ++ s2)).2 = (DP_splitb s2).2.
Lemma DP_splitb_race2 l (v:= (DP_splitb l).2): Dyck_path l -> ~~ nilp l ->
   (DP_splitb v).2 = v.
Lemma DP_splitb_prop3 l: Dyck_path l -> ~~nilp l -> Dyck_irred (DP_splitb l).2.
Lemma DP_splitb_prop4 l1 l2 (s := l1 ++ DP_raise l2):
   Dyck_path l1 -> Dyck_path l2 ->
   [&& Dyck_path s, (DP_splitb s).1 == l1 & (DP_splitb s).2 ==  DP_raise l2 ].

Lemma Dyck_irrred_nnil l: Dyck_irred l -> ~~ nilp l.
Lemma Dyck_irrred_prop1 l1 l2: Dyck_path l1 -> Dyck_path l2 ->
  Dyck_irred (l1++l2) -> (nilp l1 ) || (nilp l2).
Lemma Dyck_irrred_prop2 l: Dyck_path l -> ~~ nilp l ->
  (forall l1 l2, Dyck_path l1 -> Dyck_path l2 -> l = l1 ++ l2 ->
     (nilp l1 ) || (nilp l2)) ->
  Dyck_irred l.
```

We introduce now $k(l)$, the half of the size of $S'_4(l)$. If l is Dyck path, of size $2n+2$, then l is non-empty and $S'_4(l)$ is a Dyck path, thus have even size. Note that $S'_4(l)$ has size $2k+2$, so that $k \le n$. We now partition the set of Dyck paths, according to the value of k, into sets E_k. For the ease of the argument, we show that E_k is non-empty. Let T_i be the sequence formed of i F, followed by the same number of T. The sequence $N(T_{n-k}, T_k)$ is a solution.

Let's show the main result: (6.63). The LHS is the cardinal of the set E_{n+1} of Dyck paths of length $2n+2$. Let T_i be the set of tuples of size $2i$, so that E is a a subset of T_{n+1}. The relation $k \le n$ says that we may consider k as a function $T_{n+1} \to I_{n+1}$ Now, the cardinal of E is $\sum_{j\in J}\sum_l 1$. Here J is the set of all $k(l)$, where l is a Dyck path. As noted above, this is I_{n+1}, and we can rewrite the cardinal as $\sum_{j\le n}\sum_l 1$. The inner sum is over all Dyck paths l such that $k(l) = j$. It suffices to show that this sum is $C_j C_{n-j}$, the cardinal of $E_j \times E_{n-j}$. The function N introduced above can be considered as an injection $T_j \times T_{n-j} \to T_{n+1}$, and there is is difficulty in showing that it is a bijection $E_j \times E_{n-j} \to E_{n+1}$. Total size of the proof: 600 lines.

```
Definition DP_splitb_size l := (size (DP_lower(DP_splitb l).2))./2.

Lemma DP_splitb_size_correct l n (k :=  DP_splitb_size l):
  Dyck_path l -> size l = n.+1.*2 ->
  (size (DP_splitb l).2 == k.*2.+2) && (k <= n).
Lemma Dyck_path_exists1 n (l := (nseq n false) ++  (nseq n true)):
   (DP_Tcount l == n) && (Dyck_path l).
Lemma Dyck_path_exists2 n m (ln := (nseq n false) ++  (nseq n true))
  (lm := (nseq m false) ++  (nseq m true)) (l := ln ++ DP_raise lm):
  [&& size l == (n+m).+1.*2, (Dyck_path l) &  DP_splitb_size l == m].

Lemma catalan_rec2 n:
  catalan n.+1 = \sum_(i<n.+1) catalan i * catalan (n-i).
```

Let's show that C_n is the number of Dyck paths of size $2(n+1)$ with no peak of height 2. A peak is a local maximum of y, this means $y_{i-1} < y_i > y_{i+1}$. As mentioned above, if s has a peak of height k; then $\bar{s}$ has a peak at height $k+1$. Conversely, if l has a peak at height $k+1$; one of its components has the form $\bar{s}$ where s has a peak at height k (for $k=0$, this reads: s is empty).

We obtain the decomposition in irreducible components $S_5(l)$ by recursively applying our split algorithm: it is formed of $S_5(S_3(l))$ followed by $S_4(l)$. The recursion terminates when $S_4(l)$ is empty; in the other case $S_3(l)$ is smaller than l. The actual induction is over an integer. By construction l is the concatenation of the elements of $S_5(l)$; if l is a Dyck path, each element of $S_5(l)$ is irreducible. Conversely, If v is a list of irreducible Dyck paths, whose concatenation is l, then it is $S_5(l)$.

```
Fixpoint DP_splitc_aux n l:=
  if n is n'.+1 then
    if nilp l then [::] else let uv:= DP_splitb l in
      if nilp uv.2 then [:: l] else rcons (DP_splitc_aux n' uv.1) uv.2
  else [::].
Definition DP_splitc l:= DP_splitc_aux (size l) l.

Lemma DP_splitb_size_rec s: nilp (DP_splitb s).2 = false ->
  size (DP_splitb s).1 < size s.
Lemma DP_splitc_rec l:
   DP_splitc l = if nilp l then [::] else
     if nilp (DP_splitb l).2 then  [:: l]
     else rcons (DP_splitc (DP_splitb l).1)  (DP_splitb l).2.
Lemma DP_splitc_recover l: flatten (DP_splitc l) =  l.
Lemma DP_splitc_correct l: Dyck_path l -> all Dyck_irred (DP_splitc l).
Lemma Dyck_flatten d: all Dyck_irred d -> Dyck_path (flatten d).
Lemma DP_splitc_unique l d:
  all Dyck_irred d -> flatten d = l -> DP_splitc l = d.
```

```
Lemma DP_splitc_cat  l1 l2:  Dyck_path l1 -> Dyck_path l2 ->
  DP_splitc (l1 ++ l2) = DP_splitc l1 ++ DP_splitc l2.
Lemma DP_splitc_irred l:  Dyck_irred l -> DP_splitc l = [:: l].
```

Let's say that l has no peak at height 1, in short $\mathrm{H}_1(l)$, if no irreducible component has size 2. Since an irreducible component has the form $\bar{s}$, this means that no component is FT. This means that l is never of the form l_1 followed by FT followed by l_2, where l_1 and l_2 are Dyck paths. We give here a decomposition d of l into parts that are either FT or maximal H_1. We say that d is a decomposition of l if its concatenation is l. We assume moreover, that either d is empty or d is s followed by d_1 and (1) d_1 satisfies the condition, (2) s is FT or non-empty and $\mathrm{H}_1(\underline{s})$, and (3), either l is FT, or d_1 is empty, or d_1 starts with FT. In the algo, the condition "s is FT" is replaced by s is of size two. We first show uniqueness.

```
Fixpoint DP_ph1_aux (d: (list (list bool))) :=
  if d is l::d then
      DP_ph1_aux d && [|| size l == 2, nilp d | size (head nil d) == 2]
  else true.

Definition DP_NH1 l:= all (fun s => size s != 2) (DP_splitc l).
Definition Dyck_NH1 l:= Dyck_path l && DP_NH1 l.
Definition DP_ph1_aux2 l :=
  (l ==  DyckFT) ||  (~~ nilp l && (Dyck_NH1 l)).
Definition DP_ph1 d:= (DP_ph1_aux d) &&  (all DP_ph1_aux2 d).

Lemma DP_ph1_simp a l:  DP_ph1 (a:: l) =
    [&& DP_ph1 l, [|| size a == 2, nilp l | size (head nil l) == 2]
     & DP_ph1_aux2 a].
Lemma DP_irred_size2 l: Dyck_irred  l -> size l == 2 -> l = DyckFT.
Lemma DP_ph1_Dyck_aux l: DP_ph1_aux2 l -> Dyck_path l.
Lemma DP_ph1_Dyck d : DP_ph1 d -> Dyck_path (flatten d).
Lemma DP_ph1_nonempty d : DP_ph1 d -> all (fun z => ~~ nilp  z) d.
Lemma DP_ph1_size2 l: DP_ph1_aux2 l -> size l == 2 -> l = DyckFT.
Lemma DP_NH1_prop1 x y: Dyck_path y -> ~~ Dyck_NH1 ((y ++ DyckFT) ++ x).
Lemma DP_NH1_prop2 x: ~~ Dyck_NH1(DyckFT ++ x).
Lemma DP_ph1_unique d1 d2 :
  DP_ph1 d1 -> DP_ph1 d2 -> flatten d1 = flatten d2 -> d1 = d2.
```

We prove here existence. The idea isto merge consecutive elements, not of size one. We denote by $\mathrm{S}_6(l)$ the result. This is the only decomposition of l satisfying the previous conditions.

```
Fixpoint DP_split_resize (l: list (list bool)):=
  if l is a :: l then let l' := DP_split_resize l in
   if (size a == 2) then a::l'
   else if l' is u::v then if(size u == 2) then a :: u :: v
  else (a++u) ::v  else [:: a]
  else [::].

Lemma DP_split_resize_flatten l: flatten l = flatten (DP_split_resize l).
Lemma DP_split_resize_correct1 l: DP_ph1_aux  (DP_split_resize l).
Lemma DP_split_resize_correct2 l: all Dyck_irred l ->
   DP_ph1 (DP_split_resize l).
Lemma DP_split_resize_correct3 l (d := (DP_split_resize (DP_splitc l))):
  Dyck_path l -> DP_ph1 d && (flatten d == l).
```

Variant. Let $S_7(d)$ be the following function: we merge an element x and the element that follows y, unless y starts with FT. The result is a list such that each element (with the possible exception of the head) starts with FT. If the head of d is FT, this condition is always satisfied. Assume that, whenever x and y are consecutive in d, then at least one of them starts with FT. Note the potentiel merge is between x and the head result of the merge, so either y or the concatenation of y and something else. So, if y starts with FT we do not merge. Hence if x does not start with FT we do not merge. Assume further that each element is FT or non-empty H_1. Let's denote by $Q(l)$ the property that l is the concatenation of FT and a (possibly empty) H_1. We pretend that $S_7(d)$ is formed of element satisfying this condition, except that the head could be a FT or a H_1. The proof is a bit tricky. One deduces: if the first element of d is FT, then all elements of $S_7(d)$ satisfy Q.

So, if l is a Dyck path that starts with FT, if we apply S_6, then S_7, we get a decomposition where every terms satisfies Q.

```
Definition starts_withFT l:= if l is [:: false, true & l] then true else false.
Definition DP_FT_DH1 l := starts_withFT l && Dyck_NH1 (drop 2 l).
Definition prependFT l := [:: false, true & l].

Fixpoint DP_resizeb (l: list (list bool)) :=
  if l is a::l then let l' :=  DP_resizeb l in
    if (nilp l' || (starts_withFT (head nil l'))) then  a::l'
    else  (a ++  (head nil l')):: (behead l')
  else nil.
Definition DP_resizeb_condition s :=
  nilp s || ((all DP_FT_DH1 (behead s)) &&
    (DP_FT_DH1 (head nil s) || (DP_ph1_aux2 (head nil s)))).

Lemma DP_resizeb_flatten l: flatten l = flatten (DP_resizeb l).
Lemma starts_withFT_prop l: starts_withFT l = (l == prependFT (drop 2 l).
Lemma DP_NH1_prop3 b:  starts_withFT b -> ~~(Dyck_NH1 b).
Lemma DP_NH1_prop4 a x: Dyck_path a -> starts_withFT x ->
  ~~(Dyck_NH1 (a ++ x)).
Lemma starts_withFT_alt x: starts_withFT x = ((take 2 x) == DyckFT).
Lemma DP_resizeb_prop1 l: all starts_withFT (behead (DP_resizeb l)).
Lemma DP_resizeb_prop2 l: all starts_withFT (DP_resizeb( DyckFT ::l)).
Lemma DP_resizeb_prop3 d: DP_ph1 d -> DP_resizeb_condition (DP_resizeb d).

Lemma DP_resizeb_prop4 d:
   DP_ph1 d -> (all DP_FT_DH1 ((DP_resizeb (DyckFT :: d)))).
Lemma DP_split_resizeb_correct l
  (d := ( DP_resizeb (DP_split_resize (DP_splitc l)))):
  Dyck_path l -> starts_withFT l -> (all DP_FT_DH1 d) && (flatten d == l).

Lemma DP_FT_DH1aux_unique d1 d2:
  DP_resizeb_condition d1 ->  DP_resizeb_condition d2 ->
  flatten d1 = flatten d2 -> d1 = d2.
Lemma DP_FT_DH1_unique d1 d2:
  all DP_FT_DH1 d1 -> all DP_FT_DH1 d2 -> flatten d1 = flatten d2 -> d1 = d2.
```

We consider now two operations T'_r and T'_l (raise, lower), that are inverse functions. If l is FTs, then $T'_r(l)$ is FsT, and if l is FsT, then $T'_l(l)$ is FTs. Given a Dyck path l, we consider its irreducible components l_i, apply T'_l to each part, concatenate this results, and remove the first FT. We denote this by $T_l(l)$. Conversely, given a list l, we add FT in front, apply S_6 and S_7. This gives a list of items satisfying Q. To each item, we apply T'_r and then flatten; the result is T_r. Let u be the first list, u' the second. Denote by u^+ the concatenation of u. We have $u^+ = \mathrm{FT}l$; each element of u' is irreducible, so that the decomposition of u'^+ is u'; finally, if we apply T'_l to the elements of u', we get u. This implies $\mathrm{T}_l(\mathrm{T}_r(l)) = l$. Note that T_r increases the size by two, and T_l decreases it.

Note that $\mathrm{T}_r(l))$ satisfies H_2. Assume x satisfies H_2 and is non-empty; then its is of the form $\mathrm{T}_r(l))$; in fact $x = \mathrm{T}_r(\mathrm{T}_l(x)))$. This proves the result.; there are as many Dyck paths of size n that Dyck paths of size $n+1$ that are NH2.

```
Definition Dyck_lp_aux l := prependFT (DP_lower l).
Definition Dyck_rp_aux l := DP_raise (drop 2 l).
Definition DP_NH2 l:=
   all (fun s => (DP_NH1 (DP_lower s))) (DP_splitc l).
Definition Dyck_NH2 l:= Dyck_path l && DP_NH2 l.

Definition Dyck_lp l :=
  drop 2 (flatten [seq  Dyck_lp_aux s | s <- DP_splitc l]).
Definition Dyck_rp_aux2 l :=
  DP_resizeb (DP_split_resize (DP_splitc (prependFT l))).

Definition Dyck_rp l :=
  (flatten [seq  Dyck_rp_aux s | s <- Dyck_rp_aux2 l]).

Lemma Dyck_rp_prop l
  (u:= Dyck_rp_aux2 l) (u' := [seq Dyck_rp_aux s | s <- u]) :
  Dyck_path l ->
  [&& all DP_FT_DH1 u, flatten u == prependFT l, DP_splitc (flatten u') == u' &
   [seq Dyck_lp_aux s | s <- u'] == u ].
Lemma Dyck_lp_auxK l: Dyck_irred l ->  Dyck_rp_aux(Dyck_lp_aux l) = l.
Lemma Dyck_rp_auxK l: starts_withFT l ->
   Dyck_lp_aux(Dyck_rp_aux l) = l.
Lemma Dyck_rp_aux_size x: starts_withFT x -> size (Dyck_rp_aux x) = size x.
Lemma Dyck_lp_size n l: Dyck_path l -> size l = n.+1.*2 ->
  size (Dyck_lp l) = n.*2.
Lemma Dyck_rp_size  l: Dyck_path l ->
  size (Dyck_rp l) = (size l).+2.
Lemma Dyck_lp_Dyck n l: Dyck_path l -> size l = n.+1.*2 ->
  Dyck_path (Dyck_lp l).
Lemma Dyck_rp_Dyck l: Dyck_path l ->  Dyck_path (Dyck_rp l).
Lemma Dyck_rpK l: Dyck_path l -> Dyck_lp (Dyck_rp l) = l.
Lemma Dyck_rp_H2 l: Dyck_path l -> Dyck_NH2 (Dyck_rp l).
Lemma Dyck_lpK n l: Dyck_NH2 l -> size l = n.+1.*2 ->
   Dyck_rp (Dyck_lp l) = l.

Lemma cardinal_Dyck_NH2 n:
  #|[set s: (n.+1.*2).-tuple bool | Dyck_NH2  s ]| = catalan n.
```

6.9.8 Derangements

We introduce here a quantity p_n, defined by $p_{n+1} = n(p_n + p_{n-1})$, called the number of derangements.

```
Fixpoint nder_rec n :=
  if n is n1.+1 then
    if n1 is n2.+1 then n1 *(nder_rec n1 + nder_rec n2)
```

```
    else 0
  else 1.

Definition nder := nosimpl nder_rec.

Lemma nderE : nder = nder_rec.
Lemma nder0: nder 0 = 1.
Lemma nder1: nder 1 = 0.
Lemma nderS n : nder (n.+2) = (n.+1) * (nder n.+1 + nder n).
```

We gave a reccurence of order one: $p_{n+1} = (n+1)p_n + (-1)^{n+1}$.

```
Lemma nbder_gt0 n: 0 < nder n.+2.
Lemma nbder_even_gt0 n: ~~ odd n -> 0 < nder n.
Lemma nderS' n (m := (n.+1) *(nder n)):
   nder (n.+1) = if (odd n) then m.+1 else m.-1.
```

It follows $\sum_i \binom{n}{i} p_i = n!$. Proof: if we had $p_{n+1} = (n+1)p_n$ then the result would be obvious. The additional terms cancel because of (6.55) and (6.56).

```
Lemma split_sum_even_odd (F: nat -> nat) n:
  \sum_(i<n) (F i) = \sum_(i< (n.+1)./2) (F i.*2) + \sum_(i< n./2) (F i.*2.+1).

Lemma nbder_gt0 n: 0 < nder n.+2.
Lemma nderS' n (m := (n.+1) *(nder n)):
   nder (n.+1) = if (odd n) then m.+1 else m.-1.
Lemma nder_sum n: \sum_(i< n.+1) 'C(n,i) * nder i = n `!.
```

6.9.9 Formulas on Z

We shall now consider formulas of the form $\sum_i (-1)^i x_i$. We use a notation that makes this easier to input (in the first case, x is a natural number, and we have to specify that the -1 is in $\mathbf{Z}$; in the second case, x is in a ring and the -1 comes from the same ring).

We prove the following by induction on k. If $k = n+1$ the RHS vanishes.

$$\sum_{i=0}^{k} (-1)^i \binom{n+1}{i} = (-1)^k \binom{n}{k}. \tag{6.64}$$

```
Notation with_sign k x := ( (-1%:Z) ^+ k  *+  x ).
Notation with_sgn k x := ( (-1) ^+ k  *  x ).

Lemma F26a (k n: nat):
  \sum_(i<k.+1) ( with_sign i 'C(n.+1, i)) = with_sign k 'C(n, k).
Lemma F26b (n: nat):
  \sum_(i<n.+2) ( with_sign i 'C(n.+1, i)) = 0.
```

One deduces, thanks to (6.22),

$$\sum_{k=0}^{q} (-1)^k \binom{j+q}{j+k} \binom{j+k}{j} = \delta_{q0} \tag{6.65}$$

and, if $j \le q$ then

$$\sum_{k=0}^{q} (-1)^k \binom{q}{k} \binom{k}{j} = (-1)^j \delta_{qj}. \tag{6.66}$$

```
Lemma bin_inva (j q: nat):
  \sum_(k<q.+1) with_sign k ('C(j+q, j+k) * 'C(j+k, j)) =  (q==0%N).
Lemma bin_invb (j q: nat): j <= q ->
  \sum_(k<q.+1) with_sign k ('C(q, k) * 'C(k, j)) =
     with_sgn j (Posz(q - j == 0)%N).
```

Consider

$$f(p)=\sum_{i=0}^{p}\binom{p}{i}g(i), \tag{6.67}$$

$$g(p)=\sum_{i=0}^{p}(-1)^i\binom{p}{i}f(p-i)=(-1)^p\sum_{i=0}^{p}(-1)^i\binom{p}{i}f(i). \tag{6.68}$$

We write these formulas as $f=\mathscr{B}(g)$ and $g=\mathscr{B}'(f)$. The two formulas are equivalent. This can be restated as: the $p\times p$ matrix with entries $\binom{i}{j}$ is invertible, and its inverse is the matrix with entries $(-1)^{i+j}\binom{i}{j}$. One deduces, for every f:

$$g(p)+g(p+1)=g^+(p), \qquad [g=\mathscr{B}'(f), g^+=\mathscr{B}'(f^+), f^+(i)=f(i+1)] \tag{6.69}$$

```
Definition bin_conv_dir (g: nat -> int) n :=
  \sum_(i<n.+1) (g i) *+ 'C(n,i).
Definition bin_conv_inv (f: nat -> int) n :=
  with_sgn n (\sum_(i<n.+1)  with_sgn i  (f i) *+ 'C(n,i)).

Lemma double_pow_m1 (i: nat) (a : int):  with_sgn i (with_sgn i a) = a.
Lemma bin_conv_inv1 (f: nat -> int) n:
  bin_conv_inv f n = \sum_(i<n.+1) with_sgn i (f (n- i)%N) *+ 'C(n,i).
Lemma bin_conv_inv2 (f g: nat -> int) n:
  (forall i, i<=n -> f i = bin_conv_dir g i) ->
  (forall i, i<=n -> g i = bin_conv_inv f i).
Lemma bin_conv_inv3 (f g: nat -> int) n:
  (forall i, i<=n -> f i = bin_conv_inv g i) ->
  (forall i, i<=n -> g i = bin_conv_dir f i).
Lemma bin_conv_inv_rec (f g: nat -> int) n:
  (forall i, i<=n.+1 -> g i = bin_conv_inv f i) ->
  (g n) + (g n.+1) =  bin_conv_inv (fun i => (f i.+1)) n.
```

We consider now the case where $g(i)=n^i$. We write $\mathrm{S}_{n,p}$ instead of f and pretend that this is the number of surjections of a set of n elements onto a set of p elements. We have $\mathrm{S}_{0,0}=1$, $\mathrm{S}_{n+1,0}=0$, $\mathrm{S}_{0,p+1}=0$, $\mathrm{S}_{n+1,1}=1$. We have $\mathrm{S}_{1,p}=0$ and $\mathrm{S}_{n,2}=2^n-2$ (let f be a surjection $\mathrm{E}\to\{,;\}$ the set $f^{-1}(0)$ can be any subset of E, except E and the empty set).

```
Definition nb_surj n p := bin_conv_inv  (fun i => (i ^n)%N) p.

Lemma nb_surj_00 : nb_surj 0 0 = 1.
Lemma nb_surj_n0  n: nb_surj n.+1 0 = 0.
Lemma nb_surj_0p p: nb_surj 0 p.+1 = 0.
Lemma nb_surj_n1  n: nb_surj n.+1 1 = 1.
Lemma nb_surj_n2 n: nb_surj n.+1 2 = (2 ^n.+1 - 2)%N.
Lemma nb_surj_1p p: nb_surj 1 p.+2 = 0.
```

An easy consequence of (6.69) is $S_{n+1,p+1} = (p+1)(S_{n,p+1} + S_{n,p})$. We have already met this equation, so that $S_{n,p}$ is the number of surjections introduced above (i.e., the stirling number ties a factorial). Moreover $S_{n,n+p+1} = 0$. It follows $S_{n,n} = n!$ and

$$S_{n+1,n} = \binom{n+1}{2} n! \qquad S_{n+2,n} = (\binom{n+3}{4} + 2\binom{n+2}{2}) n! \tag{6.70}$$

```
Lemma nb_surj_rec n p:
   nb_surj n.+1 p.+1 = (nb_surj n p + nb_surj n p.+1) *+ (p.+1).
Lemma nbsurj_stirling n p: nb_surj n p = nbsurj n p.
Lemma nb_surj_pos1 n p: {k:nat | nb_surj n p = (p`!*k)%N }.
Lemma nb_surj_small n p: nb_surj n (n+p.+1) = 0.
Lemma nb_surj_nn n: nb_surj n n = n`!.
Lemma nb_surj_Snn n: nb_surj n.+1 n = ('C(n.+1,2) * n`!) %N.
Lemma nb_surj_SSnn n:
   nb_surj n.+2 n = (('C(n.+3,4) + 2 * 'C(n.+2,2)) * n`!) %N.
```

6.9.10 Formulas using polynomials

We assume from now on that R is a ring. If x and y are in R and n is an integer then $\sum_{i\leq n}(x+iy) = x(n+1) + yn(n+1)/2$.

```
Lemma F1 x y n: (\sum_(i<n.+1) (x+y*+ i))*+2  = (x*+2+y*+n)*+n.+1.
```

Consider the relation $(X+c)^n = \sum_{i\leq n}\binom{i}{n} c^{n-i} X^i$. It holds as a polynomial equality, where X is the variable, and c an element of the ring. We shall deduce: the coefficient of index i is $\binom{n}{i} c^{n-i}$ (this holds whether $i \leq n$ or $i < n$).

```
Lemma power_monom (c:R) n :
  ('X + c%:P) ^+ n = \poly_(i< n.+1) (c^+(n - i)%N *+ 'C(n, i)).
```

Consider $z^{k+l} = z^k z^l$, where $z = X+1$. If we compute the coefficient of index i, we get the Vandermonde formula (6.59) above.

Let's compute the coefficient of index i in $(X+a+b)^n$. We get (6.71) below. The formula simplifies if we consider $a = 1$ and $b = \pm 1$. Note that if $p > n$, everything is zero.

$$\binom{n}{p}(a+b)^p = \sum_{i=0}^{p}\binom{n}{i}\binom{n-i}{p-i} a^{p-i} b^i. \tag{6.71}$$

$$\sum_{i=0}^{p}\binom{n}{i}\binom{n-i}{p-i} = 2^p\binom{n}{p}. \tag{6.72}$$

$$\sum_{i=0}^{p}(-1)^i\binom{n}{i}\binom{n-i}{p-i} = 0 \qquad (p \neq 0). \tag{6.73}$$

```
Lemma bin_vandermonde (k l i:nat):
  (\sum_(j < i.+1) 'C(k, j)  *'C(l, (i - j)) = 'C(k+l , i)  )%N.
Lemma F36 k l i:
  (\sum_(j < i.+1) ('C(k, j)  *'C(l, (i - j)))) = 'C(k+l , i).
```

```
Lemma F37 n p: p<= n ->  (* 4 *)
  (\sum_(k < (n-p).+1) ('C(n, k)  *'C(n, p+k))) = 'C(2*n,n-p).
Lemma F38 n: \sum_(i<n.+1) 'C(n,i) ^2 = 'C(n.*2, n).  (* 2 *)

Lemma G6_a (n p:nat) (a b: int): (* 25 *)
  (\sum_(i < p.+1) a^+ (p-i) * b ^+ i *+ ('C(n, i) * 'C(n-i, p-i)) =
     (a+b) ^+ p *+  'C(n , p)  )%Z.
Lemma G6_b (n p:nat): p <> 0%N ->  (* 4 *)
  \sum_(i < p.+1)  (-1%:Z) ^+ i *+ ('C(n, i) * 'C(n-i, p-i)) =  0.
Lemma G6_c (n p:nat):   (* 4 *)
  (\sum_(i < p.+1) 'C(n, i) * 'C(n-i, p-i) =  2 ^ p *'C(n , p)  )%N.
```

Relation (6.74) below can be proved by induction. Alternate proof: differentiate $(X+1)^n$, and evaluate at 1. Evaluating at -1 gives (6.75).

$$\sum_{i=0}^{n} i\binom{n}{i} = n2^{n-1} \tag{6.74}$$

$$\sum_{i=0}^{n}(-1)^i i\binom{n}{i} = 0 \qquad (n \neq 1) \tag{6.75}$$

```
Lemma F27_a n: \sum_(i < n.+1)  i* 'C(n, i) = n * 2 ^ (n.-1) .
Lemma F27_b n: (\sum_(i < n.+1)  i* 'C(n, i) = n * 2 ^ (n.-1) ) %N.
Lemma F28 n: \sum_(i < n.+1)  with_sign i (i* 'C(n, i))
    =  if n is 1 then -1%:Z else 0.
```

6.9.11 Partitions

We show here that: if P is a partition of E, then Card(P) ≤ Card(E), if P is a partition of ∅, then P = ∅, and if $f : E \to F$ is a function, the set of all $f_x = \{y, y \in A, f(x) = f(y)\}$ is a partition of A. It follows that the cardinal of A is the sum of the cardinals of sets f_x; if all these sets have cardinal n, then card(A) = $n\,\text{Card}(f\langle A\rangle)$.

```
Lemma neq_sym (T: eqType) (x y: T): (x != y) = (y != x).

Section Partition.
Variable T: finType.
Variables aT rT : finType.
Implicit Type P : {set {set T}}.

Lemma disjointsU1 (T: finType) (A B: {set T}) x:
  [disjoint (x |: A)  & B]  =  (x \notin B) && [disjoint A & B].

Lemma partition_ni_set0 (T:finType) (P: {set {set T}}) E i:
   partition P E -> i \in P -> i != set0.
Lemma card_partition_aux  (T:finType) (P: {set {set T}}) E:
   partition P E -> #|P| <= #|E|.
Lemma partition_set0 (T:finType) (P: {set {set T}}) :
   partition P set0  = (P == set0).
Lemma preim_atE (aT: finType) (rT: eqType) (f: aT -> rT) (A: {set aT}) x:
  [set y \in A | f y == f x] = A :&: preim_at f x.
Lemma card_inv_im (aT rT: finType) (f: aT -> rT) (A: {set aT}) (n: nat):
   (forall x, x \in A -> #|[set y \in A | f y == f x]| = n) ->
```

```
   #|A| =  #|f @: A| * n.
Lemma stirling_partition  (T: finType) (E: {set T}) (p:nat):   (* 170 *)
  #|[set P  | partition P E && (#|P| == p) ]| = 'St(#|E|, p).
```

We propose here to count the number of surjective functions $E \to F$. There no definition of "surjective" in SSREFLECT and there is no obvious definition. Assume that T and T′ are two finite types, $E \subset T$ and $F \subset T'$ are two finite sets. We say that $f : T \to T'$ is surjective if $f(a) \in F$ whenever $a \in E$, and every $b \in F$ is of the form $f(a)$ for some $a \in E$. We can say that f is surjective if it has a right inverse g, so that (1) $f(a) \in F$ whenever $a \in E$, (2) $g(b) \in E$ whenever $b \in F$, and (3) $f(g(b)) = b$ whenever $b \in F$. Note that if (4): $g(f(a)) = a$ whenever $a \in E$, then f is injective on E, and all four conditions together say that f is a bijection $E \to F$. Note that g must be defined on T′; this is easy if T is non-empty. However, if T is the empty function $\emptyset \to \emptyset$, T is empty, and T′ is non-empty, there is no inverse for f. If we want to count the number of surjections $E \to F$, we must assume that two functions equal on E are equal. One solution is to assume that $f(a) = y$ whenever $a \not\in E$. This works when T′ is non-empty. The other solution is to assume $E = T$.

Let $f\langle E\rangle$ denote the set of $f(x)$ for $x \in E$. We say that f is surjective if $f\langle E\rangle = F$. For simplicity, we assume $E = T$. Thus f is surjective if $f\langle T\rangle = F$. Note that we cannot say $f\langle T\rangle = T'$ (since T′ is not a set), we have to cast T′ into a set.

```
Definition set_surj_fun_on (aT rT: finType) (B: {set rT} ):=
  [set f: {ffun aT -> rT} | f @: aT == B ].
Definition set_surj_fun (aT rT: finType):= set_surj_fun_on aT [set: rT].

Lemma set_surj_fun_onP  (aT rT: finType)(B: {set rT} )(f: {ffun aT -> rT}):
   reflect
      ((forall b, b \in B -> exists2 a, a \in aT & b = f a)
      /\ (forall a, f a \in B))
      (f \in set_surj_fun_on aT B).
Lemma set_surj_fun_P (aT rT: finType) (f: {ffun aT -> rT}):
   reflect (forall b, exists a, b = f a)(f \in set_surj_fun aT rT).

Definition preim_at x := f @^-1: pred1 (f x).
Definition preim_partition D := [set D :&: preim_at x | x <- D].
Lemma preim_partitionP D : partition (preim_partition D) D.
```

Chapter 7

Infinite sets

Bourbaki defines an *infinite set* as a set that is not finite, assumes via an axiom the existence of an infinite set and deduces (Theorem 1, [4, p. 184]) the existence of the set of natural integers **N**. We introduced the set of integers a long time ago, so that the first section of this chapter has no code. The second section concerns definitions of a function by induction (we already needed this possibility, so that we give only some additional properties). The third section shows that addition and multiplication of two infinite cardinals is trivial. We study some properties of countable cardinals and stationary sequences.

7.1 The set of natural integers

Assume that there is an infinite set, thus an infinite cardinal $\mathfrak{a}$. Let **N** be the set of all finite cardinals $< \mathfrak{a}$, $\aleph_0$ its cardinal. Then **N** is the set of all integers. Conversely, **N** is infinite, since the successor function is injective and not surjective. Note that, if n is finite, the set of integers $\le n$ has cardinal $n+1$. We deduce $n+1 \le \aleph_0$. Since $n < n+1$, it follows that $\aleph_0$ is not n. If we use von Neumann cardinals, we get that **N** is the least infinite ordinal ω as well as the least infinite cardinal $\aleph_0$.

Bourbaki defines a *sequence* as a family whose index set I is a subset of **N**. It is called an infinite sequence if I is infinite. Remember that a *finite sequence* is a family where I is finite and contains only integers; this means that I is a finite subset of **N**.

Let's quote Bourbaki [4, p. 184] "Let P⦃n⦄ be a relation and let I denote the set of integers n such that P⦃n⦄ is true. I is then a subset of **N**. A sequence $(x_n)_{n\in\mathrm{I}}$ is then sometimes written $(x_n)_{\mathrm{P}\{n\}}$, and x_n is called the *nth term* in the sequence." Example. Assume that P⦃n⦄ is the relation $n \in \mathbf{Z}$ where **Z** denotes the set of rational integers as a subset of **N**. According to the quote, $(x_n)_{n\in\mathbf{N}}$ and $(x_n)_{n\in\mathbf{Z}}$ are the same sequences. This may be confusing since they are obviously different families. In section 6.4, Bourbaki assumes that P⦃n⦄ implies that n is an integer; this is missing here. Other example: we consider the property "n even and $n < 10$". This is a finite sequence and the 4th term is 6; in the French version, we can read "x_4 est le terme d'indice 4" and "x_6 est le 4-ème terme". In English this is translated as "x_4 is the 4th term" and "x_6 is the 4th term". This may be confusing.

According to Bourbaki, if the property is $n \ge k$, the sequence is written as $(x_n)_{k\le n}$ or $(x_n)_{n\ge k}$ or even (x_n) if $k=0$ or $k=1$. This last notation is obviously ambiguous. The sum of such a family may be denoted as $\sum_{n=k}^{\infty} x_n$.

Two sequences $(x_n)_{n\in\mathrm{I}}$ and $(y_n)_{n\in\mathrm{I}}$ with the same index set are said to *differ only in the order of their terms* if there exists a permutation f of the index set I such that $x_{f(n)} = y_n$ for all $n \in \mathrm{I}$. This makes sense even if I is not a subset of the integers. By commutativity, two sequences that differ only in the order of their terms have same sum and product.

A *multiple sequence* is a family whose index set is a subset of a product $\mathbf{N}^p$ (p is a integer).

Let f be a bijection of $\mathbf{N}$ onto a set I. For each family $(x_i)_{i\in\mathrm{I}}$, the sequence $n \mapsto x_{f(n)}$ is said to be obtained by *arranging the family* $(x_i)_{i\in\mathrm{I}}$ *in the order defined by* f.

7.2 Definition of mappings by induction

If we instantiate Criterion C60 (see page 54) to the well-ordered set $\mathbf{N}$ we get another criterion C62; it asserts that, for any term T, there exists a unique surjective function f such that

$$\text{(TIND)} \qquad \forall n, n \in \mathbf{N} \implies f(n) = \mathrm{T}\{f^{(n)}\}$$

where $u^{(x)}$ denotes the surjective restriction of u to the segment $]\leftarrow, x[$. Recall that $]\leftarrow, x[$ is the set of all elements y such that $y <_{\mathbf{N}} x$; this is just the interval $[0, x[$. Note that the operator $\mathrm{T}\{u\}$ must be *defined* for any set u, but is *used* only when u is of the form $u_n = f^{(n)}$, this is the restriction of some unknown function to a segment of our well-ordered set.

Bourbaki deduces C63: *Let* $\mathrm{S}\{v\}$ *and a be two terms. Then there exists a set* V *and a mapping f of* $\mathbf{N}$ *onto* V *such that* $f(0) = a$ *and* $f(n) = \mathrm{S}\{f(n-1)\}$ *for each integer* $n \geq 1$. *Moreover the set* V *and the mapping f are uniquely determined by these conditions.*

It is obvious, by induction on n, that if f and f' satisfy the conditions of the criterion, then $f(n) = f'(n)$ whatever n. Since f and f' are surjective, they are equal as well as their targets. Thus, the non-trivial point is existence of f. We explain here the Bourbaki construction. Let

$$\mathrm{D}(u) = \mathscr{E}_x(x \in \mathbf{N} \text{ and } (\exists y)((x, y) \in \mathrm{pr}_1(\mathrm{pr}_1(u)))).$$

Let $\mathrm{M}(u)$ be the least upper bound of $\mathrm{D}(u)$ in $\mathbf{N}$, $\phi = (\emptyset, \emptyset, \emptyset)$ the empty mapping, and consider the relation

$$\mathrm{R}\{y, u\}: \quad (u = \phi \text{ and } y = a) \text{ or } (u \neq \phi \text{ and } y = \mathrm{S}\{u(\mathrm{M}(u))\}).$$

Let $\mathrm{T}\{u\}$ be the term $\tau_y(\mathrm{R}\{y, u\})$.

We have $f(0) = \mathrm{T}\{f^{(0)}\}$, and $f^{(0)} = \phi$, so that $f(0) = \tau_y(\mathrm{R}\{y, \phi\})$. Since $\mathrm{R}\{y, \phi\}$ is equivalent to $y = a$, we get $f(0) = a$. Assume now $n \geq 1$. In this case, $f^{(n)}$ is a function whose domain $\mathrm{D}(f^{(n)})$ is the closed interval $[0, n-1]$. In particular, this is not the empty function, $\mathrm{R}\{y, f^{(n)}\}$ is equivalent to $y = \mathrm{S}\{f^{(n)}(\mathrm{M}(f^{(n)}))\}$ so that $f(n) = \mathrm{S}\{f^{(n)}(\mathrm{M}(f^{(n)}))\}$. Since M is the least upper bound of D, it is $n-1$. Thus $f(n) = \mathrm{S}\{f^{(n)}(n-1)\} = \mathrm{S}\{f(n-1)\}$.

In a footnote, Bourbaki says: "The definition of the least upper bound can be formulated in such a way that it has a meaning even for a set which is not bounded above (it denotes a term, in the formalized language, of the form $\tau_x(\mathrm{R}\{x\})$, which the reader will have no difficulty in writing down)." Recall that Bourbaki uses the term "least upper bound" only when it is defined; as $\mathrm{D}(u)$ is a subset of $\mathbf{N}$, $\mathrm{M}(u)$ is defined when $\mathrm{D}(u)$ is bounded. A simple solution could be

$$\mathrm{M}(u) = \text{the greatest element of } \mathrm{D}(u), \text{ if it exists, zero otherwise.}$$

Alternatively, let $\mathrm{N}(u)$ be the set of upper bounds of $\mathrm{D}(u)$. This is the set of all $x \in \mathbf{N}$ such that $\forall y \in \mathrm{D}(u)$, $y \leq_{\mathbf{N}} x$ (this can be defined for any order relation). Now, $\mathrm{M}(u)$ is the least element of $\mathrm{N}(u)$; it satisfies "$x \in \mathrm{N}(u)$ and $\forall y \in \mathrm{N}(u), x \leq_{\mathbf{N}} y$". If we denote this relation by R', then the least upper bound will be $\tau_x(\mathrm{R}' \wr x \wr)$.

If we denote by $\mathrm{N}'(u)$ the set of all $x \in \mathrm{N}(u)$ such that $\forall y \in \mathrm{N}(u), x \leq_{\mathbf{N}} y$, then N' is empty or has a single element; so that $\mathrm{M}(u) = \bigcup \mathrm{N}'(u)$ if the least upper bound exists. This gives a definition for the least upper bound, that does not use the symbol τ.

Note that $\mathrm{D}(u)$ is the intersection of $\mathbf{N}$ and the domain of u. If u is a function, the domain is the source, and if the source is a subset of $\mathbf{N}$, it is the source. Thus an equivalent definition of D could be: D is the source of u, i.e., $\mathrm{D}(u) = \mathrm{pr}_2(\mathrm{pr}_1(u))$. Given the form of R, the expression $\tau_y(\mathrm{R} \wr y, u \wr)$ is : if u is ϕ then a else $\mathrm{S} \wr u(\mathrm{M}(u)) \wr$. In the first implementation of Gaia we used:

```
M := fun u => supremum Nat_order (source u)
T := fun u => Yo (u = empty_function) a (s (Vf u (M u)))
```

It is sometimes useful to consider the case where S depends on n, in order to get a recurrence relation of the form $f(n+1) = h(n, f(n))$.

This is how Bourbaki handles this case. He considers a *function* $h : \mathbf{N} \times \mathrm{E} \to \mathrm{E}$, where E is some fixed set. Denote by F the set $\mathbf{N} \times \mathrm{E}$. Consider the function $\psi : \mathrm{F} \to \mathrm{F}$ defined by $y \mapsto (\mathrm{pr}_1 y + 1, h(y))$ and the surjective function g defined by induction as $g(0) = (0, a)$ and $g(n+1) = \psi(g(n))$. By induction, g takes its values in F. Define $a_n = \mathrm{pr}_1(g(n))$ and $b_n = \mathrm{pr}_2(g(n))$. From $g(n) \in \mathrm{F}$, one deduces $a_n \in \mathbf{N}$, $b_n \in \mathrm{E}$ and $g(n) = (a_n, b_n)$, hence the two recurrence relations $a_{n+1} = a_n + 1$ and $b_{n+1} = h(a_n, b_n)$. Obviously $a_n = n$, thus $b_{n+1} = h(n, b_n)$. The mapping $\mathbf{N} \to \mathrm{E}$, $n \mapsto b_n$ is the solution.

Consider the following example. Given a sequence of cardinals (x_i), we consider $h(n, y) = x_{n+1} + y$ or $h(n, y) = x_{n+1}^y$. Using our induction scheme (IND0) defined below, there exist two sequences satisfying $y_0 = x_0$ and $y_{n+1} = x_{n+1} + y_n$ or $z_0 = x_0$ and $z_{n+1} = x_{n+1}^{z_n}$. Let's try to apply the Bourbaki method. The difficulty is to find a set E such that $h(n, x) \in \mathrm{E}$ whenever $n \in \mathbf{N}$ and $x \in \mathrm{E}$, in order to coerce h into a function $\mathbf{N} \times \mathrm{E} \to \mathrm{E}$. There is a set E stable by cardinal sum that contains the range E_0 of the sequence (x_i). If E_0 is a subset of $\mathbf{N}$, just take $\mathbf{N}$, otherwise let $\mathfrak{a}$ be the supremum of E_0 (see page 87). This is an infinite cardinal, and we shall see in the next section that the set of all cardinals $\leq \mathfrak{a}$ is stable by cardinal sum. As a consequence, we can use the Bourbaki construction and $y_n \in \mathrm{E}$. Doing the same for z_n is tricky. For any set E we define $p(\mathrm{E})$ to be the set of all x^y for $x \in \mathrm{E}$ and $y \in \mathrm{E}$, and for any cardinal $\mathfrak{a}$, we define $\mathrm{E}_{\mathfrak{a}}$ to be the set of all cardinals $\leq \mathfrak{a}$ and $s(\mathfrak{a})$ to be the supremum of $p(\mathrm{E}_{\mathfrak{a}})$. Let f be the function defined by induction via s (where $f(0)$ is any cardinal such that $\mathrm{E}_{f(0)}$ contains the range of the sequence x_i). Finally, let E be the union of the sets $\mathrm{E}_{f(i)}$. Since f is increasing, if x and y are two elements of E, there is an i such that $x \in \mathrm{E}_{f(i)}$ and $y \in \mathrm{E}_{f(i)}$ hence $x^y \in \mathrm{E}_{f(i+1)} \subset \mathrm{E}$. This is a very large set (we could reduce a bit its size by defining $p(\mathrm{E})$ to be the set of all x^y for $x \in \mathrm{E}_0$ and $y \in \mathrm{E}$). Our method is better, since there is no need to introduce this set E.

If $s(x)$ and $h(n, x)$ are two functional terms (we do not require them to be functions), and a term a , there exists a unique surjective function f defined on $\mathbf{N}$ such that:

(IND) $$f(0) = a \quad \text{and} \quad n \in \mathbf{N} \implies f(n+1) = s(f(n)),$$

respectively

(IND0) $$f(0) = a \quad \text{and} \quad n \in \mathbf{N} \implies f(n+1) = h(n, f(n)).$$

In both cases, we use transfinite induction via a term T. In the first case, we use $y = s(u(\mathrm{M}(u)))$ and in the second case, we use $y = h(\mathrm{M}(u), u(\mathrm{M}(u)))$. Let $\mathrm{M}'(u)$ be the cardinal of the source of u. Then $u = \phi$ can be replaced by $\mathrm{M}'(u) = 0$, and if $u = f'(n)$, then $\mathrm{M}'(u) = n$. Hence the following definition:

```
M' u := cardinal (source u)
T u := Yo (M' u = \0c) a (s (Vf u ((M' u) -c \1c)))
```

These definitions can be simplified further. Since $\mathrm{M}(u)$ is a von Neumann cardinal, we have $\mathrm{M}'(u) = \mathrm{M}(u)$, and $\mathrm{M}'(u) - 1$ is the predecessor of $\mathrm{M}(u)$. We obtain the following definitions, introduced in the previous chapter.

```
(*
Definition induction_defined0 (h: fterm2)  (a: Set) :=
  transfinite_defined Nat_order
   (fun u => Yo(source u = \0c) a (h
          (cpred (source u))(Vf u (cpred (source u))))).

Definition induction_defined (s: fterm)  (a: Set) :=
 transfinite_defined Nat_order
 (fun u => Yo(source u = \0c) a (s (Vf u (cpred (source u))))).
*)
```

We consider also functions that satisfy a recurrence relation on an interval $[0, m]$:

(IND1') $$f(0) = a \quad\text{and}\quad f(n+1) = g(n, f(n)) \quad\text{if}\quad n < m.$$

It is easier to define f for all integers, for instance as $f(n) = a$ whenever $n > m$, so we consider:

(IND1) $$f(0) = a \quad\text{and}\quad n < m \implies f(n+1) = g(n, f(n)) \quad\text{and}\quad n \in \mathbf{N}, n > m \implies f(n) = a.$$

```
Definition induction_defined1 (h: fterm2) a p :=
   induction_defined0 (fun n x => Yo (n <c p) (h n x) a) a.
```

We give here three additional definitions. Instead of considering a surjective function, we consider a function with target E.

```
Definition change_target_fun f t := triple (source f) t (graph f).

Definition induction_defined_set s a E:=
  change_target_fun (induction_defined s a) E.

Definition induction_defined_set0 h a E:=
  change_target_fun (induction_defined0 h a) E.

Definition induction_defined_set1 h a p E:=
  change_target_fun (induction_defined1 h a p) E.
```

We state some theorems that say that such a function exists and is unique.

```
Lemma induction_defined_pr1 h a p (f := induction_defined1 h a p):
    natp p ->
    [/\  source f = Nat, surjection f,
      Vf f \0c = a,
      (forall n,  n <c p -> Vf f (csucc n) = h n (Vf f n)) &
      (forall n, natp n -> ~ (n <=c p) -> Vf f n = a)].
Lemma integer_induction1 h a p: natp p ->
  exists! f, [/\ source f = Nat, surjection f,
    Vf f \0c = a ,
     (forall n,  n <c p -> Vf f (csucc n) = h n (Vf f n)) &
     (forall n, natp n -> ~ (n <=c p) -> Vf f n = a)].
```

We now show that the target of the function defined by induction is a subset of E under some conditions. It follows that there is a variant where the target is E. We shall not prove uniqueness, it is obvious.

```
Lemma change_target_pr f E (g:= change_target_fun f E):
   function f -> sub (target f) E
  -> (function_prop g (source f) E
     /\ forall x, inc x (source f) -> Vf g x = Vf f x).
Lemma integer_induction_stable E g  a:
  inc a E -> (forall x, inc x E -> inc (g x) E) ->
  sub (target (induction_defined g a)) E.
Lemma integer_induction_stable0 E h a:
  inc a E -> (forall n x, inc x E -> inc n Nat -> inc (h n x) E) ->
  sub (target (induction_defined0 h a)) E.
Lemma integer_induction_stable1 E h a p:
  natp p ->
  inc a E -> (forall n x, inc x E -> n <c p -> inc (h n x) E) ->
  sub (target (induction_defined1 h a p)) E.

Lemma induction_defined_pr_set E g a:
  let f := induction_defined_set g a E in
    inc a E -> (forall x, inc x E -> inc (g x) E) ->
     [/\ function_prop f Nat E, Vf f \0c = a &
     forall n, natp n -> Vf f (csucc n) = g (Vf f n)].
Lemma induction_defined_pr_set0 E h a:
  let f := induction_defined_set0 h a E in
    inc a E -> (forall n x, inc x E -> natp n -> inc (h n x) E) ->
     [/\ function_prop f Nat E, Vf f \0c  = a &
     forall n, natp n -> Vf f (csucc n) = h n (Vf f n)].
Lemma induction_defined_pr_set1 E h a p:
  let f := induction_defined_set1 h a p E in
    natp p ->
    inc a E -> (forall n x, inc x E -> n <c p -> inc (h n x) E) ->
    [/\ function_prop f Nat  E, Vf f \0c = a,
      (forall n, n <c p -> Vf f (csucc n) = h n (Vf f n)) &
      (forall n, natp n -> ~ (n <=c p) -> Vf f n = a)].
```

We consider here a variant of generic definition by induction: instead of the function $f^{(t)}$, we consider its graph, this is the functional graph of $s \mapsto f(s)$ defined on t. Let's write it $f_{(t)}$. We claim: there is a unique functional symbol f such that, for every ordinal x, $f(x) = p(f_{(x)})$.

```
Definition transdef_ord_prop p f x := f x = p (Lg x f).
Definition transdef_ord p x :=
  (Vf (transfinite_defined (ordinal_o (osucc x)) (fun f => p (graph f))) x).

Lemma transdef_ord_unique p f1 f2:
   (forall x, ordinalp x ->  transdef_ord_prop p f1 x) ->
   (forall x, ordinalp x ->  transdef_ord_prop p f2 x) ->
   f1 =1o f2.
Lemma transdef_ord_pr (p:fterm) x:
  ordinalp x -> transdef_ord_prop p (transdef_ord p) x.
```

Definition by stratified induction. We assume here that we have some property $\mathrm{W}(x)$, and a functional term $\rho(x)$ such that $\rho(x)$ is an ordinal whenever $\mathrm{W}(x)$ holds. Moreover, assume that, for every ordinal α, the relation $W(x) \wedge \rho(x) < \alpha$ is collectivizing. Using the axiom of choice, we get a set W_α such that $x \in W_\alpha \iff W(x) \wedge \rho(x) < \alpha$ (note: this set is obviously unique). We deduce a set W'_α formed of all x such that $W(x) \wedge \rho(x) = \alpha$.

Assume now that $H(x, f)$ is some functional term. We define by transfinite induction a functional term g such that $g(\alpha)$ is the functional graph with domain W'_α that maps x to $H(x, \bigcup_\alpha g)$. We set $f(x) = g(\rho(x))(x)$. This is $H(x, \bigcup_{\rho(x)} g)$. Note that the union is the set of all t such that there is some $\beta < \rho(x)$ such that $t \in g(\beta)$. Since $g(\beta)$ is a functional graph, t is a pair and $\mathrm{pr}_1 t$ is an element of the domain of $g(\beta)$, namely W'_α; thus β is unique and $\mathrm{pr}_1 t \in W_{\rho(x)}$. It follows that $f(x) = H(x, F_x)$, where F_x is the functional graph, with domain $W_{\rho(x)}$, that maps t to $f(t)$.

```
Section StratifiedInduction.
Variable W: property.
Variable H: fterm2.
Variable idx: fterm.
Hypothesis OS_idx: forall x, W x -> ordinalp (idx x).
Hypothesis Wi_coll: forall i, ordinalp i ->
     exists E, forall x,  inc x E <-> (W x /\ idx x <o i).

Definition stratified_set i :=
    choose (fun E => forall x,  inc x E <-> (W x /\ idx x <o i)).

Definition stratified_setr i :=
   Zo (stratified_set (osucc i)) (fun z => idx z = i).

Lemma stratified_setP i (E:= stratified_set i):
   ordinalp i -> (forall x,  inc x E <-> (W x /\ idx x <o i)).
Lemma stratified_setrP i (E:= stratified_setr i):
   ordinalp i -> (forall x,  inc x E <-> (W x /\ idx x = i)).

Definition stratified_fct_aux:=
  transdef_ord (fun G => Lg (stratified_setr (domain G))
    (fun z =>  (H z (unionb G)))).
Definition stratified_fct x := Vg (stratified_fct_aux (idx x)) x.

Lemma stratified_fct_aux_p1 x ( g:= stratified_fct_aux) :
  ordinalp x -> g x = Lg (stratified_setr x) (fun z => H z (unionf x g)).

Lemma stratified_fct_pr x  (f := stratified_fct):
  W x -> f x =  H x (Lg (stratified_set (idx x)) f).
```

7.3 Properties of infinite cardinals

Bourbaki claims (Lemma 1, [4, p. 186]) that every infinite set E has a subset F equipotent to **N**; the argument being that there exists a well-ordering on E. If E is isomorphic to a segment of **N**, then E is equipotent to **N** since all other segments are of the form $]\leftarrow, x[$, hence $[0, x[$, thus are finite. Otherwise, **N** is isomorphic to a segment of E, hence equipotent to a subset of E.

Our proof is simpler: it follows from $\mathrm{card}(\mathbf{N}) \subset \mathrm{card}(E)$. We shall denote by $\aleph_0$ the quantity $\mathrm{card}(\mathbf{N})$. We have $\mathbf{N} = \aleph_0 = \omega_0$.

```
Definition aleph0 :=  omega0.
```

```
Lemma cardinal_Nat: cardinal Nat = Nat.
Lemma aleph0_pr1: aleph0 = cardinal Nat.
Lemma CIS_aleph0: infinite_c aleph0.
Lemma CS_aleph0: cardinalp aleph0.
Lemma aleph0_nz: aleph0 <> \0c.
Lemma infinite_gt1 x: infinite_c x -> \1c <c x.
Lemma infinite_ge2 x: infinite_c x -> \2c <=c x.
Lemma infinite_greater_countable1 E:
  infinite_set E -> aleph0 <=c (cardinal E).
Lemma infinite_greater_countable E:
  infinite_set E -> exists F, sub F E /\ cardinal F = aleph0.
```

Bourbaki claims that $\mathbf{N}\times\mathbf{N}$ is equipotent to $\mathbf{N}$: the relation $\mathrm{Card}(\mathbf{N}) \le \mathrm{Card}(\mathbf{N}\times\mathbf{N})$ is a consequence of $\{0\}\times\mathbf{N}\subset\mathbf{N}\times\mathbf{N}$; moreover there is an injection from $\mathbf{N}\times\mathbf{N}$ into $\mathbf{N}$. He uses expansion to base 2. Assume $x = \sum x_i 2^i$ and $y = \sum y_i 2^i$, then $\sum(2x_i + y_i)4^i$ is an injective function of x and y. We use here a different function; consider

$$f(n,m) = n + \binom{n+m+1}{2} = n + g(n+m).$$

The function g is the binomial coefficient with indices $a+1$ and 2, it is also $g(a) = a(a+1)/2$; it satisfies $g(a+1) = g(a)+a+1$, hence $g(n+m) \le f(n,m) < g(n+m+1)$. This relation shows that $n+m$ is uniquely defined by $f(n,m)$, from which injectivity of f follows. Consider x and the least a such that $x < g(a)$. Then $x = f(n,m)$, where n and m are the unique integers satisfying $n+m+1 = a$ and $x = n+g(a-1)$. This shows that f is bijective.

```
Lemma equipotent_N2_N: (coarse Nat) \Eq Nat.  (* 63 *)
```

We show here Theorem 2 ([4, p. 186]): for every infinite cardinal $\mathfrak{a}$, we have $\mathfrak{a} = \mathfrak{a}^2$.

Let E be an infinite set and $\mathfrak{a}$ its cardinal. We consider all subsets F of E equipotent to $\mathrm{F}\times\mathrm{F}$. In fact we consider all bijections $f:\mathrm{F}\to\mathrm{F}\times\mathrm{F}$, and extend them as a function $g:\mathrm{F}\to\mathrm{E}\times\mathrm{E}$. If $g':\mathrm{F}'\to\mathrm{E}\times\mathrm{E}$, we say that $g\le g'$ if $\mathrm{F}\subset\mathrm{F}'$ and $g(x) = g'(x)$ for all $x\in\mathrm{F}$. This gives an ordered set $\mathfrak{M}$.

Bourbaki says “It is immediately seen that $\mathfrak{M}$ is inductive”. In fact, this is not so trivial. We first notice that there is an infinite subset F_0 of E, thus some bijection $f_0:\mathrm{F}_0\to\mathrm{F}_0\times\mathrm{F}_0$, associated to $g_0:\mathrm{F}_0\to\mathrm{E}\times\mathrm{E}$. Let $\mathfrak{M}_0$ be the subset of $\mathfrak{M}$ formed of functions that extend g_0. This set is inductive. In fact, consider a totally ordered family g_i of elements of $\mathfrak{M}_0$. It has an upper bound. If the family is empty, we can take g_0 as upper bound. Otherwise, let g be the common extension of all these g_i. It is obvious that g is injective, and that if F is its source, then its range is $\mathrm{F}\times\mathrm{F}$. Since the family is non-empty, g extends g_0. Thus g is an upper bound.

We apply Zorn’s lemma, and take a maximal element. Thus we get a bijection $\mathrm{F}\to\mathrm{F}\times\mathrm{F}$. Denote by $\mathfrak{b}$ the cardinal of F. We have $\mathfrak{b} = \mathfrak{b}^2$. Since $\mathrm{F}_0\subset\mathrm{F}$, it follows that F is infinite. In particular $\mathfrak{b}\ge 2$. Thus $\mathfrak{b}\le 2\mathfrak{b}\le\mathfrak{b}^2 = \mathfrak{b}$. This says $2\mathfrak{b} = \mathfrak{b}$. Let $\mathrm{G} = \mathrm{E}-\mathrm{F}$. If its cardinal is $\le\mathfrak{b}$, it follows that E has cardinal $\le \mathrm{card}(\mathrm{G}) + \mathrm{card}(\mathrm{F}) \le \mathrm{card}(\mathrm{F})$, so that $\mathfrak{b} = \mathfrak{a}$, and $\mathfrak{a} = \mathfrak{a}^2$ holds. If its cardinal is $\ge\mathfrak{b}$, we can find a subset F' of G with cardinal $\mathfrak{b}$. Let $\mathrm{F}'' = \mathrm{F}\cup\mathrm{F}'$, and $\mathrm{Z} = \mathrm{F}''\times\mathrm{F}'' - \mathrm{F}\times\mathrm{F}$. Note that Z has cardinal $3\mathfrak{b}$, thus is equipotent to F'. This means that we can extend the bijection $\mathrm{F}\to\mathrm{F}\times\mathrm{F}$ into a bijection $\mathrm{F}''\to\mathrm{F}''\times\mathrm{F}''$, contradicting maximality.

```
Theorem equipotent_inf2_inf E: infinite_c E ->
  E ^c \2c = E. (* 200 *)
```

By induction, $\mathfrak{a}^n = \mathfrak{a}$ if $\mathfrak{a}$ is an infinite cardinal and $n \geq 1$ is an integer. As a consequence, if $(\mathfrak{a}_i)_{i\in \mathrm{I}}$ is a finite family of non-zero cardinals, if the largest one is an infinite cardinal $\mathfrak{a}$, then the product is $\mathfrak{a}$. Finally, if $\mathfrak{a}$ and $\mathfrak{b}$ are two non-zero cardinals, one of them being infinite, the product is (according to Bourbaki) $\sup(\mathfrak{a}, \mathfrak{b})$. We state here: if $\mathfrak{a} \leq \mathfrak{b}$, $\mathfrak{b}$ infinite, and $\mathfrak{a}$ is non-zero, then the product is $\mathfrak{b}$.

We deduce some inequalities. For instance $xa = xb$ implies $a = b$ whenever x is finite, non-zero, a and b are cardinals, finite or not.

```
Lemma csquare_inf a: infinite_c a ->
  a *c a = a.
Lemma cpow_inf a n: infinite_c a -> natp n ->
  n <> \0c -> a ^c  n = a.
Lemma cpow_inf1 a n: infinite_c a -> natp n ->
   (a ^c n) <=c a.
Lemma finite_family_product a f: fgraph f ->
  finite_set (domain f) -> infinite_c a ->
  (forall i, inc i (domain f) -> (Vg f i) <=c a) ->
   card_nz_fam f ->
   (exists2 j,  inc j (domain f) & (Vg f j) = a) ->
  cprod f = a.
Lemma cprod_inf a b: b <=c a ->
  infinite_c a -> b <> \0c -> a *c b = a.
Lemma cprod_inf6 a b: cardinalp a -> cardinalp b ->
  (infinite_c a \/ infinite_c b) -> a <> \0c -> b <> \0c ->
  a *c b = cmax a b.
Lemma cprod_inf1 a b: b <=c a ->
  infinite_c a -> a *c b <=c a.
Lemma cprod_inf2 a b: finite_c b ->
  infinite_c a -> a *c b <=c a.
Lemma cprod_inf4 a b c:
   a <=c c -> b <=c c -> infinite_c c -> a *c b <=c c.
Lemma cprod_inf5 a b c:
   a <c c -> b <c c -> infinite_c c -> a *c b <c c.
Lemma cprod_inf7 a b: natp a -> a <> \0c -> infinite_c b -> a *c b = b.
Lemma cprod_eq2lx a b b': natp a -> cardinalp b -> cardinalp b' ->
  a <> \0c -> a *c b = a *c b' -> b = b'.
```

The quantity x^y is trivial or infinite when one argument is infinite.

```
Lemma CIS_pow x y: infinite_c x -> y <> \0c -> infinite_c (x ^c y).
Lemma CIS_pow2  x y: infinite_c x -> infinite_c y ->
   infinite_c (x ^c y).
Lemma CIS_pow3 x y : \2c <=c x -> infinite_c y -> infinite_c (x ^c y).
```

From $\mathfrak{a} = \mathfrak{a}^2$ we deduce: if $\mathfrak{a}$ is an infinite cardinal, $(\mathfrak{a}_i)_{i\in \mathrm{I}}$ is a family of cardinals, if $\mathrm{card}(\mathrm{I}) \leq \mathfrak{a}$ and $\mathfrak{a}_i \leq \mathfrak{a}$ then $\sum \mathfrak{a}_i \leq \mathfrak{a}$ and we have equality if one of the cardinals is $\mathfrak{a}$. As a special case, when I has two elements, we get $\mathfrak{a} = \mathfrak{a} + \mathfrak{a}$. Thus, if $\mathfrak{a}$ and $\mathfrak{b}$ are two cardinals, one of them being infinite, then the sum is the greatest of them.

```
Lemma notbig_family_sum a f:
  infinite_c a ->(cardinal (domain f)) <=c a ->
  (forall i, inc i (domain f) -> (Vg f i) <=c a) ->
  (csum f) <=c a.
Lemma notbig_family_sum1 a f:
```

```
  infinite_c a -> (cardinal (domain f)) <=c a ->
  (forall i, inc i (domain f) ->  (Vg f i) <=c a) ->
  (exists2 j,  inc j (domain f) & (Vg f j) = a) ->
  csum f = a.
Lemma csum_inf1 a: infinite_c a -> a +c a = a.
Lemma csum_inf a b: b <=c a ->
  infinite_c a -> a +c b = a.
Lemma csum_inf6 a b: cardinalp a -> cardinalp b ->
  (infinite_c a \/ infinite_c b) ->  a +c b = cmax a b.
Lemma csum_inf5 a b c:
    a <c c -> b <c c -> infinite_c c -> a +c b <c c.
Lemma csum_inf2 a b c: cardinalp c -> infinite_c a ->
  b <c a -> a = b +c c -> a = c.
```

We deduce the following properties: if $a < b$ and $c < d$, then $a+c < b+d$. This implies that if c is an infinite cardinal, A and B two sets with cardinal $< c$, then the union has cardinal $< c$. If B is an infinite set and card(A) < card(B), then B − A has the same cardinal as B. If moreover card(A′) < card(B), then B − (A ∪ A′) has the same cardinal and a fortiori is non-empty.

```
Lemma csum_M1t1t a b c d : a <c b -> c <c d -> a +c c <c b +c d.
Lemma csum2_pr6_inf1 a b X: infinite_c X ->
  cardinal a <=c X -> cardinal b <=c X ->
  cardinal (a \cup b) <=c X.
Lemma csum2_pr6_inf2 a b X: infinite_c X ->
  cardinal a <c X -> cardinal b <c X ->
  cardinal (a \cup b) <c X.
Lemma infinite_compl A B:
   infinite_set B -> cardinal A <c cardinal B ->
   cardinal (B -s A) = cardinal B.
Lemma card_setC1_inf E x:
  infinite_set E -> cardinal E = cardinal (E -s1 x).
Lemma infinite_union2 x y z:
    infinite_c z -> cardinal x <c z -> cardinal y <c z ->
    nonempty (z -s (x \cup y)).
```

If c is infinite, $a < c$ then $c - a = c$ (whatever c, if $c \leq a$, then $c - a = 0$).

If $c \leq a + b$, then $c - b \leq a$. Moreover $(a + b) - b = a$ (if b is infinite, we need $b < a$, for otherwise, the result is zero).

```
Lemma csum_lt_inf a b c: infinite_c c -> a <c c -> b <c c ->
    (a +c b) <c c.
Lemma cdiff_inf a b: infinite_c a -> b <c a -> a -c b = a.

Lemma cdiff_Mle_gen a b c:
  cardinalp a -> cardinalp b -> cardinalp c ->
  c <=c (a +c b) -> (c -c b) <=c a.
Lemma cdiff_pr1_gen a b: cardinalp a -> cardinalp b ->
  (finite_c b \/ b <c a) ->
     (a +c b) -c b = a.
Lemma cdiff_pr2_gen a b: infinite_c b -> a <=c b -> (a +c b) -c b = \0c.

Lemma cprod_Meqlt_gen a b b':
  natp a -> b <c b' -> a <> \0c -> (a *c b) <c (a *c b').
```

We shall prove later on (in an exercise) that if E is infinite, the number of permutations of E has cardinal 2^{E}. We show here that it is at most this value

```
Lemma cprod_inf3 E F: nonempty E ->
   (cardinal E) <=c (cardinal F) -> infinite_set F ->
   (F \times E) =c F.
Lemma Exercise6_5a  E F:
  (cardinal (functions E F)) <=c (cardinal (sub_functions E F)).
Lemma Exercise6_5b E F:
  (cardinal (sub_functions E F)) <=c (cardinal (\Po (product E F))).
Lemma Exercise6_5c E: infinite_set E ->
  (cardinal (permutations E)) <=c  (cardinal (\Po E)).
```

7.4 Countable sets

A countable set is one that is equipotent to a subset of **N**. Proposition 2 [4, p. 188] says that an infinite countable set is equipotent to **N**. We rewrite this as: a countable set is finite or equipotent to **N**.

Proposition 1 [4, p. 188] says that a subset of a countable set is countable; the product of a finite family of countable sets is countable; the union of a countable family of countable sets is countable.

Proposition 3 [4, p. 189] says that an infinite set E has a partition $(X_\iota)_{\iota\in I}$ where X_ι is countable infinite and I is equipotent to E. Proposition 4 [4, p. 189] says that if f is a function from E onto F, such that F is infinite and $\overset{-1}{f}\langle\{x\}\rangle$ is countable for any $x \in$ F, then F is equipotent to E.

```
Definition countable_set E:= equipotent_to_subset E Nat.
Definition countable_infinite E := countable_set E /\ infinite_set E.

Lemma countableP E:
  countable_set E <-> (cardinal E) <=c aleph0.
Lemma infinite_countableP E:
  countable_infinite E  <-> (cardinal E) = aleph0.
Lemma finite_is_countable X: finite_set X -> countable_set X.
Lemma aleph0_countable E: cardinal E = aleph0 -> countable_set E.
Lemma countable_finite_or_N E: countable_set E ->
  finite_c (cardinal E) \/ cardinal E = aleph0.
Lemma countable_infinite_Nat: countable_infinite  Nat.
Lemma countable_Nat : countable_set Nat.
Theorem countable_sub E F: sub E F -> countable_set F  ->
  countable_set E.
Lemma countable_sub_Nat x : sub x Nat -> countable_set x.
Lemma countable_fun_image z f:
  countable_set z -> countable_set (fun_image z f).
Theorem countable_product f:
  finite_set (domain f) ->
  (allf f countable_set) ->
  countable_set (productb f).
Theorem countable_union f:
  countable_set (domain f) ->
  (allf f countable_set) ->
  countable_set (unionb f).
Lemma countable_setU2 a b:
   countable_set a -> countable_set b -> countable_set (a \cup b).
Lemma countable_setX2 a b:
```

```
   countable_set a -> countable_set b -> countable_set (a \times b).
Theorem infinite_partition E: infinite_set E ->
  exists f, [/\ partition_w_fam f E, (domain f) \Eq E &
    (forall i, inc i (domain f) -> (countable_infinite (Vg f i)))].
Theorem countable_inv_image f: surjection f ->
  (forall y, inc y (target f) ->
    countable_set (Vfi1 f y)) ->
  infinite_set (target f) ->
  (source f) =c (target f).
```

Proposition 5 [4, p. 189] says that the set $\mathfrak{F}$ of finite subsets of an infinite set E is equipotent to E. The proof of Bourbaki is not clear. He defines $\mathfrak{F}_n$ as the set of all subsets with n elements of E and claims $\mathrm{Card}(\mathfrak{F}_n) \le \mathrm{Card}(E)$. Thus, the cardinal of the union of these sets is at most $\sum_{n\in\mathbf{N}} \mathrm{Card}(E) = \mathrm{Card}(E)$. Thus $\mathrm{Card}(\mathfrak{F}) \le \mathrm{Card}(E)$; equality holds because the set of singletons is equipotent to E and is a subset of $\mathfrak{F}$.

The Bourbaki claim is: for every $X \in \mathfrak{F}_n$ there is a bijection from $[1,n]$ onto X, so that the cardinal of $\mathfrak{F}_n$ is at most the cardinal of the set of functions from $[1,n]$ into X which is $\mathrm{Card}(E^n) = \mathrm{Card}(E)$. Our proof is as follows.

For every $X \in \mathfrak{F}_n$ there is a bijection $[1,n] \to X$, hence an injective function from $[1,n]$ into E with range X, but it is not unique. Let K be the set of injections from $[1,n]$ into E. Let f be the function that associates to each element of K its range. The target of this function is clearly $\mathfrak{F}_n$. Let Q be the set of permutations of $[1,n]$, and c it cardinal. This is a non-zero integer. We pretend that the cardinal of $f^{-1}\langle\{x\}\rangle$ is c. We take an element g in this set (it exists, by the remark above). For every permutation h of $[1,n]$, we consider $g \circ h$. This operation is a bijection from Q onto $f^{-1}\langle\{x\}\rangle$. Surjectivity of this operation uses the fact that for any k there exists g such that $k = g \circ h$, if the ranges are the same (since h is injective) and this function is surjective. It is bijective since it is an endomorphism of a finite set. We can now apply the shepherd's principle. The product of the cardinal a of $\mathfrak{F}_n$ and c is the cardinal b of the set of injections, that is smaller than the cardinal d of the set of functions from $[1,n]$ into E. If $n = 0$, we clearly have $a \le \mathrm{Card}(E)$; otherwise $d = \mathrm{Card}(E)$. Hence $ac = b \le \mathrm{Card}(E)$. If a is finite, we have $a \le \mathrm{Card}(E)$; but if a is infinite, we have $a = ac$ (since c is non-zero finite). This implies $a \le \mathrm{Card}(E)$.

As a corollary, the set of finite sequences with value in E is equipotent to E; in fact, this set is the union of the sets of functions from I into E (that has the same cardinal as E^I) for all finite subsets I of $\mathbf{N}$. Since E^I and I are equipotent and since the set of finite subsets of $\mathbf{N}$ is countable, the result is immediate.

```
Theorem infinite_finite_subsets E: infinite_set E ->
  (Zo (\Po E) finite_set) =c  E. (* 161 *)

Lemma infinite_finite_sequence E: infinite_set E ->
 (Zo (sub_functions Nat E) (fun z=> finite_set (source z))) =c E.
```

We give here some properties of $\aleph_0$.

```
Lemma aleph0_pr2:  aleph0 +c aleph0 = aleph0.
Lemma aleph0_pr3:  aleph0 *c aleph0 = aleph0.
Lemma aleph0_plus1: aleph0 +c \1c = aleph0.
```

A set is said to have *the power of the continuum* if it is equipotent to $\mathfrak{P}(\mathbf{N})$. In this case, its cardinal is $2^{\aleph_0}$, and the set is not countable.

7.5 Stationary sequences

A sequence $(x_n)_{n\in\mathbf{N}}$ is *stationary* if there exists an integer m such that $x_n = x_m$ for $n \geq m$. We define here the notion of increasing and decreasing sequences. It is the graph of an increasing function where the source is $\mathbf{N}$ with its natural order. Note that a decreasing sequence is increasing for the opposite order.

```
Definition stationary_sequence f :=
  [/\ fgraph f, domain f = Nat &
  exists2 m, natp m & forall n, natp n -> m <=c n ->
    Vg n f = Vg m f].
Definition increasing_sequence f r:=
  [/\ fgraph f, domain f = Nat, sub (range f) (substrate r) &
  forall n m, natp n  -> natp m -> n <=c m ->
    gle r (Vg f n) (Vg f m)].
Definition decreasing_sequence f r:=
  [/\ fgraph f, domain f = Nat, sub (range f) (substrate r) &
  forall n m, natp n -> natp m -> n <=c m ->
    gle r  (Vg f m) (Vg f n)].
```

Proposition 6 [4, p. 190] says that, if E is an ordered set, each non-empty set has a maximal element if and only if each increasing sequence is stationary. We start with a lemma: a function f such that $f(n) \leq f(n+1)$ is increasing (by induction on m, we have $f(n) \leq f(n+m)$). By definition `increasing_fun f r r'` says that the target of f is the substrate of r; in our case, it is merely a subset, so that the definition will not be used: we show that the graph of f is an increasing sequence.

The Proposition is shown as follows. Assume first that every non-empty subset has a maximal element. Given an increasing sequence, its range is non-empty. It has a maximal element x_n and $m \geq n$ says $x_n \leq x_m$ thus $x_m = x_n$. Conversely, assume that we have a set A that has no maximal element. For each x, the subset T_x of elements of A greater than x is non-empty. This means that the product $\prod \mathrm{T}_x$ is non-empty, hence there is a function $f : \mathrm{A} \to \mathrm{A}$ such that $f(x) > x$ and a sequence $x_{n+1} = f(x_n)$. This sequence is strictly increasing, absurd.

As a consequence a totally ordered set E is well-ordered if and only if each decreasing sequence is stationary (to show that it is well-ordered, we consider the opposite order; thus every non-empty set has a minimal element, this element is the least element, since all subsets of E are directed). Moreover, an increasing sequence in a finite ordered set has a maximal element.

```
Lemma increasing_seq_prop f r: order r ->
  function f -> source f = Nat -> sub (target f) (substrate r) ->
  (forall n, natp n -> gle r (Vf f n) (Vf f (csucc n))) ->
  increasing_sequence (graph f) r.
Lemma decreasing_seq_prop f r: order r ->
  function f -> source f = Nat -> sub (target f) (substrate r) ->
  (forall n, natp n -> glt r (Vf f (csucc n))  (Vf f n)) ->
  decreasing_sequence (graph f) r.
Theorem increasing_stationaryP r: order r ->
  ((forall X, sub X (substrate r) -> nonempty X ->
     exists a,  maximal (induced_order r X) a) <->
  (forall f, increasing_sequence f r -> stationary_sequence f)).
Theorem decreasing_stationaryP r: total_order r ->
  ( (worder r) <->
   (forall f, decreasing_sequence f r -> stationary_sequence f)).
```

```
Theorem  finite_increasing_stationary r: order r ->
  finite_set (substrate r) ->
  (forall f, increasing_sequence f r -> stationary_sequence f).
```

As a consequence, if E is totally ordered but not well-ordered, there is a strictly decreasing sequence.

```
Definition decreasing_strict_sequence f r :=
 [/\ fgraph f, domain f = Nat, sub (range f) (substrate r)
   & forall n m, natp n -> natp m -> n <c m -> glt r (Vg f m) (Vg f n)].

Lemma total_order_worder_dichot r: total_order r ->
   (worder r \/ exists f, decreasing_strict_sequence f r).
```

Proposition 7 [4, p. 190] says that if E is noetherian (every non-empty set has maximal element) and if F is a subset of E such that for $a \in \mathrm{E}$, if $\forall x, x > a \implies x \in \mathrm{F}$ then $a \in \mathrm{F}$; then $\mathrm{F} = \mathrm{E}$.

```
Theorem noetherian_induction r F :order r ->
  (forall X, sub X (substrate r) -> nonempty X ->
    exists a,  maximal (induced_order r X) a) ->
  sub F (substrate r) ->
  (forall a, inc a (substrate r) -> (forall x, glt r a x -> inc x F)
    -> inc  a F)
  -> F = substrate r.
```

Chapter 8

Rational integers

Bourbaki defines the set of rational integers in [6] (Algebra, Chapter I, section 2, paragraph 5) as the group of differences of **N**: the elements of **Z** are the equivalence classes of the relation between (m_1, n_1) and (m_2, n_2) which is written $m_1 + n_2 = m_2 + n_1$; if $\phi_+(m)$ is the class consisting of the elements $(m+n, n)$, where $n \in \mathbf{N}$, then an element m of **N** is identified with $\phi_+(m)$. Every integer has an opposite, so that, if $\phi_-(m) = -\phi_+(m)$, every integer is of the form $\phi_+(m)$ or $\phi_-(m)$ (both functions are injective, and only zero has both forms). One can then forget that **Z** is a quotient.

There are two different implementations of **Z** in COQ. In the standard library, a number is zero, $\phi(n)$ or $-\phi(n)$ (where n is a binary representation of a non-zero natural number). In the SSREFLECT library, there are two functions ϕ_+ and ϕ_- related by $\phi_-(n) = -\phi_+(n+1)$, where n is any natural number. In our implementation, we chose $\phi_-(0) = \phi_+(0)$. The set of integers is totally ordered by a relation compatible with addition, so that $a \le b$ when $b - a$ is positive (of the form $\phi_+(c)$).

Multiplication is defined in Bourbaki in paragraph 6. He says that if E is a monoid, and $x \in \mathrm{E}$, there is a unique homomorphism $f : \mathbf{N} \to \mathrm{E}$ such that $f(1) = x$. This is denoted in SSREFLECT by `x *+ n` or `x ^+ n`. Moreover, if x is invertible, there is a unique homomorphism $g : \mathbf{Z} \to \mathrm{E}$ such that $g(1) = x$, it coincides with f on **N**. The SSREFLECT library provides `x *- n` and `x ^- n` for the case of negative numbers, and `x *~ n` for rational integers. Multiplication can be defined by taking $(\mathbf{Z}, +)$ for E. Usually, one defines the product of positive numbers by $\phi_+(a) \cdot \phi_+(b) = \phi_+(a \cdot b)$.

In section 8, paragraph 11, Bourbaki notices that **Z** is a principal ideal domain, thus defines gcd and lcm. In 9.4, he defines **Q** as the field of fractions of **Z**; this is a short paragraph, containing the definition of addition, multiplication, and comparison. The set **Q** will be defined in the next chapter.

8.1 A definition of the set of rational integers

We define **Z** as the disjoint union of $\mathbf{N}^*$ and **N**, where $\mathbf{N}^*$ is the set of non-zero natural numbers. Elements of **Z** are sometimes called "rational integers". Such an object is a pair; the first component is `val`, the second is `sg`. We may identify the first field with the absolute value, and the second with the sign. We shall use $\mathrm{pr}_1 z$ and $\mathrm{pr}_2 z$ for simplicity here.

A *positive* number is of the form $z = \phi_+(a)$, where $a \in \mathbf{N}$, $\mathrm{pr}_1 z = a$ and $\mathrm{pr}_2 z =$ `C1`. The values $\phi_+(0)$, $\phi_+(1)$, $\phi_+(2)$, $\phi_+(3)$ and $\phi_+(4)$ will be denoted by 0_z, 1_z, 2_z, 3_z and 4_z, and usually identified with 0, 1, 2, 3 and 4.

```
Definition BZ_of_nat x := J x C1.
Definition BZ_zero := BZ_of_nat \0c.
Definition BZ_one := BZ_of_nat \1c.
Definition BZ_two := BZ_of_nat \2c.
Definition BZ_three := BZ_of_nat \3c.
Definition BZ_four := BZ_of_nat \4c.

Notation "\0z" := BZ_zero.
Notation "\1z" := BZ_one.
Notation "\2z" := BZ_two.
Notation "\3z" := BZ_three.
Notation "\4z" := BZ_four.
```

A *negative* number is of the form $z = \phi_-(a)$, where $a \in \mathbf{N}$, $\mathrm{pr}_1 z = a$ and $\mathrm{pr}_2 z = \mathtt{C0}$. Note that zero is both positive and negative; in the definition of $\mathbf{Z}$ it appears only in the positive part. We say that a number is strictly positive or strictly negative when it is non-zero and positive or negative. The most useful negative number is $\phi_-(1)$, denoted -1.

```
Definition BZm_of_nat x := Yo (x = \0c) \0z (J x C0).
Definition BZ_mone := BZm_of_nat \1c.
Notation "\1mz" := BZ_mone.
```

We define $\mathbf{N}^*$, $\mathbf{Z}$ and the following subsets. The sets $\mathbf{Z}_+$ and $\mathbf{Z}_-$ are the images of ϕ_+ and ϕ_-, while $\mathbf{Z}_+^*$ and $\mathbf{Z}_-^*$ are the same sets, without zero. We shall refer to $\mathbf{Z} - \{0\}$ as $\mathbf{Z}^*$.

```
Definition Nats:= Nat -s1 \0c.
Definition BZ := canonical_du2 Nats Nat.
Definition BZms:= Nats *s1 C0.
Definition BZp:=  Nat *s1 C1.
Definition BZps:= Nats *s1 C1.
Definition BZm:= BZms +s1 \0z.
Definition BZs := BZ -s1 \0z.
Definition intp x := inc x BZ.
```

We show some inclusions.

```
Lemma BZps_sBZp : sub BZps BZp.
Lemma BZms_sBZm : sub BZms BZm.
Lemma BZp_sBZ x : inc x BZp -> intp x.
Lemma BZps_sBZ x:inc x BZps -> intp x.
Lemma BZms_sBZ x:inc x BZms -> intp x.
Lemma BZm_sBZ x:inc x BZm -> intp x.
```

We start with some trivial properties. For any natural integer n, $\phi_+(n)$ and $\phi_-(n)$ are rational integers; these functions are injective. We have $\phi_+(n) = \phi_-(m)$ if and only if $n = m = 0$. We have $\mathrm{pr}_1 x = 0$ if and only if $x = 0$.

```
Lemma BZ0_val: BZ_val \0z = \0c.
Lemma BZ0_sg: BZ_zg \0z = C1.
Lemma BZms_nz x: inc x BZms -> x <> \0z.
Lemma BZps_nz x: inc x BZps -> x <> \0z.
Lemma BZms_iP x: inc x BZms <-> (inc x BZm /\ x <> \0z).
```

```
Lemma BZ_of_natp_i x: natp x -> inc (BZ_of_nat x) BZp.
Lemma BZ_of_nat_i x: natp x -> intp (BZ_of_nat x).
Lemma BZm_of_natms_i x: natp x -> x <> \0c ->
  inc (BZm_of_nat x) BZms.
Lemma BZm_of_natm_i x: natp x -> inc (BZm_of_nat x) BZm.
Lemma BZm_of_nat_i x: natp x -> intp (BZm_of_nat x).
Lemma BZps_valnz x: inc x BZps -> BZ_val x <> \0c.
Lemma BZps_iP x: inc x BZps <-> (inc x BZp /\ x <> \0z).
Lemma BZ_of_nat_val x: BZ_val (BZ_of_nat x) = x.
Lemma BZm_of_nat_val x: BZ_val (BZm_of_nat x) = x.
Lemma BZ_of_nat_inj x y: BZ_of_nat x = BZ_of_nat y -> x = y.
Lemma BZm_of_nat_inj x y:  BZm_of_nat x = BZm_of_nat y -> x = y.
Lemma BZm_of_nat_inj_bis x y: BZm_of_nat x = BZ_of_nat y -> (x = y /\ x = \0c).
Lemma BZ_0_if_val0 x: intp x -> BZ_val x = \0c -> x = \0z.
```

Some quantities defined above are in **Z**.

```
Lemma ZS0 : intp \0z.
Lemma ZpS0 : inc \0z BZp.
Lemma ZmS0 : inc \0z BZm.
Lemma ZS1 : intp \1z.
Lemma ZpsS1 : inc \1z BZps.
Lemma ZS2 : intp \2z.
Lemma ZS2 : intp \3z.
Lemma ZS4 : intp \4z.
Lemma ZSm1 : intp \1mz.
Lemma ZmsS_m1: inc \1mz BZms.
Lemma BZ1_nz: \1z <> \0z.
Lemma BZm1_nz: \1mz <> \0z.

Lemma BZ_valN a: intp a -> natp (BZ_val a).
Lemma BZ_sgv x: intp x -> (BZ_sg x = C0 \/ BZ_sg x = C1).
Lemma BZp_sg x: inc x BZp -> BZ_sg x = C1.
Lemma BZps_sg x: inc x BZps -> BZ_sg x = C1.
Lemma BZms_sg x: inc x BZms -> BZ_sg x = C0.

Lemma BZms_hi_pr x: inc x BZms ->
   (BZ_val x <> \0c /\ BZm_of_nat (BZ_val x) = x).
Lemma BZp_hi_pr x: inc x BZp -> BZ_of_nat (BZ_val x) = x.
Lemma BZm_hi_pr x: inc x BZm ->  BZm_of_nat (BZ_val x) = x.
Lemma BZ_hi_pr a: intp a ->
   a =  BZ_of_nat (BZ_val a) \/ a =  BZm_of_nat (BZ_val a).
```

We show that some sets are disjoint.

```
Lemma BZ_i0P x: intp x <-> (inc x  BZms \/ inc x BZp).
Lemma BZ_i1P x: intp x  <-> [\/ x = \0z, inc x BZps | inc x BZms].
Lemma BZ_i2P x: intp x  <-> (inc x  BZps \/ inc x BZm).
Lemma BZs_prop: BZs = BZms \cup BZps.
Lemma BZ_di_neg_pos x: inc x BZms -> inc x BZp -> False.
Lemma BZ_di_pos_neg x: inc x  BZps -> inc x BZm -> False.
Lemma BZ_di_neg_spos x: inc x  BZms -> inc x BZps -> False.
Lemma BZp_i a : intp a -> BZ_sg a = C1 -> inc a BZp.
Lemma BZms_i a : intp a -> BZ_sg a = C0 -> inc a BZms.
```

We show here that $\mathbf{N}^*$ and **Z** are infinite and countable.

```
Lemma cardinal_Nats: cardinal Nats = aleph0.
Lemma cardinal_BZ: cardinal BZ = aleph0.
```

8.2 Opposite and absolute value

We define the opposite of a number as:

$$-x = \begin{cases} \phi_+(\mathrm{pr}_1(x)) & \text{if } x \in \mathbf{Z}_-^* \\ \phi_-(\mathrm{pr}_1(x)) & \text{otherwise.} \end{cases}$$

This implies $\mathrm{pr}_1(-x) = \mathrm{pr}_1(x)$. As a consequence, if x belongs to $\mathbf{Z}$, $\mathbf{Z}_+^*$, or $\mathbf{Z}_-^*$, respectively, then $-x$ belongs to $\mathbf{Z}$, $\mathbf{Z}_-^*$ or $\mathbf{Z}_+^*$, respectively. The mapping $x \mapsto -x$ is involutive, hence a permutation of $\mathbf{Z}$.

```
Definition BZopp  x :=
   Yo (BZ_sg x = C0) (BZ_of_nat (BZ_val x))(BZm_of_nat (BZ_val x)).

Lemma ZSo x: intp x -> intp (BZopp x).
Lemma BZopp_0 : BZopp \0z = \0z.
Lemma BZopp_val x: BZ_val (BZopp x) = BZ_val x.
Lemma BZnon_zero_opp x: intp x ->  (x <> \0z <-> BZopp x <> \0z).

Lemma BZopp_sg x: intp x -> x <> \0z ->
   ((BZ_sg x = C0 -> BZ_sg (BZopp x) = C1)
     /\ (BZ_sg x = C1 -> BZ_sg (BZopp x) = C0)).

Lemma BZopp_positive1 x: inc x BZps -> inc (BZopp x) BZms.
Lemma BZopp_positive2 x: inc x BZp -> inc (BZopp x) BZm.
Lemma BZopp_negative1 x: inc x BZms -> inc (BZopp x) BZps.
Lemma BZopp_negative2 x: inc x BZm -> inc (BZopp x) BZp.

Lemma BZopp_K x: intp x -> BZopp (BZopp x) = x.
Lemma BZopp_inj a b: intp a -> intp b -> BZopp a = BZopp b -> a = b.
Lemma BZopp_fb: bijection (Lf BZopp BZ BZ).
Lemma BZopp_perm: inc (Lf BZopp BZ BZ) (permutations BZ).
Lemma BZopp_p x: BZopp (BZ_of_nat x) = BZm_of_nat x.
Lemma BZopp_m x: BZopp (BZm_of_nat x) = BZ_of_nat x.

Lemma ZmsS_m1: inc \1mz BZms.
Lemma BZopp_m1: BZopp \1mz = \1z.
Lemma BZopp_1: BZopp \1z = \1mz.
```

The *absolute value* of x, denoted by $|x|$, is $\phi_+(\mathrm{pr}_1(x))$. It satisfies

$$|x| = \begin{cases} x & \text{if } x \in \mathbf{Z}_+ \\ -x & \text{if } x \in \mathbf{Z}_-^*. \end{cases}$$

We have $|x| \in \mathbf{Z}_+$ and $|-x| = |x|$. The absolute value is not injective, however $|x|$ is zero if and only if x is zero.

```
Definition BZabs x := BZ_of_nat (BZ_val x).

Lemma BZabs_pos x: inc x BZp -> BZabs x = x.
```

```
Lemma BZabs_neg x: inc x BZms -> BZabs x = BZopp x.
Lemma BZabs_iN x: intp x -> inc (BZabs x) BZp.
Lemma ZSa x: intp x -> intp (BZabs x).
Lemma BZabs_val x: BZ_val (BZabs x) = BZ_val x.
Lemma BZabs_sg x: intp x -> BZ_sg (BZabs x) = C1.
Lemma BZabs_abs x: BZabs (BZabs x) = BZabs x.
Lemma BZabs_opp x:  BZabs (BZopp x) = BZabs x.
Lemma BZabs_0 : BZabs \0z = \0z.
Lemma BZabs_0p x: intp x -> BZabs x = \0z  -> x = \0z.
Lemma BZabs_m1: BZabs \1mz = \1z.
```

8.3 Ordering the integers

Consider the ordinal sum of $\mathbf{N}^*$ (ordered by $\geq$) and $\mathbf{N}$ (ordered by $\leq$). The substrate of this ordering is, by construction, $\mathbf{Z}$, so that $\mathbf{Z}$ is a totally ordered set.

```
Definition BZ_ordering:=
  order_sum2 (opp_order (induced_order Nat_order Nats)) Nat_order.

Lemma BZ_order_aux1: sub Nats (substrate Nat_order).
Lemma BZ_order_aux:
  order (opp_order (induced_order Nat_order Nats))
  /\ order Nat_order.
Lemma BZor_or: order BZ_ordering.
Lemma BZor_sr: substrate BZ_ordering = BZ.
Lemma BZor_tor: total_order BZ_ordering.
```

Expanding definitions, it is clear that comparison on $\mathbf{Z}$ is the following relation $x \leq_{\mathbf{Z}} y$:

```
Definition BZ_le x y:= [/\ intp x, intp y  &
  [\/ [/\ BZ_sg x = C0, BZ_sg y = C0 & (BZ_val y) <=c (BZ_val x)],
      [/\ BZ_sg x = C0 & BZ_sg y = C1] |
      [/\ BZ_sg x = C1, BZ_sg y = C1 &  (BZ_val x) <=c (BZ_val y)]]].
Definition BZ_lt x y:= BZ_le x y /\ x <> y.

Notation "x <=z y" := (BZ_le x y) (at level 60).
Notation "x <z y" := (BZ_lt x y) (at level 60).
```

The following lemmas express that $\leq_{\mathbf{Z}}$ is a total ordering:

```
Lemma zle_P x y: gle BZ_ordering x y <->  x <=z y.
Lemma zlt_P x y: glt BZ_ordering x y <-> x <z y.
Lemma zleT a b c:  a <=z b -> b <=z c ->  a <=z c.
Lemma zleR a: inc a BZ -> a <=z a.
Lemma zleA a b:  a <=z b -> b <=z a -> a = b.
Lemma zleNgt a b: a <=z b -> ~(b <z a).
Lemma zlt_leT a b c: a <z b -> b <=z c -> a <z c.
Lemma zle_ltT a b c: a <=z b -> b <z c -> a <z c.
Lemma zleT_ee a b:  intp a -> intp b -> a <=z b  \/ b <=z a.
Lemma zleT_ell a b:  intp a -> intp b -> [\/ a = b, a <z b | b <z a].
Lemma zleT_el a b:  intp a -> intp b ->  a <=z b  \/ b <z a.
```

Let x and y be integers, $x' = \mathrm{pr}_1 x$ and $y' = \mathrm{pr}_1 y$. We have $x \leq_{\mathbf{Z}} y$ if and only if, either x and y are in $\mathbf{Z}_+$ and $x' \leq_{\mathrm{card}} y'$ or x and y are in $\mathbf{Z}^*_-$, and $y' \leq_{\mathrm{card}} x'$, or x is in $\mathbf{Z}^*_-$ and y in $\mathbf{Z}_+$. In particular the injection $\phi_+ : \mathbf{N} \to \mathbf{Z}$ is strictly increasing.

```
Lemma zle_P1 x y: inc x BZp ->  inc y BZp ->
  (x <=z y <-> (BZ_val x) <=c (BZ_val y)).
Lemma zlt_P1 x y: inc x BZp ->  inc y BZp ->
  (x <z y <->  (BZ_val x) <c (BZ_val y)).
Lemma zle_pr2 x y: inc x BZp -> inc y BZms ->  y <z x.
Lemma zle_P3 x y: inc x BZms -> inc y BZms ->
  (x <=z y <-> (BZ_val y) <=c (BZ_val x)).
Lemma zle_pr4 x y: inc x BZp -> inc y BZms -> ~ (x <=z y).
Lemma zle_P0 x y:
   x <=z y <->
   [\/ [/\ inc x BZms, inc y BZms & (BZ_val y) <=c (BZ_val x)],
       [/\ inc x BZms & inc y BZp] |
       [/\ inc x BZp, inc y BZp &  (BZ_val x) <=c (BZ_val y)]].
Lemma zle_pr5 x y: inc x BZp ->  inc y BZp ->
  (x <=z y =  (BZabs x) <=z (BZabs y)).
Lemma zle_cN a b: natp a -> natp b ->
    (a <=c b <-> BZ_of_nat a <=z BZ_of_nat b).
Lemma zlt_cN a b: natp a -> natp b ->
    (a <c b <-> BZ_of_nat a <z BZ_of_nat b).
Lemma zlt_24: \2z <z \4z.
Lemma zle_24: \2z <=z \4z.
```

We show that $x \leq 0$, $x < 0$, $x \geq 0$ and $x > 0$ are equivalent to say that x is, respectively, in $\mathbf{Z}_-$, $\mathbf{Z}_-^*$, $\mathbf{Z}_+$ and $\mathbf{Z}_+^*$. We then show that $x \leq y$ if and only if $-y \leq -x$, this means that the opposite function is an order isomorphism of $(\mathbf{Z}, \leq)$ onto $(\mathbf{Z}, \geq)$.

```
Lemma zle0xP x: \0z <=z x <-> inc x BZp.
Lemma zlt0xP x: \0z <z x <-> inc x BZps.
Lemma zgt0xP x: x <z \0z <->  inc x BZms.
Lemma zge0xP x: x <=z \0z <-> inc x BZm.
Lemma zle_P6 x y: inc x BZm -> inc y BZm ->
  (x <=z y <->  (BZ_val y) <=c (BZ_val x)).
Lemma BZabs_positive b: intp b -> b <> \0z -> \0z <z (BZabs b).
Lemma zle_opp x y: x <=z y ->  (BZopp y) <=z  (BZopp x).
Lemma zlt_opp x y: x <z y -> (BZopp y) <z (BZopp x).
Lemma zle_oppP x y: intp x -> intp y ->
  (BZopp y <=z BZopp x <-> x <=z y).
Lemma zlt_oppP x y: intp x -> intp y ->
    (BZopp y <z BZopp x <-> x <z y).
Lemma zle_opp_iso:
  order_isomorphism (Lf BZopp BZ BZ) BZ_ordering (opp_order BZ_ordering).
```

8.4 The sum of two integers

The definition of the sum of two integers is complicated, but straightforward. We want the sum to be commutative, agree with the sum on $\mathbf{N}$, and satisfy $-(a+b) = (-a)+(-b)$. Let $\mathrm{A} = \mathrm{pr}_1(a)$ and $\mathrm{B} = \mathrm{pr}_1(b)$. If a and b are positive, the sum is $\phi_+(\mathrm{A}+\mathrm{B})$; if a and b are negative, the sum is $\phi_-(\mathrm{A}+\mathrm{B})$. Otherwise, we consider C to be $\mathrm{A}-\mathrm{B}$ (if $\mathrm{A} \geq \mathrm{B}$) or $\mathrm{B}-\mathrm{A}$ (if $\mathrm{A} \leq \mathrm{B}$); the sum is $\phi_+(\mathrm{C})$ or $\phi_-(\mathrm{C})$, it has the same sign as the number with the greatest absolute value.

We provide an alternate definition, more suited for proving associativity (see comments below). It happens that Coq sometimes unfolds the definitions and gets lost (i.e., takrs a long time to decide that two objects are different). For this reason we lock the definition.

```
Definition BZsum_v2 x y:=
```

```
  let abs_sum :=  (BZ_val x) +c (BZ_val y) in
    let abs_diff1 :=  (BZ_val x) -c (BZ_val y) in
      let abs_diff2 :=  (BZ_val y) -c (BZ_val x) in
        Yo (inc x BZp /\ inc y BZp) (BZ_of_nat abs_sum)
        (Yo ( ~ inc x BZp /\ ~ inc y BZp) (BZm_of_nat abs_sum)
          (Yo (inc x BZp /\ ~ inc y BZp)
            (Yo ( (BZ_val y) <=c (BZ_val x))
              (BZ_of_nat abs_diff1) (BZm_of_nat abs_diff2))
            (Yo ( (BZ_val x) <=c (BZ_val y))
              (BZ_of_nat abs_diff2) (BZm_of_nat abs_diff1)))).
Definition BZsum_v1 x y :=
  let f := fun x => Yo (inc x BZp) (J \0c (BZ_val x)) (J (BZ_val x) \0c) in
  let g := fun x => Yo ((P x) <=c (Q x))
                (BZ_of_nat((Q x) -c (P x))) (BZm_of_nat ((P x) -c (Q x))) in
  let h := fun x y => J ( (P x) +c (P y)) ( (Q x) +c (Q y)) in
  g (h (f x) (f y)).

Definition BZsum := locked BZsum_v2.
Notation "x +z y" := (BZplus x y) (at level 50).
```

We show here the that two definitions are the same.

```
Lemma csubn0 x: cardinalp x -> x -c \0c = x.
Lemma BZsum_spec x: cardinalp x ->
 Yo (x <=c \0c) (BZ_of_nat (\0c -c x)) (BZm_of_nat (x -c \0c)) = BZm_of_nat x.
Lemma BZsum_alt x y: intp x -> intp y -> BZsum_v2 x y = BZsum_v1 x y.
Lemma BZsumE x y :  x +z y = BZsum_v2 x y.
```

Immediate consequences.

```
Lemma BZsum_pp x y: inc x BZp -> inc y BZp
  x +z y = BZ_of_nat ((BZ_val x) +c (BZ_val y)).
Lemma BZsum_mm x y: inc x BZms -> inc y BZms ->
  x +z y = BZm_of_nat ((BZ_val x) +c (BZ_val y)).
Lemma BZsum_pm x y: inc x BZp -> inc y BZms ->
  x +z y = (Yo ((BZ_val y) <=c (BZ_val x))
    (BZ_of_nat ((BZ_val x) -c (BZ_val y)))
    (BZm_of_nat ((BZ_val y) -c (BZ_val x)))).
Lemma BZsum_pm1 x y: inc x BZp -> inc y BZms ->
  (BZ_val y) <=c (BZ_val x) -> x +z y = BZ_of_nat((BZ_val x) -c (BZ_val y)).
Lemma BZsum_pm2 x y: inc x BZp -> inc y BZms ->
  (BZ_val x) <c (BZ_val y) -> x +z y = (BZm_of_nat ((BZ_val y) -c (BZ_val x))).
```

We start here with some easy properties. In particular, we have $a+0=0+a$ and $a+(-a)=0$. Almost all theorems assume that their arguments are in **Z**. There is one exception: commutativity of the sum.

```
Lemma BZsumC x y: x +z y = y +z x.
Lemma ZSs x y: intp x -> intp y -> intp (x +z y).
Lemma BZsum_cN x y: natp x -> natp y ->
  BZ_of_nat x +z BZ_of_nat y = BZ_of_nat (x +c y).
Lemma BZsum_0l x: intp x -> \0z +z x  = x.
Lemma BZsum_0r x: intp x -> x +z \0z = x.
Lemma BZsum_11 : \1z +z \1z = \2z.
Lemma BZsum_opp_r x: intp x ->  x +z (BZopp x) = \0z.
Lemma BZsum_opp_l x: intp x -> (BZopp x) +z x = \0z.
```

We show here that $\mathbf{Z}_+$ is stable by addition, as well as some other subsets of $\mathbf{Z}$.

```
Lemma BZoppD x y: intp x -> intp y ->
  BZopp (x +z y) = (BZopp x) +z (BZopp y).

Lemma BZsum_N_Ns x y: natp x -> inc y Nats ->  inc (x +c y) Nats.
Lemma ZpS_sum x y: inc x BZp -> inc y BZp -> inc (x +z y) BZp.
Lemma ZpsS_sum_r x y: inc x BZp -> inc y BZps -> inc (x +z y) BZps.
Lemma ZpsS_sum_l x y: inc x BZps -> inc y BZp ->  inc (x +z y) BZps.
Lemma ZpsS_sum_rl x y: inc x BZps -> inc y BZps -> inc (x +z y) BZps.
Lemma ZmsS_sum_r x y: inc x BZm -> inc y BZms ->  inc (x +z y) BZms.
Lemma ZmsS_sum_l x y: inc x BZms -> inc y BZm -> inc (x +z y) BZms.
Lemma ZmS_sum x y: inc x BZm -> inc y BZm -> inc (x +z y) BZm.
```

One could prove associativity of the sum by case analysis. The indirect method is a bit shorter, but needs some preparation. It relies on the fact that $\mathbf{Z}$ is a group of differences, and the law of a group is associative. We consider the following three functions.

$$f : x \in \mathbf{Z} \mapsto \begin{cases} (0, x) & \text{if } x \geq 0 \\ (-x, 0) & \text{if } x < 0, \end{cases}$$

$$g : (x, y) \in \mathbf{N}^2 \mapsto \begin{cases} y - x & \text{if } x \leq y \\ -(x - y) & \text{otherwise,} \end{cases}$$

$$h : (x, y) \times (x', y') \in \mathbf{N}^2 \times \mathbf{N}^2 \mapsto (x + x', y + y').$$

We shall consider f as a function with values in $\mathbf{N}^2$ (so, if $x < 0$, $f(x)$ is $(\mathrm{pr}_1 x, 0)$) and consider g as a function with values in $\mathbf{Z}$ (so, if $x > y$, $g(x, y)$ is $\phi_-(x - y)$). This means that we can compose f and g. We have $g(f(x)) = x$, the converse being false. The key relation is the following:

$$g(x, y) = g(x', y') \text{ if and only if } x + y' = x' + y. \tag{8.1}$$

Another important relation is $a + b = g(h(f(a), f(b)))$ (in other terms: the two definitions of addition on $\mathbf{Z}$ are the same). If we write $f(a) = (a_1, a_2)$ and $f(b) = (b_1, b_2)$, this reduces to

$$a + b = g(a_1 + b_1, a_2 + b_2). \tag{8.2}$$

Associativity is

$$\forall x, y, z \in \mathbf{Z}, \quad x + (y + z) = (x + y) + z. \tag{8.3}$$

Write this as $x + \bar{x} = \bar{z} + z$, where $\bar{z} = x + y$ and $\bar{x} = y + z$. Using (8.2) gives $g(x_1 + \bar{x}_1, x_2 + \bar{x}_2) = g(\bar{z}_1 + z_1, \bar{z}_2 + x_2)$ then (8.1) gives

$$x_1 + \bar{x}_1 + \bar{z}_2 + z_2 = \bar{z}_1 + z_1 + x_2 + \bar{x}_2. \tag{8.4}$$

From $g(f(\bar{x})) = \bar{x} = g(y_1 + z_1, y_2 + z_2)$ and (8.1) we deduce

$$\bar{x}_1 + y_2 + z_2 = y_1 + z_1 + \bar{x}_2,$$

and similarly

$$\bar{z}_1 + x_2 + y_2 = x_1 + y_1 + \bar{z}_2.$$

Combing these two equations, using commutativity and associativity of addition on $\mathbf{N}$, and simplifying by $y_1 + y_2$ yields the result.

```
Lemma BZsumA x y z: intp x -> intp y -> intp z ->  (* 117 *)
  x +z (y +z z) = (x +z y) +z z.
Lemma BZsum_AC x y z: intp x -> intp y -> intp z ->
  (x +z y) +z z = (x +z z) +z y.
Lemma BZsum_CA x y z: intp x -> intp y -> intp z ->
  x +z (y +z z) = y +z (x +z z).
Lemma BZsum_ACA a b c d: intp a -> intp b -> intp c -> intp d ->
  (a +z b) +z (c +z d) = (a +z c) +z (b +z d).
```

8.4.1 Subtraction

We now define subtraction as $x - y = x + (-y)$. If x is an integer, its successor is $x + 1$ and its predecessor is $x - 1$.

```
Definition BZdiff x y := x +z (BZopp y).

Notation "x -z y" := (BZdiff x y) (at level 50).

Definition BZsucc x  :=  x +z \1z.
Definition BZpred x  :=  x -z \1z.
```

The quantities introduced above are in **Z**. The successor on **Z** is compatible with the successor on **N**.

```
Lemma ZS_diff x y: intp x -> intp y -> intp (x -z y) BZ.
Lemma ZS_succ x: intp x -> intp (BZsucc x).
Lemma ZS_pred x: intp x BZ -> intp (BZpred x).
Lemma BZsucc_N x: natp x ->  BZsucc (BZ_of_nat x) = BZ_of_nat (csucc x).
Lemma BZprec_N x: inc x Nats -> BZpred (BZ_of_nat x) = BZ_of_nat (cpred x).
```

We have $(a + b) - b = (a - b) + b = a$. A special case is when $b = 1$. We deduce that if $a + c = b + c$ then $a = b$. We show also that $a \neq a + 1$. We have $a - 0 = a$ and $0 - a = -a$. We have $(a + b) - (a + c) = b - c$. We have $(a + b) - c = (a - c) + b$. Taking $b = 1$ and denoting $a + 1$ by $\mathrm{S}a$ gives $\mathrm{S}(a - c) = \mathrm{S}a - c$.

```
Section BZdiffProps.
Variables (x y z: Set).
Hypotheses (xz: intp x)(yz: intp y)(zz: intp z).

Lemma BZsucc_sum : (BZsucc x +z y) = BZsucc (x +z y).
Lemma BZpred_sum: (BZpred x +z y) = BZpred (x +z y).
Lemma BZsucc_pred: BZsucc (BZpred x) = x.
Lemma BZpred_succ: BZpred (BZsucc x) = x.
Lemma BZdiff_sum : (x +z y) -z x = y.
Lemma BZsum_diff: x +z (y -z x) = y.
Lemma BZdiff_sum1: (y +z x) -z x = y.
Lemma BZsum_diff1: (y -z x) +z x = y.
Lemma BZdiff_diag : x -z x = \0z.
Lemma BZdiff_0r: x -z \0z = x.
Lemma BZdiff_0l: \0z -z x = BZopp x.
Lemma BZdiff_sum_simpl_l:  (x +z y) -z  (x +z z) = y -z z.
Lemma BZdiff_sum_comm: (x +z y) -z z = (x -z z) +z y.
Lemma BZoppB: BZopp (x -z y) = y -z x.
```

```
End BZdiffProps.

Section BZdiffProps2.
Variables (x y z: Set).
Hypotheses (xz: intp x)(yz: intp y)(zz: intp z).

Lemma BZsucc_disc:  x <> BZsucc x.
Lemma BZsum_diff_ea: x = y +z z -> z = x -z y.
Lemma BZdiff_diag_rw:  x -z y = \0z -> x = y.
Lemma BZdiff_sum_simpl_r:  (x +z z) -z (y +z z) = x -z y.
Lemma BZdiff_succ_l: BZsucc (x -z y) =  (BZsucc x) -z y.
Lemma BZsum_eq2r:  x +z z = y +z z -> x = y.
Lemma BZsum_eq2l:  x +z y = x +z z -> y = z.
End  BZdiffProps2.


Lemma BZdiff_diff a b c: intp a -> intp b -> intp c ->
   a -z (b -z c)  = (a -z b) +z c.
Lemma BZdiff_diff2 a b c: intp a -> intp b -> intp c ->
   a -z (b +z c)  = (a -z b) -z c.
```

8.4.2 The sign function

The sign function $\operatorname{sgn}(x)$ is 0 if $x = 0$, -1 if $x < 0$ and 1 otherwise.

```
Definition BZsign x:= Yo (BZ_val x = \0c) \0z (Yo (BZ_sg x = C1) \1z \1mz).

Lemma BZsign_trichotomy a: BZsign a = \1z \/ BZsign a = \1mz \/ BZsign a = \0z.
Lemma ZS_sign x: inc (BZsign x) BZ.
Lemma BZsign_pos x: inc x BZps -> BZsign x = \1z.
Lemma BZsign_neg x: inc x BZms -> BZsign x = \1mz.
Lemma BZsign_0: BZsign \0z = \0z.
Lemma BZopp_sign x: intp x -> BZsign (BZopp x) = BZopp (BZsign x).
```

8.5 Multiplication

Given two numbers a and b, we define the product, denoted ab or $a \cdot b$, by taking the product of the absolute value, then the opposite of this, if the numbers have a different sign. Note that $xy = yx$ and $0 \cdot x = 0$, whatever x and y. Note that ab is positive if $a = 0$, if $b = 0$, or if a and b have the same sign; it is negative otherwise. The product of two non-zero numbers is non-zero.

```
Definition BZprod x y :=
  let aux := BZ_of_nat ((BZ_val x) *c (BZ_val y)) in
     (Yo (BZ_sg x = BZ_sg y) aux (BZopp aux)).
Definition BZprod_sign_aux x y:=
  Yo (x = \0z) C1 (Yo (y= \0z) C1 (Yo (BZ_sg x = BZ_sg y) C1 C0)).

Notation "x *z y" := (BZprod x y) (at level 40).

Lemma BZprodC x y: x *z y = y *z x.
Lemma BZprod_0r x: x *z \0z = \0z.
Lemma BZprod_0l x: \0z *z x = \0z.
```

```
Lemma BZprod_22: \2z *z \2z = \4z.
Lemma BZprod_val x y: BZ_val (x *z y) = (BZ_val x) *c (BZ_val y).
Lemma BZprod_nz x y: intp x -> intp y ->
  x <> \0z -> y <> \0z -> x  *z y <> \0z.
Lemma BZprod_abs2 x y: intp x -> intp y ->
   x *z y = J ((BZ_val x) *c (BZ_val y)) (BZprod_sign_aux x y).
```

Obviously $ab \in \mathbf{Z}$. We consider the cases where a and b are positive or negative.

```
Lemma ZSp x y: intp x -> intp y ->  intp (x *z y).
Lemma BZprod_pp x y: inc x BZp -> inc y BZp ->
    x *z y = BZ_of_nat ((BZ_val x) *c (BZ_val y)).
Lemma BZprod_cN x y: natp x ->  natp y ->
  BZ_of_nat x *z BZ_of_nat y = BZ_of_nat (x *c y).

Lemma BZprod_mm x y: inc x BZms -> inc y BZms ->
    x *z y = BZ_of_nat ((BZ_val x) *c (BZ_val y)).
Lemma BZprod_pm x y: inc x BZp -> inc y BZms->
   x *z y = BZm_of_nat ((BZ_val x) *c (BZ_val y)).
Lemma BZprod_mp x y: inc x BZms -> inc y BZp ->
  x *z y = BZm_of_nat ((BZ_val x) *c (BZ_val y)).
```

We how here how the product behaves regarding the partition $\mathbf{Z} = \mathbf{Z}_+ \cup \mathbf{Z}_-$.

```
Lemma ZpS_prod a b: inc a BZp -> inc b BZp -> inc (a *z b) BZp.
Lemma ZpsS_prod a b: inc a BZps -> inc b BZps -> inc (a *z b) BZps.
Lemma ZmsuS_prod a b: inc a BZms -> inc b BZms -> inc (a *z b) BZps.
Lemma ZmuS_prod a b: inc a BZm -> inc b BZm -> inc (a *z b) BZp.
Lemma ZpmsS_prod a b: inc a BZps -> inc b BZms -> inc (a *z b) BZms.
Lemma ZpmS_prod a b: inc a BZp -> inc b BZm -> inc (a *z b) BZm.
Lemma BZps_stable_prod1 a b: intp a -> intp b -> inc (a *z b) BZps ->
  ((inc a BZps <-> inc b BZps) /\ (inc a BZms <-> inc b BZms)).
```

We have $1 \cdot x = x$, $-1 \cdot x = -x$, $(-x) \cdot y = -(x \cdot y)$. Let $a(x)$ be the absolute value of x, and $s(x)$ the sign. We have $x = s(x) \cdot a(x)$ and $s(x) \cdot x = a(x)$. We have $s(x \cdot y) = s(x) \cdot s(y)$ and $a(x \cdot y) = a(x) \cdot a(y)$.

```
Lemma BZprod_1l x: intp x ->  \1z *z x = x.
Lemma BZprod_1r x: intp x ->  x *z \1z  = x.
Lemma BZprod_m1r x: intp x ->  x *z \1mz = BZopp x.
Lemma BZprod_m1l x: intp x -> \1mz *z x  = BZopp x.
Lemma BZsign_abs x: intp x -> x *z (BZsign x) = BZabs x.
Lemma BZabs_sign x: intp x -> x = (BZsign x) *z (BZabs x).
Lemma BZprod_abs x y: intp x -> intp y ->
  BZabs (x *z y) = (BZabs x) *z (BZabs y).
Lemma BZprod_sign x y: intp x -> intp y ->
  BZsign (x *z y) = (BZsign x) *z (BZsign y).
Lemma BZopp_prod_r x y: intp x -> intp y ->
  BZopp (x *z y) = x *z (BZopp y).
Lemma BZopp_prod_l x y: intp x -> intp y ->
    BZopp (x *z y) = (BZopp x) *z y.
Lemma BZprod_opp_comm x y: intp x -> intp y ->
  x *z  (BZopp y) =  (BZopp x) *z y.
Lemma BZprod_opp_opp x y: intp x -> intp y ->
  (BZopp x) *z (BZopp y) = x  *z y.
```

We have $a(bc) = (ab)c$ and $a(b+c) = ab+ac$, thus $a(b-c) = ab-ac$.

```
Lemma BZprodA x y z: intp x -> intp y -> intp z ->
  x *z (y *z z) = (x *z y) *z z.
Lemma BZprod_AC x y z: intp x -> intp y -> intp z ->
  (x *z y) *z z = (x *z z) *z y.
Lemma BZprod_CA x y z: intp x -> intp y -> intp z ->
  x *z (y *z z) = y *z (x *z z).
Lemma BZprod_ACA a b c d: intp a -> intp b -> intp c -> intp d ->
    (a *z b) *z (c *z d) = (a *z c) *z (b *z d).
Lemma BZprodDr n m p: intp n -> intp m -> intp p ->   (* 54 *)
   n *z ( m +z p) = (n *z m) +z (n *z p).
Lemma BZprodDl n m p: intp n -> intp m -> intp p ->
   (m +z p) *z n = (m *z n) +z (p *z n).
Lemma BZprodBr x y z: intp x -> intp y -> intp z  ->
   x *z (y -z z) = (x *z y) -z (x *z z).
Lemma BZprodBl x y z: intp x -> intp y -> intp z  ->
  (y -z z) *z x =  (y *z x) -z  (z *z x).
Lemma BZdoublep x: intp x -> \2z *z x = x +z x.
```

We deduce regularity of multiplication. Moreover, if $ab = 1$, then $a = \pm 1$ and $a = b$, so if $a = abc$, then either $a = 0$ or $|b| = 1$. We have $n(m+1) = nm+n$.

```
Lemma BZprod_eq2r x y z: intp x -> intp y -> intp z  -> z <> \0z ->
   x *z  z =  y *z z ->  x = y.
Lemma BZprod_eq2l x y z: intp x -> intp y -> intp z  -> z <> \0z ->
   z *z x =  z *z y ->  x = y.
Lemma BZprod_1_inversion_l x y : intp x -> intp y -> x *z y = \1z ->
   (x = y /\ (x = \1z \/ x = \1mz)).
Lemma BZprod_1_inversion_s x y : intp x -> intp y -> x *z y = \1z ->
   (BZabs y = \1z).
Lemma BZprod_1_inversion_more a b c:
  intp a -> intp b -> intp c -> a = a *z (b *z c) ->
  [\/ a = \0z, b = \1z | b = \1mz].
Lemma BZprod_succ_r  n m: intp n -> intp m ->
   n *z (BZsucc m) = (n *z m)  +z n.
Lemma BZprod_succ_l n m: intp n -> intm m ->
  (BZsucc n) *z m = (n *z m) +z m.
```

8.6 The principle of induction

We have $a \le b$ if and only if $b - a \in \mathbf{Z}_+$. Thus $a \le b$ if and only if $a + c \le b + c$. As a consequence, $\mathbf{Z}$ is an ordered group.

```
Lemma zle_diffP a b: intp a -> intp b ->  (a <=z b <-> inc (b -z a) BZp).
Lemma zle_diffP1 a b: intp a -> intp b -> (\0z <=z (b -z a) <-> a <=z b).
Lemma zlt_diffP a b: intp a -> intp b ->  (a <z b <-> inc (b -z a) BZps).
Lemma zlt_diffP1 a b: intp a -> intp b -> ((\0z  <z (b -z a) <-> a <z b).
Lemma zlt_diffP2 a b: intp a -> intp b -> (a <z b <-> inc (a -z b) BZms).

Lemma BZsum_le2l a b c: intp a -> intp b -> intp c ->
  ((c +z a) <=z (c +z b) <-> a <=z b).
Lemma BZsum_le2r a b c: intp a -> intp b -> intp c ->
  (a <=z b <->  (a +z c) <=z (b +z c)).
```

```
Lemma BZsum_lt2l a b c: intp a -> intp b -> intp c ->
  (a <z b <-> (c +z a) <z (c +z b)).
Lemma BZsum_lt2r a b c: intp a -> intp b -> intp c ->
   (a +z c <z b +z c <-> a <z b ).
```

The principle of induction on **N** is: if P(a) and P(n) $\Longrightarrow$ P($n+1$) for $n \geq a$ then P(n) is true for $n \geq a$. We have shown it by applying the basic induction principle (with $a = 0$) to Q(n) = P($n-a$). This remains true for a and n in **Z**. We may replace $n+1$ by $n-1$ and $\leq$ by $\geq$. This gives a second induction principle.

The general induction principle is: if P(a) and if P(n) $\Longrightarrow$ P($n+1$) for $n \geq a$ and if P(n) $\Longrightarrow$ P($n-1$) for $n \leq a$ then P(n) is true for $n \in \mathbf{Z}$. We give a weaker variant where $a = 0$, and the conditions $n \geq a$ and $n \leq a$ are removed.

```
Lemma BZ_induction_pos a (r:property):
  (r a) -> (forall n,  a <=z n -> r n -> r (BZsucc n)) ->
  (forall n, a <=z n -> r n).
Lemma BZ_induction_neg a (r:property):
  (r a) -> (forall n,  n <=z a -> r n -> r (BZpred n)) ->
  (forall n, n <=z a -> r n).
Lemma BZ_ind1 a (p:property):
  intp a -> p a ->
  (forall x, BZ_le a x -> p x -> p (BZsucc x)) ->
  (forall x, BZ_le x a -> p x -> p (BZpred x)) ->
  forall n, inc n BZ -> p n.
Lemma BZ_ind (p:property):
  p \0z ->
  (forall x, intp x -> p x -> p (BZsucc x)) ->
  (forall x, intp x -> p x -> p (BZpred x)) ->
  forall n, inc n BZ -> p n.
```

8.7 Properties of order

We state here a bunch of lemmas that state how the sum behaves with order.

```
Lemma BZsum_Mlele a b c d: a <=z c ->  b <=z d -> (a +z b) <=z (c +z d).
Lemma BZsum_Mlelt a b c d: a <=z c -> b <z d -> (a +z b) <z (c +z d).
Lemma BZsum_Mltle a b c d: a <z c -> b <=z d -> (a +z b) <z (c +z d).
Lemma BZsum_Mltlt a b c d: a <z c -> b <z d -> (a +z b) <z (c +z d).

Lemma BZsum_Mlege0 a c d: a <=z c ->  \0z <=z d -> a <=z (c +z d).
Lemma BZsum_Mlegt0 a c d: a <=z c ->  \0z <z d -> a <z (c +z d).
Lemma BZsum_Mltge0 a c d: a <z c ->  \0z <=z d -> a <z (c +z d).
Lemma BZsum_Mltgt0 a c d: a <z c ->  \0z <z d -> a <z (c +z d).

Lemma BZsum_Mlele0 a b c : a <=z c ->  b <=z \0z -> (a +z b) <=z c.
Lemma BZsum_Mlelt0 a b c : a <=z c ->  b <z \0z -> (a +z b) <z c.
Lemma BZsum_Mltle0 a b c : a <z c ->  b <=z \0z -> (a +z b) <z c.
Lemma BZsum_Mltlt0 a b c : a <z c ->  b <z \0z -> (a +z b) <z c.

Lemma BZsum_Mp a b: intp a -> inc b BZp ->  a <=z (a +z b).
Lemma BZsum_Mps a b: intp a -> inc b BZps ->  a <z (a +z b).
Lemma BZsum_Mm a b: intp a -> inc b BZm->  (a +z b) <=z a.
Lemma BZsum_Mms a b: intp a -> inc b BZms ->  (a +z b) <z a.
```

```
Lemma zlt_succ n: intp n -> n <z (BZsucc n).
Lemma zlt_pred n: intp n -> (BZpred n) <z n.
Lemma zlt_succ1P a b: intp a -> intp b ->
  (a <z (BZsucc b) <-> a <=z b).
Lemma zlt_succ2P a b:  intp a -> intp b ->
    (BZsucc a <=z b <-> a <z b).
```

We show $a \le |a|$ and the triangular inequality $|a+b| \le |a|+|b|$.

```
Lemma zle_abs n: intp n ->  n <=z (BZabs n).
Lemma zle_triangular n m:  intp n -> intp m ->
  (BZabs (n +z m)) <=z (BZabs n) +z (BZabs m).
```

We show here that the only order isomorphisms of **Z** are the functions $x \mapsto x+a$. Let f be an order isomorphism. We have $f(n)+1 = f(n+1)$. The argument is as follows: Since f is surjective, $f(n)+1 = f(m)$ for some m. Now $m \le n$ contradicts the fact that f is increasing. We deduce $n < m$, thus $n+1 < m+1$, thus $n+1 \le m$. Assume $n+1 < m$. Since f is strictly increasing, we get $f(n) < f(n+1)$ and $f(n+1) < f(m)$. The second relation is equivalent to $f(n+1) \le f(n)$; contradiction. We deduce $f(n)-1 = f(n-1)$. By induction $f(a) = f(0)+a$ for all a. Thus f is $x \mapsto x+f(0)$. These functions are obviously order isomorphisms.

Note: let N be the ordering on **N**, N^* the opposite of this ordering, and Z the ordinal sum of N^* and N. Then Z is isomorphic to the ordering of **Z** (this is nearly trivial, except that zero appears twice in Z). Consider order types (to be defined later). The order type of N is ω, the order type of N^* is $^*\omega$, so that the order-type of **Z** is $^*\omega+\omega$. Cantor says: the order-type is similar to itself in an infinite manner; he shows that $x \mapsto x+a$ is an order isomorphism, but not the converse. He deduces: **Z** is not well-ordered (there is unique order isomorphism on a well-ordered set).

```
Lemma BZ_order_isomorphism_P f:  (* 54 *)
 (order_isomorphism f BZ_ordering BZ_ordering) <->
 (exists2 u, inc u BZ & f = Lf (fun z => z +z u) BZ BZ).
```

We give now the characteristic property of the ordering of **Z**.

Let's say that x and y are consecutive when $x < y$, and there is no z such that $x < z < y$. In this case, we say that y is the successor of x and that x is the predecessor of y. In a totally ordered set, these quantities are unique. In **Z**, the successor is $x+1$, the predecessor is $x-1$.

```
Definition consecutive r x y :=
   glt r x y /\ forall z, inc z (substrate r) -> ~( glt r x z /\ glt r z y).
Definition or_succ r x := select (fun z => consecutive r x z) (substrate r).
Definition or_pred r x := select (fun z => consecutive r z x) (substrate r).

Lemma conseq_unique_right r x y y': total_order r ->
  consecutive r x y -> consecutive r x y' -> y = y'.
Lemma conseq_unique_left r x x' y: total_order r ->
  consecutive r x y -> consecutive r x' y -> x = x'.
Lemma or_succ_prop r x: total_order r ->
   (exists y, consecutive r x y) -> consecutive r x (or_succ r x).
Lemma or_pred_prop r x: total_order r ->
   (exists y, consecutive r y x) -> consecutive r (or_pred r x) x.
Lemma or_succ_prop' r x y: total_order r ->
```

```
  consecutive r x y ->  y = or_succ r x.
Lemma or_pred_prop' r x y: total_order r ->
  consecutive r x y ->  x = or_pred r y.
Lemma BZ_succ_pred (r := BZ_ordering) x: intp x ->
  [/\ consecutive r x (BZsucc x), consecutive r (BZpred x) x,
      or_succ r x = BZsucc x & or_pred r x = BZpred x].
```

We say that E is complete if every element has a successor and a predecessor, and connected if no non-trivial subset of E is stable by successor and predecessor. The set **Z** is totally ordered, complete and connected (proof by induction). Conversely, if E is totally ordered, complete, connected, non-empty, then it is order isomorphic to **Z**. We first notice that a function $f : \mathbf{Z} \to \mathrm{E}$ is strictly increasing when $f(z) < f(z+1)$. Consider now: $x_0 \in \mathrm{E}$, x_{n+1} the successor of x_n, $y_0 =$, y_{n+1} the predecessor of y_n; define z_n ($n \in \mathbf{Z}$) to be x_n if $n > 0$ and y_{-n} otherwise. Since E is totally ordered and complete, the successor and predecessor satisfy the desired properties; in particular, $z_n \in \mathrm{E}$, and the mapping $z \to z_n$ is an order isomorphism. It remains to show that its image is E.

```
Definition or_complete r := forall x, inc x (substrate r) ->
   (exists y, consecutive r x y) /\ (exists y, consecutive r y x).
Definition or_stable r E:= forall x, inc x E ->
   inc (or_succ r x) E /\ inc  (or_pred r x) E.
Definition or_connected r:= forall E, sub E (substrate r) -> or_stable r E ->
   E = emptyset \/ E = substrate r.
Definition or_likeZ r := [/\ total_order r, or_complete r & or_connected r].

Lemma BZ_order_props: or_likeZ BZ_ordering.
Lemma BZ_order_sfinc  f r' (r:= BZ_ordering) :
   function_prop f BZ (substrate r') -> order r' ->
   (forall z, intp z -> glt r' (Vf f z) (Vf f (BZsucc z))) ->
   strict_increasing_fun f r r'.
Lemma BZ_order_props_bis r: nonempty (substrate r) -> or_likeZ r ->
  r \Is BZ_ordering. (* 82 *)
```

We show now how the ordering behaves with product. In particular, assume $c > 0$. Then $ac \le bc$ if and only if $a \le b$ and $ac < bc$ if and only if $a < b$.

```
Lemma BZprod_Mlege0 a b c: inc c BZp -> a <=z b -> (a *z c)  <=z (b *z c).
Lemma BZprod_Mltgt0 a b c: inc c BZps -> a <z b -> (a *z c) <z (b *z c).
Lemma BZprod_Mlele0 a b c: inc c BZm -> a <=z b -> (b *z c)  <=z (a *z c).
Lemma BZprod_Mltlt0 a b c: inc c BZms -> a <z b -> (b *z c) <z (a *z c).

Lemma BZ1_small c: inc c BZps -> \1z <=z c.
Lemma BZprod_Mpp b c: inc b BZp ->  inc c BZps -> b <=z (b *z c).
Lemma BZprod_Mlepp a b c: inc b BZp -> inc c BZps -> a <=z b -> a <=z (b *z c).
Lemma BZprod_Mltpp a b c: inc b BZp -> inc c BZps -> a <z b -> a <z (b *z c).

Lemma BZprod_Mlelege0 a b c d: inc b BZp -> inc c BZp ->
  a <=z b -> c <=z  d -> (a *z c)  <=z (b *z d).
Lemma BZprod_Mltltgt0 a b c d: inc b BZps -> inc c BZps ->
  a <z b -> c <z  d -> (a *z c)  <z (b *z d).
Lemma BZprod_Mltltge0 a b c d: inc a BZp -> inc c BZp ->
  a <z b -> c <z  d -> (a *z c)  <z (b *z d).

Lemma BZprod_ple2r a b c: intp a -> intp b ->  inc c BZps ->
```

```
   ((a *z c)  <=z (b *z c) <-> a <=z b).
Lemma BZprod_plt2r a b c: intp a -> intp b ->  inc c BZps ->
  ((a *z c)  <z (b *z c) <-> a <z b).
```

8.8 Euclidean Division

Euclidean division is defined, as in the case of **N**, by $a = bq + r$, where $0 \le r < |b|$. Let $\mathrm{Q}(a,b)$ and $\mathrm{R}(a,b)$ denote the quotient and remainder of a by b. We have $a = (-b)(-q) + r$, hence $\mathrm{Q}(a,-b) = -\mathrm{Q}(a,b)$ and $\mathrm{R}(a,-b) = \mathrm{R}(a,b)$. What happens when we change the sign of a is more complicated. Assume first that b divides a. We have $\mathrm{Q}(-a,b) = -\mathrm{Q}(a,b)$. Assume now $r \neq 0$. We have $0 \le |b| - r < |b|$, so that $\mathrm{R}(-a,b) = |b| - \mathrm{R}(a,b)$. Assume $a > 0$ and $b > 0$ (if $a = 0$ division is exact). We have $-a = b(-q-1) + b - r$ and $-a = (-b)(q+1) + b - r$, so that the quotient is $-s(q+1)$ where s is the sign of b, and q the quotient of a by $|b|$. Since sq is the quotient of a by b we get $\mathrm{Q}(-a,b) = -\mathrm{Q}(a,b) - s(b)$.

When $b = 0$, there is no solution to $0 \le r < |b|$. We define the quotient to be zero as in the case of **N**.

```
Definition BZdivision_prop a b q r :=
  [/\ a = (b *z q) +z r, r <z (BZabs b) & inc r BZp].

Definition BZquo a b :=
  let  q:= BZ_of_nat ((BZ_val a) %/c (BZ_val b)) in
    Yo (b = \0z) \0z
    (Yo (BZ_sg a= C1) (Yo (BZ_sg b = C1) q (BZopp q))
      (Yo ((BZ_val a %%c BZ_val b) = \0c)  (Yo (BZ_sg b = C1) (BZopp q) q)
        (Yo (BZ_sg b = C1) (BZopp (BZsucc q)) (BZsucc q)))).

Notation "x %/z y" := (BZquo x y) (at level 40).

Definition BZrem a b := a -z b *z (a %/z b).
Notation "x %%z y" := (BZrem x y) (at level 40).
```

We first study how the quotient behaves when a sign changes. The non-trivial point is that, if $|b|$ does not divide $|a|$, then the remainder of the division of a by b cannot be zero. Moreover, $\mathrm{R}(-a,b) = |b| - \mathrm{R}(a,b)$. It suffices to prove this for positive b; it is then equivalent to $-\mathrm{Q}(-a,b) = 1 + \mathrm{Q}(a,b)$. We deduce, that, in any case $0 \le \mathrm{R}(a,b) < |b|$.

```
Lemma BZquo_val a b (q1:= ((BZ_val a) %/c (BZ_val b))) (q2 := BZ_of_nat q1):
   intp a -> intp b -> [/\ inc q1 Nat, inc q2 BZp & inc q2 BZ].
Lemma BZ_quo0 a: a %/z \0z = \0z.
Lemma BZ_quorem0 a: intp a -> (a %%z \0z = a /\ a %/z \0z = \0z).
Lemma BZ_quorem00 b: intp b -> (\0z %%z b = \0z /\ \0z %/z b = \0z).

Lemma ZS_quo a b: intp a -> intp b -> intp (a %/z b).
Lemma ZS_rem a b: intp a -> intp b -> intp (a %%z b).
Lemma ZpS_quo a b: intp ap -> inc b BZp -> inc (a %/z b) BZp.
Lemma BZquo_opp_b a b: intp a -> intp b -> a %/z (BZopp b) = BZopp (a %/z b).
Lemma BZrem_opp_b a b: intp a -> intp b -> a %%z (BZopp b) = a %%z b.
Lemma BZquo_div1 a b: intp a -> intp b -> b <> \0z ->
   (BZ_val a) %%c (BZ_val b) = \0c -> a = b *z  (a %/z b).
Lemma BZrem_div1 a b: intp a -> intp b -> b <> \0z ->
   (BZ_val a) %%c (BZ_val b) = \0c -> (a %%z b) = \0z.
Lemma BZquo_opp_a1 a b: intp a -> intp b -> b <> \0z ->
```

```
   (BZ_val a) %%c (BZ_val b) = \0c -> (BZopp a) %/z b = BZopp (a %/z b).
Lemma BZdivision_opp_a2 a b:
   intp a -> intp b -> b <> \0z -> (P a %%c P b) <> \0c ->
  ( (BZopp a) %%z b <> \0z
   /\ (BZopp a) %%z b =  (BZabs b) -z (a %%z b)).
Lemma BZdvd_correct a b: intp a -> intp b -> b <> \0z ->
  [/\ inc (a %/z b) BZ, inc (a %%z b) BZp &
  (BZdivision_prop a b (a %/z b) (a %%z b))].
```

We deduce $0 \le \mathrm{R}(a,b) < |b|$. It follows $bq \le a < b(q+s)$, where s is the sign of b and $q = \mathrm{Q}(a,b)$. This equation has a unique solution (if $b > 0$, $bq \le a$ and $a < b(q'+s)$ says $q < q'+1$, thus $q \le q'$). Thus $a = bq + r$ and $0 \le r < |b|$ uniquely define q and r.

```
Lemma ZpS_rem a b: intp a -> intp b -> b <> \0z -> inc (a %%z b) BZp.
Lemma BZrem_small a b:  intp a -> intp b -> b <> \0z ->
   (a %%z b) <z (BZabs b).
Lemma BZdvd_exact b q: intp q -> intp b -> b <> \0z ->
  ((q *z b) %/z b = q /\  (q *z b) %%z b = \0z).
Lemma BZdvd_unique a b q r q' r': intp a -> intp b ->  b <> \0z ->
  intp q -> intp r -> intp q' -> intp r' ->
  BZdivision_prop a b q r -> BZdivision_prop a b q' r' ->
  (q = q' /\ r =r').
Lemma BZdvd_unique1 a b q r: intp a -> intp b ->
  intp q -> intp r -> b <> \0z ->
  BZdivision_prop a b q r -> (q = a %/z b /\ r  = a %%z b).

Lemma BZquo_cN a b: natp a -> natp b ->
  (BZ_of_nat a) %/z  (BZ_of_nat b) =  BZ_of_nat (a%/c b).
Lemma BZrem_cN a b: natp a -> natp b ->
  (BZ_of_nat a) %%z  (BZ_of_nat b) =  BZ_of_nat (a%%c b).
```

8.8.1 Divisibility

We say that b divides a if the remainder of the division is zero. The quotient q is then denoted by a/b and satisfies $a = bq$ (we allow b to be zero, but in this case, a has to be zero).

```
Definition BZdivides b a :=
  [/\ intp a, intp b & BZrem a b = \0z].

Notation "x %|z y" := (BZdivides x y) (at level 40).
```

Conversely, b divides bq. Since b divides a if only only if $\mathrm{pr}_1 b$ divides $\mathrm{pr}_1 a$ (as elements of **N**), we get that b divides a if and only if $\pm b$ divides $\pm a$.

```
Lemma BZdvds_trivial: \0z %|z \0z.
Lemma BZdvds_trivial_rec x: \0z %|z x -> x = \0z.
Lemma BZdvds_pr a b: b %|z a -> a = b *z  (a %/z b).
Lemma BZdvds_pr1 a b: intp a -> intp b ->  b %|z (a *z b).
Lemma BZdvds_pr1' a b: intp a -> intp b -> b %|z (b *z a).
Lemma BZdvd_pr2 a b q: inc q BZ -> intp b -> b <> \0z ->
  a = b *z q -> q = a %/z b.
Lemma BZdvds_pr0 a b: b %|z a ->  (BZ_val b) %|c (BZ_val a).
Lemma BZdvds_pr3 a b: intp a -> intp b ->
  (b %|z a <->  (BZ_val b) %|c  (BZ_val a)).
```

```
Lemma BZdiv_cN a b: natp a -> natp b ->
   ((a %|c b) <-> (BZ_of_nat a) %|z (BZ_of_nat b)).
Lemma BZdvds_opp1 a b: intp b -> (b %|z a <->  (BZopp b) %|z a).
Lemma BZdvds_opp2 a b: intp a -> (b  %|z a  <-> b %|z (BZopp a)).
Lemma BZquo_opp2 a b: b %|z a -> (BZopp a) %/z b = BZopp (a %/z b).

Lemma BZdvds_one a: intp a -> \1z %|z a.
Lemma BZdvds_mone a: intp a -> \1mz %|z a.
Lemma BZquo_one a: intp a -> a %/z \1z = a.
Lemma BZquo_mone a: intp a -> a %/z \1mz = BZopp a.

Lemma BZdvds_pr4 a b q: b %|z a -> q = a %/z b -> a = b *z q.
Lemma BZdvds_pr5 b q:  intp b -> inc q BZ -> b <> \0z -> (b *z q) %/z b = q.
Lemma BZdvd_itself a: intp a -> a <> \0z ->  (a %|z a /\ a %/z a = \1z).
Lemma BZdvd_opp a: intp a -> a <> \0z ->
   (a %|z (BZopp a) /\  (BZopp a)  %/z a = \1mz).
Lemma BZdvd_zero1 a: intp a -> (a %|z \0z /\ \0z %/z a = \0z).
Lemma BZdvds_trans a b a': a %|z a'-> b %|z a ->  b %|z  a'.
Lemma BZdvds_trans1 a b a': a %|z a'-> b %|z a ->
    a' %/z b = (a' %/z a) *z (a %/z b).
Lemma BZdvds_trans2 a b c: intp c -> b %|z a -> b %|z (c *z a).
```

If $a = bq + r$ we have $ac = b(qc) + rc$. We deduce, when $c > 0$, that $(ac/bc) = (a/b)$. Whatever $c \neq 0$, b divides a if and only if bc divides ac. We have $(a+b)/c = (a/c)+(b/c)$ when division is exact.

```
Lemma BZquo_simplify a b c: intp a -> intp b -> inc c BZps ->
   ( (a *z c) %/z(b *z c) = a %/z b /\
     (a *z c) %%z (b *z c) =  (a %%z b) *z c).
Lemma BZdvds_prod a b c: intp a -> intp b -> intp c -> c <> \0z ->
   ( a %|z b <->  (a *z c) %|z (b *z c)).
Lemma BZdvd_and_sum a a' b: b %|z a -> b %|z a' ->
  ( b %|z (a +z a') /\ (a +z a') %/z b =  (a %/z b) +z (a' %/z b)).
Lemma BZdvd_and_diff a a' b: b %|z a -> b %|z a'
  -> ( b %|z (a -z a') /\ (a -z a') %/z b =  (a %/z b) -z (a' %/z b)).
```

8.8.2 Ideals and Gcd

An *ideal* I is a set stable by addition and by multiplication by an element of **Z**.

```
Definition BZ_ideal x:=
  [/\ (forall a, inc a x -> intp a),
      (forall a b, inc a x -> inc b x -> inc (a +z b) x) &
      (forall a b, inc a x -> intp b -> inc (a *z  b) x)].
Definition BZ_ideal2 a b := Zo BZ (fun z =>
  exists u v, [/\ intp u, intp v & z = (a *z u) +z (b *z v)]).
Definition BZ_ideal1 a := fun_image BZ (fun z => a *z z).
```

We denote by $\mathrm{I}(a,b)$ the set of elements of the form $au+bv$. It contains a, b and is an ideal. We denote by $\mathrm{I}(a)$ the set of all multiples of a. It is $\mathrm{I}(a,a)$. It is the ideal *generated* by a.

```
Lemma BZ_in_ideal1 a b: intp a -> intp b ->
  (inc a (BZ_ideal2 a b) /\ inc b (BZ_ideal2 a b)).
Lemma BZ_is_ideal2 a b: intp a -> intp b -> BZ_ideal (BZ_ideal2 a b).
Lemma BZ_in_ideal3 a: intp a ->  BZ_ideal1 a = BZ_ideal2 a a.
```

```
Lemma BZ_in_ideal4 a: intp a ->
 (BZ_ideal (BZ_ideal1 a) /\ inc a (BZ_ideal1 a)).
```

An ideal is stable by taking the opposite, absolute value and remainder.

```
Lemma BZ_idealS_opp a x: BZ_ideal x -> inc a x -> inc (BZopp a) x.
Lemma BZ_idealS_abs a x: BZ_ideal x -> inc a x -> inc (BZabs a) x.
Lemma BZ_idealS_diff a b x:
 BZ_ideal x -> inc a x ->  inc b x ->  inc (a -z b) x.
```

We show that $\mathbf{Z}$ is *principal*: this means that every ideal is of the form $\mathrm{I}(a)$ for some a. We first state that any non-empty subset of $\mathbf{Z}_+$ has a least element. Consider a non-empty ideal x. If it contains only 0, it is $\mathrm{I}(0)$. Otherwise, its intersection with $\mathbf{Z}_+^*$ is non-empty, and has a least element a. For any b in the ideal, the remainder of b by a has to be zero, hence $b \in \mathrm{I}(a)$. This element a is unique, modulo its sign, for if $\mathrm{I}(a) = \mathrm{I}(a')$, we have x and y such that $a = xa'$ and $a' = ya$, thus $a = a(1 - xy)$. This shows that either $a = 0$ (thus $a' = 0$) or $xy = 1$, thus $x = \pm 1$, hence $a' = \pm a$, so that $|a| = |a'|$.

```
Lemma BZ_N_worder X: sub X BZp -> nonempty X ->
  exists2 a, inc a X & forall b, inc b X -> BZ_le a b.
Lemma BZ_ideal_0P a: inc a (BZ_ideal1 \0z) <->  (a = \0z).
Theorem BZ_principal x: BZ_ideal x -> nonempty x ->
  exists2 a, inc a BZp & BZ_ideal1 a = x.
Lemma BZ_ideal_unique_gen a b: intp a -> intp b ->
  BZ_ideal1 a = BZ_ideal1 b -> BZabs a = BZabs b.
Lemma BZ_ideal_unique_gen1 a b: inc a BZp -> inc b BZp ->
  BZ_ideal1 a = BZ_ideal1 b -> a = b.
```

The unique positive generator of $\mathrm{I}(a, b)$ is called the *greatest common divisor* of a and b. If g is the gcd, the quantity ab/g is called the *least common multiple* of a and b.

```
Definition BZgcd a b := select (fun z => BZ_ideal1 z = BZ_ideal2 a b) BZp.
Definition BZlcm a b := (a *z b) %/z (BZgcd a b).
```

Since $\mathbf{Z}$ is principal, the gcd g of a and b satisfies $\mathrm{I}(g) = \mathrm{I}(a, b)$. We deduce $a \in \mathrm{I}(g)$ and $b \in \mathrm{I}(g)$, and rewrite it as $a = g \cdot (a/g)$ and $b = g \cdot (b/g)$. The two integers a/g and b/g are sometimes called the cofactors of a and b. If $a \geq 0$ or $a \neq 0$ so is a/g.

Conversely, $g \in \mathrm{I}(a, b)$, so that $g = au + bv$ for some u and v. We have $\gcd(a, 0) = |a|$. We have $\gcd(a, b) = \gcd(-a, b)$. The gcd is zero if and only if $a = b = 0$.

```
Lemma BZgcd_prop1 a b: intp a -> intp b ->
  (inc (BZgcd a b) BZp /\ BZ_ideal1  (BZgcd a b) = BZ_ideal2 a b).
Lemma ZpS_gcd a b: intp a -> intp b -> inc (BZgcd a b) BZp.
Lemma ZS_gcd a b: intp a -> intp b -> intp (BZgcd a b).
Lemma BZgcd_unq a b g: intp a -> intp b ->
  inc g BZp -> BZ_ideal1 g = BZ_ideal2 a b ->
   g = (BZgcd a b).
Lemma BZgcd_x1 x: intp x -> BZgcd x \1z = \1z.
Lemma BZgcd_div a b (g:= (BZgcd a b)): intp a -> intp b ->
  a = g *z  (a %/z g) /\ b = g *z  (b %/z g).
Lemma BZgcd_s2 a b (g:= (BZgcd a b)): intp a -> intp b ->
  [/\ inc g BZ, inc (a %/z g) BZ & inc (b %/z g) BZ].
Lemma BZgcd_nz a b: intp a ->  intp b ->
```

```
  BZgcd a b = \0z ->  (a = \0z /\ b = \0z).
Lemma BZgcd_nz1 a b: intp a ->  intp b ->
  (a <> \0z \/ b <> \0z) -> BZgcd a b <> \0z.
Lemma BZ_nz_quo_gcd a b: intp a -> intp b -> a <> \0z ->
   a %/z (BZgcd a b) <> \0z.
Lemma BZ_positive_quo_gcd a b: inc a BZp -> intp b ->
  inc (a %/z (BZgcd a b)) BZp.
Lemma BZgcd_prop2 a b: intp a -> intp b ->
  (exists x y, [/\ intp x, intp y  &
    (BZgcd a b =  (a *z x) +z (b *z y))]).
Lemma BZgcd_opp a b: intp a -> intp b -> BZgcd a b = BZgcd (BZopp a) b.
Lemma BZgcd_C a b: BZgcd a b = BZgcd b a.

Lemma BZgcd_id a: intp a -> BZgcd a a = BZabs a.
Lemma BZgcd_rem a b q: intp a -> intp b -> inc q BZ ->
    BZgcd a (b +z a *z q)  = BZgcd a b.
Lemma BZgcd_diff a b: intp a -> intp b ->  BZgcd a (b -z a)  = BZgcd a b.
Lemma BZgcd_zero a: intp a -> BZgcd a \0z = BZabs a.
Lemma BZgcd_div2 a b: intp a -> intp b -> BZgcd a b %|z a.
```

Basic properties of lcm.

```
Lemma ZS_lcm a b: intp a -> intp b -> intp (BZlcm a b).
Lemma BZlcm_C a b: BZlcm a b = BZlcm b a.
Lemma BZlcm_zero a: intp a -> BZlcm a \0z = \0z.
Lemma BZlcm_prop1 a b (g := BZgcd a b) (l := BZlcm a b):
  intp a -> intp b ->
  [/\ l =  (a %/z g) *z b, l = a *z (b %/z g) &
    l =  ((a %/z g) *z (b %/z g)) *z g].
Lemma BZlcm_nz a b: intp a -> intp b ->
  a <> \0z -> b <> \0z -> BZlcm a b <> \0z.
```

Given a and b, the Bezout relation is

$$\exists u, v \quad au + bv = 1$$

This says that $1 \in \mathrm{I}(a, b)$, and is equivalent to say that a and b are *coprime* (the gcd is one). If one of a or b is non-zero, the gcd g of a and b is non-zero, there is a Bezout relation between a/g and b/g. (There are algorithms, not given here, that compute the gcd and the Bezout relation).

```
Definition BZcoprime a b :=  BZgcd a b = \1z.
Definition Bezout_rel a b u v :=  (a *z u) +z (b *z v) = \1z.
Definition BZBezout a b :=
  exists u v, [/\ intp u, intp v & Bezout_rel a b u v].

Lemma BZcoprime_sym a b: BZcoprime a b -> BZcoprime b a.
Lemma BZcoprime_add a b: intp a -> intp b ->
  BZcoprime a b -> BZcoprime a (a +z b).
Lemma BZcoprime_diff a b: intp a -> intp b ->
  BZcoprime a b -> BZcoprime a (b -z a).
Lemma BZ_Bezout_if_coprime a b: intp a -> intp b ->
  BZcoprime a b -> BZBezout a b.
Lemma BZ_coprime_if_Bezout a b: intp a -> intp b ->
  BZBezout a b -> BZcoprime a b.
```

```
Lemma  BZ_Bezout_cofactors a b: intp a -> intp b ->
 (a<> \0z \/ b <> \0z) ->
  BZBezout (a %/z (BZgcd a b)) (b %/z (BZgcd a b)).
Lemma BZ_coprime1r  a: intp a -> BZcoprime a \1z.
Lemma BZ_coprime1l a: intp a -> BZcoprime \1z a.
```

The relation "a divides b" is an order on $\mathbf{Z}_+^*$. Note that, if a divides b and $b > 0$, then $a \le b$. Note that a divides b if and only if $\mathrm{I}(b) \subset \mathrm{I}(a)$.

The greatest lower bound of a and b for this relation is the gcd. We state this as: let $\mathrm{P}(x)$ be the property: x divides a, x divides b and any y that divides a and b divides y. Let P' be the same relation with the constraint that x and y are positive. Then the gcd satisfies P and P'. If x satisfies P, then its absolute value is the gcd, if x satisfies P', it is the gcd.

The lcm is the least upper bound. In particular, if a and b are coprime, the lcm is the product, so that if a and b divide x, so does the product.

Note that the gcd is associative (consider the ideal generated by a, b and c), and distribute with the product: if $c \ge 0$ then $c \cdot \gcd(a,b) = \gcd(ca, cb)$. We have $\gcd(a, bc) = \gcd(a, c)$ whenever a and b are coprime.

```
Definition BZdvdordering := graph_on BZdivides BZps.
Definition BZgcd_prop a b p :=
   [/\ p %|z a, p %|z b  & forall t, t %|z a  -> t %|z b  -> t %|z p].
Definition BZgcdp_prop a b p :=
   [/\ inc p BZp, p %|z a, p %|z b  &
    forall t, inc t BZp ->  t %|z a  -> t %|z b  -> t %|z p].

Lemma BZdvds_pr6 a b: intp a -> intp b -> ->
  (a %|z b <-> sub (BZ_ideal1 b)(BZ_ideal1 a)).
Lemma BZdvds_pr6' a b: a %|z b -> sub (BZ_ideal1 b)(BZ_ideal1 a).
Lemma BZdvds_monotone a b: inc b BZps -> a %|z b -> a <=z b.
Lemma BZdvdordering_or: order BZdvdordering.
Lemma BZgcd_prop3 a b: intp a -> intp b ->
   (BZgcd_prop a b (BZgcd a b)
       /\ forall g,  BZgcd_prop a b g -> (BZgcd a b) = BZabs g).
Lemma BZgcd_prop3' a b: intp a -> intp b ->
   (BZgcdp_prop a b (BZgcd a b)
       /\ forall g,  BZgcdp_prop a b g -> (BZgcd a b) = g).
Lemma BZlcm_prop2 a b (l := BZlcm a b):
  intp a -> intp b ->
  [/\ a %|z l, b %|z  l & forall u, a %|z  u ->  b  %|z u ->  l  %|z  u].
Lemma BZ_lcm_prop3 a b u: BZcoprime a b ->
  a  %|z u -> b  %|z u ->  (a *z b)  %|z u.

Lemma BZpsS_gcd x y: inc x BZps -> inc y BZps -> inc (BZgcd x y) BZps.
Lemma BZpsS_lcm x y: inc x BZps -> inc y BZps -> inc (BZlcm x y) BZps.
Lemma BZgcd_A a b c: intp a -> intp b -> intp c ->
   (BZgcd a (BZgcd b c)) = (BZgcd (BZgcd a b) c).
Lemma BZgcd_prodD a b c: intp a -> intp b -> inc c BZp ->
   (BZgcd (c *z a) (c *z b)) = c *z (BZgcd a b).
Lemma BZgcd_simp a b c: intp a -> intp b -> intp c ->
   BZcoprime a b -> BZgcd a (b *z c) = BZgcd a c.
```

We show here that the set $\mathbf{Z}_+^*$ of strictly positive integers, ordered by divisibility, is a lattice, where the sup and inf are the gcd and lcm.

```
Lemma BZdvdordering_sr: substrate BZdvdordering = BZps.
Lemma BZdvdordering_gle x y:
  gle BZdvdordering x y <-> [/\ inc x BZps, inc y BZps & x %|z y].

Lemma BZdvd_lattice_aux x y: inc x BZps -> inc y BZps ->
  (least_upper_bound BZdvdordering (doubleton x y) (BZlcm x y)
   /\ (greatest_lower_bound BZdvdordering (doubleton x y) (BZgcd x y))).
Lemma BZdvd_lattice: lattice BZdvdordering.
Lemma BZdvd_sup x y: inc x BZps -> inc y BZps ->
  sup  BZdvdordering x y = BZlcm x y.
Lemma BZdvd_inf x y: inc x BZps -> inc y BZps ->
  inf BZdvdordering x y = BZgcd x y.
```

Note that the set $\mathbf{Z}_+^*$ of strictly positive integers, ordered by divisibility, is a distributive lattice. This means $\gcd(a, \mathrm{lcm}(b,c)) = \mathrm{lcm}(\gcd(a,b), \gcd(a,c))$. Proof: consider the set F of functions $\mathrm{P} \to \mathbf{N}$ (here P is any set, $f \le f'$ means $\forall p \in \mathrm{P}, f(p) \le f'(p)$). This is a product of totally ordered set, thus is a distributive lattice. Consider the subset F′ of functions with finite support (the set of all p such that $f(p) \neq 0$ is finite); given two elements of F′, they have the same sup and inf, considered in F′ or F. Thus F′ is a distributive lattice. Let P be the set of prime numbers; for any function f with finite support consider the product $\prod_p p^{f(p)}$; this is a non-zero natural number, thus can be consider in $\mathbf{Z}_+^*$. We get an order isomorphism $\mathrm{F}' \to \mathbf{Z}_+^*$.

Direct proof. According to Exercise 1.16, $\mathbf{Z}_+^*$ is a distributive lattice if, and only if, for all x, y and z, the relation $\inf(z, \sup(x,y)) \le \sup(x, \inf(y,z))$ holds. This can be restated as $\gcd(z, \mathrm{lcm}(x,y))$ divides $\mathrm{lcm}(x, \gcd(y,z))$. All quantities introduced below are > 0, as we assume that x, y and z are > 0. Let g be the gcd of x and y, write $x = ag$, and $y = bg$. The quantities a and b are coprime and we want to show: $\gcd(z, abg)$ divides $\mathrm{lcm}(ag, \gcd(bg,z))$. Let p be the gcd of ag and $\gcd(bg,z))$, so that the lcm becomes the product of the three factors $(ag)/p$, $\gcd(bg,z)/p$ and p. By associativity, p is the gcd of $\gcd(ag,bg)$ and z. The first quantity is $g \cdot \gcd(a,b)$ so that $p = \gcd(g,z)$. Let $g = g'p$ and $z = z'p$, where g' and p' are coprime. We have $\gcd(z, abg) = \gcd(z'p, abg'p) = p \cdot \gcd(z', abg')$ This simplifies to $p \cdot \gcd(z', ab)$. Note that $(ag)/p = (ag'p)/p = ag'$. Moreover $\gcd(bg,z)/p = \gcd(bg'p, z'p)/p = \gcd(bg', z') = \gcd(b, z')$. We have to show that $p \cdot \gcd(z', ab)$ divides $p \cdot g' a \gcd(z', b)$. We can get rid of p. We can forget about g'. It suffices to show that $\gcd(z', ab)$ divides $a \gcd(z', b) = \gcd(az', ab)$.

```
Lemma BZdvd_sup x y: inc x BZps -> inc y BZps ->
  sup  BZdvdordering x y = BZlcm x y.
Lemma BZdvd_inf x y: inc x BZps -> inc y BZps ->
  inf BZdvdordering x y = BZgcd x y.

Lemma BZdvd_latticeD: distributive_lattice1 BZdvdordering.
```

The Bezout relation is not unique: if $au + bv = 1$, $u' = u + qb$, $v' = v - qa$, then $au' + bv' = 1$. We show the converse: this amounts to: if $au + bv = 0$, then there is q such that $u = bq$ and $v = -aq$. Proof. Not first that if $a = 0$, then $b = \pm 1$ and the result holds. Otherwise, write $v = aq + r$ and the relation as $a(u + bq) + br = 0$. Let g be the gcd of a and this quantity. This is $\gcd(a, r) = \gcd(a, r)$. Since the quantity is zero, it is also $|a|$. We may choose $0 \le r < |a|$. But this relation says that $|a|$ cannot divide r.

We deduce the following

$$\gcd(a,b) = 1, a \neq 0 \implies \exists u, v, \quad 0 \le v < |a|, \quad au + bv = 1. \tag{8.5}$$

Here we have uniqueness. Assume moreover b non-zero; then $|u| \le |b|$. Assume further $1 < |b|$; then $|u| < |b|$. (in the case $a = b = 1$, we have the solution $u = 1$, $v = 0$).

```
Definition Bezout_pos a b u v :=
    [/\ inc u BZ, inc v BZp, v <z BZabs a & Bezout_rel a b u v ].

Lemma Bezout_non_unique1 a b u v: intp a -> intp b ->
   intp u -> intp v -> BZcoprime a b -> (a *z u) +z (b *z v) = \0z ->
   exists q, [/\ intp q, u = q *z b & v = BZopp(q *z a) ].
Lemma Bezout_non_unique2 a b u v u' v': intp a -> intp b ->
   intp u -> intp v -> intp u' -> intp v' ->
    Bezout_rel a b u v -> Bezout_rel a b u' v' ->
   exists q, [/\ intp q, u' = u +z q *z b & v' = v -z q *z a].

Lemma Bezout_pos_exists a b: intp a -> intp b -> a <> \0z ->
    BZcoprime a b -> exists u v, Bezout_pos a b u v.
Lemma Bezout_pos_unique a b u v u' v': intp a -> intp b ->
   Bezout_pos a b u v  -> Bezout_pos a b u' v' ->
   (u = u' /\ v = v').
Lemma Bezout_pos_aux a b u v : intp a -> intp b -> b <> \0z ->
   Bezout_pos a b u v  -> BZabs u <=z BZabs b.
Lemma Bezout_pos_aux2 a b u v : intp a -> intp b -> b <> \0z ->
    BZabs b <> \1z ->  Bezout_pos a b u v  -> BZabs u <z BZabs b.
```

8.8.3 Gcd of natural numbers

We define here the gcd of two natural numbers a and b as the absolute value of the gcd considered on **Z**.

```
Definition Ngcd n m := BZ_val (BZgcd (BZ_of_nat n)(BZ_of_nat m)).
Definition Ncoprime a b := Ngcd a b = \1c.
Definition Ngcd_prop a b p :=
   [/\ natp p, p %|c a, p %|c b  &
    forall t, natp t  ->  t %|c a  -> t %|c b  -> t %|c p].

Lemma NS_gcd n m: natp n -> natp m -> natp (Ngcd n m).
Lemma Ngcd_C n m: Ngcd n m = Ngcd m n.
Lemma Ngcd_n1 n: natp n ->  Ngcd n \1c = \1c.
Lemma Ngcd_1n n: natp n ->  Ngcd \1c n = \1c.
Lemma Ngcd_n0 n: natp n ->  Ngcd n \0c = n.
Lemma Ngcd_0n n: natp n ->  Ngcd \0c n = n.
Lemma Ngcd_nn n: natp n ->  Ngcd n n = n.

Lemma Ngcd_div a b (g:= (Ngcd a b)): natp a -> natp b ->
  a = g *c  (a %/c g) /\ b = g *c  (b %/c g).
Lemma Ngcd_nz a b: natp a -> natp b ->
  Ngcd a b = \0c ->  (a = \0c /\ b = \0c).
Lemma Ngcd_nz1 a b: natp a -> natp b->
  (a <> \0c \/ b <> \0c) -> Ngcd a b <> \0c.
Lemma Ngcd_rem a b q: natp a ->natp b -> natp q ->
    Ngcd a (b +c a *c q)  = Ngcd a b.
Lemma Ngcd_sum a b: natp a -> natp b ->  Ngcd a (a +c b) = Ngcd a b.
Lemma Ngcd_diff a b: natp a -> natp b -> a <=c b ->
  Ngcd a (b -c a)  = Ngcd a b.
Lemma Ngcd_simp a b c: natp a -> natp b -> natp  c ->
   Ncoprime a b -> Ngcd a (b *c c) = Ngcd a c.

Lemma Ngcd_P a b: natp a -> natp b ->
```

```
  (Ngcd_prop a b (Ngcd a b)
      /\ forall g,  Ngcd_prop a b g -> (Ngcd a b) = g).
```

Assume a and b coprime. We may express the Bezout relation as: there are two natural numbers u and v such that $au = 1 + bv$ (this requires $a \neq 0$). Proof. If $b = 0$, we know $a = 1$, and the result is trivial. Otherwise, we know that there is a solution with $u < b$, and $v \in \mathbf{Z}$. Assume $v < 0$. From $1 + bv \geq 0$ we deduce $u = 0$, $b = 1$, and the problem has a solution. We may assume $u < b$ (except in the case where $b = 0$ or $b = 1$).

```
Lemma Ngcd_simp a b c: natp a -> natp b -> natp  c ->
   Ncoprime a b -> Ngcd a (b *c c) = Ngcd a c.
Lemma Nbezout a b: natp a -> natp b -> a <> \0c -> Ncoprime a b ->
   exists u v, [/\ natp u, natp v, a *c u = \1c +c b *c v&
      (b <=c \1c \/ u <c b)].
```

Application $\gcd(F_n, F_m) = F_{\gcd(n,m)}$, where F_n is the n-th Fibonacci number. Assume first $m = n+1$. The relation says that two consecutive Fibonacci numbers are coprime. The proof is by induction and follows from $F_{n+2} = F_n + F_{n+1}$. The general case is by induction on the maximum of n and m. We may assume $m < n$, as $m = n$ is trivial. We may also assume $m \neq 0$. So, we can write $n = m + r$, where r is non-zero, $r < n$. In particular, the induction hypothesis applies and gives $\gcd(F_r, F_m) = F_k$, where $k = \gcd(r, m) = \gcd(n, m)$. Now, $F_{m+r} = F_m F_{r-1} + F_{m+1} F_r$. If we take the gcd with F_m, we can ignore the first term of the sum, and the factor of F_r (since F_{m+1} is coprime with F_m). Thus $\gcd(F_n, F_m) = \gcd(F_r, F_m)$.

```
Lemma Ncoprime_Sn_fib n: natp n -> Ncoprime (Fib n) (Fib (csucc n)).
Lemma Ngcd_fib n m: natp n -> natp m ->
  Ngcd (Fib n) (Fib m) = Fib (Ngcd n m).
```

8.9 Equivalence of definitions

We prove that our integers are the same as those of COQ. There are two implementations: in `ZArith`, an integer is zero, positive or the opposite of a positive, where a "positive" is a binary representation of a non-zero natural number. The SSREFLECT library provides an alternative version where a positive number is a `nat` (and there is a coercion from `nat` to `int`), and a negative number `Negz n` is the opposite of $n+1$, where n is a `nat`. In fact, there is a lemma `NegzE` that says that `Negz n` is equal to '`ssralg.GRing.opp n.+1`' (there is an implicit argument, namely `int_ZmodType`). Loading the `ssralg` library simplifies this to '`(- n.+1)%R`'; opening the ring scope simplifies it further to '`- n.+1`'. The multiplication is commutative: this is the consequence of some hidden theorem; one just uses the fact that $\mathbf{Z}$ is a commutative ring. Moreover, $\mathbf{Z}$ is a RealDomainType: it is totally ordered, and there is a norm $|x|$. This explains the following lines.

```
Section Conversions.

Require Import ssralg ssrnum.
Import GRing.Theory.
Local Open Scope ring_scope.
```

We define here a function that maps the set $\mathbb{Z}$ of SSREFLECT integers onto $\mathbf{Z}$. We show that it is a morphism.

```
Definition BZ_of_Z (n:int) :=
  match n with
    | Posz p => BZ_of_nat (nat_to_B p)
    | Negz p => BZm_of_nat (nat_to_B p.+1)
end.

Lemma positive_non_zero (p: positive) :
  inc (nat_to_B (nat_of_P p)) Nats.


Lemma positive_non_zero1 p :
  inc (BZ_of_nat (nat_to_B p)) BZp.
Lemma positive_non_zero2 p :
  inc (BZm_of_nat (csucc (nat_to_B p))) BZms.
Lemma positive_non_zero2bis p :
  inc (BZm_of_nat (nat_to_B p.+1)) BZms.

Lemma BZ_of_Z_inc x: inc (BZ_of_Z x) BZ.
Lemma BZ_of_Z_injective x y :  BZ_of_Z x = BZ_of_Z y ->  x = y.
Lemma BZ_of_Zp_surjective x : inc x BZp -> exists y:nat, BZ_of_Z y = x.
Lemma BZ_of_Z_surjective x : intp x -> exists y, BZ_of_Z y = x.
```

We get an inverse by using the axiom of choice.

```
Fact nonempty_int: inhabited int.
Definition Z_of_BZ x :=  (chooseT (fun y => BZ_of_Z y = x) nonempty_int).

Lemma Z_of_BZ_pa x: intp x -> (BZ_of_Z (Z_of_BZ x)) = x.
Lemma Z_of_BZ_pb p : Z_of_BZ (BZ_of_Z p) = p.

Lemma BZ_Z_0: BZ_of_Z 0 = \0z.
Lemma BZ_Z_1: BZ_of_Z 1 = \1z.
Lemma BZ_Z_1n: BZ_of_Z 1%N = \1z.
Lemma BZ_Z_m1: BZ_of_Z (- 1)%Z = \1mz.
Lemma Z_of_BZ_zero: Z_of_BZ \0z = 0%Z.
Lemma Z_of_BZ_one: Z_of_BZ \1z = 1%Z.
Lemma Z_of_BZ_mone: Z_of_BZ \1mz = (-1)%Z.
```

We show that the function defined above preserves all properties.

```
Lemma BZ_of_Z_opp x: BZ_of_Z (- x) = BZopp (BZ_of_Z x).
Lemma BZ_of_Z_neg p: BZ_of_Z (Negz p) = BZopp (BZ_of_Z (Posz p.+1)).
Lemma BZ_of_Z_abs x: BZ_of_Z (`|x|) = BZabs (BZ_of_Z x).
Lemma BZ_of_Z_succ (x:int): BZ_of_Z (1 + x) = \1z +z BZ_of_Z x.
Lemma BZ_of_Z_sum x y: BZ_of_Z (x + y) = (BZ_of_Z x) +z (BZ_of_Z y).
Lemma BZ_of_Z_diff x y: BZ_of_Z (x - y ) = (BZ_of_Z x) -z (BZ_of_Z y).
Lemma BZ_of_Z_pred (x:int): BZ_of_Z (x -1) =  BZ_of_Z x -z \1z.
Lemma BZ_of_Z_sign x: BZ_of_Z (Zsgn x) = BZsign (BZ_of_Z x).
Lemma BZ_of_Z_prod x y: BZ_of_Z (x * y ) = (BZ_of_Z x) *z (BZ_of_Z y).
Lemma BZ_of_Z_le (x y: int):  x <= y  <-> ( (BZ_of_Z x) <=z (BZ_of_Z y)).
Lemma BZ_of_Z_lt (x y: int): x < y  <-> ( (BZ_of_Z x) <z (BZ_of_Z y)).
```

Let's notice that division on $\mathbb{Z}$ is defined in a weird place, so we shall not use it. In the lemmas that follow, if a has type `nat` it is promoted to `Z` inside `BZ_of_Z`; if this gives A we have: a divides b (in `nat`) is the same as A divides B (in **Z**). If g is the gcd of a and b, then G is the gcd of A and B; if a and b are coprime, then A and B are coprime.

```
Require Import div.

Lemma BZ_of_div (a b:nat): a %| b <-> (BZ_of_Z a) %|z (BZ_of_Z b).
Lemma BZ_of_gcd (n m: nat) :
  BZgcd (BZ_of_Z n) (BZ_of_Z m) = BZ_of_Z (gcdn n m).
Lemma BZ_of_coprime (n m: nat) :
  BZcoprime (BZ_of_Z n) (BZ_of_Z m) <-> (coprime n m).
End Conversions.
```

Chapter 9

Rational numbers

The set **Q** of rational numbers is the quotient field of **Z**; as such, it is the quotient of some equivalence relation. An element x of a class X has the form (a, b), where a and b are in **Z** and b is non-zero. Since $(-a, -b) \in \mathrm{X}$, we may assume $b > 0$. Since $(a/g, b/g) \in \mathrm{X}$ where $g = \gcd(a, b)$, we may assume a and b coprime. Now, there is only one element in each class satisfying these properties, this leads to a direct definition.

9.1 Definition

We introduce the set **Q** of all pairs of rational integers that are coprime and such that the second component (the *denominator*) is strictly positive. The first element of the pair is called the *numerator*. We shall write a/b instead of (a, b), see justification below.

We consider the subsets $\mathbf{Q}_+$, $\mathbf{Q}_-$, $\mathbf{Q}_+^*$ and $\mathbf{Q}_-^*$ formed of numbers with numerator in $\mathbf{Z}_+$, $\mathbf{Z}_-$, $\mathbf{Z}_+^*$ and $\mathbf{Z}_-^*$.

```
Notation Qnum := P (only parsing).
Notation Qden := Q (only parsing).
Definition BQ := Zo (BZ \times BZps) (fun z => BZcoprime (Qnum z) (Qden z)).
Definition BQms:= Zo BQ (fun z => inc (Qnum z) BZms).
Definition BQps:= Zo BQ (fun z => inc (Qnum z) BZps).
Definition BQp:= Zo BQ (fun z => inc (Qnum z) BZp).
Definition BQm:= Zo BQ (fun z => inc (Qnum z) BZm).

Definition ratp x := inc x BQ.

Lemma BQ_P q: ratp q <->
  [/\ pairp q, intp (Qnum q), inc (Qden q) BZps
   & BZcoprime (Qnum q) (Qden q)].
```

We have some inclusions.

```
Lemma BQP_sBQ : sub BQp BQ.
Lemma BQps_sBQ : sub BQps BQ.
Lemma BQms_sBQ : sub BQms BQ.
Lemma BQm_sBQ : sub BQm BQ.
Lemma BQps_sBQp : sub BQps BQp.
Lemma BQms_sBQm : sub BQms BQm.
```

9.1.1 Basic properties

Consider a pair $x = (a,b) \in \mathbf{Z} \times \mathbf{Z}_+^*$. Let p be the gcd of a and b, $a' = a/p$, $b' = b/p$. By construction, the two numbers a' and b' are coprime, so that (a',b') is in $\mathbf{Q}$. We shall write $i_{\mathrm{Q}}(x) = i_{\mathrm{Q}}(a,b) = (a',b')$. In the SSREFLECT library, this function is defined for all b, if $b < 0$, then $i_{\mathrm{Q}}(a,b) = i_{\mathrm{Q}}(-a,-b)$; as an exception $i_{\mathrm{Q}}(a,0) = i_{\mathrm{Q}}(1,1)$. Consider a second pair $y = (c,d)$. Let's say that x and y are *equivalent* if $ad = bc$, and denote it by $x \equiv y$. If q is the gcd of c and d, since p and q are non-zero, $ad = bc$ is equivalent to $a'd' = b'c'$. If this relation holds, the Bezout relation between a' and b' implies that b' divides d', similarly d' divides b'. Since "x divides y" is an order on $\mathbf{Z}_+^*$, it follows $b' = d'$, thus $a' = c'$. Thus i_{Q} satisfies

$$(*) \qquad i_{\mathrm{Q}}(x) = i_{\mathrm{Q}}(y) \iff x \equiv y.$$

```
Definition BQ_of_pair a b :=
   J (a %/z (BZgcd a b)) (b %/z (BZgcd a b)).

Lemma BQ_of_pair_prop1 a b
   (a' := a %/z (BZgcd a b)) (b' := b %/z (BZgcd a b)):
   intp a -> inc b BZps ->
  [/\ inc a' BZ, inc b' BZps, BZcoprime a' b' & BZBezout a' b'].
Lemma BQ_of_pair_prop2 a b c d:
  intp a -> inc b BZps -> intp c -> inc d BZps ->
  BZBezout a b ->  BZBezout c d ->
  a *z d = b *z c -> a = c /\ b = d.
Lemma BQ_of_pair_prop3 a b c d
   (a' := a %/z (BZgcd a b)) (b' := b %/z (BZgcd a b))
   (c' := c %/z (BZgcd c d)) (d' := d %/z (BZgcd c d)):
   intp a -> inc b BZps -> intp c -> inc d BZps ->
   (a *z d = b *z c <-> (a' = c' /\ b' = d')).
Lemma BQ_of_pair_prop4 a b:
  intp a -> inc b BZps -> ratp (BQ_of_pair a b).
Lemma BQ_of_pair_pos a b:
   inc a BZp -> inc b BZps -> inc (BQ_of_pair a b) BQp.
Lemma BQ_of_pair_prop5 a b c d:
  intp a -> inc b BZps -> intp c -> inc d BZps ->
   (a *z d = b *z c <-> BQ_of_pair a b = BQ_of_pair c d).
```

If $x = a/b$ is in $\mathbf{Q}$ then $i_{\mathrm{Q}}(a,b) = x$. More generally, if a and b are coprime, $i_{\mathrm{Q}}(a,b)$ is the pair (a,b). For instance, $a \mapsto i_{\mathrm{Q}}(a,1)$ is the natural injection $\mathbf{Z} \to \mathbf{Q}$. We also introduce $b \mapsto i_{\mathrm{Q}}(1,b)$. This allow us to define one-half, and, later on, to show that the set numbers x such that $0 < x < 1$ is infinite countable. For the moment being, we prove that $\mathbf{Q}$ is infinite countable.

```
Definition BQ_of_Z x := (J x \1z).
Definition BQ_of_Zinv x := (J \1z x).

Lemma BQ_pr2 q: ratp q -> BQ_of_pair (Qnum q) (Qden q) = q.
Lemma BQ_of_Z_pr0 z: intp z -> BQ_of_Z z = BQ_of_pair z \1z.
Lemma BQ_of_Z_iQ z: intp z -> ratp (BQ_of_Z z).
Lemma BQ_of_Z_iQps z: inc z BZps -> inc (BQ_of_Z z) BQps.
Lemma BQ_of_Z_iQp z: inc z BZp -> inc (BQ_of_Z z) BQp.
Lemma BQ_of_Z_iQm z: inc z BZm -> inc (BQ_of_Z z) BQm.
Lemma BQ_of_Z_iQms z: inc z BZms -> inc (BQ_of_Z z) BQms.
Lemma BQ_of_Zi_pr0 z: intp z -> BQ_of_Zinv z = BQ_of_pair \1z z.
```

```
Lemma BQ_of_Zi_iQps z: inc z BZps -> inc (BQ_of_Zinv z) BQps.
Lemma BQ_of_Zi_iQ z: inc z BZps -> rqtp (BQ_of_Zinv z).
Lemma cardinal_Q : cardinal BQ = aleph0.
Lemma cardinal_Qps : cardinal BQps = aleph0.
```

We consider here some constants: 0_q, 1_q, 2_q, -1_q, of the form $i_{\mathrm{Q}}(t,1)$, and $1/2 = i_{\mathrm{Q}}(2,1)$ called one-half.

```
Definition BQ_zero := BQ_of_Z \0z.
Definition BQ_one := BQ_of_Z \1z.
Definition BQ_two := BQ_of_Z \2z.
Definition BQ_four := BQ_of_Z \4z.
Definition BQ_mone := BQ_of_Z \1mz.
Definition BQ_half := BQ_of_Zinv \2z.

Notation "\0q" := BQ_zero.
Notation "\1q" := BQ_one.
Notation "\2q" := BQ_two.
Notation "\4q" := BQ_four.
Notation "\1mq" := BQ_mone.
Notation "\2hq" := BQ_half.
```

We show that some quantities are in some sets.

```
Lemma QS0 : ratp \0q.
Lemma QpS0 : inc \0q BQp.
Lemma QmS0 : inc \0q BQm.
Lemma QpsS1 : inc \1q BQps.
Lemma QS1 : ratp \1q.
Lemma QpsS2 : inc \2q BQps.
Lemma QS2 : ratp \2q.
Lemma QmsSm1 : inc \1mq BQms.
Lemma QSm1 : ratp \1mq.
Lemma QpsSh2 : inc \2hq BQps.
Lemma QSh2 : ratp \2hq.
Lemma QpsS4 : inc \4q BQps.
Lemma QS4 : ratp \4q.
```

We show here some simple properties, as: x is in exactly one of $\mathbf{Q}_+$ and $\mathbf{Q}_-^*$, or : if $x \in \mathbf{Q}_+^*$, then $x \neq 0$, or x is zero if and only if its numerator is zero.

```
Lemma BQnum0 : Qnum \0q = \0z.
Lemma BQden0 : Qden \0q = \1z.
Lemma BQnum0_P q : ratp q -> Qnum q = \0z -> q = \0q.
Lemma BQnum0_P1 x: inc x BZps -> (BQ_of_pair \0z x) = \0q.
Lemma BQ1_nz: \1q <> \0q.
Lemma BQ2_nz :  \2q <> \0q.
Lemma BQms_nz  x: inc x BQms -> x <> \0q.
Lemma BQps_nz  x: inc x BQps -> x <> \0q.

Lemma BQ_i0P x: ratp x <-> (inc x  BQms \/ inc x BQp).
Lemma BQ_i1P x: ratp x  <-> [\/ x = \0q, inc x BQps | inc x BQms].
Lemma BQ_i2P x: ratp x <-> (inc x  BQps \/ inc x BQm).
Lemma BQ_di_neg_pos x: inc x BQms -> inc x BQp -> False.
Lemma BQ_di_pos_neg x: inc x BQps -> inc x BQm -> False.
```

```
Lemma BQ_di_neg_spos x: inc x BQms -> inc x BQps -> False.
Lemma BQps_iP x: inc x BQps <-> inc x BQp /\ x <> \0q.
Lemma BQms_iP x: inc x BQms <-> inc x BQm /\ x <> \0q.
```

9.1.2 Opposite

We define the *opposite* of a/b as $(-a)/b$ and denote it by $-(a/b)$. All properties shown here are trivial (we just restate the properties of the opposite in **Z**).

```
Definition BQopp x:= J (BZopp (Qnum x)) (Qden x).

Lemma QSo x: ratp x -> ratp (BQopp x).
Lemma BQopp_0 : BQopp \0q = \0q.
Lemma BQopp0_bis x: ratp x ->  (x = \0q <-> BQopp x = \0q).
Lemma BQopp_num x: Qnum (BQopp x) = BZopp (Qnum x).
Lemma BQopp_den x: Qden (BQopp x) = Qden x.

Lemma BQopp_positive1 x: inc x BQps -> inc (BQopp x) BQms.
Lemma BQopp_positive2 x: inc x BQp -> inc (BQopp x) BQm.
Lemma BQopp_negative1 x: inc x BQms -> inc (BQopp x) BQps.
Lemma BQopp_negative2 x: inc x BQm -> inc (BQopp x) BQp.

Lemma BQopp_K x: ratp x -> BQopp (BQopp x) = x.
Lemma BQ_of_pair_opp a b:intp a -> inc b BZps ->
   (BQ_of_pair (BZopp a) b)  = BQopp (BQ_of_pair a b).
Lemma BQ_of_pair_zero a b: intp a -> inc b BZps ->
   (BQ_of_pair a b) = \0q -> a = \0z.
Lemma BQ_of_pair_spos a b: inc a BZps -> inc b BZps ->
   inc (BQ_of_pair a b) BQps.
Lemma BQ_of_pair_sneg a b: inc a BZms -> inc b BZps ->
   inc (BQ_of_pair a b) BQms.
Lemma BQ_of_pair_neg a b: inc a BZm -> inc b BZps -> inc (BQ_of_pair a b) BQm.
Lemma BQopp_inj a b: ratp a -> ratp b -> BQopp a = BQopp b -> a = b.
Lemma BQopp_fb: bijection (Lf BQopp BQ BQ).
Lemma BQopp_perm: inc (Lf BQopp BQ BQ) (permutations BQ).
Lemma BQopp_cZ x: BQopp (BQ_of_Z x) = BQ_of_Z (BZopp x).
Lemma BQopp_m1: BQopp \1mq = \1q.
Lemma BQopp_1: BQopp \1q = \1mq.
```

9.1.3 Absolute value

The *absolute value* of a rational number a/b is $|a|/b$, denoted $|a/b|$. If x is positive then $|x| = x$, otherwise $|x| = -x$. All properties shown here are trivial.

```
Definition BQabs x:= J (BZabs (Qnum x)) (Qden x).

Lemma BQabs_num x: Qnum (BQabs x) = BZabs (Qnum x).
Lemma BQabs_den x: Qden (BQabs x) = Qden x.
Lemma BQabs_abs x: BQabs (BQabs x) = BQabs x.
Lemma BQabs_opp x: BQabs (BQopp x) = BQabs x.

Lemma BQabs_pos x: inc x BQp -> BQabs x = x.
Lemma BQabs_negs x: inc x BQms -> BQabs x = BQopp x.
Lemma BQabs_neg x: inc x BQm -> BQabs x = BQopp x.
```

```
Lemma QpSa x: ratp x -> inc (BQabs x) BQp.
Lemma QSa x: ratp x -> ratp (BQabs x).
Lemma BQabs_0 : BQabs \0q = \0q.
Lemma BQabs_m1: BQabs \1mq = \1q.
Lemma BQabs_1: BQabs \1q = \1q.
Lemma BQabs0_bis x: ratp x ->  (x = \0q <-> BQabs x = \0q).
```

9.1.4 Order

Let $x \le_q y$ be the relation "$x \in \mathbf{Z}\times\mathbf{Z}^*_+$ is the pair (a,b), $y \in \mathbf{Z}\times\mathbf{Z}^*_+$ is the pair (c,d) and $ad \le_{\mathbf{Z}} bc$", and G be the graph of this relation on $\mathbf{Q}$. Let $x \le_{\mathbf{Q}} y$ be the relation "$x \in \mathbf{Q}$, $y \in \mathbf{Q}$ and $x \le_q y$". By definition, this relation is equivalent to $(x, y) \in$ G.

Note that $\le_q$ is a preorder relation compatible with the equivalence $x \equiv y$ introduced above, and that $x \le_q y \wedge y \le_q x$ is equivalent to $x \equiv y$. We simplify a bit the definition of $x \le_q y$, by removing the assumption that x and y are pairs of rational integers; this does not change the relation $x \le_{\mathbf{Q}} y$.

```
Definition BQle_aux x y := (Qnum x) *z (Qden y) <=z  (Qden x) *z (Qnum y).
Definition BQ_le x y := [/\ ratp x, ratp y & BQle_aux x y].
Definition BQ_lt x y :=  BQ_le x y /\ x <> y.
Definition BQ_ordering := graph_on BQle_aux BQ.

Notation "x <=q y" := (BQ_le x y) (at level 60).
Notation "x <q y" := (BQ_lt x y) (at level 60).
```

The relation $x \le_{\mathbf{Q}} y$ is a total ordering on $\mathbf{Q}$. It will be written $x \le y$ if no confusion arises.

```
Lemma qleR_aux a: ratp a -> BQle_aux a a.
Lemma BQle_P x y: gle BQ_ordering  x y <-> x <=q y.
Lemma BQlt_P x y: glt BQ_ordering  x y <-> x <q y.
Lemma qleR a: ratp a -> a <=q a.
Lemma qleA x y:  x <=q y -> y <=q x -> x = y.
Lemma qleT y x z:  x <=q y -> y <=q z -> x <=q z.
Lemma BQor_or: order BQ_ordering.
Lemma BQor_sr: substrate BQ_ordering = BQ.
Lemma BQor_tor: total_order BQ_ordering.
Lemma BQ_le_pair a b c d:
  intp a -> inc b BZps -> intp c -> inc d BZps ->
  ((BQ_of_pair a b) <=q (BQ_of_pair c d) <-> (a *z d) <=z (b *z c)).
```

We express here that $\le$ is transitive and total.

```
Lemma qleNgt a b: a <=q b -> ~ (b <q a).
Lemma qlt_leT b a c: a <q b -> b <=q c -> a <q c.
Lemma qle_ltT b a c: a <=q b -> b <q c -> a <q c.
Lemma qlt_ltT b a c: a <q b -> b <q c -> a <q c.
Lemma qleT_ee a b:  ratp a -> ratp b -> a <=q b  \/ b <=q a.
Lemma qleT_ell a b: ratp a -> ratp b -> [\/ a = b, a <q b | b <q a].
Lemma qleT_el a b: ratp a -> ratp b -> a <=q b  \/ b <q a.
Lemma qltP x y: x <q y <->  [/\ ratp x, ratp y &
  (Qnum x) *z (Qden y) <z  (Qden x) *z (Qnum y) ].
```

If $x \in \mathbf{Q}^*_-$ and $y \in \mathbf{Q}^*_+$, then $x < 0 < y$. We have $a < b$ if and only if $-b < -a$.

```
Lemma qge0xP x:  x <=q \0q <-> inc x BQm.
Lemma qgt0xP x:  x <q \0q <-> inc x BQms.
Lemma qle0xP x: \0q <=q x <-> inc x BQp.
Lemma qlt0xP x:  \0q <q x <-> inc x BQps.
Lemma qle_par1 x y: inc x BQps -> inc y BQm ->  y <q x.
Lemma qle_par2 x y: inc x BQp -> inc y BQms ->  y <q x.
Lemma qle_par3 x y: inc x BQp -> inc y BQm ->  y <=q x.
Lemma qle_cZ a b: intp a -> intp b ->
    (a <=z b <-> BQ_of_Z a <=q BQ_of_Z b).
Lemma qlt_cZ a b: intp a -> intp b ->
    (a <z b <-> BQ_of_Z a <q BQ_of_Z b).
Lemma qlt_12:  \1q <q \2q.
Lemma qlt_24: \2q <q \4q.
Lemma qle_24: \2q <=q \4q.
Lemma BQabs_positive b: ratp b -> b <> \0q -> \0q <q (BQabs b).
Lemma qle_opp x y: x <=q y ->  (BQopp y) <=q  (BQopp x).
Lemma qlt_opp x y: x <q y -> (BQopp y) <q (BQopp x).
Lemma qle_oppP x y: ratp x -> ratp y ->
  ((BQopp y) <=q  (BQopp x) <-> x <=q y).
Lemma qlt_oppP x y: ratp x -> ratp y ->
  ((BQopp y) <q  (BQopp x) <-> x <q y).
Lemma BQabs_prop1 x y: ratp x -> ratp y ->
  (BQabs x <=q y <->  (BQopp y <=q x /\ x <=q y)).
Lemma BQabs_prop2 x y: ratp x -> ratp y ->
  (BQabs x <q y <->  (BQopp y <q x /\ x <q y)).
Lemma qle_opp_iso:
  order_isomorphism (Lf BQopp BQ BQ) BQ_ordering (opp_order BQ_ordering).
```

9.2 Field operations

We show here that $\mathbf{Q}$ is a commutative field.

9.2.1 Addition

If $x = a/b$ and $y = c/d$, we define $x + y$ by $i_Q(ad + bc, bd)$.

```
Definition BQsum x y:=
  BQ_of_pair ((Qnum x) *z (Qden y) +z (Qden x) *z (Qnum y))
    ((Qden x) *z  (Qden y)).

Notation "x +q y" := (BQsum x y) (at level 50).
```

Commutativity is trivial; associativity follows from: if $x = a/b$, $y = a'/b'$ and $z = a''/b''$ then $x + y + z = i_Q(ab'b'' + a'bb'' + a''bb', bb'b'')$.

```
Lemma QSs x y: ratp x -> ratp y -> ratp (x +q y).
Lemma BQsumC x y:  x +q y = y +q x.
Lemma BQsumA x y z: ratp x -> ratp y -> ratp z ->
  x  +q (y +q z) = (x +q y) +q z.
Lemma BQsum_AC x y z: ratp x -> ratp y -> ratp z ->
  (x +q y) +q z = (x +q z) +q y.
```

```
Lemma BQsum_CA x y z: ratp x -> ratp y -> ratp z ->
  x +q (y +q z) = y +q (x +q z).
Lemma BQsum_ACA a b c d: ratp a -> ratp b -> ratp c -> ratp d ->
    (a +q b) +q (c +q d) = (a +q c) +q (b +q d).
```

Note that $a/b+c/b$ simplifies to $i_{\mathbf{Q}}(a+c,b)$, so that addition on **Q** is compatible with that on **Z**.

```
Lemma BQsum_same_den x y: ratp x -> ratp y -> Qden x = Qden y ->
  x +q y = BQ_of_pair (Qnum x +z Qnum y) (Qden x).
Lemma BQsum_cZ x y: intp x -> intp y ->
  BQ_of_Z x +q  BQ_of_Z y = BQ_of_Z (x +z y).
Lemma BQsum_0l x: ratp x -> \0q +q x  = x.
Lemma BQsum_0r x: ratp x -> x +q \0q = x.
Lemma BQsum_11 : \1q +q \1q = \2q.
Lemma BQsum_opp_r x: ratp x ->  x +q (BQopp x) = \0q.
Lemma BQsum_opp_l x: ratp x -> (BQopp x) +q x = \0q.
Lemma BQsum_opp_rev a b: ratp a -> ratp b -> a +q b = \0q ->
   a = BQopp b.
Lemma BQoppD x y: ratp x -> ratp y ->
  BQopp (x +q y) = (BQopp x) +q (BQopp y).
```

Some sets are stable by addition.

```
Lemma QpS_sum x y: inc x BQp -> inc y BQp -> inc (x +q y) BQp.
Lemma QpsS_sum_r x y: inc x BQp -> inc y BQps -> inc (x +q y) BQps.
Lemma QpsS_sum_l x y: inc x BQps -> inc y BQp ->  inc (x +q y) BQps.
Lemma QpsS_sum_rl x y: inc x BQps -> inc y BQps -> inc (x +q y) BQps.
Lemma QmsS_sum_rl x y: inc x BQms -> inc y BQms -> inc (x +q y) BQms.
Lemma QmsS_sum_r x y: inc x BQm -> inc y BQms ->  inc (x +q y) BQms.
Lemma QmsS_sum_l x y: inc x BQms -> inc y BQm -> inc (x +q y) BQms.
Lemma QmS_sum x y: inc x BQm -> inc y BQm -> inc (x +q y) BQm.
```

9.2.2 Subtraction

We introduce here subtraction: $a-b$ is $a+(-b)$.

```
Definition BQdiff x y := x +q (BQopp y).
Notation "x -q y" := (BQdiff x y) (at level 50).
Lemma QS_diff x y: ratp x -> ratp y -> ratp (x -q y).
```

Some properties.

```
Section BQdiffProps.
Variables (x y z: Set).
Hypotheses (xq: ratp x)(yq: ratp y)(zq: ratp z).

Lemma BQdiff_sum:  (x +q y) -q x = y.
Lemma BQsum_diff: x +q (y -q x) = y.
Lemma BQdiff_sum1:  (y +q x) -q x = y.
Lemma BQsum_diff1: (y -q x) +q x = y.
Lemma BQdiff_xx :  x -q x = \0q.
Lemma BQdiff_0r: x -q \0q = x.
Lemma BQdiff_0l: \0q -q x = BQopp x.
```

```
Lemma BQdiff_sum_simpl_l:  (x +q y) -q  (x +q z) = y -q z.
Lemma BQdiff_sum_comm: (x +q y) -q z = (x -q z) +q y.
Lemma BQoppB: BQopp (x -q y) = y -q x.
End BQdiffProps.
```

More properties, including regularity of addition.

```
Section BQdiffProps2.
Variables (x y z: Set).
Hypotheses (xq: ratp x)(yq: ratp y)(zq: ratp z).

Lemma BQsum_diff_ea: x = y +q z -> z = x -q y.
Lemma BQdiff_xx_rw:  x -q y = \0q -> x = y.
Lemma BQdiff_sum_simpl_r:  (x +q z) -q (y +q z) = x -q y.
Lemma BQsum_eq2l:  x +q y = x +q z -> y = z.
Lemma BQsum_eq2r:  x +q z = y +q z -> x = y.
End BQdiffProps2.

Lemma BQdiff_diff a b c: ratp a -> ratp b -> ratp c ->
   a -q (b -q c)  = (a -q b) +q c.
Lemma BQdiff_diff_simp a b: ratp a -> ratp b ->  a -q (a -q b) = b.
Lemma BQdiff_le1 a b c: ratp a -> ratp b -> ratp c ->
  (a -q b <=q c <-> a -q c <=q  b).
Lemma BQdiff_diff2 a b c: ratp a -> ratp b -> ratp c ->
   a -q (b +q c)  = (a -q b) -q c.
Lemma BQdiff_cZ x y: intp x -> intp y ->
  BQ_of_Z x -q  BQ_of_Z y = BQ_of_Z (x -z y).
```

9.2.3 Multiplication

We define the product of a/b and c/d as $i_Q(ac, bd)$. The product of x and y is denoted $x \cdot y$, $x.y$ or xy, as usual.

```
Definition BQprod x y :=
  BQ_of_pair ((Qnum x) *z (Qnum y)) ((Qden x) *z  (Qden y)).

Notation "x *q y" := (BQprod x y) (at level 40).
```

Associativity of multiplication follows from: if $x = a/b$, $y = a'/b'$ and $z = a''/b''$ then $xyz = i_Q(aa'a'', bb'b'')$. The proof of $x(y+z) = xy + xz$ is similar.

```
Lemma QSp x y: ratp x -> ratp y -> ratp (x *q y).
Lemma BQprodC x y: x *q y = y *q x.
Lemma BQprodA x y z: ratp x -> ratp y -> ratp z ->
  x  *q (y *q z) = (x *q y) *q z.
Lemma BQprod_AC x y z: ratp x -> ratp y -> ratp z ->
  (x *q y) *q z = (x *q z) *q y.
Lemma BQprod_CA x y z: ratp x -> ratp y -> ratp z ->
  x *q (y *q z) = y *q (x *q z).
Lemma BQprod_ACA a b c d: ratp a -> ratp b -> ratp c -> ratp d ->
    (a *q b) *q (c *q d) = (a *q c) *q (b *q d).
Lemma BQprodDr x y z: ratp x -> ratp y -> ratp z  ->  (* 63 *)
   x *q (y +q z) = (x *q y) +q (x *q z).
Lemma BQprodDl x y z: ratp x -> ratp y -> ratp z  ->
```

```
   (y +q z) *q x = (y *q x) +q (z *q x).
 Lemma BQprod_cZ x y: intp x -> intp y ->
  BQ_of_Z x *q  BQ_of_Z y = BQ_of_Z (x *z y).
```

We have $1 \cdot x = x$ and $-1 \cdot x = -x$. It follows $-(x \cdot y) = (-x) \cdot y$.

```
Lemma BQprod_22: \2q *q \2q = \4q.
Lemma BQprod_0r x: ratp x -> x *q \0q = \0q.
Lemma BQprod_0l x: ratp x ->  \0q *q x = \0q.
Lemma BQprod_1l x: ratp x ->  \1q *q x = x.
Lemma BQprod_1r x: ratp x ->  x *q \1q  = x.
Lemma BQprod_m1r x: ratp x ->  x *q \1mq = BQopp x.
Lemma BQprod_m1l x: ratp x -> \1mq *q x  = BQopp x.

Lemma BQopp_prod_r x y: ratp x -> ratp y ->
  BQopp (x *q y) = x *q (BQopp y).
Lemma BQopp_prod_l x y: ratp x -> ratp y ->
  BQopp (x *q y) = (BQopp x) *q y.
Lemma BQprod_opp_comm x y: ratp x -> ratp y ->
  x *q  (BQopp y) =  (BQopp x) *q y.
Lemma BQprod_opp_opp x y: ratp x -> ratp y ->
  (BQopp x) *q (BQopp y) = x  *q y.
```

Some sets are invariant by multiplication.

```
Lemma QpsS_prod a b: inc a BQps -> inc b BQps -> inc (a *q b) BQps.
Lemma QmsuS_prod a b: inc a BQms -> inc b BQms -> inc (a *q b) BQps.
Lemma QpmsS_prod a b: inc a BQps -> inc b BQms -> inc (a *q b) BQms.
Lemma QpS_prod a b: inc a BQp -> inc b BQp -> inc (a *q b) BQp.
Lemma QmuS_prod a b: inc a BQm -> inc b BQm -> inc (a *q b) BQp.
Lemma QpmS_prod a b: inc a BQp -> inc b BQm -> inc (a *q b) BQm.

Lemma BQps_stable_prod1 a b: ratp a -> ratp b -> inc (a *q b) BQps ->
  ((inc a BQps <-> inc b BQps) /\ (inc a BQms <-> inc b BQms)).
Lemma BQprod_nz x y: ratp x -> ratp y ->
  x <> \0q -> y <> \0q -> x *q y <> \0q.
Lemma BQprod_abs x y: ratp x -> ratp y ->
  BQabs (x *q y) = (BQabs x) *q (BQabs y).
Lemma BQprodBr x y z: ratp x -> ratp y -> ratp z  ->
   x *q (y -q z) = (x *q y) -q (x *q z).
Lemma BQprodBl x y z: ratp x -> ratp y -> ratp z  ->
  (y -q z) *q x =  (y *q x) -q  (z *q x).
```

We show some properties of the injection $\mathbf{N} \to \mathbf{Q}$.

```
Definition BQ_of_nat n := BQ_of_Z (BZ_of_nat n).

Lemma QpsS_of_nat n: natp n -> n <> \0c -> inc (BQ_of_nat n) BQps.
Lemma QpS_of_nat n: natp n -> inc (BQ_of_nat n) BQp.
Lemma QS_of_nat n: natp n -> inc (BQ_of_nat n) BQ.
Lemma BQsum_cN n m: natp n -> natp m ->
  BQ_of_nat n +q  BQ_of_nat m = BQ_of_nat (n +c m).
Lemma BQ_of_nat_succ n: natp n ->  BQ_of_nat n +q \1q =  BQ_of_nat (csucc n).
Lemma qle_cN a b: natp a -> natp b ->
    (a <=c b <-> BQ_of_nat a <=q BQ_of_nat b).
```

```
Lemma qlt_cN qlt_cN qlt_cN a b: natp a -> natp b ->
    (a <c b <-> BQ_of_nat a <q BQ_of_nat b).
Lemma BQ_of_nat_injective: injective  BQ_of_nat.
```

Write x^2 instead of $x \cdot x$. If $x = a/b$ this is a^2/b^2. In particular, if $x^2 \in \mathbf{Z}$, we have $b = 1$ so that $x \in \mathbf{Z}$. Thus $x^2 = 1$ is equivalent to $x = \pm 1$ and $x^2 = 2$ has no solution.

```
Definition BQsquare x := x *q x.

Lemma BQpS_square x: ratp x -> inc (BQsquare x) BQp.
Lemma BQ_squarep x: ratp x ->
   BQsquare x = J (Qnum x *z Qnum x) (Qden x *z Qden x).
Lemma BQ_square_Z x y: ratp x -> BQsquare x = BQ_of_Z y ->
   [/\ inc y BZp, Qnum x *z Qnum x = y & Qden x = \1z].
Lemma BQ_square_1 x: ratp x ->
   (BQsquare x = \1q <-> (x = \1q \/ x = \1mq) ).
Lemma BQ_square_2 x : ratp x -> BQsquare x = \2q -> False.
Lemma BQprod_cN a b: natp a -> natp b ->
  BQ_of_nat (a *c b) =  BQ_of_nat a *q (BQ_of_nat b).
Lemma BQ_nat_square_monotone n (x := BQ_of_nat n) : natp n ->
  x <=q BQsquare x.
```

We have

$$(a+b)^2 = a^2 + b^2 + 2ab = (a-b)^2 + 4ab.$$

```
Definition BQdouble x := \2q *q x.

Lemma BQdouble_p x : ratp x -> x +q x = BQdouble x.
Lemma QSdouble x : ratp x -> inc (BQdouble x) BQ.
Lemma BQsum_square a b: ratp a -> ratp b ->
   BQsquare (a +q b) = BQsquare a +q BQsquare b +q (BQdouble (a *q b)).
Lemma BQdiff_square a b: ratp a -> ratp b ->
   BQsquare (a -q b) = BQsquare a +q BQsquare b -q (BQdouble (a *q b)).
Lemma BQsumdiff_square a b: ratp a -> ratp b ->
   BQsquare (a +q b) = \4q *q (a *q b) +q BQsquare (a -q b).
Lemma BQdouble_opp x: ratp x -> BQdouble (BQopp x) = BQopp (BQdouble x).
Lemma BQdoubleD x y: ratp x -> ratp y ->
   BQdouble (x +q y) = (BQdouble x) +q (BQdouble y).
```

9.2.4 Inverse

The *inverse* of x is denoted x^{-1}; if $x = a/b$ and is positive, then $x^{-1} = b/a$, if x is negative, then $x^{-1} = -(-x)^{-1}$. For simplicity, the inverse of zero will be zero.

```
Definition BQinv x :=
  Yo (inc (Qnum x) BZps) (J (Qden x) (Qnum x))
    (Yo (Qnum x = \0z) \0q (J (BZopp (Qden x)) (BZopp (Qnum x)))).

Lemma BQinv_0: BQinv \0q = \0q.
Lemma BQinv_pos x: inc x BQps -> BQinv x = (J (Qden x) (Qnum x)).
Lemma BQinv_neg x: inc x BQms ->
   BQinv x = J (BZopp (Qden x)) (BZopp (Qnum x)).
Lemma BQinv_opp x: ratp x -> BQinv (BQopp x) = BQopp (BQinv x).
Lemma QpsS_inv x: inc x BQps -> inc (BQinv x) BQps.
```

```
Lemma QmsS_inv x: inc x BQms -> inc (BQinv x) BQms.
Lemma QS_inv x: ratp x -> ratp (BQinv x).
Lemma BQinv_K x: ratp x -> BQinv (BQinv x)  = x.
Lemma BQinv_inj x y: ratp x -> ratp y -> BQinv x = BQinv y -> x = y.
```

Some properties.

```
Lemma BQinv_Z x: inc x BZps -> BQinv (BQ_of_Z x)  = BQ_of_Zinv x.
Lemma BQinv_1: BQinv \1q = \1q.
Lemma BQinv_m1: BQinv \1mq = \1mq.
Lemma BQinv_2: BQinv \2q = \2hq.
```

More properties. Note that $xy = 1$ says that y is the inverse of x.

```
Lemma BQinv_abs x: ratp x -> BQabs (BQinv x) = BQinv (BQabs x).
Lemma BQprod_inv1 x : ratp x -> x <> \0q -> (x *q  (BQinv x)) = \1q.
Lemma BQ_inv_prop a b: ratp a -> ratp b -> a *q b = \1q -> b = BQinv a.
Lemma BQprod_inv x y:ratp x -> ratp y ->
   BQinv (x *q y) = BQinv x *q BQinv y.
```

9.2.5 Division

We define the quotient of x and y by $x \cdot y^{-1}$, and we denote it by x/y. Note that $0/x = x/0 = 0$.

```
Definition BQdiv x y := x *q  (BQinv y).
Notation "x /q y" := (BQdiv x y) (at level 40).

Lemma BQdiv_0x x : ratp x -> \0q /q x = \0q.
Lemma BQdiv_x0 x : ratp x -> x /q \0q = \0q.
Lemma BQdiv_1x x : ratp x -> \1q /q x = BQinv x.
Lemma BQdiv_x1 x : ratp x -> x /q \1q = x.

Lemma QS_div x y: ratp x -> ratp y -> ratp (x /q y).
Lemma QpsS_div a b: inc a BQps -> inc b BQps -> inc (a /q b) BQps.
Lemma QmsuS_div a b: inc a BQms -> inc b BQms -> inc (a /q b) BQps.
Lemma QpmsS_div a b: inc a BQps -> inc b BQms -> inc (a /q b) BQms.
Lemma QmpsS_div a b: inc a BQms -> inc b BQps -> inc (a /q b) BQms.
Lemma QpS_div a b: inc a BQp -> inc b BQp -> inc (a /q b) BQp.
Lemma QmuS_div a b: inc a BQm -> inc b BQm -> inc (a /q b) BQp.
Lemma QpmS_div a b: inc a BQp -> inc b BQm -> inc (a /q b) BQm.
Lemma QmpS_div a b: inc a BQm -> inc b BQp -> inc (a /q b) BQm.
```

Division, opposite and absolute value.

```
Lemma BQopp_div_r x y: ratp x -> ratp y ->
  BQopp (x /q y) = x /q (BQopp y).
Lemma BQopp_div_l x y: ratp x -> ratp y ->
  BQopp (x /q y) = (BQopp x) /q y.
Lemma BQdiv_opp_comm x y: ratp x -> ratp y ->
  x /q  (BQopp y) =  (BQopp x) /q y.
Lemma BQdiv_opp_opp x y: ratp x -> ratp y ->
  (BQopp x) /q (BQopp y) = x /q y.
Lemma BQdiv_abs x y: ratp x -> ratp y ->
  (BQabs x) /q (BQabs y) = BQabs (x /q y).
```

Note that division is compatible with the notation a/b introduced above. We have $x/x = 1$ (unless x is zero), so that $x = x^{-1}$ holds only if x is zero, or ± 1.

```
Lemma BQdiv_numden x: ratp x ->
  (BQ_of_Z (Qnum x)) /q (BQ_of_Z (Qden x)) = x.
Lemma BQdiv_numden1 a b (q := BQ_of_Z a /q  BQ_of_Z b):
   intp a -> inc b BZps -> BZcoprime a b ->
  (Qnum q = a /\ Qden q = b).
Lemma BQdiv_xx x : ratp x -> x <> \0q -> (x *q  (BQinv x)) = \1q.
Lemma BQ_self_inv x: ratp x ->
   (x = BQinv x <-> [\/ x= \0q, x = \1q | x = \1mq]).
Lemma BQdiv_square a b: ratp a -> ratp b ->
    BQsquare (a /q b) = (BQsquare a) /q (BQsquare b).
```

Some properties.

```
Section BQdiffProps3.
Variables (x y z: Set).
Hypotheses (xq: ratp x)(yq: ratp y)(zq: ratp z).

Lemma BQdiv_sumDl: (y +q z) /q x = (y /q x) +q (z /q x).
Lemma BQdiv_prod_simpl_l: x <> \0q ->  (x *q y) /q  (x *q z) = y /q z.
Lemma BQdiv_prod_comm: (x *q y) /q z = (x /q z) *q y.
Lemma BQinv_div: BQinv (x /q y) = y /q x.
Lemma BQdiv_prod: x <> \0q -> (x *q y) /q x = y.
Lemma BQprod_div: x <> \0q -> x *q (y /q x) = y.
End BZdiffProps3.
```

We deduce regularity of the product.

```
Section BQdiffProps4.
Variables (x y z: Set).
Hypotheses (xq: ratp x)(yq: ratp y)(zq: ratp z).

Lemma BQprod_div_ea: y <> \0q -> x = y *q z -> z = x /q y.
Lemma BQdiv_diag_rw:  x /q y = \1q -> x = y.
Lemma BQdiv_prod_simpl_r:  z <> \0q -> (x *q z) /q (y *q z) = x /q y.
Lemma BQprod_eq2r: z <> \0q ->  x *q z = y *q z -> x = y.
Lemma BQprod_eq2l:  x <> \0q -> x *q y = x *q z -> y = z.
End BZdiffProps4.
```

Note that $a + b/c = (ac + b)/c$.

```
Lemma BQdiv_div_simp a b c: ratp a -> ratp b -> ratp c -> b <> \0q ->
    (a /q b) /q (c /q b) = a /qc.
Lemma BQsum_div a b c: ratp a -> ratp b -> ratp c -> c <> \0q ->
   a +q (b /q c) = (a *q c +q b) /q c.
Lemma BQdiff_div a b c: ratp a -> ratp b -> ratp c -> c <> \0q ->
   a -q (b /q c) = (a *q c -q b) /q c.
Lemma BQdiv_div2 a b c:
   ratp a -> ratp b -> ratp c -> a /q (b *q c) = (a /q b) /q c.
```

9.2.6 Sign

We define the sign of x as the sign of its numerator. This is in **Z**. If s is the sign of x converted to **Q** we have $x = s|x|$ and $|x| = sx$.

```
Definition BQsign x:= BZsign (Qnum x).
Definition BQQsign x := BQ_of_Z (BQsign x).

Lemma QS_sign x: intp (BQsign x).
Lemma QS_qsign x: ratp (BQQsign x).
Lemma BQsign_trichotomy a:
  BQsign a = \1z \/ BQsign a = \1mz \/ BQsign a = \0z.
Lemma BQsign_pos x: inc x BQps -> BQsign x = \1z.
Lemma BQsign_neg x: inc x BQms -> BQsign x = \1mz.
Lemma BQsign_0: BQsign \0q = \0z.
Lemma BQopp_sign x: ratp x -> (BQsign (BQopp x)) = BZopp (BQsign x).
Lemma BQinv_sign x: ratp x -> BQsign (BQinv x) = BQsign x.
Lemma BQ_sign_prop x: ratp x ->
   [/\ inc x BQps <-> BQsign x = \1z,
       inc x BQms <-> BQsign x = \1mz &
       x = \0q <->  BQsign x = \0z].

Lemma BQabs_sign x: ratp x -> x = (BQQsign x) *q (BQabs x).
Lemma BQsign_abs x: ratp x -> x *q (BQQsign x) = BQabs x.
Lemma BQprod_sign x y: ratp x -> ratp y ->
  BQsign (x *q y) = (BQsign x) *z (BQsign y).
Lemma BQdiv_sign x y: ratp x -> ratp y ->
  BQsign (x /q y) = (BQsign x) *z (BQsign y).
```

9.2.7 Compatibility of order and operations

We first show that $x \le y$ if $y - x \ge 0$, then deduce the other formulas.

```
Lemma qle_diffP a b: ratp a -> ratp b ->
   (a <=q b <-> inc (b -q a) BQp).
Lemma qle_diffP1 a b: ratp a -> ratp b ->
  (a <=q b <-> \0q <=q  (b -q a)).
Lemma qlt_diffP a b: ratp a -> ratp b ->
  (a <q b <-> inc (b -q a) BQps).
Lemma qlt_diffP1 a b: ratp a -> ratp b ->
   (\0q  <q (b -q a) <-> a <q b)
Lemma qlt_diffP2 a b: ratp a -> ratp b ->
  (a <q b <-> inc (a -q b) BQms).
Lemma qlt_diffP3 a b c: ratp a -> ratp b -> ratp c ->
  (a <q b -q c <->  a +q c <q b).
Lemma qlt_diffP4 a b c: ratp a -> ratp b -> ratp c ->
  (b -q c <q a <-> b <q a +q c).
Lemma qgt_diffP  a b: ratp a -> ratp b -> (a -q b <q \0q <-> a <q b).
Lemma BQsum_le2l a b c: ratp a -> ratp b -> ratp c ->
  ((c +q a) <=q (c +q b) <-> a <=q b).
Lemma BQsum_le2r a b c: ratp a -> ratp b -> ratp c ->
   ((a +q c) <=q (b +q c) <-> a <=q b).
Lemma BQsum_lt2l a b c: ratp a -> ratp b -> ratp c ->
  (c +q a <q c +q b <-> a <q b).
Lemma BQsum_lt2r a b c: ratp a -> ratp b -> ratp c ->
  (a +q c <q b +q c <-> a <q b).
Lemma BQdiff_lt1P a b c: ratp a -> ratp b -> ratp c ->
  (a -q b <q c <-> a -q c <q  b).
Lemma BQdiff_lt2P a b c: ratp a -> ratp b -> ratp c ->
  (c <q a -q b <-> b <q a -q c).
Lemma BQdiff_le1P a b c: ratp a -> ratp b -> ratp c ->
```

```
  (a -q b <=q c <-> a -q c <=q  b).
Lemma BQdiff_le2P a b c: ratp a -> ratp b -> ratp c ->
  (c <=q a -q b <-> b <=q a -q c).
Lemma BQabs_prop3 x y e: ratp x -> ratp y -> ratp e ->
  (BQabs (x -q y) <=q e <->  y -q e <=q x /\ x <=q y +q e).
Lemma BQabs_prop4 x y e: ratp x -> ratp y -> ratp e ->
  (BQabs (x -q y) <q e <->  y -q e <q x /\ x <q y +q e).
```

More lemmas including comparison and sum.

```
Lemma BQsum_Mlele a b c d: a <=q c ->  b <=q d -> (a +q b) <=q (c +q d).
Lemma BQsum_Mlelt a b c d: a <=q c -> b <q d -> (a +q b) <q (c +q d).
Lemma BQsum_Mltle a b c d: a <q c -> b <=q d -> (a +q b) <q (c +q d).
Lemma BQsum_Mltlt a b c d: a <q c -> b <q d -> (a +q b) <q (c +q d).
Lemma BQsum_Mlege0 a c d: a <=q c ->  \0q <=q d -> a <=q (c +q d).
Lemma BQsum_Mlegt0 a c d: a <=q c ->  \0q <q d -> a <q (c +q d).
Lemma BQsum_Mltge0 a c d: a <q c ->  \0q <=q d -> a <q (c +q d).
Lemma BQsum_Mltgt0 a c d: a <q c ->  \0q <q d -> a <q (c +q d).
Lemma BQsum_Mlele0 a b c : a <=q c ->  b <=q \0q -> (a +q b) <=q c.
Lemma BQsum_Mlelt0 a b c : a <=q c ->  b <q \0q -> (a +q b) <q c.
Lemma BQsum_Mltle0 a b c : a <q c ->  b <=q \0q -> (a +q b) <q c.
Lemma BQsum_Mltlt0 a b c : a <q c ->  b <q \0q -> (a +q b) <q c.
Lemma BQsum_Mp a b: ratp a -> inc b BQp ->  a <=q (a +q b).
Lemma BQsum_Mps a b: ratp a -> inc b BQps ->  a <q (a +q b).
Lemma BQsum_Mm a b: ratp a -> inc b BQm ->  (a +q b) <=q a.
Lemma BQsum_Mms a b: ratp a -> inc b BQms ->  (a +q b) <q a.
Lemma qlt_succ x: ratp x ->  x <q x +q \1q.
```

Case of a product. The result depends on the sign of some quantities.

```
Lemma BQprod_Mlege0 a b c: inc c BQp -> a <=q b -> (a *q c)  <=q (b *q c).
Lemma BQprod_Mltgt0 a b c: inc c BQps -> a <q b -> (a *q c) <q (b *q c).
Lemma BQprod_Mlele0 a b c: inc c BQm -> a <=q b -> (b *q c)  <=q (a *q c).
Lemma BQprod_Mltlt0 a b c: inc c BQms -> a <q b -> (b *q c) <q (a *q c).


Lemma BQprod_Mpp  b c: inc b BQp -> \1q <=q c -> b <=q (b *q c).
Lemma BQprod_Mpp1 a b: \1q <=q a -> \1q <=q b -> \1q <=q a *q b.
Lemma BQprod_Mlepp a b c: inc b BQp -> \1q <=q c -> a <=q b -> a <=q (b *q c).
Lemma BQprod_Mltpp a b c: inc b BQp -> \1q <=q c  -> a <q b -> a <q (b *q c).

Lemma BQprod_Mlelege0 a b c d: inc b BQp -> inc c BQp ->
  a <=q b -> c <=q  d -> (a *q c)  <=q (b *q d).
Lemma BQprod_Mltltgt0 a b c d: inc b BQps -> inc c BQps ->
  a <q b -> c <q  d -> (a *q c)  <q (b *q d).

Lemma BQprod_Mltltge0 a b c d: inc a BQp -> inc c BQp ->
  a <q b -> c <q  d -> (a *q c)  <q (b *q d).
Lemma BQprod_ple2r a b c: ratp a -> ratp b -> inc c BQps ->
  ((a *q c)  <=q (b *q c) <-> a <=q b).
Lemma BQprod_plt2r a b c: ratp a -> ratp b ->  inc c BQps ->
   ((a *q c)  <q (b *q c) <-> a <q b).
Lemma BQprod_mle2r a b c: ratp a -> ratp b -> inc c BQms ->
 ((b *q c)  <=q (a *q c) <-> a <=q b).
Lemma BQprod_mlt2r a b c: ratp a -> ratp b ->  inc c BQms ->
  ((b *q c)  <q (a *q c) <-> a <q b).
```

```
Lemma BQprod_Mlt1 a b: ratp a -> inc b BQps ->
  (a /q b <q \1q <-> a<q b).
```

Now division.

```
Lemma BQdiv_Mlelege0 a b c d:
   ratp a -> inc b BQps -> ratp c -> inc d BQps ->
   ( a /q b <=q c /q d <-> a *q d <=q b *q c).
Lemma BQdiv_Mltltge0 a b c d:
   ratp a -> inc b BQps -> ratp c -> inc d BQps ->
   ( a /q b <q c /q d <-> a *q d <q b *q c).
Lemma BQdiv_Mle1 a b c: ratp a -> ratp b -> inc c BQps ->
     ( a <=q b *q c <->  a /q c <=q b).
Lemma BQinv_mon a b: inc a BQps -> inc b BQps  ->
   (\1q /q a <=q \1q /q b <-> b <=q a).
Lemma BQinv_mon1 a b: inc a BQps -> inc b BQps  ->
   (BQinv a <=q BQinv b <-> b <=q a).
Lemma BQdiv_lt1P  a b c:   ratp a -> inc b BQps -> inc c BQps ->
    (a /q b <q c <-> a /q c <q b)
Lemma BQdiv_le1P a b c: ratp a -> inc b BQps -> inc c BQps ->
  (a /q b <=q c <-> a /q c <=q  b).
Lemma Qdiv_Mlelege1 a c d: ratp a ->  ratp c -> inc d BQps ->
   (a <=q c /q d <-> a *q d <=q c).
Lemma Qdiv_Mltltge1 a c d: ratp a ->  ratp c -> inc d BQps ->
   (a <q c /q d <-> a *q d <q c).
Lemma Qdiv_Mltltge2 a b c: ratp a ->  inc b BQps -> ratp c ->
   (a /q b  <q c <-> a <q b *q c).
```

We show here the triangular inequality.

```
Lemma qle_abs x: ratp x ->  x <=q (BQabs x).
Lemma qle_triangular n m: ratp n -> ratp m ->
  (BQabs (n +q m)) <=q (BQabs n) +q (BQabs m)
```

9.2.8 Floor

If $x = a/b$, and $a = bq+r$ (Euclidean division on **Z**), then q is called the *floor* of x, denoted $\lfloor x \rfloor$. If z is q considered in **Q**, then $z \le x < z+1$. No other integer satisfies this property. The floor function is increasing. To say that **Q** is Archimedean means: for every $x \in \mathbf{Q}$, there is $n \in \mathbf{N}$ such that $x < n$. One deduces: if $\epsilon > 0$ in **Q**, there is $n \in \mathbf{N}$ such that $1/(n+1) < \epsilon$.

```
Definition BQfloor x := (Qnum x) %/z (Qden x).

Lemma ZS_floor x: ratp x -> intp (BQfloor x).
Lemma BQ_floor_aux x: intp x  ->
   (BQ_of_Z x) +q \1q = BQ_of_Z (x +z \1z).
Lemma BQ_floorp x (y := (BQ_of_Z (BQfloor x))) : ratp x ->
    (y <=q x /\ x <q y +q \1q).
Lemma BQ_floorp2 x z (y := (BQ_of_Z z)) : ratp x -> intp z ->
    (y <=q x /\ x <q y +q \1q) -> z = BQfloor x.
Lemma BQ_floorp3 x: ratp x -> exists2 y, intp y & x <q (BQ_of_Z y).
Lemma BQ_floorp4 x: ratp x -> exists2 y, inc y Nat & x <q (BQ_of_nat y).
Lemma BQpsS_fromN_large e: inc e BQps ->
    exists2 n, natp n &  BQinv  (BQ_of_nat (csucc n)) <q e.
Lemma BQfloor_M x y: x <=q y -> BQfloor x <=z BQfloor y.
```

```
Lemma BQfloor_Z x: intp x -> BQfloor (BQ_of_Z x) = x.
Lemma BQfloor_0: BQfloor \0q = \0z.
Lemma QpS_floor x: inc x BQp -> inc (BQfloor x) BZp.
Lemma BQ_floor_zero a b: inc a BQp -> a <q b -> BQfloor (a/q b) = \0z.
Lemma BQfloor_pos x  (m := BQfloor x): inc x BQp ->
  ( BQ_of_Z m = BQ_of_nat (P m) /\ natp (P m)).
Lemma BQfloor_pos2 x (y := BQ_of_nat (P (BQfloor x))):
   inc x BQp -> y <=q x /\  x <q y +q \1q.
```

We say that $x/2$ is the *half* of x and $(x+y)/2$ is the *middle* of x and y.

```
Definition BQhalf x := x *q \2hq.
Definition BQmiddle x y := BQhalf (x +q y).

Lemma QS_half x:  ratp x -> ratp (BQhalf x).
Lemma QS_middle x y:  ratp x -> ratp y -> inc (BQmiddle x y) BQ.
Lemma BQdouble_half2:  \2hq +q \2hq = \1q.
Lemma BQdouble_half1 x: ratp x -> BQhalf x +q BQhalf x = x.
Lemma BQdouble_half x: ratp x -> BQdouble (BQhalf x) = x.
Lemma BQhalf_double x: ratp x -> BQhalf (BQdouble x) = x.

Lemma BQhalf_pos x: inc x BQps -> inc (BQhalf x) BQps.
Lemma BQhalf_pos1 x: inc x BQps ->  (BQhalf x) <q x.
Lemma BQmiddle_comp x y: x <q y -> x <q BQmiddle x y /\  BQmiddle x y <q y.
Lemma BQ_middle_prop1 a b: ratp a -> ratp b ->
   b -q (BQmiddle a b) = BQhalf (b -q a).
Lemma BQ_middle_prop2 a b: ratp a -> ratp b ->
   (BQmiddle a b) -q a = BQhalf (b -q a).
Lemma BQhalf_mon x y : x <=q y -> BQhalf x <=q BQhalf y.
Lemma BQhalf_prop x: BQhalf x = x /q \2q.
```

9.2.9 Sums

We define here sums of the form $\sum_{i<n} f(i)$ when $f(i) \in \mathbf{Q}$, and study some properties. For instance, the sum is left unchanged if $f(i)$ is replaced by $f(n-i-1)$, the sum is n if every term is one. One can split over even and odd indices.

```
Definition qsum f n := induction_term (fun n v => (f n) +q v) \0q n.
Definition rat_below f n := forall i, i<c n -> inc (f i) BQ.
Definition same_below (e e': fterm) n:= (forall i, i <c n -> e i = e' i).


Lemma qsum0 f: qsum f \0c = \0q.
Lemma qsum_rec f n: natp n ->
   qsum f (csucc n) =  f n +q  qsum f n.
Lemma QS_qsum f n: natp n -> rat_below f n -> ratp (qsum f n).
Lemma qsum_exten f g n: natp n -> same_below f g n -> qsum f n = qsum g n.
Lemma qsum1 f: inc (f \0c) BQ -> qsum f \1c = f \0c.
Lemma qsum_An f n m: natp n -> natp m ->
  (rat_below f (n +c m)) ->
  qsum f (n +c m) = (qsum f n) +q (qsum (fun i => f (n +c i)) m).
Lemma qsum_An1 f m: natp m ->
  (rat_below f (csucc m)) ->
  qsum f (csucc m) = (f \0c) +q (qsum (fun i => f (csucc i)) m).
Lemma qsum_rev f n: natp n -> rat_below f n ->
```

```
  qsum f n  = qsum (fun i => f (n -c (csucc i))) n.
Lemma qsum_one f n: natp n -> (forall i, i<c n -> f i = \1q) ->
   qsum f n = BQ_of_nat n.
Lemma qsum_sum f1 f2 n: natp n ->
  rat_below f1 n -> rat_below f2 n ->
  qsum f1 n +q qsum f2 n = qsum (fun i => (f1 i +q f2 i)) n.
Lemma qsum_even_odd f n: natp n ->
  rat_below f (cdouble n) ->
  qsum f (cdouble n) = qsum (fun i => f (cdouble i)) n +q
     qsum (fun i => f (csucc(cdouble i))) n.
```

9.3 The ordering of Q

Cantor defines η as the order-type of the set of rational numbers greater than zero, less than one, ordered by increasing magnitude. This means that η is some ordered set, order-isomorphic to $(\mathrm{I}, \leq_{\mathrm{I}})$, where I is the set of all rational numbers x such that $0 < x < 1$ and $\leq_{\mathrm{I}}$ is the order induced on I by $\leq_{\mathbf{Q}}$.

He says: $(\mathrm{E}, \leq_{\mathrm{E}})$ is order-isomorphic to η if and only if

- the set is totally ordered;
- the cardinal of E is $\aleph_0$;
- E has no least and no greatest element;
- E is everywhere dense (meaning, between two elements, there is at least other one).

An ordering satisfying these properties is said to be "like η". The third property says that, if E is non-empty, it has to be infinite; so that, if E is countable, its cardinal is zero or $\aleph_0$.

```
Definition eta_like0 r  (E:=substrate r) :=
  [/\ total_order r, countable_set E,
     (forall x, inc x E ->  exists y, glt r y x),
     (forall x, inc x E ->  exists y, glt r x y) &
     (forall x y, glt r x y -> exists z, glt r x z /\ glt r z y)].
Definition eta_like r := eta_like0 r  /\ cardinal (substrate r) = aleph0.
Definition BQ_int01 := Zo BQ (fun x => \0q <q x /\ x <q \1q).
Definition BQps_ordering := (induced_order BQ_ordering BQps).
Definition BQ_int01_ordering := induced_order BQ_ordering BQ_int01.

Lemma eta_like_pr1 r: eta_like0 r ->
    substrate r = emptyset \/ cardinal (substrate r) = aleph0.
```

The three sets $\mathbf{Q}$, $\mathbf{Q}_+^*$ and $]0,1[$, ordered by $\leq_{\mathbf{Q}}$, are like η.

```
Lemma cardinal_BQ_int01: cardinal BQ_int01 = aleph0.
Lemma BQps_or_osr: order_on BQps_ordering BQps.
Lemma BQ_int01_or_osr: order_on BQ_int01_ordering BQ_int01.
Lemma eta_likeQ: eta_like BQ_ordering.
Lemma eta_likeQps: eta_like BQps_ordering.
Lemma eta_likeQp_int01: eta_like BQ_int01_ordering.
```

We show here: two orders that are like η are isomorphic. The idea is the following: let $(x_i)_i$ be an enumeration of the first set, $(y_i)_i$ an enumeration of the second set, and ϕ the isomorphism; it induces an isomorphism h on the indices. Fix n. There is a permutation σ of the interval I_n of integers $< n$ such that $i \mapsto x_{\sigma(i)}$ is strictly increasing. Then $i \mapsto y_{h(\sigma(i))}$ is also strictly increasing. Let k be the least index such that x_n is less than $x_{\sigma(k)}$. We deduce a new permutation σ' on I_{n+1} such that $i \mapsto x_{\sigma'(i))}$ is also strictly increasing. In order for $i \mapsto y_{h(\sigma'(i)}$ to be strictly increasing, the quantity $h(n)$ has to satisfy a given condition. We take the least integer satisfying the condition and hope for the best.

We first show how to extend a permutation of I_n to a permutation of I_{n+1}.

```
Definition EPperm_extend_aux n k s :=
  fun z => Yo (z <c k) (Vf s z) (Yo (z = k) n (Vf s (cpred z))).

Definition EPperm_extend n k s :=
  Lf (EPperm_extend_aux n k s) (csucc n) (csucc n).

Lemma perm_ints n f: natp n ->
  surjection f -> source f = n -> target f = n ->
  inc f  (permutations n).


Lemma EPperm_extend_perm n k s : natp n -> k <=c n -> inc s (permutations n) ->
  [/\ lf_axiom (EPperm_extend_aux n k s) (csucc n) (csucc n)
     &  inc (EPperm_extend n k s) (permutations (csucc n))].
```

For simplicity, we assume here that E and F are ordered sets like η, and that we have bijections $f : \mathbf{N} \to \mathrm{E}$ and $g : \mathbf{N} \to \mathrm{F}$. We shall write x_i and y_i instead of $f(i)$ and $g(i)$.

```
Section EtaProp.

Variables r1 r2 f g: Set.
Hypothesis bij_f:  bijection_prop f Nat (substrate r1).
Hypothesis bij_g:  bijection_prop g Nat (substrate r2).
Hypothesis eta_like_r1: eta_like r1.
Hypothesis eta_like_r2: eta_like r2.
```

We shall denote by $\mathrm{P}(\sigma)$ the condition that σ is a permutation on I_n such that $i \mapsto x_{\sigma(i)}$ is strictly increasing. There is exactly one such permutation, and we shall describe it. We introduce the condition $\mathrm{C}(\le, \psi, n, k, x)$ that says essentially $\psi(k-1) < x < \psi(k)$; here $\le$ is the ordering of E or F; if $k = 0$, we drop the first inequality, if $k = n$ we drop the second; we assume $k \le n$ so that the argument of ψ is $< n$.

Obviously, at most one k can satisfy the condition. If the set is totally ordered, x not of the form $\psi(i)$, then k exists. As a special case, we consider the case where $x = x_n$ and $\psi(i) = x_{\sigma(i)}$. The quantity k will be denoted by $k_{\mathrm{C}}(\sigma, n)$.

```
Definition EPperm_M s n:=
   inc s (permutations n) /\
   forall i j, i<c j -> j <c n -> glt r1 (Vf f (Vf s i)) (Vf f (Vf s j)).
Definition EPperm_compat0 r h n k x :=
   [/\ k <=c n,
    (k = \0c -> glt r x  (h \0c)),
    (k = n -> k <>\0c -> glt r (h (cpred n)) x) &
```

```
    (k <c n -> k <> \0c -> (glt r (h (cpred k)) x) /\ glt r x (h k))].
Definition EPperm_compat s n k :=
   EPperm_compat0 r1 (fun i =>  (Vf f (Vf s i))) n k (Vf f n).
Definition EPperm_next_index  n s :=
  Yo (n= \0c) \0c (select (EPperm_compat s n) Nat).

Lemma EPperm_compat_uniq s n i j:  (* 55 *)
  natp n -> EPperm_M s n ->
  EPperm_compat s n i ->  EPperm_compat s n j -> i = j.
Lemma EPperm_compat0_exists r n h x:
  natp n -> n <> \0c -> total_order r ->
  (inc x (substrate r)) ->
  (forall i, i <c n -> inc (h i) (substrate r)) ->
  (forall i, i <c n -> (h i) <> x) ->
  exists k, (EPperm_compat0 r h n k x).
Lemma EPperm_compat_exists  n s:
  natp n -> n <> \0c -> EPperm_M s n ->
  exists k, EPperm_compat s n k.
Lemma EPperm_next_indexP n s (k:= EPperm_next_index n s):
  natp n -> n <> \0c -> EPperm_M s n ->
  (EPperm_compat s n k).
Lemma EPperm_next_unique2 n s l (k:= EPperm_next_index n s):
  natp n -> n <> \0c -> EPperm_M s n ->
  EPperm_compat s n l -> l = k.
```

Let's extend the permutation σ with $\sigma(n) = k_C(\sigma, n)$. The new permutation satisfies P on I_{n+1}.

```
Lemma EPperm_extend_M n s k:
  natp n -> n <> \0c ->
  EPperm_M s n -> (EPperm_compat s n k) ->
  EPperm_M (EPperm_extend n k s) (csucc n).
Lemma EPperm_extend_M2 n s
    (k:= EPperm_next_index n s) (s':= (EPperm_extend n k s)):
  natp n -> n <> \0c -> EPperm_M  s n -> EPperm_M s' (csucc n).
Lemma EPperm_extend_M3: EPperm_M empty_function \0c.
Lemma EPperm_extend_M4: EPperm_M (identity \1c) \1c.
```

We define now, by induction, a permutation σ_n on each I_n. In case $n = 0$ and $n = 1$, the interval has a unique permutation.

```
Definition EPperm_rec :=
   induction_term (fun n s =>
       (Yo (n = \0c) (identity \1c)
          (EPperm_extend n (EPperm_next_index n s) s))) empty_function.

Lemma EPperm_rec0: (EPperm_rec \0c) =  empty_function.
Lemma EPperm_rec1: (EPperm_rec \1c) = (identity \1c).
Lemma EPperm_recs n: natp n -> n <> \0c ->
   (EPperm_rec (csucc n)) =
   EPperm_extend n (EPperm_next_index n (EPperm_rec n)) (EPperm_rec n).
Lemma EPperm_recs_mon n: natp n -> EPperm_M (EPperm_rec n) n.
```

Recall that $C(k,z)$ roughy says $\psi(k-1) < z < \psi(k)$. If ψ is strictly increasing, and the ordering is like η, there is z satisfying this condition. We shall later on use it in E; but we consider first the case of F; here there is some i such that $z = y_i$. We can take the least such i, and call it $N(\psi, k)$. We show that if $\psi(i) = \psi'(i)$ for each $i < n$, then $N(\psi, k) = N(\psi', k)$.

```
Definition EPpermi_fct h n k :=
    intersection (Zo Nat (fun j => EPperm_compat0 r2 h n k (Vf g j))).

Lemma EPpermi_pr0  h n k: natp (EPpermi_fct h n k).
Lemma EPpermi_compat0_exists r h n k G :
  natp n -> n <> \0c -> k <=c n ->
  eta_like r -> bijection_prop G Nat (substrate r) ->
  (forall i j, i<c j -> j <c n -> glt r (h i) (h j)) ->
  (forall j, j <c n -> inc (h j) (substrate r))  ->
  exists2 i, inc i Nat &   EPperm_compat0 r h n k (Vf G i).
Lemma EPpermi_pr  h n k (j := EPpermi_fct h n k)
   (P := fun m => EPperm_compat0 r2 h n k (Vf g m)) :
   (forall i j, i<c j -> j <c n -> glt r2 (h i) (h j)) ->
   (forall j, j <c n -> inc (h j) (substrate r2)) ->
   natp n -> k <=c n -> n <> \0c ->
  [/\ inc j Nat, P j & forall k, inc k Nat -> P k -> j <=c k].
Lemma EPpermi_exten h1 h2 n k:
  natp n -> n <> \0c ->
  (forall i, i<c n -> h1 i = h2 i) ->
  EPpermi_fct h1 n k = EPpermi_fct h2 n k.
```

We define h by transfinite induction via $h(n) = F(h')$, where h' is the restriction of h to I_n and F some expression. Note that the source of h' is I_n, thus n, so F may depend on n. We consider the permutation $\sigma = \sigma_n$ of I_n studied above, $k = k_C(\sigma, n)$ and $\psi(i) = g(h'(\sigma(i)))$. We take $F = k$ (except for $n = 0$, where k is undefined, in this case, we take zero). Consider the following property $R_n(h')$: h' is defined for $i < n$ and whenever i and j are indices less than n, $x_i < x_j$ is equivalent to $y_{h'(i)} < y_{h'(j)}$. If this property holds, k is as above, h'' is the extension of h' to I_{n+1} defined by $h''(n) = k$, then $R_{n+1}(h'')$ holds.

```
Definition EPpermi_next ph n (s:= EPperm_rec n)
   (h := fun i => Vf g (Vf ph (Vf s i)))
   (k := (EPperm_next_index n s)) :=
  Yo (n = \0c) \0c (EPpermi_fct h n k).

Definition EPpermi_prop ph n :=
  [/\ function ph, source ph = n, sub (target ph) Nat &
     (forall i j, i <c n -> j <c n ->
       (glt r1 (Vf f i) (Vf f j) <->
       glt r2 (Vf g (Vf ph i)) (Vf g (Vf ph j))))].
Lemma EPpermi_next_pr1 ph n   (* 140 *)
  (k1 := EPpermi_next ph n)
  (ph1 := extension ph n k1):
  natp n -> (EPpermi_prop ph n) ->
  [/\ (Vf ph1 n) = k1, forall i, i <c n -> Vf ph1 i = Vf ph i &
    (EPpermi_prop ph1 (csucc n))].
```

We consider now the function h defined by transfinite induction. The previous lemma says: whatever i and j, we have $f(i) < f(j)$ if and only if $g(h(i)) < g(h(j))$. This shows also that $\phi = g \circ h \circ f^{-1}$ is strictly increasing.

Let's consider now the following condition: the number of terms is m, the permutation σ is σ_m, the index k is k_0. We consider (C) on E with $\psi(i) = f(\sigma(i))$, and assume that k_1 is the least index i such that (C) holds for $x = f(i)$, and we consider (C) on F with $\psi(i) = g(h(\sigma(i)))$, $x = g(n)$. Let k_2 be $C_k(\sigma, m)$ (this is the position of x_m). These two conditions will be denoted by (P). We assume $k_0 \neq k_2$ and define k'_0 to be k_0 if it is $< k_2$, $k_0 + 1$ otherwise. Then (P) holds if we replace m and k_0 by k'_0.

Let's show that ϕ is the desired order isomorphism. All we need to do is show that every integer n is in the range of h. Assume integers $< n$ are in the range. Let's say that they are of the form $h(i)$, and take the supremum m' of these indices i. Consider the set of all $h(i)$ for $i < m$ (where $m = m' + 1$). If n is on the list, then n is in the range. Otherwise there is k_0 satisfying the second condition (P). The first is also satisfied. We proceed by induction.

```
Definition EPfun_aux :=
  transfinite_defined Nat_order (fun u => (EPpermi_next u (source u))).
Definition EP_fun := g \co (EPfun_aux) \co  (inverse_fun f).

Lemma EPfun_aux_pr1 (h := EPfun_aux) :
  [/\ surjection h, source h = Nat, sub (target h) Nat,
      Vf h \0c = \0c &
     forall n, natp n -> Vf h n = EPpermi_next (restriction1 h n) n].
Lemma EPfun_aux_pr2 n: natp n ->
  EPpermi_prop (restriction1  EPfun_aux n) n.
lemma EPfun_aux_M i j (h:= EPfun_aux): natp i -> natp j ->
  (glt r1 (Vf f i) (Vf f j) <->
   glt r2 (Vf g (Vf h i)) (Vf g (Vf h j))).


Definition EPperm_2pos m k1 q n
   (s :=  EPperm_rec m)   (f1 := fun i => Vf f (Vf s i))
   (h:= fun i => Vf g (Vf EPfun_aux (Vf s i)))
   (P1 := fun i => EPperm_compat0 r1 f1 m q (Vf f i)) :=
   (P1 k1 /\ (forall i, natp i -> P1 i -> k1 <=c i))
  /\ EPperm_compat0 r2 h m q (Vf g n).

Lemma EPpermi_extension2 m k0 k1 n    (* 98 *)
  (k2 := EPperm_next_index m (EPperm_rec m))
  (k0' := Yo (k0 <c k2) k0 (csucc k0)):
  natp m -> m <> \0c -> k2 <> k0 ->
  EPperm_2pos m k1 k0 n ->
  EPperm_2pos (csucc m) k1 k0' n.
Lemma EPfun_aux_bij : bijection_prop EPfun_aux Nat Nat.  (* 198 *)
Lemma EP_fun_pr: order_isomorphism EP_fun r1 r2.

End EtaProp.
```

The result is now trivial.

```
Lemma Cantor_eta_pr r1 r2:
  eta_like r1 -> eta_like r2 -> r1 \Is r2.
```

The three ordered sets considered above are order isomorphic, since they are like η. We give here an explicit function. For $\mathbf{Q}_+^* \to]0,1[$, we consider $x \mapsto x/(1+x)$. For $\mathbf{Q} \to \mathbf{Q}_+^*$ we consider $1 - 1/x$ for $x < 0$ and $x + 1$ otherwise.

```
Lemma BQ_moebius_props1  x
  (f := fun z=> z /q (\1q +q z))
```

```
  (g := fun z => z /q (\1q -q z)):
  ratp x ->
  ((x <> \1mq -> g (f x) = x) /\ (x <> \1q -> f (g x) = x)).
Lemma BQ_moebius_props2  x
  (f := fun z=> \1q /q (\1q -q z))
  (g := fun z => \1q -q  (\1q /q z)):
  ratp x ->
  ((x <> \1q -> g (f x) = x) /\ (x <> \0q -> f (g x) = x)).
Lemma BQ_iso1: order_isomorphism
   (Lf (fun z => z /q (\1q +q z)) BQps BQ_int01)
    BQps_ordering  BQ_int01_ordering.
Lemma BQ_iso2: order_isomorphism  (* 52 *)
   (Lf (fun z => Yo (z <q \0q) (\1q /q (\1q -q z)) (z +q \1q)) BQ BQps)
    BQ_ordering BQps_ordering.
```

The isomorphisms given above are not unique: for instance $x \mapsto ax+b$ is an order isomorphism on $\mathbf{Q}$ when $a>0$. Let's count the number c of order isomorphisms $\mathbf{Q}\to\mathbf{Q}$: there are $2^{\aleph_0}$. We first note that an order isomorphism is a permutation, thus $c \le 2^{\aleph_0}$. Conversely, $2^{\aleph_0}$ is the number of functions $\mathbf{N}\to\{0,1\}$. Given such a function f, we construct a function g by: $g(0)=0$, and $g(n+1)=g(n)+f(n)+1$.

Now, consider $h_{n,u,v}(x)=(v-u)(x-n)+u$. Whatever u,v,n, $h(n)=u$ and $h(n+1)=v$; if $u\ne v$, then h is a bijection $\mathbf{Q}\to\mathbf{Q}$; if moreover $u<v$, then h is strictly increasing (thus is injective), and maps the interval $[n,n+1]$ onto the interval $[u,v]$. Let f and g be as above, and define h by $h(x)=h_{\lfloor x\rfloor,g(\lfloor x\rfloor),g(\lfloor x\rfloor+1)}(x)$ for $x\ge 0$ and $h(x)=x$ otherwise. Note that $\lfloor x\rfloor$ is a priori in $\mathbf{Z}$, but has to be considered in $\mathbf{N}$, and $\mathbf{Q}$. This is easily seen to be an order isomorphism $\mathbf{Q}\to\mathbf{Q}$, and $f\mapsto h$ is injective.

```
Definition simple_interpolation n u v :=
   fun x => (v -q u) *q (x -q n) +q u.
Definition multiple_interpolation f x:=
   let y :=(BQfloor x) in
     simple_interpolation (BQ_of_Z y) (f (P y)) (f (csucc (P y))) x.

Lemma interpolation_prop1 n u v (f := simple_interpolation n u v):
  ratp n -> u <q v ->
  [/\ f n = u, f (n +q \1q) = v,
      (forall x, n <=q x -> x <=q (n +q \1q) ->
        (u <=q f x /\ f x <=q v)),
      (forall y,  u <=q y -> y <=q v -> exists x,
       [/\ n <=q x,  x <=q (n +q \1q) & y = f x]) &
   (forall x y, x <q y -> f x <q f y)].
Lemma interpolation_prop2 n u v (f := simple_interpolation n u v):
  ratp n -> u <q v ->
      (forall x, n <=q x -> x <q (n +q \1q) ->
        (u <=q f x /\ f x <q v)) /\
      (forall y,  u <=q y -> y <q v -> exists x,
       [/\ n <=q x,  x <q (n +q \1q) & y = f x]).

Lemma  multiple_interpolation_prop f   (* 101 *)
   (g := multiple_interpolation  (fun z => (BQ_of_nat (f z)))):
  (forall n, natp n -> natp (f n)) ->
  (forall n, natp n -> f n <c f(csucc n)) ->
  (f \0c) = \0c ->
  [/\ forall x, inc x BQp -> inc (g x) BQp,
```

```
      (forall y, inc y BQp -> exists2 x, inc x BQp & g x = y),
      (forall n, natp n -> g (BQ_of_nat  n) = BQ_of_nat  (f n)) &
      forall x y, inc x BQp -> x <q y -> g x <q g y].

Lemma BQ_iso3 a b: inc a BQps -> ratp b ->
    order_isomorphism (Lf (fun z => a*q z +q b) BQ BQ)
      BQ_ordering  BQ_ordering.
Lemma BQ_iso4 (E := Zo (permutations BQ)
       (fun f => order_isomorphism f BQ_ordering  BQ_ordering)):
   cardinal E = \2c ^c aleph0.
```

We now construct an order isomorphism $f : \mathbf{Q}^* \to \mathbf{Q}$. Let $b_n = f(1/(n+1))$. This sequence is strictly decreasing; let B be the set of all t such that $b_n < t$ for at least one n. Then B satisfies the following properties: it is non-empty, not $\mathbf{Q}$, is an initial segment (i.e., $t \in \mathrm{B}$ and $t < x$ implies $x \in \mathrm{B}$) and has no least upper bound. In the next chapter, such a set B will be called an irrational number and we shall also see how to obtain a sequence b_n, given an irrational number B.

We explain here how to construct a function from a sequence b_n. On each interval $[1/(n+1), 1/n]$ we consider the unique function f of the form $z \mapsto az+b$ such that $f(1/(n+1)) = b_n$ and $f(1/n) = b_{n-1}$ (the function is as before, a bit more complicated because the interval is not of unit length); also we have to assume $n > 0$. For $n = 0$, the interval is $]1,\infty[$ and we consider $f(z) = z + b$ for some b. We can merge these functions together, so as to obtain a function f_b, an order isomorphism $\mathbf{Q}^*_+ \to \mathrm{B}$. We get an order isomorphism $\mathbf{Q}^* \to \mathbf{Q}$ if $\mathbf{Q}$ is the disjoint union of A and B.

```
Definition simple_interpolation2  i u v :=
   fun z => u -q (BQ_of_nat (csucc i) *q z -q \1q)
   *q (BQ_of_nat i) *q ( u -q v).
Definition multiple_interpolation2 f x :=
  let n := (BZ_val (BQfloor (BQinv x))) in
  Yo (n = \0c) (x +q (f \0c) -q \1q)
    (simple_interpolation2 n (f n) (f (cpred n))  x).


Lemma simple_interpolation2_pa i u v   (* 58 *)
  (yk := fun k => (BQinv (BQ_of_nat (csucc k))))
  (ya := yk i) (yb := yk (csucc i))
  (f := simple_interpolation2  (csucc i) u v):
  natp i -> u <q v ->
  [/\ f yb = u, f ya = v,forall z1 z2, z1 <q z2 -> f z1 <q f z2 &
     forall a,  (f yb) <q a /\ a <q (f ya) ->
     exists z, [/\ yb <q z, z <q ya  & a = f z]].
Lemma multiple_interpolation_prop2 f   (* 172 *)
   (g := multiple_interpolation2 f)
   (Z := Zo BQ (fun z => exists2 n, natp n & f n <q z)):
  (forall n, natp n -> f (csucc n) <q f n) ->
  order_isomorphism (Lf g BQps Z)
    (induced_order BQ_ordering BQps) (induced_order BQ_ordering Z).
Lemma multiple_interpolation_prop3 f1 f2 (* 100 *)
   (Zb := Zo BQ (fun z => exists2 n, natp n & f1 n <q z))
   (Za := Zo BQ (fun z => exists2 n, natp n & z <q f2 n)):
  (forall n, natp n -> f1 (csucc n) <q f1 n) ->
  (forall n, natp n -> f2 n <q f2 (csucc n)) ->
  (disjoint Za Zb) -> (Za \cup Zb = BQ) ->
```

```
  exists2 g, order_isomorphism g
    (induced_order BQ_ordering (BQ -s1 \0q)) BQ_ordering &
   Vfs g BQps = Zb.
```

9.4 Stern Brocot sequences

We have shown above that **N** and **Q** are equipotent. We give here an explicit enumeration of $\mathbf{Q}_+$, via the so called Stern diatomic sequence (or Stern-Brocot sequence, named "fusc" by Dijkstra); it satisfies

$$a_0 = 1, \quad a_1 = 1, \quad a_{2n} = a_n, \quad a_{2n+1} = a_n + a_{n+1}. \tag{9.1}$$

One may assume $n > 0$, since the case $n = 0$ is trivial, so that $n < 2n$ and $n+1 < 2n+1$, this shows that there is a unique solution. Defining the sequence by induction is not possible, since, if $h(m)$ is the half of m, one has to show $h(m) < m$ for m even, and $h(m)+1 < m$ for m odd; these properties are true, but not by structural induction. However, the definition by transfinite induction always makes sense. We only have to check a posteriori that the argument n of the auxiliary function F is called with arguments in its domain, in the recursive calls needed to evaluate the fusc function on integer arguments.

```
Definition fusc_next F n:=
  Yo (n = \0c) \0c (Yo (n = \1c) \1c (Yo (evenp n) (Vf F (chalf n))
   ((Vf F (chalf n)) +c (Vf F (csucc (chalf n)))))).
Definition fusc :=
  Vf (transfinite_defined Nat_order (fun u => (fusc_next u (source u)))).
Definition fusc_prop f:=
  [/\ f \0c = \0c, f \1c = \1c,
   (forall n, natp n -> f (cdouble n) = f n) &
   (forall n, natp n -> f (csucc (cdouble n)) =  f n +c f (csucc n)) ].

Lemma fusc_pr: fusc_prop fusc.
```

The equation (9.1) has a unique solution; it will be denoted by s_n in what follows. Clearly s_n is a non-zero integer (for $n > 0$). The sequence s_n satisfies some amusing properties, for instance:

$$p = 2^n \implies s_p = 1, \quad s_{p-1} = n, \quad s_{p+1} = n+1.$$

```
Lemma fusc_unique f g: fusc_prop f -> fusc_prop g ->
   {inc Nat, f =1 g}.
Lemma NS_fusc n: natp n -> natp (fusc n).
Lemma fusc_even n: natp n -> fusc (cdouble n) = fusc n.
Lemma fusc_odd n: natp n ->
   fusc (csucc (cdouble n)) = fusc n +c fusc (csucc n).
Lemma fusc_nz n: natp n -> n <> \0c -> (fusc n) <> \0c.
Lemma fusc_nz' n: natp n -> fusc (csucc n) <> \0c.
Lemma fusc0: fusc \0c = \0c.
Lemma fusc1: fusc \1c = \1c.
Lemma fusc2: fusc \2c = \1c.
Lemma fusc3: fusc \3c = \2c.
Lemma fusc_pow2 n: natp n -> fusc (\2c ^c n) = \1c.
Lemma fusc_pred_pow2 n: natp n ->  fusc (\2c ^c n -c \1c) = n.
Lemma fusc_succ_pow2 n: natp n ->  fusc (\2c ^c n +c \1c) = csucc n.
```

Bijection beween N and Q. Consider $x_n = s_n / s_{n+1}$; we have $x_0 = 0$, $x_1 = 1$, $x_2 = 1/2$, and if $n = 2^p$, then $x_{p-1} = n$, $x_p = 1/(n+1)$. The induction principle is:

$$x_{2n} = \frac{1}{1+1/x_n} (\text{or } \frac{1}{x_{2n}} = \frac{1}{x_n} + 1) \qquad x_{2n+1} = x_n + 1. \tag{9.2}$$

```
Definition fuscz n := BZ_of_nat (fusc n).
Definition fusc_quo n:= BQ_of_nat (fusc n) /q  BQ_of_nat (fusc (csucc n)).

Lemma QpS_fusc_quo n: natp n -> inc (fusc_quo n) BQp.
Lemma QpsS_fusc_quo n: natp n -> n <> \0c -> inc (fusc_quo n) BQps.
Lemma QS_fusc_quo n: natp n -> inc (fusc_quo n) BQ.
Lemma fusc_quo_0: fusc_quo \0c = \0q.
Lemma fusc_quo_1: fusc_quo \1c = \1q.
Lemma fusc_quo_2: fusc_quo \2c = \2hq.
Lemma fusc_quo_pow2 n: natp n ->
   fusc_quo (\2c ^c n) = BQinv (BQ_of_nat (csucc n)).
Lemma fusc_quo_pow2p n: natp n ->  fusc_quo (\2c ^c n -c \1c) = BQ_of_nat n.
Lemma fusc_quo_even n: natp n -> n <> \0c ->
  fusc_quo (cdouble n)  = BQinv(\1q +q BQinv (fusc_quo n)).
Lemma fusc_quo_odd n: natp n ->
  fusc_quo (csucc (cdouble n)) = \1q +q (fusc_quo n).
```

Let F be the function defined by the code that follows. We have $F(x) \ge 0$, and $F(x) > 0$ if $x > 0$ and $x \ne 1$; moreover $F(x_{2n}) = x_n$ and $F(x_{2n+1}) = x_n$. To each real number x, we can associate a sequence of bits as follows: If $x > 1$, the first bit is one, and we continue with the bits of $F(x)$; if $0 < x < 1$, the first bit is zero and we continue with the bits of $F(x)$; otherwise the sequence is empty. If x has the form x_k, the sequence of bits associated to x is the sequence of binary digits of k (minus the leading digit). On can show that if x is rational, the sequence is always finite (this is because every positive rational number has the form x_k). If x is a positive real irrational number, then the sequence is never finite (all arguments to F are > 0). If we take the n first bits, the associated integer k and $y_n = x_k$, then the sequence y_n converges to x.

```
Definition fusc_quo_inv x:=
   Yo (\1q <=q x) (x -q \1q)
   (Yo (\0q <q x) (BQinv (BQinv x -q \1q)) \0q).

Lemma fusc_quo_inv_props (F := fusc_quo_inv):
  [/\ (forall q, ratp q -> inc (F q) BQp),
      (forall x, inc x BQps -> x <> \1q -> inc (F x) BQps),
      (forall n, natp n -> F(fusc_quo (csucc (cdouble n))) = (fusc_quo n))
        &  (forall n, natp n -> n <> \0c ->
        F(fusc_quo (cdouble n)) = (fusc_quo n))].
```

We show that $n \mapsto x_n$ is a bijection $\mathbf{N} \to \mathbf{Q}_+$. The idea of the proof is explained above. We first show that s_n and s_{n+1} are coprime (if s_n and s_{n+1} are coprime then $s_{2n} = s_n$ and $s_{2n+1} = s_n + s_{n+1}$ are coprime, the other case is similar). Assume $x_n = x_m$; then n and m have the same parity (otherwise, the numbers compare differently with 1) and $s_n = s_m$, $s_{n+1} = s_{m+1}$ (the fractions are reduced). Assume for instance $n = 2p$ and $m = 2q$. We have $s_p = s_q$ and $s_p + s_{p+1} = s_q + s_{q+1}$, then $s_{p+1} = s_{q+1}$, and we conclude by induction; the other case is similar. Every rational number has the form x_n: if we apply F, the sum of the numerator and denominator is strictly decreasing. If the sum is ≤ 1, the fraction is 0/1, thus x_0.

```
Lemma fusc_coprime n: natp n -> BZcoprime (fuscz n) (fuscz (csucc n)).
Lemma fusc_quo_numden n (q :=  fusc_quo n): natp n ->
   Qnum q = (fuscz n) /\ Qden q = (fuscz (csucc n))
Lemma fusc2_injective n m: natp n -> natp m ->
  fusc n = fusc m -> fusc (csucc n) = fusc (csucc m) -> n = m.
Lemma fusc2_surjective a b:
  natp a -> natp b -> b <> \0c -> BZcoprime (BZ_of_nat a) (BZ_of_nat b) ->
 exists n, [/\ natp n, fusc n = a & fusc (csucc n) = b].
Lemma fusc_quo_bijection:
   bijection_prop (Lf fusc_quo Nat BQp) Nat BQp.
```

Even Stern numbers. Let's show:

$$s_n \text{ is even} \iff n = 0 \mod 3.$$

If n is even, this reduces to: n and $2n$ are both zero or both non-zero mod 3. In the odd case, we have to consider $s_{2m+1} = s_m + s_{m+1}$. Asume s_m even, so m is zero mod 3, $m+1$ is not zero mod 3, $2m+1$ is not zero mod 3, s_{m+1} is odd, s_{2m+1} is odd. The case s_{m+1} even is similar. Otherwise, the two terms are odd and s_{2m+1} is even. The quantity m mod 3 has to be one; so that $2m+1$ is a multiple of 3.

```
Lemma fusc_is_even n: natp n -> (evenp (fusc n) <-> n %%c \3c = \0c).
```

The Stern diatomic sequence In [19], Stern consider a table, defined by two strictly positive integers m and n. The first row contains m and n, the second row contains m, $m+n$, n, etc. If a row contains a_1, a_2, a_3, etc, then the next row contains a_1, a_1+a_2, a_2 , a_2+a_3, a_3, etc. Elements that are copied from the previous row will be called “old”, other elements will be called “new”. Denote by $\mathrm{A}_{pi}(m,n)$ the i-th element of the p-th row of the table, initialized with (m,n). Row p has $2^{p-1}+1$ elements. We have $\mathrm{A}_{1,0} = m$, $\mathrm{A}_{1,1} = n$, $\mathrm{A}_{p+1,2i} = \mathrm{A}_{p,i}$, and $\mathrm{A}_{p+1,2i+1} = \mathrm{A}_{p,i} + \mathrm{A}_{p,i+1}$. Asume $m = n = 1$; then the first row contains s_1 and s_2, the second row contains s_2, s_3 and s_4, the third row contains all s_i with index i between 4 and 8, etc. Note that if $i = 2^p$, then s_i appears twice: at the end of row p and the start of row $p+1$. Thus $\mathrm{A}(1,1)$ is a representation of the sequence s_i as a table.

Assume $m = s_k$ and $n = s_{k+1}$; then $\mathrm{A}_{p,i}(m,n) = s_{2^{p-1}k+i}$ (by induction). Thus $\mathrm{A}(m,n)$ is a subtable of $\mathrm{A}(1,1)$. If m and n are two numbers, g their gcd, there is k such that $m/g = s_k$ and $n/g = s_{k+1}$, so that $\mathrm{A}_{p,i}(m,n) = g s_{2^{p-1}k+i}$. Thus, all properties of $\mathrm{A}(m,n)$ can be deduced from the sequence s_i.

Let's assume for a moment $m = n = 1$. Here new elements have the form s_{2k+1}, old elements have the form s_{2k}. If $x = s_{2k}/s_{2k+1}$, then its numerator is old, its denominator is new. We have seen above that every rational number x with $0 < x < 1$ has this form.

In §4 Stern says: if a, b and c are three consecutive numbers in the $\mathrm{A}(1,1)$ table, then b divides $a+c$, the quotient being odd. Since the quotient is odd, we get $c = a+b$ modulo 2, so that one out of three Stern numbers is even.

In §5, he deduces that a and b are coprime (if x divides s_k and s_{k+1} it divides s_{k+2}, and by induction all s_i for $i \ge k$, but one of these s_i is one). He also deduces: $b = a+c$, then a and c are coprime.

We first show that a non-zero integer k can be uniquely written as $k = 2^n(2p+1)$. The relation $s_{k-1} + s_{k+1} = s_k(2n+1)$ follows by induction on n. Thus, if $a = s_{k_1}$, $b = s_k$ and $c = s_{k+1}$, it follows that b divides a and c. If $a+c = b$, then $n = 0$, $a = s_{2p} = s_p$ and $c = s_{2p+2} = s_{p+1}$. These two quantities are hence coprime.

```
Lemma even_odd_factor b: natp b -> b <> \0c -> exists n p,
  [/\ natp n, natp p &  b = (\2c ^c n) *c csucc (cdouble p)].
Lemma even_odd_factor_uniq n p n' p':
  natp n -> natp p -> natp n' -> natp p' ->
  (\2c ^c n) *c csucc (cdouble p) = (\2c ^c n') *c csucc (cdouble p') ->
  (n = n' /\ p = p').
Lemma fusc_mean_div n p (b := (\2c ^c n) *c csucc (cdouble p))
   (a := cpred b) (c := csucc b):
   natp n -> natp p ->
   fusc a +c fusc c = (fusc b) *c  csucc (cdouble n).
Lemma fusc_mean_div1 b (a := cpred b) (c := csucc b):
  natp b -> b <> \0c ->
  exists2 n, natp n &fusc a +c fusc c = (fusc b) *c  csucc (cdouble n).
Lemma fusc_mean_div2 b (a := cpred b) (c := csucc b):
  natp b -> b <> \0c ->
  fusc a +c fusc c = fusc b -> BZcoprime (fuscz a) (fuscz c).
```

An Iterative Formula. Dijkstra [9] proposes the following procedure to compute s_{N}. Initially, we have $n = \mathrm{N}$, $a = 1$, $n = 0$. While n is non-zero, it is replaced by its half; in the even case a is replaced by $a+b$, in the odd case b is replaced by $b+a$; finally the return value is b.

We can rewrite this as: there is a function $\mathrm{F}_n(a,b)$, such that

$$\mathrm{F}_0(a,b) = b, \quad \mathrm{F}_{2n}(a,b) = \mathrm{F}_n(a+b,b), \quad \mathrm{F}_{2n+1}(a,b) = \mathrm{F}_n(a,b+a) \tag{9.3}$$

and $\mathrm{F}_n(1,0) = s_n$. Proving that this function exists is tricky (we must use the same trick as for s_n). In order to show $\mathrm{F}_n(1,0) = s_n$, it is convenient to use a loop invariant.

$$\mathrm{F}_n(a,b) = as_n + bs_{n+1}.$$

```
Definition Fusci n a b := a *c (fusc n) +c b*c (fusc (csucc n)).

Lemma fusci_even n a b: natp n -> natp a -> natp b ->
   Fusci (cdouble n) a b = Fusci n (a +c b) b.
Lemma fusci_odd n a b: natp n -> natp a -> natp b ->
   Fusci (csucc (cdouble n)) a b = Fusci n a (a +c b).
Lemma fusci_zero a b: natp b -> Fusci \0c a b = b.
Lemma fusci_val n: natp n -> Fusci n \1c \0c = fusc n.
```

The Palindrome. Denote s_{2^i+j} by $a_{i,j}$. Consider the table whose i-th row is formed by all the $a_{i,j}$ where $j \le 2^i$ (note that powers of two appear twice in the table, as 2^{i+1} appears on row i (with $j = 2^i$) and row $i+1$ (with $j = 0$). We prove here that each row is a palindrome:

$$p = 2^n, \quad p \le a, b \le 2p, \quad a+b = 3p \implies s_a = s_b. \tag{9.4}$$

The proof is by induction on n. If $n = 0$, the condition $p \le a \le 2p$ says $a = 1$ or $a = 2$, case where $s_a = 1$, similarly $s_b = 1$. Assume the property true for n, and let's show it for $n+1$. Write $p = 2^n$, so that $a+b = 6p$. This shows that a and b have the same parity, and the case a, b even holds trivially by induction. Assume $a = 2a'+1$ and $b = 2b'+1$. We have $p \le a' \le a'+1 \le 2p$. The conclusion follows from $s_{a'} = s_{b'+1}$ and $s_{a'+1} = s_{b'}$.

```
Lemma fusc_palindrome n a b (p := \2c ^c n): natp n ->
   p <=c a -> a <=c cdouble p -> p <=c b -> b <=c cdouble p ->
   a +c b = \3c *c p -> fusc a = fusc b.
```

The Fractal Structure. If we consider the sequence s_n as a table $a_{i,j}$, then each column forms an arithmetic progression; moreover, the common difference is the Stern number of the column. So $a_{i+1,j} = a_{i,j} + s_j$, or:

$$s_{2p+j} = s_{p+j} + s_j \qquad (p = 2^i, j \le p). \tag{9.5}$$

Proof. Denote the equality by $\mathrm{H}(i, j)$. It implies $\mathrm{H}(i+1,2j)$ (all arguments of s are even). If moreover $\mathrm{H}(i, j+1)$ holds, then $\mathrm{H}(i+1,2j+1)$ holds (all arguments of s are odd). In the case $i = 0$, we have $j = 0$ or $j = 1$, and the result is trivial. Note that the case $j = p$ follows from: $s_{pk} = s_k$, whenever p is a power of two.

```
Lemma fusc_kpow2n k n : natp n -> natp k -> fusc (\2c ^c n *c k) = fusc k.
Lemma fusc_col_progression i j: natp i -> natp j -> j <=c \2c ^c i ->
  fusc(\2c ^c (csucc i) +c j) = fusc(\2c ^c i +c j) +c fusc j.
```

Maximum of the rows. The maximum of the n-th row is the Fibonacci number F_{n+1}. First, if $k \le 2^n$ then $s_k \le \mathrm{F}_{n+1}$ (This is obvious by induction if k is even. Otherwise, let $k = 2m+1$, $s_k = s_m + s_{m+1}$; one of m and $m+1$ is even, so that one of s_m, s_{m+1} is $\le \mathrm{F}_{n-1}$, both are $\le \mathrm{F}_n$). Consider

$$2^n \le c_{n+2} = \frac{4.2^n - (-1)^n}{3} < c'_{n+2} = \frac{5.2^n + (-1)^n}{3} \le 2.2^n$$

(in case $n = 1$, we must replace $<$ by $=$). Since $c_n + c'_n = 3.2^{n-2}$, we have $s(c_n) = s(c'_n)$. It happens that the maximum of s_k in the range $[2^n, 2^{n+1}]$ is reached at c_n and c'_n, and at no other point. We give here an alternate definition of c_n, and show that $s(c_n) = \mathrm{F}_n$. Consider

$$c_0 = d_0 = 0, c_{n+1} = 2d_n + 1, d_{n+1} = c_n + d_n.$$

Write $c = c_n$, $c' = c_{n+1}$ $d = c_n$, $d' = c_{n+1}$. We have $s(c_{n+2}) = s(d') + s(d'+1)$. If n is even, then $c = d$, otherwise $c = d+1$. Thus, the pair $(d', d'+1)$ is respectively $(2c, c')$ and $(c', 2c)$. So $s(c_{n+2}) = s(c') + s(2c) = s(c') + s(c)$. The result follows.

```
Definition Fib_fusc_rec :=
 induction_term  (fun _ v => (J (csucc (cdouble (Q v))) (P v +c Q v)))
   (J \0c \0c).
Definition Fib_fusc n := P ( Fib_fusc_rec n).

Lemma fusc_bound1 n k: natp n -> k <=c \2c ^c n -> fusc k <=c Fib (csucc n).

Lemma Fib_fusc_recS n (v := Fib_fusc_rec n) (a := P v) (b := Q v) :
   natp n -> Fib_fusc_rec (csucc n) = J (csucc (cdouble b)) (a +c b).
Lemma Fib_fusc_rec0: Fib_fusc_rec \0c = (J \0c \0c).
Lemma Fib_fusc_rec1: Fib_fusc_rec (csucc \0c) = (J \1c \0c).
Lemma NS_Fib_fusc_rec n (v := Fib_fusc_rec n) (a := P v) (b := Q v): natp n ->
  [/\ (pairp v), natp a & natp b].
Lemma Fib_fusc_rec_eo n
  (a := P (Fib_fusc_rec n)) (b := Q (Fib_fusc_rec n))
  (a' := P (Fib_fusc_rec (csucc n))) (b' := Q (Fib_fusc_rec (csucc n))):
  natp n ->
  ((evenp n -> b' = cdouble a /\ (succ b') = a')
  /\ (oddp n -> b' = a'/\  (csucc b') = cdouble a)).
Lemma Fib_fusc_rec_frec n (F := fun k => (fusc (Fib_fusc k))) : natp n ->
  F (csucc (csucc n)) = F (csucc n) +c  F n.
Lemma Fib_fusc_val n: natp n -> (fusc (Fib_fusc n)) = Fib n.
Lemma Fib_fusc_bound n (p := \2c ^c n) (v := Fib_fusc (csucc (csucc n))):
   natp n -> (p <=c v /\ v <=c (cdouble p)).
```

Sum of simplicities. We show here

$$\sum_{p\leq i<2p} \frac{1}{s_i s_{i+1}} = 1, \qquad (p = 2^n).$$

Denote the sum by C_n, and C'_{ni} be the sum of the i first terms. If $x = a/b$ is a rational number in reduced form, the quantity $1/(ab)$ is sometimes called the the simplicity of x. We compute here the sum of the simplicities of the fractions s_i/s_{i+1}. If $n = 0$, we have a single term $1/(s_1 s_2)$ and the result is one. Otherwise, let $p = 2q$. We shall use (proof by induction on i):

$$s_i s_{2q+i+1} + 1 = s_{i+1} s_{2q+i} \qquad i < q, q = 2^n.$$

One deduces $C'_{ni} = s_i/s_{p+i}$, thus $C'_{ny} = s_q/s_{3q} = 1/2$.

```
Definition fsimpl_sum p i :=
  qsum(fun j => BQinv (BQ_of_nat (fusc (p +c j) *c fusc (p +c (csucc j))))) i.

Lemma fusc_rec_spec n i (p := \2c ^c  n) : natp n -> i <c p ->
   fusc i *c fusc(csucc (cdouble p) +c i) +c \1c =
    fusc (csucc i) *c fusc((cdouble p) +c i).
Lemma fusc_rec_spec1 n i (q := \2c ^c  n)(p := cdouble q) :
  natp n -> i <=c q ->
  fsimpl_sum p i = BQdiv (BQ_of_nat (fusc i)) (BQ_of_nat (fusc (p +c i))).
Lemma fusc_rec_spec2 n (q := \2c ^c  n)(p := cdouble q) :
  natp n -> fsimpl_sum p q =  \2hq.
Lemma fusc_sum_simpl n (p := \2c ^c  n):
  natp n -> fsimpl_sum p p = \1q.
```

We show here

$$\sum_{p\leq i<2p} \frac{s_i}{s_{i+1}} = (3p-1)/2, \qquad p = 2^n.$$

If $n = 0$, then this is 1; in all other cases, it is not an integer. Write this as $A_n + B_n$, where A contains terms with even index. If $n = 1$, there is a single term in A_n, namely $s_2/s_3 = 1/2$. Otherwise, we may assume $p = 2q$ and $A_n = \sum_i s_{2q+2i}/s_{2q+2i+1} = \sum_i s_{q+i}/(s_{q+i} + s_{q+i+1})$. We split this sum in two, in the first part $i < q/2$, in the second part we replace $f(i)$ by $f(q-i-1)$. If the generic term of the first part is $a/(a+b)$, by the palindrome condition, the generic term of the second part is $b/(b+a)$. The sum of these two terms is one; it follows $A_n = q/2$. The generic term of B_n has the form $(a+b)/b = 1 + a/b$. Now, $\sum a/b$ is the sum at order $n-1$; the result follows by induction.

```
Lemma qsum_fusc1 n (p := \2c ^c  n) : natp n ->
 qsum (fun i => fusc_quo ( (cdouble p)  +c (cdouble i))) p =
  BQhalf (BQ_of_nat p).
Lemma qsum_fusc n (p := \2c ^c  n) : natp n ->
 qsum (fun i => fusc_quo (p +c i)) p =
   BQhalf (BQ_of_nat (cpred (\3c *c p))).
```

Weak base two representations. Let's consider the decomposition of a natural number n as

$$n = \sum_{i<k} a_i 2^i, \quad \forall i, a_i \leq 2 \quad k = 0 \vee a_{k-1} \neq 0.$$

The condition $k = 0 \vee a_{k-1} \neq 0$ will be referred to as "the leading term condition"; it implies $n \geq 2^{k-1}$ (it also says that $n = 0$ is equivalent to $k = 0$). We know that there is a unique decomposition if we require $a_i < 2$. We count here the number of decompositions with $a_i \leq 2$.

```
Definition expansion_ext f k:= expansion f \3c k.
Definition expansion_ext_of f k a :=
  expansion_ext f k /\ expansion_value f \2c = a.
Definition expansion_ext_normal_of f k a :=
  expansion_ext_of f k a /\  exp_boundary f k.
Definition expansion_ext_of_a f a :=
  expansion_ext_normal_of f (cardinal (domain f)) a.
Definition expansions_ext_of n :=
   Zo (sub_fgraphs Nat \3c) (fun z => (expansion_ext_of_a z n)).
Definition Nbexp n := cardinal (expansions_ext_of n).
```

Let F be an expansion of length k and value n, and E_n the set of all expansions (of arbitrary length) with value n. We study here its cardinal f_n. The leading term condition gives $k \le n$, so that E_n is finite, and f_n is an integer. Obviously $f_0 = 1$,

```
Lemma expe_p1 f k: expansion_ext f k -> inc f (sub_fgraphs Nat \3c).
Lemma expe_p2 f n: expansion_ext_of_a f n -> inc f (sub_fgraphs Nat \3c).
Lemma expe_p3 f n: expansion_ext_of_a f n -> inc f (expansions_ext_of n).

Lemma expe_bounded1 f k a:
  natp k -> expansion_ext_normal_of f (csucc k) a ->
   (\2c ^c k) <=c a.
Lemma expe_bounded2 n f: natp n -> inc f (expansions_ext_of n)  ->
  cardinal (domain f) <=c n.
Lemma expe_0 : Nbexp \0c = \1c.
Lemma expe_nat n: natp n -> natp (Nbexp n).
```

If G is the restriction of F to its $k-1$ first terms with value m, if a is the omitted term, then $n = a+2^{k-1}+m$. We consider here $\bar{F}$, the restriction of F to its $k-1$ last terms, shifted by one. If m is the value and a the omitted term, we have now $n = 2m + a$. Moreover $\bar{F}$ satisfies the leading term condition. The inverse construction is denoted by F_a: its first term is a, and its restriction is F. Note: when we restrict, we need $n \neq 0$, conversely, if we augment, we need $a \neq 0$ when $n = 0$ (together with the obvious $a < 3$).

```
Lemma NS_expe_val f k: expansion_ext f k -> natp (expansion_value f \2c).

Definition expansion_ext_aug f i (k := (cardinal (domain f))):=
   Lg (csucc k) (fun z => Yo (z = \0c) i (Vg f (cpred z))).
Definition expansion_ext_dim f (k := (cardinal (domain f))):=
   Lg (cpred k) (fun z => (Vg f (csucc z))).

Lemma expansion_ext_aug_p1 f i (n:= expansion_value f \2c)
   (f' :=  expansion_ext_aug f i) (m:= expansion_value f' \2c):
  expansion_ext_of_a f n ->  i<c \3c -> (n <> \0c \/ i <> \0c) ->
  (expansion_ext_of_a f' m /\ m = \2c *c n +c i).
Lemma expansion_ext_dim_p1 f (n:= expansion_value f \2c)
  (f' :=  expansion_ext_dim f) (m:= expansion_value f' \2c) (i := Vg f \0c):
  expansion_ext_of_a f n -> n <> \0c ->
  [/\ i<c \3c, expansion_ext_of_a f' m & n = \2c *c m +c i].
Lemma expansion_ext_dim_aug f n i:
   inc f (expansions_ext_of n) ->
   expansion_ext_dim (expansion_ext_aug f i) = f.
Lemma expansion_ext_aug_dim f n:
   inc f (expansions_ext_of n) -> n <> \0c ->
   expansion_ext_aug (expansion_ext_dim f) (Vg f \0c) = f.
```

Assume n odd, F an expansion with sum n, $\bar{\mathrm{F}}$, the restriction withe value m, and a the constant term, so that $n = 2m + a$. Since n is odd, it follows $a = 1$. Thus $\mathrm{F} \mapsto \bar{\mathrm{F}}$ is a bijection $\mathrm{E}_n \to \mathrm{E}_m$. It follows $f_{2n+1} = f_n$.

With the same notations, assume n even, non-zero, say $n = 2k+2$, so that $2k+2 = 2m+a$. We have $a = 0$ or $a = 2$, so that $m = k+1$ or $m = k$. Now $\mathrm{F} \mapsto \bar{\mathrm{F}}$ is a bijection between E_n and the disjoint union of E_{k+1} and E_k. It follows $f_{2n+2} = f_{n+1} + f_{n+2}$.

We deduce $f_n = s_{n+1}$.

```
Lemma expe_odd n: natp n -> Nbexp(csucc (cdouble n)) = Nbexp n.
Lemma expe_even n: natp n ->
    Nbexp(cdouble (csucc n)) = Nbexp (csucc n) +c (Nbexp n).
Lemma expe_fusc n: natp n -> Nbexp n = fusc (csucc n).
```

Diagonals of the Pascal triangle. Set $b(n,k) = \binom{n}{k}$. The diagonal of the Pascal triangle is the set of all $b(n,k)$ where the sum $n+k$ is fixed, say m. If $m = 2q$ or $m = 2q+1$, there are q terms on the diagonal. We pretend that the sum is F_{m+1}, and the number of odd terms on the diagonal is s_{n+1}.

The key relation is the binomial formula (6.13). It says that an element of diagonal $m+2$ is the sum of an element of diagonal m and an element of the diagonal $m+1$. The first formula follows easily.

```
Lemma sum_diag_pascal n: natp n ->
  csumb (Nintc n) (fun k => binom (n -c k) k)  = Fib (csucc n).
```

Let's compute the binomial coefficient modulo two. We start with the following relation [apply (6.13) three times, and notice that $b(2n, 2k+1)$ appears twice]:

$$\binom{n+2}{k+2} = \binom{n}{k} + \binom{n}{k+2} \quad (\text{mod } 2).$$

We have $b(2n, 2k) = b(n,k)$ (by induction, using (6.13)) and $b(2n, 2k+1) = 0$ (also by induction). The two other formulas given here follows from (6.13).

$$\binom{2n}{2k} = \binom{n}{k}, \qquad \binom{2n}{2k+1} = 0, \quad \binom{2n+1}{2k+1} = \binom{2n+1}{2k} = \binom{n}{k} \quad (\text{mod } 2). \tag{9.6}$$

```
Lemma bin_mod2_rec1 n k : natp n -> natp k ->
  eqmod \2c (binom (csucc (csucc n)) (csucc (csucc k)))
  ((binom n (csucc (csucc k))) +c (binom n k)).
Lemma bin_mod2_rec2 n k: natp n -> natp k ->
  eqmod \2c (binom (cdouble n) (cdouble k)) (binom n k).
Lemma bin_mod2_prop1 n k: natp n -> natp k ->
  (binom (cdouble n) (csucc (cdouble k))) %%c \2c =  \0c.
Lemma bin_mod2_prop2 n k: natp n -> natp k ->
  eqmod \2c (binom (csucc (cdouble n)) (cdouble k)) (binom n k).
Lemma bin_mod2_prop3 n k: natp n -> natp k ->
  eqmod \2c (binom (csucc (cdouble n)) (csucc (cdouble k))) (binom n k).
```

Let q be a prime number, $q_i(n)$ and $r_i(n)$ the quotient and remainder in the division of n by q^i. The exponent of q in the decomposition into prime numbers of $n!$ is $\sum q_i$, so that so that the exponent of q in $b(n,k)$ is $\sum_i(q_i(n)-q_i(k)-q_i(n-k))$. Each term here is ≥ 0, so that q divides $b(n,k)$ if and only if at least one term is non-zero. One can restate this as: there is i such that $r_i(n) < r_i(k)$.

We show here: the binomial coefficient is even if and only if, for some i, we have $r_i(n) < r_i(k)$. We do not compute the exponent of two in the factorial, but use the previous relations. The key relation is the following: Assume $a = q2^i + r$ by Euclidean division. Then $2a = q2^{i+1} + 2r$ and $2a+1 = q2^{i+1}+2r+1$. Thus $r_{i+1}(2a) = 2r_i(a)$ and $r_{i+1}(2a+1) = 2r_i(a)+1$.

Assume n' is $2n$ or $2n+1$, k' is $2k$ or $2k+1$. We deduce that $r_i(n) < r_i(k)$ is equivalent to $r_{i+1}(n') < r_{i+1}(k')$, except when n' is even and k' is odd. In this case, it happens that $r_i(n) < r_i(k)$ is false for $i = 0$ as division by one is exact, and $r_{i+1}(n') < r_{i+1}(k')$ is true (the first remainder is zero, the second remainder is one). This agrees with the fact that $b(2n, 2k+1)$ is even. Otherwise, we proceed by induction, reducing the case (n', k') to the case (n, k).

```
Lemma rem_two_prop1 m i: natp m -> natp i ->
   (cdouble m) %%c \2c ^c (csucc i) = cdouble (m %%c \2c ^c i).
Lemma rem_two_prop2 m i: natp m -> natp i ->
   (csucc (cdouble m)) %%c \2c ^c (succ i) = csucc (cdouble (m %%c \2c ^c i)).
Lemma binom_evenP n k: natp n -> natp k ->  (* 75 *)
  (evenp (binom n k) <->
   exists2 i, natp i & n %%c (\2c ^c i) <c  k %%c (\2c ^c i)).
```

Let S_n denote the sum of $b(n-k,k)$ modulo two. Consider S_{2n+1} and split the sum according to whether k is even or odd. If k is odd, the binomial coefficient is even. Otherwise, the generic term is $b(2n+1-2k, 2k)$, this is $b(n-k,k)$, so that $\mathrm{S}_{2n+1} = \mathrm{S}_n$. Consider S_{n+2}. The even term is $b(2n+2-2k, 2k) = b(n+1-k, k)$, and the odd term is $b(2n+2-(2k+1), 2k+1) = b(n-k,k)$. Thus $\mathrm{S}_{2n+2} = \mathrm{S}_n + \mathrm{S}_{n+1}$. As $\mathrm{S}_0 = 1$, it follows that $\mathrm{S}_n = s_{n+1}$.

```
Definition sum_diag_pascal2 n :=
  csumb (Nintc n) (fun k => (binom (n -c k) k) %%c \2c).

Lemma sum_diag_pascal2_0: sum_diag_pascal2 \0c = \1c.
Lemma sum_diag_pascal_mod2_odd n (S := sum_diag_pascal2) : natp n ->
  S (csucc (cdouble n)) = S n.
Lemma sum_diag_pascal_mod2_even n (S := sum_diag_pascal2) : natp n ->
  S (csucc (csucc (cdouble n))) = S (csucc n) +c S n.
Lemma sum_diag_pascal_prop n: natp n ->
   (sum_diag_pascal2 n) = fusc (csucc n).

Lemma csum_fusc_row n: natp n ->
  csumb (interval_co Nat_order (\2c ^c n)  (\2c ^c (csucc n))) fusc
  = \3c ^c n.
```

Dijkstra [9] expresses the palindrome condition (9.4) as: “the value of the function fusc does not change if we invert in the binary representation of the argument all “internal” digits; for instance $s_{19} = s_{29}$”.

Consider a such that $p \leq a \leq 2p$. If a is a power of two, then $a = p$ or $a = 2p$, so that $a+b = 3p$ says that b is a power of two, and $s_a = s_b = 1$. Let’s exclude this case. Let $\mathrm{A}(u,v,w) = (2(2^v+u)+1)2^w$. We can uniquely write $a = \mathrm{A}(u,v,w)$ where $u < 2^v$. The digits of a are, starting with the most significant one: 1, followed by the v digits of u, padded with zero if necessary, followed by 1, followed by w zeros. The internal digits are those of u. Let u' be the number obtained by inverting these digits, and $b = \mathrm{A}(u', v', w')$. Since v' is the size of u', we have $v = v'$. On the other hand the value of s_b is independent of w'. The relation between u and u' is $u+u'+1 = 2^v$, it implies $u < 2^v$ and $u' < 2^v$.

Set $A = (2^v + u)$ and $B = (2^v + u')$. We have $s_a = F_A(1,1)$ and $s_b = F_B(1,1)$. Let's apply v times the rules of F. At stage k, we have $s_a = F_{A_k}(x_k, y_k)$ and a similar formula for s_b. In fact we have $s_b = F_{B_k}(y_k, x_k)$ (if A_k is even then B_k is odd). If $k = v$, then $A_k = B_k = 1$, and we conclude by $F_1(x, y) = x + y$.

```
Lemma fusci_one a b: natp a -> natp b -> Fusci \1c a b = a +c b.
Lemma fusci_palindrome_aux u u' v a b:
   natp a -> natp b -> natp u -> natp u' -> natp v ->
   csucc (u +c u') = \2c ^c v ->
   Fusci (\2c ^c v +c u) a b = Fusci (\2c ^c v +c u') b a.
Lemma fusc_palindrome_bis u u' v w w'
   (aux := fun u v w => csucc(cdouble (\2c ^c v +c u)) *c (\2c ^c w)):
   natp u -> natp u' -> natp v -> natp w ->  natp w' ->
   csucc (u +c u') = \2c ^c v ->
   fusc (aux u v w)  = fusc (aux u' v w').
```

Dijkstra [9] says: "The next property is more surprising. (At least, I think so.) Let us try to represent the pair (a, b) by the single value m, according to $a = s_{m+1}$ and $b = s_m$". Define $G_n(m) = F_n(s_{m+1}, s_m)$. Then

$$G_0(m) = s_m, \qquad G_{2n}(m) = G_n(2m), \qquad G_{2n+1}(m) = G_n(2m+1) \tag{9.7}$$

and $G_n(0) = s_n$.

```
Definition Fuscj n m := Fusci n (fusc (csucc m)) (fusc m).

Lemma fuscj_zero m: natp m -> Fuscj \0c m = fusc m.
Lemma fuscj_val n: natp n -> Fuscj n \0c = fusc n.
Lemma fuscj_even n m: natp n -> natp m ->
   Fuscj (cdouble n) m = Fuscj n (cdouble m).
Lemma fuscj_odd n m: natp n -> natp m ->
   Fuscj (csucc (cdouble n)) m = Fuscj n (csucc (cdouble m)).
Lemma fuscj_one n: natp n -> Fuscj \1c n = Fuscj \0c (csucc (cdouble n)).
```

Dijkstra concludes "the fusc-value does not change if we write the binary digits of the argument in reverse order". Example: from $G_{19}(0) = G_0(25)$ we deduce $s_{19} = s_{25}$. Let $r(n)$ be the number obtained by reverting the order of the bits, so that $r(19) = 25$. Then $s_a = s_{r(a)}$ follows from $G_a(0) = G_0(r(a))$. More generally, we have $G_a(b) = G_0(a2^{\ln b} + r(b))$.

```
Lemma fuscj_reverse n: natp n ->
  Fuscj n \0c = Fuscj \0c (base_two_reverse n).
```

The next rational number. There is an explicit function F such that

$$F(x_n) = x_{n+1}. \tag{9.8}$$

It is given by

$$F(x) = \frac{1}{1 + 2\lfloor x \rfloor - x}$$

Let's first note the curious result: if $T(z) = -1/z$, then $F(T(F(x))) = T(x)$ for $x > 0$ (set $y = F(x)$, so that $T(y) = x - (1 + 2\lfloor x \rfloor)$, and $\lfloor T(y) \rfloor = -1 - \lfloor x \rfloor$).

If what follows, we assume the argument of F to be positive. Since $\lfloor x\rfloor < x+1$, the denominator of F is $> \lfloor x\rfloor$, so that $F(x) > 0$. Assume $0 < x < 1$, let's say $x = a/(a+b)$ with $b > 0$. In this case we have $\lfloor x\rfloor = 0$, so

$$F(\frac{a}{a+b}) = \frac{a+b}{b} \tag{9.9}$$

Assume $x \geq 1$, say, $x = (a+b)/b$ with $b > 0$. Let r be the remainder in the division of a by b, and q the quotient. We have $\lfloor x\rfloor = q+1$, so that

$$F(\frac{a+b}{b}) = \frac{b}{b+c}, \qquad (c = (a+b) - 2R_b(a)) \tag{9.10}$$

Note that $a+b-2r = (a-r)+(b-r) = bq+(b-r)$. From $a \geq 0$ and $b > 0$ it follows $bq \geq 0$; from $r < b$ it follows $c > 0$. Thus: if $x <$ then $F(x) > 1$ and if $x \geq 1$ then $F(x) < 1$.

```
Definition rat_iterator x :=
   BQinv (\1q +q (BQdouble (BQ_of_Z(BQfloor x))) -q x).

Lemma QS_rati x: ratp x -> inc (rat_iterator x) BQ.

Lemma rati_0: rat_iterator \0q = \1q.
Lemma rati_1: rat_iterator \1q = \2hq.
Lemma rati_pos x: inc x BQp -> inc (rat_iterator x) BQps.
Lemma BQfloor_spec_sum a b:
  ratp a -> (exists2 c, intp c & b = BQ_of_Z c) ->
  BQ_of_Z (BQfloor (a +q b)) = BQ_of_Z (BQfloor a) +q b.
Lemma rati_neg x (f:= rat_iterator) (T:= fun x => BQinv (BQopp x)) (y := f x) :
   inc x BQps -> f (T y) = T x.
Lemma rati_lt1 a b (A := BQ_of_Z a)  (B := BQ_of_Z b):
  inc a BZp -> inc b BZps ->
  rat_iterator (A /q (A +q B)) = (A +q B) /q B.
Lemma rati_gt1 a b (c := (a +z b) -z \2z *z (a %%z b))
  (A := BQ_of_Z a) (B := BQ_of_Z b)  (C := BQ_of_Z c):
  inc a BZp -> inc b BZps ->
  (inc c BZps /\ rat_iterator ((A +q B) /q B) =  B /q (B +q C)).
```

Let's show (9.8). The result is obvious when x is even (case $x_n < 1$ and (9.9) applies). Otherwise, we use (9.10), and our property follows from: if r is the remainder in the division of s_n by s_{n+1} then $s_{n+2}+2r = (s_n+s_{n+1})$. If n is even, then $s_n < s_{n+1}$ and $r = s_n$; the conclusion is easy. Now $s_{2m+1} \bmod s_{2m}$ is $s_m + s_{m+1} \bmod s_m$, thus $s_{m+1} \bmod s_m$, and we conclude by induction.

```
Lemma fusc_rem n (A :=fusc n) (B :=  fusc (csucc n)) : natp n ->
   fusc (csucc (csucc n)) +c cdouble (A %%c B) = (A +c B).
Lemma rat_fusc n: natp n -> rat_iterator (fusc_quo n) = fusc_quo (csucc n).
```

Chapter 10

Real numbers

We define here the set of real numbers as the set of all Dedekind cuts of **Q**.

10.1 Definition and basic properties

10.1.1 Dedekind cuts

A *cut* in a totally ordered set E is a partition (A, B) of E such that every element of A is less than any element of B. Note that the cut is uniquely determined by A or by B, we shall use B in what follows. If $x \in A$ and $y < x$ then $y \in A$ so that A is an initial segment. Similarly, $x \in B$ and $x < y$ says $y \in B$. We can order the set of cuts by: $(A, B) \le (A', B')$ when $A \subset A'$ (this is the same as $B' \subset B$). This is a total ordering, and a complete lattice (the supremum corresponds to the intersection of B, the infimum to the union, the least element is $A = \emptyset$, the greatest element is $B = \emptyset$).

```
Definition or_cut r B :=
   sub B (substrate r) /\ (forall x y, inc x B -> glt r x y -> inc y B).
Definition or_cuts r := Zo (powerset (substrate r)) (or_cut r).
Definition or_cut_order r := opp_order (sub_order (or_cuts r)).

Lemma or_cutsP r B: inc B (or_cuts r) <-> (or_cut r B).
Lemma or_cut_osr r: order_on (or_cut_order r) (or_cuts r).
Lemma or_cut_tor r: total_order r -> total_order (or_cut_order r).
Lemma or_cut_gleP r x y:
   gle (or_cut_order r) x y <-> [/\ ( or_cut r x),  or_cut r y & sub y x].
Lemma or_cut_gle_least r : least (or_cut_order r) (substrate r) .
Lemma or_cut_gle_greatest r : greatest (or_cut_order r) emptyset.
Lemma or_cut_P r B : sub B (substrate r) ->
    (or_cut r B <-> segmentp r (substrate r -s B)).
Lemma or_cut_prop2 r B : order r ->  sub B (substrate r) ->
   (forall x y, inc x (substrate r -s B) -> inc y B -> glt r x y) ->
    or_cut r B.
Lemma or_cut_P2 r B : total_order r -> sub B (substrate r) ->
 (or_cut r B <-> forall x y, inc x (substrate r -s B) -> inc y B -> glt r x y).
Lemma or_cut_supinf r X:
  order r -> (forall x, inc x X -> or_cut r x) ->
  (or_cut r (union X) /\ or_cut r (intersection X)).
```

Let D be the set of cuts of E. Let $C_x =]x, \to[$ and $C'_x = [x, \to[$. If $x \in E$, then $B = C_x$ and $B = C'_x$ are in D. Both $x \mapsto C_x$ and $x \mapsto C'_x$ are strictly increasing injections of $E \to D$. If Y is another cut, then $C_x < Y$ is equivalent to $C'_x < Y$.

```
Lemma or_cut_segment r x : order r ->
   or_cut r (Zo (substrate r) (fun t => glt r x t)).
Lemma or_cut_segmente r x : order r ->
   or_cut r (Zo (substrate r) (fun t => gle r x t)).
Lemma or_cut_segment_cp r x y
  (X :=  Zo (substrate r) (fun t => glt r x t))
  (Y :=  Zo (substrate r) (fun t => glt r y t)):
  total_order r -> inc x (substrate r) -> inc y (substrate r) ->
  (glt r x y <-> glt (or_cut_order r) X Y).
Lemma or_cut_segmente_cp r x y
  (X :=  Zo (substrate r) (fun t => gle r x t))
  (Y :=  Zo (substrate r) (fun t => gle r y t)):
  total_order r -> inc x (substrate r) -> inc y (substrate r) ->
  (glt r x y <-> glt (or_cut_order r) X Y).
Lemma or_cut_segment_irrelevant r x Y
  (X :=  Zo (substrate r) (fun t => glt r x t))
  (X' :=  Zo (substrate r) (fun t => gle r x t)):
  order r ->  Y <> X -> Y <> X' ->
 (glt (or_cut_order r) X Y <-> glt (or_cut_order r) X' Y).
```

We consider here the question: is there a complete totally ordered lattice F, and an order preserving injection $i : E \to F$? Is there a least such F (modulo order isomorphism)? The answer to the first question is yes, it suffices to consider D. But D is not the least: the set D_1 formed of all cuts not of the form C'_x, and the set D_2 formed of all cuts not of the form C_x, are two candidates (they are isomorphic, but not to D).

On the other hand, consider some i and F. Let (A, B) be a cut, and $j(B)$ the infimum of $i\langle B\rangle$. If B has the form C'_x, then $j(B)$ is $i(x)$. If B if neither C_x nor C'_x, then $a \in A$ and $b \in B$ says $i(a) < j(B) < i(b)$. This says there is an injection $D_1 \to F$, so that D_1 is the least solution. [This should be easy to prove formally.]

10.1.2 Rational cuts

In what follows, we shall consider the set of cuts of **Q** and denote it $\overline{\mathbf{R}}$. For the reasons given above, we shall not consider the cuts of the form C'_x (but identify C'_x with C_x). The set C_x is called a *rational* cut, all other cuts are called *irrational*. The least element is denoted $-\infty$ and the greatest is $+\infty$. We shall define **R** as the set of all other cuts: thus $B \in \mathbf{R}$ if (1) B is non-empty, (2) B is different from **Q**, (3) whatever x and y, $x \in B$, $x < y$ implies $y \in B$; and (4) B has no least element. For an irrational cut, the complement of B has no greatest element.

```
Definition real_dedekind B :=
   [/\ sub B  BQ, nonempty B, B <> BQ,
      (forall x y, inc x B -> x <q y -> inc y B) &
      (forall x, inc x B -> exists2 y, inc y B & y <q x)].
Definition irrationalp B :=  real_dedekind B /\
  (forall x, inc x (BQ -s B) -> exists2 y, inc y (BQ -s B) & x<q y).
Definition rationalp x := real_dedekind x /\ ~ (irrationalp x).
Definition BR_of_Q x := Zo BQ (fun z => x <q z).
```

```
Lemma BR_of_Q_prop1 x: ratp x -> rationalp (BR_of_Q x).
Lemma BR_of_Q_prop2 X: rationalp X ->
  exists2 x, ratp x & X = BR_of_Q x.
```

We first consider the (partial) inverse of the canonical injection $\mathbf{Q} \to \mathbf{R}$ defined for rational numbers.

```
Definition BQ_of_R x := (select (fun y => x = BR_of_Q y) BQ).

Lemma BR_of_Q_inj1: {inc BQ &, injective BR_of_Q}.
Lemma BQ_of_R_prop x:  rationalp x ->
  x =  BR_of_Q (BQ_of_R x) /\ inc (BQ_of_R x) BQ.
Lemma BQ_of_R_prop2 x: ratp x -> BQ_of_R (BR_of_Q x) = x.
```

An example of an irrational cut is $\sqrt{2}$, this is the set B of all x such that $x > 0$ and $2 < x^2$. We have already seen that $x^2 = 2$ has no solution. Assume $a \in \mathbf{Q} - \mathrm{B}$ and $b \in \mathrm{B}$. We can find a' and b' such that $a < a' < \sqrt{2} < b' < b$ (this is short for $a < a'$, $a' \in \mathbf{Q} - \mathrm{B}$, $b' < b$ and $b' \in \mathrm{B}$). This is obvious by continuity[1] of the square function.

Let's first show how to obtain a'. We may assume $a > 0$, since otherwise $a' = 1$ is a solution. Assume $a = n/d$, and take $a' = (n + 1/4n)/d$. This is $a + 1/(4nd)$, so $a' > a$. We have $a'^2 = (n^2 + 1/2 + 1/(4n)^2)/d^2$. Note that $1/(4n)^2) \le 1/2$; this is equivalent to $2 \le (4n)^2$, and holds since $1 \le n$. Thus $a'^2 \le (n^2+1)/d^2$. Now, $a^2 < 2$ says $n^2 < 2d^2$, and since n and d are integers, we get $n^2 + 1 \le 2d^2$. We deduce $a'^2 \le 2$, thus $a'^2 < 2$.

The same argument can be used for b'. There is a better solution: Let $f(x) = (x^2+2)/(2x)$. We have $f(b) = b - (b^2-2)/2b$; so that if $b \in \mathrm{B}$ then $0 < f(b) < b$. Moreover, $f(b)^2 - 2 = [(b^2-2)/2b]^2$. In particular $f(b)^2 > 2$. Consider the sequence $x_{n+1} = f(x_n)$. This converges fast to $\sqrt{2}$ (here "fast" means that $\delta_n = x_n^2 - \sqrt{2}$ satisfies $\delta_{n+1} \approx C\delta_n^2$, and "converges" means: the set of all t such that $t \ge x_n$ for at least one n is B). One can construct a sequence that converges in $\mathbf{Q} - \mathrm{B}$ to $\sqrt{2}$. Let $g(x) = (x + f(x))/2$. If $x^2 < 2$ then $f(x) \ge x$, so $g(x) \ge x$. Let $z = g(x)$, $t = 2 - x^2$, then $z = (3x^2+2)/4x$, and $z^2 - 2 = (9t^2-16)/(4x)^2$. If $t < 1$ it follows $z^2 < 2$. Thus, if $z_{n+1} = g(z_n)$, and $z_0 > 1$, the sequence z_n is in $\mathbf{Q} - \mathrm{B}$ and increasing. Let z be the set of all t such that $z_n \le t$ for every n. This is some element of $\mathbf{R}$; we shall see below how to extend g to elements of $\mathbf{R}$, we then have $g(z) = z$ so that $z^2 = 2$, so that $z = \sqrt{2}$.

```
Definition BRsqrt2 := (Zo BQps (fun z => \2q <q z *q z)).
Lemma sqrt2_irrational: irrationalp BRsqrt2. (* 107 *)
```

10.1.3 Real numbers

From now on, we shall say "real number" instead of "cut". We define here some constants, 0_r, 1_r, 2_r, etc.

```
Definition BR := Zo (powerset BQ) real_dedekind.
Definition realp x := inc x BR.

Definition BR_zero := BR_of_Q \0q.
Definition BR_one := BR_of_Q \1q.
Definition BR_two := BR_of_Q \2q.
Definition BR_three := BR_of_Q \3q.
```

[1] We shall not define a topology on $\mathbf{Q}$, see below

```
Definition BR_four := BR_of_Q \4q.
Definition BR_mone := BR_of_Q \1mq.
Definition BR_half := BR_of_Q \2hq.

Notation "\0r" := BR_zero.
Notation "\1r" := BR_one.
Notation "\2r" := BR_two.
Notation "\3r" := BR_three.
Notation "\4r" := BR_four.
Notation "\1mr" := BR_mone.
Notation "\2hr" := BR_half.

Lemma RS0 : realp \0r.
Lemma RS1 : realp \1r.
Lemma RS2 : realp \2r.
```

By definition, a real number x has no lower bound. However, whenever $\delta > 0$ is a rational number, we can find $y \in x$ such that $y - \delta \not\in x$ (let $a \in x$, $b \not\in x$, so that $b < a$; let $n > (a - b)/\delta$ be an integer, and consider the least k such that $a - k\delta \not\in x$).

```
Lemma BR_P x: realp x <-> real_dedekind x.
Lemma BRi_sQ x y: realp x -> sub x BQ.
Lemma BRi_segment x y z :realp x -> inc y x -> y <q z -> inc z x.
Lemma BRi_no_lowbound x y: realp x -> inc y x -> exists2 z, inc z x & z <q y.
Lemma BRi_lowbound x d: realp x -> inc d BQps ->
   exists2 y, inc y x & forall z, inc z x -> y -q d <q z.
Lemma BR_rational_dichot x: realp x ->
    rationalp x \/ irrationalp x.
Lemma RS_of_Q x: ratp x -> realp (BR_of_Q x).
```

We shall say that a real is positive if all its members are positive. So, we define $\mathbf{R}_-$, $\mathbf{R}_+$, $\mathbf{R}_+^*$ and $\mathbf{R}_-^*$.

```
Definition BRp := Zo BR (fun z => sub z BQp).
Definition BRps := BRp -s1 \0r.
Definition BRms := BR -s BRp.
Definition BRm := BR -s BRps.

Lemma BRp_sBR : sub BRp BR.
Lemma BRps_sBR : sub BRps BR.
Lemma BRms_sBR : sub BRms BR.
Lemma BRm_sBR : sub BRm BR.
Lemma BRps_sBRp : sub BRps BRp.
Lemma BRms_sBRm : sub BRms BRm.

Lemma RmS0: inc \0r BRm.
Lemma RpS0: inc \0r BRp.

Lemma BR_i0P x: realp x <-> (inc x BRms \/ inc x BRp).
Lemma BR_i1P x: realp x  <-> [\/ x = \0r, inc x BRps | inc x BRms].
Lemma BR_i2P x: realp x <-> (inc x BRps \/ inc x BRm).
Lemma BR_di_neg_pos x: inc x BRms -> inc x BRp -> False.
Lemma BR_di_pos_neg x: inc x  BRps -> inc x BRm -> False.
Lemma BR_di_neg_spos x: inc x BRms -> inc x BRps -> False.
Lemma BRms_nz  x: inc x BRms -> x <> \0r.
Lemma BRps_nz  x: inc x BRps -> x <> \0r.
```

```
Lemma BRps_iP x: inc x BRps <-> inc x BRp /\ x <> \0r.
Lemma BRms_iP x: inc x BRms <-> inc x BRm /\ x <> \0r.
```

10.1.4 Order

The previous study says that **R** has a natural ordering, and the canonical injection $\mathbf{Q} \to \mathbf{R}$, $x \mapsto \mathrm{C}_x$, is strictly increasing.

```
Definition BR_order := opp_order (sub_order BR).
Definition BR_le x y := [/\ realp x, realp y  & sub y x].
Definition BR_lt x y :=  BR_le x y /\ x <> y.


Notation "x <=r y" := (BR_le x y) (at level 60).
Notation "x <r y" := (BR_lt x y) (at level 60).

Lemma BR_of_Q_inj1: {inc BQ &, injective BR_of_Q}.
Lemma BR_of_Q_inj: injection_prop (Lf (BR_of_Q) BQ BR) BQ BR.
Lemma BR_osr: order_on BR_order BR.
Lemma BR_tor: total_order BR_order.
Lemma BR_gleP x y: gle BR_order x y <-> x <=r y.
Lemma rle_cQ x y: ratp x -> ratp y ->
  (x <=q y <-> (BR_of_Q x <=r BR_of_Q y)).
Lemma rlt_cQ x y: ratp x -> ratp y ->
  (x <q y <-> (BR_of_Q x <r BR_of_Q y)).
```

Basic properties of order.

```
Lemma rleR a: realp a -> a <=r a.
Lemma rleA x y: x <=r y -> y <=r x -> x = y.
Lemma rleT y x z:  x <=r y -> y <=r z -> x <=r z.
Lemma rleNgt a b: a <=r b -> ~(b <r a).
Lemma rlt_leT b a c: a <r b -> b <=r c -> a <r c.
Lemma rle_ltT b a c: a <=r b -> b <r c -> a <r c.
Lemma rlt_ltT b a c: a <r b -> b <r c -> a <r c.

Lemma rleT_ee a b:  realp a -> realp b -> a <=r b  \/ b <=r a.
Lemma rleT_ell a b: realp a -> realp b -> [\/ a = b, a <r b | b <r a].
Lemma rleT_el a b: realp a -> realp b -> a <=r b  \/ b <r a.
Lemma rleT_el a b: realp a -> realp b -> a <=r b  \/ b <r a.
```

Recall that the union or intersection of cuts is a cut. This means that $\overline{\mathbf{R}}$ is a complete lattice. However, **R** has no greatest element. We can prove a stronger statement: for any $x \in \mathbf{R}$, there is a natural integer n such that $x < n$ (let $y \in x$ be any rational number, $n = \lfloor y \rfloor + 1$; then $x < y$ and $y < n$; if $n < 0$ we can take zero instead). We say that **R** is *Archimedean.* Every nonempty bounded subset X of **R** has a supremum and an infimum (intersection and union; the intersection could be of the form C'_x, in that case the supremum is C_x).

```
Lemma BR_le_aux1 x a: realp x -> (exists2 b,  inc b x & b <q a) ->
    x <r (BR_of_Q a).
Lemma BR_le_aux2 x a: realp x -> inc a x -> x <r (BR_of_Q a).
Lemma BR_le_aux3 x a: realp x -> ratp a -> ~(inc a x) ->
  (BR_of_Q a) <=r x.
Lemma BR_le_aux4 x: realp x -> (inc \0q x <-> x <r \0r).
```

```
Theorem BR_archimedean  x: realp x ->
    exists2 n, natp n & x <r (BR_of_Q (BQ_of_nat n)).
Lemma BR_no_greatest x : ~ (greatest BR_order x).
Lemma BR_no_least x : ~ (least BR_order x).
Lemma BR_sup_exists X: sub X BR -> nonempty X ->
  bounded_above BR_order X -> has_supremum BR_order X.
Lemma BR_inf_exists X: sub X BR -> nonempty X ->
  bounded_below BR_order X -> has_infimum BR_order X.
```

Since no element of $\mathbf{R}$ is $\mathbf{Q}_+$, the condition $x \in \mathbf{R}_+$ is equivalent to $x \in \mathbf{R}$ and $x \subset \mathbf{Q}_+^*$. It happens that $\mathbf{Q}_+^*$ is C_0, i.e., 0_r, so that $x \in \mathbf{R}_+$ is equivalent to $0 \le x$.

```
Lemma BRzero_prop:  \0r = BQps.
Lemma BR_hi_Qps x: inc x BRp -> sub x BQps.
Lemma BR_hi_Qps' x: inc x BRps -> ssub x BQps.
Lemma BRcompare_zero x:  inc x BRps ->
   exists2 y, inc y BQps & BR_of_Q y <r x.
Lemma BRcompare_zero' e: inc e BQps ->
   exists2 e', inc e' BRps & e' <r (BR_of_Q e).

Lemma rle0xP x: \0r <=r x <-> inc x BRp.
Lemma rlt0xP x:  \0r <r x <-> inc x BRps.
Lemma rgt0xP x:  x <r \0r <-> inc x BRms.
Lemma rge0xP x:  x <=r \0r <-> inc x BRm.
Lemma rle_par1 x y: inc x BRps -> inc y BRm ->  y <r x.
Lemma rle_par2 x y: inc x BRp -> inc y BRms ->  y <r x.
Lemma rle_par3 x y: inc x BRp -> inc y BRm ->  y <=r x.
Lemma infimum_BRp: infimum BR_order BRp = \0r.
```

The map $x \mapsto C_x$ respects the partitions of $\mathbf{Q}$ and $\mathbf{R}$. For instance $2_r \in \mathbf{R}_+^*$.

```
Lemma RpsS_of_Q x: inc x BQps -> inc (BR_of_Q x) BRps.
Lemma RmsS_of_Q x: inc x BQms -> inc (BR_of_Q x) BRms.
Lemma RpS_of_Q x: inc x BQp -> inc (BR_of_Q x) BRp.
Lemma RmS_of_Q x: inc x BQm -> inc (BR_of_Q x) BRm.

Lemma RpsS1 : inc \1r BRps.
Lemma RpsS2 : inc \2r BRps.
Lemma RmsSm1 : inc \1mr BRms.
Lemma RSm1 : realp \1mr.
Lemma RpsSh2 : inc \2hr BRps.
Lemma RSh2 : realp \2hr.
Lemma RpsS4 : inc \4r BRps.
Lemma RS4 : realp \4r.
Lemma RpsS3 : inc \3r BRps.
Lemma RS3 : realp \3r.
```

10.2 Field Structure

10.2.1 Opposite

If x is a cut, say (A, B), we define its opposite to be the cut $(-B, -A)$ where $-A$ is the set of all elements of the form $-z$ for $z \in A$. If x is an irrational number, this will be the opposite of x. On the other hand, if x is C_t its opposite is C_{-t}.

With this definition, the opposite of a real number x, denoted $-x$ will be real (rational if x is rational, irrational otherwise).

```
Definition BRopp x := Yo (rationalp x)
   (BR_of_Q (BQopp (BQ_of_R x)))  (fun_image (BQ -s x) BQopp).
Lemma BRopp_Q x: ratp x -> BRopp (BR_of_Q x) = BR_of_Q (BQopp x)
Lemma BRopp_irrational x: irrationalp x ->
   BRopp x = (fun_image (BQ -s x) BQopp).
Lemma RSo x: realp x -> realp (BRopp x).
Lemma RSIo x: irrationalp x ->  irrationalp (BRopp x).
Lemma BRopp_K x: realp x -> BRopp (BRopp x) = x.
Lemma BRopp_inj a b: realp a -> realp b -> BRopp a = BRopp b -> a = b.
Lemma BRopp_fb: bijection (Lf BRopp BR BR).
Lemma rle_opp x y: x <=r y ->  (BRopp y) <=r  (BRopp x).
Lemma rlt_opp x y: x <r y -> (BRopp y) <r (BRopp x).
Lemma rle_oppP x y: realp x -> realp y ->
   ((BRopp y) <=r  (BRopp x) <-> x <=r y).
Lemma rlt_oppP x y: realp x -> realp y ->
   ((BRopp y) <r  (BRopp x) <-> x <r y).
Lemma rle_opp_iso:
  order_isomorphism (Lf BRopp BR BR) BR_order (opp_order BR_order).
```

If X is a non-empty set, bounded above, it has a supremum x. If Y is the set of opposites, it has an infimum y, and $y = -x$.

```
Lemma BR_supremum_opp X a (x := supremum BR_order X):
   nonempty X -> (forall t, inc t X -> t <=r a) ->
   x = BRopp (infimum BR_order (fun_image X BRopp)).
```

Opposite and partition.

```
Lemma BRopp_0 : BRopp \0r = \0r.
Lemma BRopp_1 : BRopp \1r = \1mr.
Lemma BRopp_m1 : BRopp \1mr = \1r.

Lemma BRopp_positive1 x: inc x BRps -> inc (BRopp x) BRms.
Lemma BRopp_positive2 x: inc x BRp -> inc (BRopp x) BRm.
Lemma BRopp_negative1 x: inc x BRms -> inc (BRopp x) BRps.
Lemma BRopp_negative2 x: inc x BRm -> inc (BRopp x) BRp.
Lemma BRopp0_bis x: realp x ->  (x = \0r <-> BRopp x = \0r).
```

10.2.2 Addition

We define $x+y$ as the set of sums of elements of x and y. This is easily seen to be real if x and y are real. The operation is trivially associative and commutative. Assume $x = \mathrm{C}_a$. In this case, $x+y$ is the set of all $a+b$, with $b \in y$ (note: if $b \in y$, there is $b' \in y$ with $b' < b$, so that $a+b = (a+b-b')+b'$ where $a+b-b' \in \mathrm{C}_a$). If follows that the sum of a rational and an irrational is irrational while the sum of two rationals is rational ($\mathrm{C}_a + \mathrm{C}_b = \mathrm{C}_{a+b}$). Note that $x - x = 0$: we have to show that every $\delta > 0$ (an element of zero) is $a-b$, where $a \in x$ and $b \not\in x$; take a as above, $b = a - \delta$.

```
Definition BRsum x y :=
   union (fun_image x (fun z => (fun_image y (fun t => z +q t)))).
```

```
Notation "x +r y" := (BRsum x y) (at level 50).

Lemma BR_sump x y:
   forall a, inc a (x +r y) <->
    exists2 z, inc z x & exists2 t, inc t y & a = z +q t.
Lemma BRsumC x y: x+r y = y +r x.
Lemma BRsumA x y z: realp x -> realp y -> realp z ->
  x  +r (y +r z) = (x +r y) +r z.
Lemma BRsum_AC x y z: realp x -> realp y -> realp z ->
  (x +r y) +r z = (x +r z) +r y.
Lemma BRsum_CA x y z: realp x -> realp y -> realp z ->
  x +r (y +r z) = y +r (x +r z).
Lemma BRsum_ACA a b c d: realp a -> realp b -> realp c -> realp d ->
    (a +r b) +r (c +r d) = (a +r c) +r (b +r d).
Lemma RSs x y: realp x -> realp y -> realp (x +r y).
Lemma BR_sumQ_aux x y: ratp x -> realp y ->
   (BR_of_Q x) +r y = fun_image y (fun z => x +q z).
Lemma BR_sumQ_aux1 x y: rationalp x -> irrationalp y ->
  irrationalp (x +r y).
Lemma BRsum_cQ x y: ratp x ->  ratp y ->
  BR_of_Q x +r BR_of_Q y = BR_of_Q (x +q y).
Lemma BR_plus21: (\2r +r \1r) = \3r.
Lemma BR_plus31: (\3r +r \1r) = \4r.

Lemma BRsum_opp_r x: realp x ->  x +r (BRopp x) = \0r.
Lemma BRsum_opp_l x: realp x -> (BRopp x) +r x = \0r.
Lemma BRsum_0l x: realp x -> \0r +r x  = x.
Lemma BRsum_0r x: realp x -> x +r \0r = x.
Lemma BRsum_11 : \1r +r \1r = \2r.

Lemma BRsum_2p4 a b c d:
  realp a -> realp b -> realp c -> realp d ->
  (a +r b) +r (c +r d) = (a +r c) +r (b +r d).
Lemma BRsum_opp_rev a b: realp a -> realp b -> a +r b = \0r ->
   a = BRopp b.
Lemma BRoppD x y: realp x -> realp y ->
  BRopp (x +r y) = (BRopp x) +r (BRopp y).
```

The sum of two positive numbers is trivially positive.

```
Lemma RpS_sum x y: inc x BRp -> inc y BRp -> inc (x +r y) BRp.
Lemma RpsS_sum_r x y: inc x BRp -> inc y BRps -> inc (x +r y) BRps.
Lemma RpsS_sum_l x y: inc x BRps -> inc y BRp ->  inc (x +r y) BRps.
Lemma RpsS_sum_rl x y: inc x BRps -> inc y BRps -> inc (x +r y) BRps.

Lemma RmsS_sum_rl x y: inc x BRms -> inc y BRms -> inc (x +r y) BRms.
Lemma RmsS_sum_r x y: inc x BRm -> inc y BRms ->  inc (x +r y) BRms.
Lemma RmsS_sum_l x y: inc x BRms -> inc y BRm -> inc (x +r y) BRms.
Lemma RmS_sum x y: inc x BRm -> inc y BRm -> inc (x +r y) BRm.
```

10.2.3 Difference

The following properties are easy.

```
Definition BRdiff x y := x +r (BRopp y).
```

```
Notation "x -r y" := (BRdiff x y) (at level 50).

Lemma RS_diff x y: realp x -> realp y -> realp (x -r y).
Lemma BRdiff_diff a b c: realp a -> realp b -> realp c ->
   a -r (b -r c)  = (a -r b) +r c.
Lemma BRdiff_diff2 a b c: realp a -> realp b -> realp c ->
  (a -r b) -r c = a -r (b +r c).

Section BQdiffProps5.
Variables (x y z: Set).
Hypotheses (xr: realp x)(yr: realp y)(zr: realp z).

Lemma BRdiff_sum:  (x +r y) -r x = y.
Lemma BRsum_diff: x +r (y -r x) = y.
Lemma BRdiff_xx :  x -r x = \0r.
Lemma BRdiff_0r: x -r \0r = x.
Lemma BRdiff_0l: \0r -r x = BRopp x.
Lemma BRdiff_sum_simpl_l:  (x +r y) -r  (x +r z) = y -r z.
Lemma BRdiff_sum_comm: (x +r y) -r z = (x -r z) +r y.
Lemma BRoppB: BRopp (x -r y) = y -r x.
End BQdiffProps5.

Section BQdiffProps6.
Variables (x y z: Set).
Hypotheses (xr: realp x)(yr: realp y)(zr: realp z).

Lemma BRsum_diff_ea: x = y +r z -> z = x -r y.
Lemma BRdiff_xx_rw:  x -r y = \0r -> x = y.
Lemma BRdiff_sum_simpl_r:  (x +r z) -r (y +r z) = x -r y.
Lemma BRsum_eq2r:  x +r z = y +r z -> x = y.
Lemma BRsum_eq2l:  x +r y = x +r z -> y = z.
End  BQdiffProps6.

Lemma BRdiff_diff_simp a b: realp a -> realp b ->  a -r (a -r b) = b.
Lemma BRdiff_cQ x y: ratp x ->  ratp y ->
  BR_of_Q x -r BR_of_Q y = BR_of_Q (x -q y).
```

Addition and comparison. Note that $a+c \le b+c$ is equivalent to $a \le b$. The proof is the following. Assume $t \in b$. There is $t' \in b$ such that $t' < t$; let $\delta = t' - t$, and $y \in c$ such that $y - \delta \not\in c$. Assume that $b+c$ is a subset of $a+c$; since $t'+q \in b+c$, it follows $t = u+(v-y)$ for some $u \in a$ and $v \in c$. By definition of y we get $u < t$, so $t \in a$.

```
Lemma BRsum_le2r a b c: realp a -> realp b -> realp c ->
    (a +r c <=r b +r c <-> a <=r b).
Lemma BRsum_le2l a b c: realp a -> realp b -> realp c ->
  ((c +r a) <=r (c +r b) <-> a <=r b).
Lemma rle_diffP a b: realp a -> realp b ->  (a <=r b <-> inc (b -r a) BRp).
Lemma rle_diffP1 a b: realp a -> realp b ->
  (a <=r b <-> \0r <=r  (b -r a)).
Lemma rle_diffP2 a b: realp a -> realp b ->
  (a <=r b <-> inc (a -r b) BRm).
Lemma rlt_diffP a b: realp a -> realp b ->
  (a <r b <-> inc (b -r a) BRps).
Lemma rlt_diffP1 a b: realp a -> realp b ->
```

```
  (\0r  <r (b -r a) <-> a <r b).
Lemma rlt_diffP2 a b: realp a -> realp b ->
  (a <r b <-> inc (a -r b) BRms).
Lemma rgt_diffP  a b: realp a -> realp b -> (a -r b <r \0r <-> a <r b).
Lemma BRsum_lt2l a b c: realp a -> realp b -> realp c ->
  (c +r a <r c +r b <-> a <r b).
Lemma BRsum_lt2r a b c: realp a -> realp b -> realp c ->
  (a +r c <r b +r c <-> a <r b).
```

More lemmas.

```
Lemma BRsum_Mlele a b c d: a <=r c ->  b <=r d -> (a +r b) <=r (c +r d).
Lemma BRsum_Mlelt a b c d: a <=r c -> b <r d -> (a +r b) <r (c +r d).
Lemma BRsum_Mltle a b c d: a <r c -> b <=r d -> (a +r b) <r (c +r d).
Lemma BRsum_Mltlt a b c d: a <r c -> b <r d -> (a +r b) <r (c +r d).
Lemma BRsum_Mlege0 a c d: a <=r c ->  \0r <=r d -> a <=r (c +r d).
Lemma BRsum_Mlegt0 a c d: a <=r c ->  \0r <r d -> a <r (c +r d).
Lemma BRsum_Mltge0 a c d: a <r c ->  \0r <=r d -> a <r (c +r d).
Lemma BRsum_Mltgt0 a c d: a <r c ->  \0r <r d -> a <r (c +r d).
Lemma BRsum_Mlele0 a b c : a <=r c ->  b <=r \0r -> (a +r b) <=r c.
Lemma BRsum_Mlelt0 a b c : a <=r c ->  b <r \0r -> (a +r b) <r c.
Lemma BRsum_Mltle0 a b c : a <r c ->  b <=r \0r -> (a +r b) <r c.
Lemma BRsum_Mltlt0 a b c : a <r c ->  b <r \0r -> (a +r b) <r c.
Lemma BRsum_Mp a b: realp a -> inc b BRp ->  a <=r (a +r b).
Lemma BRsum_Mps a b: realp a -> inc b BRps ->  a <r (a +r b).
Lemma BRsum_Mm a b: realp a -> inc b BRm ->  (a +r b) <=r a.
Lemma BRsum_Mms a b: realp a -> inc b BRms ->  (a +r b) <r a.
Lemma BRdiff_lt1P a b c: realp a -> realp b -> realp c ->
  (a -r b <r c <-> a -r c <r  b).
Lemma BRdiff_le1P a b c: realp a -> realp b -> realp c ->
  (a -r b <=r c <-> a -r c <=r  b).
Lemma BRdiff_lt2P a b c: realp a -> realp b -> realp c ->
  (c <r a -r b <-> b <r a -r c).
Lemma BRdiff_le2P a b c: realp a -> realp b -> realp c ->
  (c <=r a -r b <-> b <=r a -r c).
```

10.2.4 Multiplication

We define the product xy of two positive real numbers as the set of all products of elements of x and y, and extend this operation to the whole set of real numbers. The product of two strictly positive reals is strictly positive. The product is compatible with opposite. In particular the product of two real numbers is real.

```
Definition BRprod_aux x y :=
   union (fun_image x (fun z => (fun_image y (fun t => z *q t)))).

Definition BRprod x y:=
  Yo (x = \0r) \0r (Yo (inc x BRps)
     (Yo (y = \0r) \0r (Yo (inc y BRps) (BRprod_aux x y)
            (BRopp (BRprod_aux x (BRopp y)))))
   (Yo (y = \0r) \0r (Yo (inc y BRps) (BRopp (BRprod_aux (BRopp x) y))
            (BRprod_aux (BRopp x) (BRopp y))))).
Notation "x *r y" := (BRprod x y) (at level 40).
```

```
Fact BR_prod_auxP x y a:
   inc a (BRprod_aux x y) <->
    exists2 z, inc z x & exists2 t, inc t y & a = z *q t.
Lemma BRprod_auxC x y: (BRprod_aux x y) = (BRprod_aux y x).
Lemma BRprodC x y:  BRprod x y = BRprod y x.
Lemma BRprod_0l x:  \0r *r x = \0r.
Lemma BRprod_0r x: x *r \0r = \0r.

Lemma BR_pos_prop x:
   inc x BRps <-> (realp x /\ exists2 y, inc y BQps & ~ inc y x).
Lemma BR_prod_aux1 x y : inc x BRps -> inc y BRps ->
  (x *r y) =  (BRprod_aux x y).
Lemma RpsS_prod x y : inc x BRps -> inc y BRps -> inc (x *r y) BRps.
Lemma RmsuS_prod x y : inc x BRms -> inc y BRms -> inc (x *r y) BRps.
Lemma RpmsS_prod x y : inc x BRps -> inc y BRms -> inc (x *r y) BRms.
Lemma RpS_prod x y: inc x BRp -> inc y BRp -> inc (x *r y) BRp.
Lemma RmuS_prod x y: inc x BRm -> inc y BRm -> inc (x *r y) BRp.
Lemma RpmS_prod x y: inc x BRp -> inc y BRm -> inc (x *r y) BRm.
Lemma RSp x y: realp x -> realp y -> realp (x *r y).

Lemma BRopp_prod_r x y: realp x -> realp y ->
  BRopp (x *r y) = x *r (BRopp y).
Lemma BRopp_prod_l x y: realp x -> realp y ->
  BRopp (x *r y) = (BRopp x) *r y.
Lemma BRprod_opp_comm x y: realp x -> realp y ->
  x *r  (BRopp y) =  (BRopp x) *r y.
Lemma BRprod_opp_opp x y: realp x -> realp y ->
  (BRopp x) *r (BRopp y) = x  *r y.
```

As in the case of the sum, the product simplifies when one argument is rational. In particular $1 \cdot x = x$. The product of a non-zero rational by an irrational is irrational. The product on **R** is compatible with that on **Q**. It is associative and distributive. In the case of associativity, we show it (in case arguments are positive) as in the case of addition then take opposites. Let's now show : $x(y+z) = xy + xz$. The result is trivial if one quantity is zero. By taking opposites, we are reduced to the case $x > 0$. Assume first all quantities positive. We must show that a subset of **Q** is equal to another one. One implication is trivial. Assume that a and a' are in x, b and c are in y and z, we must show $ab + a'c \in x(y+z)$. This quantity is $a(b+ca'/a)$ or $a'(ba/a'+c)$. Depending of the size of a and a', ca'/a is in z or ba/a' is in z (note: $x > 0$ implies that x is a subset of $\mathbf{Q}_+^*$). By taking opposites, the result is true if $y < 0$ and $z < 0$. Consider the case where exactly one of these is < 0, by symmetry we may assume its is z. We have to show $x(y+z) - xz = xy$ or $x(y+z) - xy = xz$. In at least one formula all quantities (but x) have the same sign.

Let's show that the square of $\sqrt{2}$ is 2. Given a rational number $t > 2$; we have to find u and v such that $t = uv$, both u and v being in $\sqrt{2}$, i.e., $u^2 > 2$ and $v^2 > 2$. Write $t = a/b$ as a quotient of two integers. Set $d = 2a+1$. We have $d^2 > 2ad + 1$. We have $(a-2b)b \geq 1$ since both factors are strictly positive integers. We deduce $abd^2 - 2ad - 1 > 2(bd)^2$. Let n be the integer such that $n^2 \leq abd^2 < (n+1)^2$. Let c be n considered in **Q**. We have $c/(bd) \leq (ad)/c$, by the first inequality. Since $2b < a$, the first inequality also shows $c \leq ad$. The second inequality gives $abd^2 - (2ad+1) < c^2$. It follows $2(bd)^2 < c^2$. Let $u = c/(bd)$. We get $2 < u^2$. Let $v = (ad)/c$; since $u \leq v$, it follows $2 < v^2$.

```
Lemma BR_prodQ_aux x y: inc x BQps -> inc y BRps ->
```

```
   (BR_of_Q x) *r y = fun_image y (fun z => x *q z).
Lemma BRprod_1l x: realp x -> \1r *r x = x.
Lemma BRprod_1r x: realp x ->  x *r \1r  = x.
Lemma BRprod_m1r x: realp x ->  x *r \1mr = BRopp x.
Lemma BRprod_m1l x: realp x -> \1mr *r x  = BRopp x.
Lemma BR_prodQ_aux1 x y: x <> \0r -> rationalp x -> irrationalp y ->
  irrationalp (x *r y).
Lemma BRprod_cQ x y: ratp x ->  ratp y ->
  BR_of_Q x *r BR_of_Q y = BR_of_Q (x *q y).
Lemma BRprodA x y z: realp x -> realp y -> realp z ->
  x *r (y *r z) = (x *r y) *r z.
Lemma BRprod_2p4 a b c d:
  realp a -> realp b -> realp c -> realp d ->
  (a *r b) *r (c *r d) = (a *r c) *r (b *r d).
Lemma BRprod_AC x y z: realp x -> realp y -> realp z ->
  (x *r y) *r z = (x *r z) *r y.
Lemma BRprod_CA x y z: realp x -> realp y -> realp z ->
  x *r (y *r z) = y *r (x *r z).
Lemma BRprod_ACA a b c d: realp a -> realp b -> realp c -> realp d ->
    (a *r b) *r (c *r d) = (a *r c) *r (b *r d).
Lemma BRprod_CA x y z: realp x -> realp y -> realp z ->
  z *r (x *r y) = y *r (x *r z).
Lemma BRprodDr x y z: realp x -> realp y -> realp z  ->
   x *r (y +r z) = (x *r y) +r (x *r z).
Lemma BRprodDl x y z: realp x -> realp y -> realp z  ->
   (y +r z) *r x = (y *r x) +r (z *r x).
Lemma BRprodBr x y z: realp x -> realp y -> realp z  ->
   x *r (y -r z) = (x *r y) -r (x *r z).
Lemma BRprodBl x y z: realp x -> realp y -> realp z  ->
  (y -r z) *r x =  (y *r x) -r  (z *r x).
Lemma BRprod_nz x y: realp x -> realp y ->
  x <> \0r -> y <> \0r -> x *r y <> \0r.
Lemma BRprod_nz_bis x y: realp x -> realp y ->
  (x *r y = \0r) -> x = \0r \/ y = \0r.
```

Product and comparison. These results are trivial.

```
Lemma BRprod_Mlege0 a b c: inc c BRp -> a <=r b -> (a *r c)  <=r (b *r c).
Lemma BRprod_Mltgt0 a b c: inc c BRps -> a <r b -> (a *r c) <r (b *r c).
Lemma BRprod_Mlele0 a b c: inc c BRm -> a <=r b -> (b *r c)  <=r (a *r c).
Lemma BRprod_Mltlt0 a b c: inc c BRms -> a <r b -> (b *r c) <r (a *r c).
Lemma BRprod_Mpp  b c: inc b BRp -> \1r <=r c -> b <=r (b *r c).
Lemma BRprod_Mlepp a b c: inc b BRp -> \1r <=r c -> a <=r b -> a <=r (b *r c).
Lemma BRprod_Mltpp a b c: inc b BRp -> \1r <=r c  -> a <r b -> a <r (b *r c).
Lemma BRprod_Mlelege0 a b c d: inc b BRp -> inc c BRp ->
  a <=r b -> c <=r  d -> (a *r c)  <=r (b *r d).
Lemma BRprod_Mltltgt0 a b c d: inc b BRps -> inc c BRps ->
  a <r b -> c <r  d -> (a *r c) <r (b *r d).
Lemma BRprod_Mltltge0 a b c d: inc a BRp -> inc c BRp ->
  a <r b -> c <r  d -> (a *r c)  <r (b *r d).
Lemma BRprod_ple2r a b c: realp a -> realp b -> inc c BRps ->
  ((a *r c)  <=r (b *r c) <-> a <=r b).
Lemma BRprod_plt2r a b c: realp a -> realp b ->  inc c BRps ->
  ((a *r c) <r (b *r c) <-> a <r b).
Lemma BRprod_mle2r a b c: realp a -> realp b -> inc c BRms ->
  ((b *r c)  <=r (a *r c) <-> a <=r b).
```

```
Lemma BRprod_mlt2r a b c: realp a -> realp b ->  inc c BRms ->
  ((b *r c)  <r (a *r c) <-> a <r b).
```

We study here the function $z \mapsto z^2$.

```
Definition BRsquare x := x *r x.

Lemma RpS_square x: realp x -> inc (BRsquare x) BRp.
Lemma BRsquare_mon1 x y:
  inc x BRp -> inc y BRp -> x <=r y -> BRsquare x <=r BRsquare y.
Lemma BRsqrt_unique x: inc x BRp ->
  singl_val2 (inc^~ BRp) (fun z => x = BRsquare z).

Lemma BRsquare_mon2 x y:
Lemma BRsqrt2_prop: inc  BRsqrt2 BRps /\ BRsquare  BRsqrt2 = \2r. (* 142 *)
```

10.2.5 Inverse and division

The definition of inverse is a bit complicated. We first consider the case where x is rational, and define the inverse of C_x to be $C_{1/x}$. The inverse of $-x$ will be $-x^{-1}$, so it suffices to consider the case of a positive irrational number x. If Y is the set of all $1/x$ for $x \in X$, then x^{-1} is $\mathbf{Q}_+^* - Y$ (note that $Y \subset \mathbf{Q}_+^*$ and if $0 < z < y$ and $y \in Y$, then $z \in Y$).

The inverse of a rational number C_x is $C_{1/x}$, the inverse of an irrational number is irrational. The non-trivial property given here is $x \cdot x^{-1} = 1$, for non-zero x. It suffices to show it for positive irrational numbers. We must show: if $t > 1$, then $t = a/b$ for some $a \in x$ and $b \not\in x$. Since $x > 0$, there is $w \notin x$ with $w > 0$. Let $\delta = w(1 - 1/t)$; there is a such that $a \in x$ and $a - \delta \not\in x$. Take $b = a/t$, so that $t = a/b$. Note that $w < a$; so that $\delta < a - b$, which is $b < a - \delta$.

```
Definition BRinv x (aux:= fun z => BQps -s fun_image z BQinv) :=
   Yo (rationalp x)
   (BR_of_Q (BQinv (BQ_of_R x)))
    (Yo (inc x BRps) (aux x) (BRopp (aux (BRopp x)))).

Lemma BRinv_Q x: ratp x -> BRinv (BR_of_Q x) = BR_of_Q (BQinv x).
Lemma BRinv_0: BRinv \0r = \0r.
Lemma BRinv_irrational x (aux:= fun z => BQps -s fun_image z BQinv):
   irrationalp x ->
   BRinv x = (Yo (inc x BRps) (aux x) (BRopp (aux (BRopp x)))).
Lemma RpsS_inv x: inc x BRps -> inc (BRinv x) BRps.
Lemma BRinv_opp x: realp x -> BRinv (BRopp x) = BRopp (BRinv x).
Lemma RmsS_inv x: inc x BRms -> inc (BRinv x) BRms.
Lemma RS_inv x: realp x -> realp (BRinv x).
Lemma RIS_inv x: irrationalp x -> irrationalp (BRinv x).
Lemma BRinv_K x: realp x -> BRinv (BRinv x) = x.
Lemma BRinv_eq0 x: realp x -> BRinv x = \0r -> x = \0r.
Lemma BRinv_inj x y: realp x -> inc yR BR -> BRinv x = BRinv y -> x = y.
Lemma BRinv_1: BRinv \1r = \1r.
Lemma BRinv_m1: BRinv \1mr = \1mr.
Lemma BRinv_2: BRinv \2r = \2hr.

Lemma BRprod_inv1 x : realp x -> x <> \0r -> (x *r (BRinv x)) = \1r. (* 61 *)
Lemma BR_inv_prop a b: realp a -> realp b -> a *r b = \1r -> b = BRinv a.
Lemma BRprod_inv x y:realp x -> realp y ->
   BRinv (x *r y) = BRinv x *r BRinv y.
```

No difficulty here.

```
Definition BRdiv x y := x *r  (BRinv y).

Notation "x /r y" := (BRdiv x y) (at level 40).

Lemma RS_div x y: realp x -> realp y -> realp (x /r y).
Lemma BRdiv_0x x : \0r /r x = \0r.
Lemma BRdiv_x0 x : x /r \0r = \0r.
Lemma BRdiv_1x x : realp x -> \1r /r x = BRinv x.
Lemma BRdiv_x1 x : realp x -> x /r \1r = x.
Lemma RpsS_div a b: inc a BRps -> inc b BRps -> inc (a /r b) BRps.
Lemma RmsuS_div a b: inc a BRms -> inc b BRms -> inc (a /r b) BRps.
Lemma RpmsS_div a b: inc a BRps -> inc b BRms -> inc (a /r b) BRms.
Lemma RmpsS_div a b: inc a BRms -> inc b BRps -> inc (a /r b) BRms.
Lemma RpS_div a b: inc a BRp -> inc b BRp -> inc (a /r b) BRp.
Lemma RmuS_div a b: inc a BRm -> inc b BRm -> inc (a /r b) BRp.
Lemma BRpmS_div a b: inc a BRp -> inc b BRm -> inc (a /r b) BRm.
Lemma BRmpS_div a b: inc a BRm -> inc b BRp -> inc (a /r b) BRm.
Lemma BRopp_div_r x y: realp x -> realp y ->
  BRopp (x /r y) = x /r (BRopp y).
Lemma BRopp_div_l x y: realp x -> realp y ->
  BRopp (x /r y) = (BRopp x) /r y.
Lemma BRdiv_opp_comm x y: realp x -> realp y ->
  x /r  (BRopp y) =  (BRopp x) /r y.
Lemma BRdiv_opp_opp x y: realp x -> realp y ->
  (BRopp x) /r (BRopp y) = x /r y.
```

Note that $x = 1/x$ is equivalent to $x = 0$, $x = 1$ or $x = -1$.

```
Lemma BRdiv_xx x : realp x -> x <> \0r -> (x /r x) = \1r.
Lemma BQ_ltinv1 x: inc x BRps ->
   (x <r \1r <-> \1r <r BRinv x).
Lemma BR_square_1 x: realp x ->
   (x *r x = \1r <-> (x = \1r \/ x = \1mr)).
Lemma BR_self_inv x: realp x ->
   (x = BRinv x <-> [\/ x= \0r, x = \1r | x = \1mr]).
Lemma BRdiv_square a b: realp a -> realp b ->
    BRsquare (a /r b) = (BRsquare a) /r (BRsquare b).
Lemma BRdiv_sumDl x y z: realp x -> realp y -> realp z ->
   (y +r z) /r x = (y /r x) +r (z /r x).
Lemma BRdiv_prod_simpl_l x y z: realp x -> realp y -> realp z ->
   x <> \0r ->  (x *r y) /r  (x *r z) = y /r z.
Lemma BRdiv_prod_comm x y z: realp x -> realp y -> realp z ->
  (x *r y) /r z = (x /r z) *r y.
Lemma BRinv_div x y: realp x -> realp y ->  BRinv (x /r y) = y /r x.
Lemma BRdiv_prod x y:realp x -> realp y ->  x <> \0r -> (x *r y) /r x = y.
Lemma BRprod_div x y: realp x -> realp y -> x <> \0r -> x *r (y /r x) = y.
Lemma BRprod_div1 x y: realp x -> realp y -> x <> \0r -> (y /r x) *r x = y.
Lemma BRprod_div_ea x y z: realp x -> realp y -> realp z ->
   y <> \0r -> x = y *r z -> z = x /r y.
Lemma BRdiv_diag_rw x y: realp x -> realp y ->  x /r y = \1r -> x = y.
Lemma BRdiv_prod_simpl_r x y z: realp x -> realp y -> realp z -> z <> \0r ->
  (x *r z) /r (y *r z) = x /r y.
Lemma BRprod_eq2r x y z: realp x -> realp y -> realp z -> z <> \0r ->
    x *r z = y *r z -> x = y.
```

```
Lemma BRprod_eq2l x y z: realp x -> realp y -> realp z -> z <> \0r ->
   z *r x = z *r y -> x = y.
Lemma BRdiv_div_simp a b c: realp a -> realp b -> realp c -> b <> \0r ->
    (a /r b) /r (c /r b) = a /r c.
Lemma BRsum_div a b c: realp a -> realp b -> realp c -> c <> \0r ->
   a +r (b /r c) = (a *r c +r b) /r c.
Lemma BRdiff_div a b c: realp a -> realp b -> realp c -> c <> \0r ->
   a -r (b /r c) = (a *r c -r b) /r c.
Lemma BRinv_diff x y: realp x -> realp y ->x <> \0r -> y <> \0r ->
 (BRinv x -r BRinv y) = (y -r x) /r (x *r y).
```

Comparison.

```
Lemma BRdiv_Mlelege0 a b c d:
   realp a -> inc b BRps -> realp c -> inc d BRps ->
   ( a /r b <=r c /r d <-> a *r d <=r b *r c).
Lemma BMdiv_Mltltge0 a b c d:
   inc a BR -> inc b BRps -> realp c -> inc d BRps ->
   ( a /r b <r c /r d <-> a *r d <r b *r c).
Lemma BRinv_mon a b: inc a BRps -> inc b BRps  ->
   (\1r /r a <=r \1r /r b <-> b <=r a).
Lemma BRinv_mon1 a b: inc a BRps -> inc b BRps  ->
   (BRinv a <=r BRinv b <-> b <=r a).
Lemma BRinv_mon2 a b: inc a BRps -> inc b BRps ->
   (BRinv a <r BRinv b <-> b <r a).
Lemma BRdiv_Mle1 a b c: realp a -> realp b -> inc c BRps ->
     ( a <=r b *r c <->  a /r c <=r b).
```

10.2.6 Absolute value

We define

$$|x| = \begin{cases} x & \text{if } x \in \mathbf{R}^+ \\ -x & \text{otherwise}. \end{cases}$$

```
Definition BRabs x:= Yo (inc x BRp) x (BRopp x).
```

The absolute value of **R** has the same properties as that on **Q**.

```
Lemma BRabs_pos x: inc x BRp -> BRabs x = x.
Lemma BRabs_poss x: inc x BRps -> BRabs x = x.
Lemma BRabs_0 : BRabs \0r = \0r.
Lemma BRabs_negs x: inc x BRms -> BRabs x = BRopp x.
Lemma BRabs_neg x: inc x BRm -> BRabs x = BRopp x.
Lemma RSa x: realp x -> realp (BRabs x).
Lemma BRabs_abs x: realp x -> BRabs (BRabs x) = BRabs x.
Lemma BRabs_opp x: realp x -> BRabs (BRopp x) = BRabs x.
Lemma BRabs_m1: BRabs \1mr = \1r.
Lemma BRabs_1: BRabs \1r = \1r.
Lemma BRabs0_bis x: realp x ->  (x = \0r <-> BRabs x = \0r).
Lemma RpsSa x: realp x -> x <> \0r -> inc (BRabs x) BRps.
Lemma BRabs_cQ x: ratp x ->
  BRabs (BR_of_Q x) = BR_of_Q (BQabs x).
Lemma BRprod_abs x y: realp x -> realp y ->
  BRabs (x *r y) = (BRabs x) *r (BRabs y).
```

```
Lemma BRinv_abs x: realp x -> BRabs (BRinv x) = BRinv (BRabs x).
Lemma BRdiv_abs x y: realp x -> realp y ->
  (BRabs x) /r (BRabs y) = BRabs (x /r y).
Lemma rle_abs x: realp x -> x <=r (BRabs x).
Lemma BRabs_prop1 x y: realp x -> realp y ->
  (BRabs x <=r y <->  (BRopp y <=r x /\ x <=r y)).
Lemma BRabs_prop2 x y: realp x -> realp y ->
  (BRabs x <r y <->  (BRopp y <r x /\ x <r y)).
Lemma BRabs_prop3 x y e: realp x -> realp y -> realp e ->
  (BRabs (x -r y) <=r e <->  y -r e <=r x /\ x <=r y +r e).
emma BRabs_prop4 x y e: realp x -> realp y -> realp e ->
  (BRabs (x -r y) <r e <->  y -r e <r x /\ x <r y +r e).
Lemma rle_triangular x y: realp x -> realpy ->
  (BRabs (x +r y)) <=r (BRabs x) +r (BRabs y).
```

10.2.7 Half and middle

We say that $x/2$ is the half of x and $(x+y)/2$ is the middle of x and y.

```
Definition BRhalf x := x *r \2hr.
Definition BRmiddle x y := BRhalf (x +r y).
Definition BRdouble x := \2r *r x.

Lemma RSdouble x: realp x -> realp (BRdouble x).
Lemma BRdouble_C x : BRdouble x = x *r \2r.
Lemma BRdouble_s x: realp x -> x +r x = x *r \2r.
Lemma BR2_nz : \2r <> \0r.
Lemma BRdouble_half2:  \2hr +r \2hr = \1r.
Lemma RS_half x:  realp x -> realp (BRhalf x).
Lemma BR_middle x y:  realp x -> realp y -> realp (BRmiddle x y).
Lemma BRdouble_half1 x: realp x -> BRhalf x +r BRhalf x = x.
Lemma BRdouble_half x: realp x -> (BRhalf x) *r \2r = x.
Lemma BRhalf_double x: realp x -> BRhalf (x *r \2r) = x.
Lemma BRhalf_opp x: realp x -> BRhalf (BRopp x)  = BRopp (BRhalf x).
Lemma BRhalf_pos x: inc x BRps -> inc (BRhalf x) BRps.
Lemma BRhalf_neg x: inc x BRms -> inc (BRhalf x) BRms.
Lemma BRhalf_pos1 x: inc x BRps ->  (BRhalf x) <r x.
Lemma BRmiddle_comp x y: x <r y -> x <r BRmiddle x y /\  BRmiddle x y <r y.
Lemma BR_middle_prop1 a b: realp a -> realp b ->
   b -r (BRmiddle a b) = BRhalf (b -r a).
Lemma BR_middle_prop2 a b: realp a -> realp b ->
   (BRmiddle a b) -r a = BRhalf (b -r a).
Lemma BRhalf_prop x: BRhalf x = x /r \2r.
Lemma BRhalf_mon x y : x <=r y -> BRhalf x <=r BRhalf y.
```

Some properties of the square functions. Note that $x^2 = y^2$ is equivalent to $x = \pm y$.

```
Lemma BRprod_22: \2r *r \2r = \4r.
Lemma BRsum_square a b: realp a -> realp b ->
   BRsquare (a +r b) = BRsquare a +r BRsquare b +r BRdouble (a *r b).
Lemma BRdiff_square a b: realp a -> realp b ->
   BRsquare (a -r b) = BRsquare a +r BRsquare b -r (BRdouble (a *r b)).
Lemma BRsumdiff_square a b: realp a -> realp b ->
   BRsquare (a +r b) = \4r *r (a *r b) +r BRsquare (a -r b).
Lemma BRsquare_diff x y: realp x -> realp y ->
```

```
   BRsquare x -r BRsquare  y = (x -r y) *r (x +r y).
Lemma BRsquare_inj x y: realp x -> realp y ->
   (BRsquare x = BRsquare  y  <->  (x = y \/ x = BRopp y)).
```

Min and max. We define here min and max (in fact, we only need the next two lemmas).

```
Definition rmin x y := Yo (x <=r y) x y.

Lemma rmin_prop1 x y (r := rmin x y): realp x ->  realp y ->
  [/\ realp r  , r <=r x & r <=r y].
Lemma rmin_prop2 x y: inc x BRps ->  inc y BRps ->
  inc (rmin x y) BRps.
```

10.3 Cauchy sequences and continuity

10.3.1 Adjacent sequences

We say that (x, y) is a pair of *adjacent sequences*, if x and y are two sequences $\mathbf{N} \to \mathbf{Q}$, such that x_n is strictly increasing, y_n is strictly decreasing, and an additional condition C holds (the sequences have the "same limit"). We associate two sets L and R, defined by: z is in L if $z < x_n$ for some n, and z is in R if $y_n < z$ for some n.

```
Definition BQpair_aux C :=
   [/\ fgraph C, domain C = Nat,
       forall n, natp n -> inc (Vg C n) (BQ \times BQ),
       forall n, natp n -> P (Vg C n) <q P (Vg C (csucc n))&
       forall n, natp n -> Q (Vg C (csucc n)) <q Q (Vg C n)].

Definition BQpair_aux2a C :=
   forall n, natp n -> P (Vg C n) <q Q (Vg C n).
Definition BQpair_aux2b C :=
   forall n m, natp n -> natp m -> P (Vg C n) <q Q (Vg C m).
Definition BQpair C  :=  BQpair_aux C /\ BQpair_aux2b C.

Definition BQpairL C :=
   Zo BQ (fun x => exists2 n, natp n & x <q P (Vg C n)).
Definition BQpairR C :=
   Zo BQ (fun x => exists2 n, natp n & Q (Vg C n) <q x).
```

The first part of condition C mentioned above can be stated as: L and R are disjoint, or as: $x_n < y_m$, whatever n and m. It implies that R is a real number.

```
Lemma BQpair_mon C n m:  BQpair_aux C ->
   natp m -> n <c m ->
   P (Vg C n) <q P (Vg C m) /\ Q (Vg C m) <q Q (Vg C n).
Lemma BQpair_aux2a_equiv C:
   BQpair_aux C -> ( BQpair_aux2a C <-> BQpair_aux2b C).
Lemma BQpair_aux2a_equiv2 C:
  BQpair_aux C ->
  (BQpair_aux2b C <->disjoint (BQpairL C)(BQpairR C)).
Lemma BQpair_real C:  BQpair C -> real_dedekind (BQpairR C).
```

Note that the complement of L is a real number, provided that L has no supremum (if $t = \sup L$, then t is the least element of the complement of L). This is the same number as R if $\sup L = \inf R$ (these two quantities exist as real numbers).

First case: R has no lower bound in $\mathbf{Q}$. The condition mentioned above is $L \cup R = \mathbf{Q}$. It is equivalent to: for any rational number t, there an integer n such that $t < x_n$ or $y_n < t$. In this case, R is irrational. The converse holds: assume $a \not\in x$ and $b \in x$. We have $a < b$. Let c be the middle of a and b. If $c \in x$, we set $b' = c$ and consider a' such that $a < a'$ and $a' \not\in x$. Otherwise we set $a' = c$ and consider $b' \in x$ such that $b' < b$. Iterating this process, we find two sequences x_n and y_n, x_n is strictly increasing in the complement of x, and y_n is strictly decreasing in x. Moreover $y_n - x_n$ can be made arbitrarily small (it is $\le (y_0 - x_0)/2^n$). We have $x = R$ (and the complement of x is L, same proof): the relation $R \subset x$ is obvious. Take $t \in x$ and $u \in x$ with $u < t$. Take n such that $y_n - x_n < t - u$. We have $x_n < u$ (since $x_n \not\in x$ and $u \in x$). It follows $y_n < t$. This says $t \in R$.

```
Lemma BQpair_irrational C:
  BQpair C ->
  (BQpairL C \cup BQpairR C = BQ) ->
  irrationalp (BQpairR C).

Lemma BQpair_irrational2  C:
  (BQpairL C \cup BQpairR C = BQ <->
  forall x, ratp x -> exists2 n, natp n &
     (x <q P (Vg C n) \/ Q (Vg C n) <q x)).
Lemma BQpair_irrational3 x: irrationalp x -> (* 117 *)
  exists C, [/\  BQpairR C = x,  BQpair C  & BQpairL C \cup BQpairR C = BQ].
```

Second case: R has a lower bound t in $\mathbf{Q}$, and this is the upper bound of L. As t is neither in L not R, we get $L \cup R = \{t\}$. Here R is rational, namely equal to C_t. Conversely, if t is rational, we may consider $x_n = t - 1/(n+1)$ and $y_n = t + 1/(n+1)$. Here L (resp., R) is the set of rational numbers $< t$ (resp. $> t$).

```
Definition BQpair_aux3 C :=
   singletonp (BQ -s ((BQpairL C \cup BQpairR C))).

Lemma BQpair_single3 C:
  BQpair_aux2b C ->
  (BQpair_aux3 C <->
   exists t, [/\ ratp t,
            forall x, x <q t ->  exists2 n, natp n & x <q P (Vg C n) &
            forall x, t <q x ->  exists2 n, natp n & Q (Vg C n) <q x ]).
Lemma BQpair_rational C : BQpair C -> BQpair_aux3 C ->
  exists2 x, ratp x & BQpairR C = BR_of_Q x.
Lemma BQpsS_fromN1 n: natp n -> inc (BZ_of_nat (csucc n))  BZps.
Lemma BQpsS_fromN n: natp n -> inc (BQ_of_nat (csucc n)) BQps.


Lemma BQpair_rational2 x: ratp x ->  (* 71 *)
  exists C, [/\ BQpairR C = BR_of_Q x, BQpair C & BQpair_aux3 C].
```

Assume x irrational. An adjacent sequence gives an order isomorphism $f : \mathbf{Q}^* \to \mathbf{Q}$, such that the image of $\mathbf{Q}^*_+$ is x. Since $\sqrt{2}$ is irrational, there is at least one such f.

```
Section BQorder.
```

```
Let r := BQ_ordering.
Let r' := induced_order r (BQ -s1 \0q).

Lemma BQ_order_isomorphisms_spec x:  irrationalp x ->
  exists2 f, order_isomorphism f r' r & Vfs f BQps  = x.
Lemma BQ_order_isomorphisms_spec2: exists f, order_isomorphism f r' r.
End BQorder.
```

10.3.2 The cardinal of R

Let's show that the cardinal of **R** is $2^{\aleph_0}$. Let E be the set of functions $\mathbf{N} \to 2$, its cardinal is $2^{\aleph_0}$. Given a function $f \in \mathrm{E}$, we consider $\mathrm{F}_n = \sum_{i \le n} f_i/3^i$. We also consider $\mathrm{F}_n + 1/3^n$; these two pairs are though of as a pair of adjacent sequences that define a real number.

```
Definition CR_inv3n n := BQinv (BQ_of_nat (\3c ^c n)).
Definition CR_next_term f n := (BQ_of_nat (Vf f n)) *q (CR_inv3n n).
Definition CR_partial_sum f n := qsum (CR_next_term f) (csucc n).
Definition CR_set := functions Nat \2c.
Definition CR_limit f := Zo BQ (fun z => exists2 n, natp n &
    CR_partial_sum f n +q (CR_inv3n n) <q z).


Lemma CR_prop0 n: natp n -> inc (CR_inv3n n) BQps.
Lemma CR_prop1 n: natp n -> ratp (CR_inv3n n).
```

We fix such a function f. Note that f_i is 0 or 1, so $f_i/3^i$ is 0 or $1/3^i$. It follows that $0 \le \mathrm{F}_i$ (and $\mathrm{F}_i \in \mathbf{Q}$). Moreover $\mathrm{F}_i \le \mathrm{F}_{i+1} \le \mathrm{F}_i + 1/3^{i+1}$. A non-trivial relation is that $\mathrm{F}_i + 1/(2\cdot 3^i)$ is decreasing. As a consequence, it is not possible that $\mathrm{F}_k \le \mathrm{F}_n + 1/3^n$.

```
Section Aux.
Variable f:Set.
Hypothesis fI: inc f CR_set.


Lemma CR_prop2 n: natp n ->
   CR_next_term f n = \0q \/ CR_next_term f n = (CR_inv3n n).
Lemma CR_prop3  n: natp n -> inc (CR_next_term f n) BQp.
Lemma CR_prop4 n: natp n ->  inc (qsum (CR_next_term f) n) BQp.
Lemma CR_prop5 n : natp n ->
  CR_partial_sum f n <=q  CR_partial_sum f (csucc n).
Lemma CR_prop6 n : natp n ->
  CR_partial_sum f (csucc n) <=q  CR_partial_sum f n +q (CR_inv3n (csucc n)).
Lemma CR_prop7 n k (h := fun i => CR_partial_sum f i  +q BQhalf (CR_inv3n i)):
   natp n -> natp k -> (h (n+c k) <=q h n).
Lemma CR_prop8 n k: natp n -> natp k ->
  CR_partial_sum f k <=q CR_partial_sum f n +q  (CR_inv3n n).
```

Let $\mathrm{B}(f)$ be the set of all t such that $\mathrm{F}_k + 1/3^k < t$ for at least one k. Note that $\mathrm{F}_k + 1/3^k \in \mathrm{B}$ (consider $k+1$) and $\mathrm{F}_k \not\in \mathrm{B}$ (by prop8). Obviously B is a real number. Let's note that $f \to \mathrm{B}$ is injective. Consider two different functions f and g. For some k, $f_k \neq g_k$ but $f_i = g_i$ whenever $i < k$. We have $\mathrm{F}_k = x + f_k/3^k$ and $\mathrm{G}_k = x + g_k/3^k$. Since f_k and g_k are zero or one, we may assume by symmetry $f_k = 0$ and $g_k = 1$; say $\mathrm{G}_k = \mathrm{F}_k + 1/3^k$. We deduce that G_k is in $\mathrm{B}(f)$ but not in $\mathrm{B}(g)$, so $\mathrm{B}(f) \neq \mathrm{B}(g)$. The injectivity says $\mathrm{card}(\mathbf{R}) \le 2^{\aleph_0}$. The result follows from $\mathbf{R} \subset \mathfrak{P}(\mathbf{Q})$.

```
Lemma CR_prop9 n: natp n ->
  inc (CR_partial_sum f n +q (CR_inv3n n)) (CR_limit f).
Lemma CR_prop10 k: natp k -> ~inc (CR_partial_sum f k) (CR_limit f).
Lemma CR_prop11 : realp (CR_limit f).
End Aux.

Lemma CR_prop12: injection (Lf CR_limit CR_set BR).
Lemma card_R: cardinal BR = \2c ^c aleph0.
```

10.3.3 Cauchy sequences

We say that a sequence x_n is Cauchy if whatever $\epsilon > 0$, there is N such that $|x_n - x_m| < \epsilon$, whenever $n > \mathrm{N}$ and $m > \mathrm{N}$. We say that a sequence x_n converges to x if, whatever $\epsilon > 0$, there is N such that $|x_n - x| < \epsilon$, whenever $n > \mathrm{N}$.

```
Definition BQ_seq x := [/\ fgraph x, domain x = Nat & sub (range x) BQ].
Definition BR_seq x := [/\ fgraph x, domain x = Nat & sub (range x) BR].

Definition CauchyQ x := BQ_seq x /\
      forall e, inc e BQps -> exists2 N, natp N &
         forall n m, natp n -> natp m -> N <=c n -> N <=c m ->
            BQabs ((Vg x n) -q (Vg x m)) <q e.

Definition CauchyR x := BR_seq x /\
      forall e, inc e BQps -> exists2 N, natp N &
         forall n m, natp n -> natp m -> N <=c n -> N <=c m ->
            BRabs ((Vg x n) -r (Vg x m)) <r (BR_of_Q e).
Definition limitQ x v:=
   forall e, inc e BQps -> exists2 N, natp N &
   forall n, natp n ->  N <=c n -> BQabs ((Vg x n) -q v) <q e.

Definition limitR x v:=
   forall e, inc e BQps -> exists2 N, natp N &
   forall n, natp n ->  N <=c n -> BRabs ((Vg x n) -r v) <r  (BR_of_Q e).
```

We first show: in the definitions of limit and Cauchy in the real case, we may consider real or rational ϵ. We show that the limit of a sequence is unique. We also show that a sequence that has a limit is Cauchy.

```
Lemma BR_seq_prop s n: BR_seq s -> natp n -> realp (Vg s n).
Lemma BQ_seq_prop s n: BQ_seq s -> natp n -> ratp (Vg s n).
Lemma CauchyR_alt x: CauchyR x <->
   (BR_seq x /\  forall e, inc e BRps -> exists2 N, natp N &
         forall n m, natp n -> natp m -> N <=c n -> N <=c m ->
            BRabs ((Vg x n) -r (Vg x m)) <r e).
Lemma limitR_alt x v : (limitR x v) <->
  forall e, inc e BRps -> exists2 N, natp N &
  forall n, natp n ->  N <=c n -> BRabs ((Vg x n) -r v) <r e.
Lemma limitQ_unique x v1 v2: BQ_seq x -> ratp v1 -> ratp v2 ->
   limitQ x v1 -> limitQ x v2 -> v1 = v2.
Lemma limitR_unique x v1 v2: BR_seq x -> realp v1 -> realp v2 ->
   limitR x v1 -> limitR x v2 -> v1 = v2.
Lemma CauchyQ_when_limit x: BQ_seq x -> (exists2 y, ratp y & limitQ x y) ->
   CauchyQ x.
```

```
Lemma CauchyR_when_limit x: BR_seq x -> (exists2 y, realp y & limitR x y) ->
   CauchyR x.
```

Question: does a Cauchy sequence have a limit? we first show that if x_n is a sequence of rational numbers, y_n the same sequence considered as a sequence of real numbers, then y_n is Cauchy when x_n is Cauchy. We have shown above that for any real number x there is a pair of adjacent sequences x_n and x'_n satisfying some properties. These are Cauchy sequences (proof for the sequence that is decreasing: for any $\epsilon > 0$, there are a, b such that $a \not\in x$, $b \in x$ and $b - a < \epsilon$. There is N such that $x_N < b$; if $n \ge N$ and $m \ge N$, then $a < x_n < b$ and $a < x_m < b$ so that $|x_n - x_m| < \epsilon$). If y_n is the sequence of real numbers associated to it, then y_n converges to x (fix ϵ, let a, b, N be as above; we have $x_n - \epsilon < b - \epsilon < a$. This says $x_n - \epsilon < x$, thus $|x - x_n| < \epsilon$).

Example: Take $x = \sqrt{2}$. There is a Cauchy sequence x_n that converges (in **R**) to x; since x is irrational, the sequence has no limit in **Q**.

Let's show that **R** is complete: every Cauchy sequence x_n in **R** converges. Assume first that y_n is a sequence such that $|y_n - x_n|$ can be made arbitrarily small. If y_n converges to y, then x_n converges to y. Note that **Q** is dense in **R**: for ever $\epsilon > 0$, every real x, there is a rational y such that $|x - y| < \epsilon$. So, given any Cauchy sequence x_n in **R**, there is a Cauchy sequence y_n in **Q**, such that $|y_n - x_n|$ can be made arbitrarily small. Consider a Cauchy sequence y_n in **Q**. Let A be the set of all t such that $x_n < t$ for large n. Since every Cauchy sequence is bounded, this set is neither empty nor **Q**. Assume that this set has a least element y; then y_n converges to y. Otherwise, A is a real number, and y_n converges to A in **R**.

```
Definition BR_seq_of_Q s :=  Lg Nat (fun n => (BR_of_Q (Vg s n))).
Definition similar_seq x y :=
 (forall e, inc e BQps -> exists2 N, natp N &
      forall n, natp n -> N <=c n ->
            BRabs ((Vg x n) -r (Vg y n))  <r (BR_of_Q e)).

Lemma BR_seq_prop1 f: (forall n, natp n -> realp (f n)) ->
   BR_seq (Lg Nat f).
Lemma BR_seq_of_Q_seq s: BQ_seq s -> BR_seq (BR_seq_of_Q s).
Lemma BR_seq_of_Q_cauchy s: CauchyQ s -> CauchyR (BR_seq_of_Q s).
Lemma BR_limit_of_rat x: realp x ->   (* 57 *)
   exists2 s, CauchyQ s & limitR (BR_seq_of_Q s) x.
Lemma similar_seq_sym x y: BR_seq x -> BR_seq y -> similar_seq x y ->
   similar_seq y x.
Lemma limitR_same x y z:  BR_seq x  -> BR_seq y -> realp z ->
   similar_seq x y  ->  limitR x z ->  limitR y z.
Lemma BQ_denseR x e: realp x -> inc e BQps ->
   exists2 y, ratp y & BRabs (x -r (BR_of_Q y)) <r (BR_of_Q e).
Lemma BQ_denseR2 x: realp x ->
  (exists2 y, ratp y & x <r (BR_of_Q y))
  /\ (exists2 y, ratp y & (BR_of_Q y) <r x).
Lemma limitR_same2 x: CauchyR x ->  (* 70 *)
   exists2 y, CauchyQ y & similar_seq x (BR_seq_of_Q y).
Lemma BR_Cauchy_bounded s: CauchyQ s -> exists2 b, ratp b &
   forall n, natp n -> BQabs (Vg s n) <=q b.
Lemma BR_complete1 s: CauchyQ s -> exists2 x, (* 92 *)
    realp x & limitR (BR_seq_of_Q s) x.
Lemma BR_complete s: CauchyR s -> exists2 x,
    realp x & limitR s x.
```

10.3.4 Continuity

We say that f is continuous at x it $f(t)$ is near $f(x)$ when t is near x. Here near means $|x-y| \le \epsilon$. We could use $<$ instead of $\le$, or restrict ϵ to be rational.

If f and g agree near x, and one function is continuous so is the other; the composition of two continuous functions is continuous. We say we that f is continuous if it is continuous at every x. We say that a function of two variables $g(x,y)$ is continuous if it is continuous at every (x,y), both as a function of x and y. This simplifies if $f(x,y) = f(y,x)$ [note: one may also consider: for every $\epsilon > 0$, there is η such that $|x-x'| \le \eta$, $|y-y'| \le \eta$ implies $|f(x,y) - f(x',y')| \le \epsilon$].

```
Definition BR_near x e y:= realp y /\  BRabs (x -r y) <=r e.
Definition continuous_at f x:=
   forall e, inc e BRps -> exists2 d, inc d BRps &
   forall y,  BR_near x d y ->  BR_near (f x) e (f y).
Definition continuous f:= forall x, realp x -> continuous_at f x.
Definition continuous2 (f:fterm2):=
  forall x y, realp x -> realp y  ->
    continuous_at (f x) y /\ continuous_at (f ^~y) x.


Lemma BR_nearP x e y: realp x -> realp e ->
  ((BR_near x e y) <-> ( x -r e <=r y /\ y <=r x +r e)).
Lemma BR_near_trans x e e' y: e <=r e' ->
  (BR_near x e y) -> (BR_near x e' y).
Lemma continuous_local f g x: realp x ->
   (exists2 e, inc e BRps & forall y, BR_near x e y -> f y = g y) ->
   continuous_at f x ->  continuous_at g x.
Lemma continuous_comp f g x:
  continuous_at f x ->  continuous_at g (f x) ->
  continuous_at (g \o f) x.
Lemma continuous2_sym (f:fterm2) :
  (forall x y, realp x -> realp y  -> f x y = f y x) ->
  (forall x y, realp x -> realp y  ->
    continuous_at (f x) y) ->  continuous2 f.
Lemma continuous_real f x: realp x ->  continuous_at f x -> realp (f x).
```

We show here continuity of some functions. In the case of $x \mapsto 1/x$, given ϵ, we chose $\delta = \min(|x|/2, \epsilon x^2/2)$. We have $(1/x - 1/y) = (x-y)/(xy)$. If $|x-y| < \delta$, then $|y| \ge |x|/2$, so that $1/|xy| \le 2|x|^2 \le \epsilon/\delta$. The function x/y is continuous in x everywhere, in y when $y \ne 0$.

```
Lemma continuous_id: continuous id.
Lemma continuous_opp : continuous BRopp.
Lemma continuous2_sum : continuous2 BRsum.
Lemma continuous2_diff : continuous2 BRdiff.
Lemma continuous2_prod : continuous2 BRprod.
Lemma continuous_inv x: realp x -> x <> \0r -> continuous_at BRinv x.
Lemma continuous2_div x y: realp x -> realp y ->
    (continuous_at (BRdiv ^~y) x) /\ (y <> \0r -> continuous_at (BRdiv x) y).

Lemma continuous_sum f g x:  realp x ->
   continuous_at f x -> continuous_at g x ->
   continuous_at (fun z => f z +r g z) x.
Lemma continuous_diff f g x:  realp x ->
```

```
   continuous_at f x -> continuous_at g x ->
   continuous_at (fun z => f z -r g z) x.
Lemma continuous_prod f g x:  realp x ->
   continuous_at f x -> continuous_at g x ->
   continuous_at (fun z => f z *r g z) x.
Lemma continuous_div f g x:  realp x -> g x <> \0r ->
   continuous_at f x -> continuous_at g x ->
   continuous_at (fun z => f z /r g z) x.
Lemma continuous_square: continuous BRsquare.
```

Some properties of continuous functions.

```
Lemma continuous_prop1 f x a: continuous_at f x -> a <r f x ->
  exists2 e, inc e BRps & forall y, (BR_near x e y) -> a <r f y.
Lemma continuous_prop2 f x a: continuous_at f x -> f x <r a ->
  exists2 e, inc e BRps & forall y, (BR_near x e y) -> f y <r a.
```

Let's show (Bolzano, intermediate value theorem): if f is continuous on $[a,b]$, $f(a) \le 0$ and $f(b) \ge 0$, then $f(x) = 0$ for some $x \in [a,b]$. Note that we do not require f to be defined outside $[a,b]$.

Let I be the interval $[a,b]$, E the set of all x in I such that $f(x) \ge 0$. Let z be the greatest lower bound of E; this exists and is in I. Assume $f(z) > 0$; this says $z \ne a$. There is t near z, $t < z$, such that $f(t) > 0$. We may assume $t \in$ I. This says $t \in$ E, absurd. Assume $f(z) < 0$; this says $z \ne b$. There is $\epsilon > 0$ so that f is negative on $[z, z+\epsilon]$, and $z + \epsilon < b$. If $u \in$ E, then $f(u)$ is positive, so that u is not in $[z, z+\epsilon]$. Since z is a lower bound, we have $z \le u$, thus $z + \epsilon \le u$; this says that $z + \epsilon$ is a lower bound, contradicting the fact that z is the greatest lower bound. It follows $f(z) = 0$.

We deduce: if f is continuous on $[a,b]$, v is between $f(a)$ and $f(b)$ then $f(x) = v$ for some $x \in [a,b]$.

```
Definition BR_between x a b: (a <=r x /\ x <=r b) \/ (b <=r x /\ x <=r a).

Definition continuous_right f x:=
   forall e, inc e BRps -> exists2 d, inc d BRps &
   forall y,  BR_near x d y ->  x <=r y -> BR_near (f x) e (f y).

Definition continuous_left f x:=
   forall e, inc e BRps -> exists2 d, inc d BRps &
   forall y,  BR_near x d y -> y <=r x -> BR_near (f x) e (f y).

Definition Bolzano_hyp f x y:=
    [/\ continuous_right f x, continuous_left f y &
    (forall z, x <r z -> z <r y -> continuous_at f z)].

Lemma Bolzano_hyp_simp f x y: x <=r y ->
    (forall z, x <=r z -> z <=r y -> continuous_at f z) ->
    Bolzano_hyp f x y.

Lemma Bolzano f x y:  x <=r y -> Bolzano_hyp f x y ->  (* 102 *)
    f x <=r \0r -> \0r <=r f y ->
    exists2 z, (x <=r z /\ z <=r y) & f z = \0r.
Lemma Bolzano1 f x y:  x <=r y ->
    (forall z, x <=r z -> z <=r y -> continuous_at f z) ->
```

```
    f x <=r \0r -> \0r <=r f y ->
    exists2 z, (x <=r z /\ z <=r y) & f z = \0r.
Lemma Bolzano2 f x y v:  x <=r y -> Bolzano_hyp f x y ->
    (BR_between v (f x) (f y)) ->
    exists2 z, (x <=r z /\ z <=r y) & f z = v.
```

Application: if $x \geq 0$, there is $y \geq 0$ such that $x = y^2$.

```
Definition BRsqrt x := select (fun z => x = BRsquare z) BRp.

Lemma BRsqrt_exists x: inc x BRp -> exists2 y, inc y BRp & x = BRsquare y.
Lemma BRsqrt_prop x: inc x BRp ->
   x = BRsquare (BRsqrt x) /\ inc (BRsqrt x) BRp.
Lemma sqrt2_prop : BRsqrt2 = BRsqrt \2r.
```

Let's show that $f(\lim x_i) = \lim f(x_i)$, assuming that $(x_i)_i$ is a sequence that converges to a point x where f is continuous. The result is clear when each $f(x_i)$ is real. If y is near x, then $f(y)$ is near $f(x)$, in particular is real. But if n is big enough, then x_n is near x. So we state: there is N, such that if $n \geq \mathrm{N}$ then $f(x_n)$ is real. We also state: if $\lim x_n = x$ then $\lim x_{n+\mathrm{N}} = x$, where $x_{n+\mathrm{N}}$ stands for the sequence $n \mapsto x_{n+\mathrm{N}}$. Let y_n be the sequence whose general term is $f(x_n)$ when this is real, zero otherwise. If $i \geq \mathrm{N}$ then $y_i = f(x_i)$ so that $\lim y_{n+\mathrm{N}} = f(x)$. The conclusion follows from: if $y_{n+\mathrm{N}}$ has a limit y, then y_n has the same limit.

```
Lemma limit_of_continuous xn x f (yn := Lg Nat (fun i => f (Vg xn i))):
   BR_seq xn -> continuous_at f x -> limitR xn x -> realp x ->
   (forall n, natp n -> inc  (f (Vg xn n)) BR) -> realp (f x) ->
   (BR_seq yn /\ limitR yn (f x)).
 Lemma limit_of_continuous_prop xn x f:
   BR_seq xn -> continuous_at f x -> limitR xn x -> realp x ->
   exists2 N, natp N & forall n, natp n -> N <=c n -> realp (f (Vg xn n)).
Lemma limit_of_subset xn x n (yn:= Lg Nat (fun i => (Vg xn (n +c i)))):
   BR_seq xn -> limitR xn x -> natp n ->
   (BR_seq yn /\  limitR yn x).
Lemma limit_of_subset2 xn x n (yn:= Lg Nat (fun i => (Vg xn (n +c i)))):
   limitR yn x -> natp n -> limitR xn x.
Lemma limit_of_continuous2 xn x f
   (coerce := fun z => Yo (realp z) z \0r)
   (yn := Lg Nat (fun i => coerce (f (Vg xn i)))):
   BR_seq xn -> continuous_at f x -> limitR xn x -> realp x -> realp (f x) ->
   (BR_seq yn /\ limitR yn (f x)).
```

Consider now a function $f : \mathbf{R} \to \mathbf{R}$, and the sequence $x_{n+1} = f(x_n)$, If x_0 is real, so are all the x_i. Assume that the sequence converges to x, and that f is continuous at x. Then $f(x) = x$.

If $x_n \geq 0$ then the limit is ≥ 0 (for otherwise there is a rational number $\epsilon > 0$ such that $x < -\epsilon < 0$, and if n is large enough $|x_n - x| \leq \epsilon$; note that $x_n - x \geq 0$). In particular if f is a function $\mathbf{R}_+ \to \mathbf{R}_+$ and $x_0 \geq 0$, then x is a positive fix point of f. The condition $x \geq 0$ can be replaced by $x \geq a$, whatever a.

```
Lemma limit_of_continuous_fix x0 x f (seq:= induction_defined f x0)
      (xn := Lg Nat (Vf seq)):
   (forall x, realp x -> inc (f x) BR) -> inc x0 BR ->
```

```
   continuous_at f x -> limitR xn x -> realp x -> f x = x.
Lemma limit_positive xn x:
   (forall n, natp n -> \0r <=r (Vg xn n)) ->
   BR_seq xn -> limitR xn x -> realp x -> \0r <=r x.
Lemma limit_of_continuous_fix_pos x0 x f (seq:= induction_defined f x0)
      (xn := Lg Nat (Vf seq)):
   (forall x, inc x BRp -> inc (f x) BRp) -> inc x0 BRp ->
   continuous_at f x -> limitR xn x -> realp x ->
   f x = x /\ inc x BRp.
Lemma limit_of_continuous_fix_gea a x0 x f (seq:= induction_defined f x0)
      (xn := Lg Nat (Vf seq)):
   realp a ->
   (forall x, a <=r x -> a <=r (f x)) -> a <=r x0 ->
   continuous_at f x -> limitR xn x -> realp x ->
   f x = x /\ a <=r x.
```

Let $(x_n)_n$ be a decreasing sequence bounded below by a, and x the infimum of the range of he sequence. Obviously $a \le x$. If $\epsilon > 0$ then $x+\epsilon$ is not a lower bound, so that for some N we have $x_N \le x+\epsilon$. Since the sequence is decreasing, it follows that $|x_n - x| \le \epsilon$ when $n \ge N$, so that $\lim x_i = x$. By considering the sequence of opposites, if $(x_n)_n$ is increasing and bounded, it has a limit, the supremum of the range. In the first case, if f is a function such that $a \le t$ implies $a \le f(t) \le t$, if the sequence $x_{n+1} = f(x_n)$ (initialized with a value $\ge a$) has an infimum x at which f is continuous, then the limit of the sequence is x and x is a fix-point of f such that $a \le x$.

```
Lemma decreasing_bounded_limit a xn (x := infimum BR_order (range xn)):
  BR_seq xn ->
  (forall n, natp n -> a <=r (Vg xn n)) ->
  (forall n, natp n -> Vg xn (csucc n) <=r  Vg xn n) ->
  (a <=r x /\ limitR xn x).
Lemma increasing_bounded_limit a xn (x := supremum BR_order (range xn)):
  BR_seq xn ->
  (forall n, natp n -> (Vg xn n) <=r a) ->
  (forall n, natp n -> (Vg xn n) <=r Vg xn (csucc n))  ->
  (x <=r a /\ limitR xn x).
Lemma decreasing_limit_bounded_fix a x0 f
  (seq:= induction_defined f x0)  (xn := Lg Nat (Vf seq))
  (x := infimum BR_order (range xn)):
  (forall x, a <=r x ->  a <=r f x /\  f x <=r x) -> a <=r x0 ->
  (continuous_at f x) ->
  [/\ a <=r x, f x = x & limitR xn x].
```

Application. Let $a \ge 0$ and b its square root. Let $f(t) = (t^2 + a)/(2t)$, and $x_{n+1} = f(x_n)$. If $x \ge 0$ then $f(x) \ge b$. If we choose $x_0 \ge 1+a$ (thus $\ge c$), the sequence decreases to a fix-point of x, thus to b. Note that f is continuous everywhere but at zero,so that case $a = 0$ is exceptional. If $g(x) = (x + f(x)/2)$, then $y_{n+1} = g(y_n)$ is increasing if the $b/3 \le y_0 \le b$ and each term satisfies this condition. Thus y_n converges to b.

```
Lemma square_root_cv1 a b (f := fun z => (BRsquare z +r a) /r (\2r *r z))
  (seq:= induction_defined f b)  (xn := Lg Nat (Vf seq))
  (x := infimum BR_order (range xn)):
  inc a BRp -> \1r +r a <=r b ->
  [/\ inc x BRp, limitR xn x & BRsquare x = a]. (* 98 *)
Lemma square_root_cv2 a (f := fun z => (BRsquare z +r a) /r (\2r *r z))
```

```
  (g := fun z => BRhalf ((f z) +r z)) (s := BRsqrt a):
  inc a BRp ->
  [/\ forall x,realp x -> x <> \0r -> (g x = x <-> x = s \/ x = BRopp s),
  (forall x, inc x BRp -> inc (f x) BRp)
  (forall x, \0r <=r x -> x <=r s -> x <=r (g x)) &
  (forall x, (s /r \3r) <=r x -> x <=r s -> g x <=r s)].
Lemma square_root_cv3 a b (f := fun z => (BRsquare z +r a) /r (\2r *r z))
  (g := fun z => BRhalf ((f z) +r z)) (s := BRsqrt a)
  (seq:= induction_defined g b)  (xn := Lg Nat (Vf seq))
  (x := supremum BR_order (range xn)):
  inc a BRp ->  (s /r \3r) <=r b -> b <=r s ->
  [/\ inc x BRp, limitR xn x & x = s].
```

10.3.5 The power function

We define x^n by induction on n (for finite n and real x) and show usual properties.

```
Section FinitePower.

Variable x: Set.
Hypothesis xr: realp x.

Definition BRnpow :=  induction_term (fun _ : Set => BRprod x) \1r.
Lemma BRnpowx0: BRnpow \0c = \1r.
Lemma BRnpowxS n : natp n -> BRnpow (csucc n) = x *r BRnpow n.
Lemma BRnpowx1: BRnpow \1c = x.
Lemma RS_Brnpow n: natp n -> realp (BRnpow n).
Lemma BRnpow_prop1 n m: natp n -> natp m ->
  (BRnpow n) *r (BRnpow m)  = (BRnpow (n +c m)).
End FinitePower.

Notation "x ^r y" := (BRnpow x y) (at level 30).

Lemma BRnpow00: \0r ^r \0c = \1r.
Lemma BRnpow0Sn n : natp n -> \0r ^r (csucc n) = \0r.
Lemma BRnpow1n n : natp n -> \1r ^r n = \1r.
Lemma BRnpow_prop2 x y n : realp x -> realp y -> natp n ->
```

10.3.6 Fibonacci

We first define 5 and $\phi = (1+\sqrt{5})/2$.

```
Definition BR_five := BR_of_Q (BQ_of_Z (BZ_of_nat \5c)).
Notation "\5r" := BR_five.
Lemma RpsS5 : inc \5r BRps.
Lemma RS5 : realp \5r.
Lemma BR_succ4 : \5r = \4r +r \1r.
Definition BR_phi := BRhalf(\1r +r BRsqrt \5r).
```

Let $x = (1+\sqrt{a})/2$. Then $x^2 = (a-1)/4$. If $a = 5$, we get $x^2 = 1+x$. So $\phi^2 = \phi+1$. It follows that ϕ^n satisfies $x_{n+2} = x_{n+1} + x_n$. Note that $-1/phi$ satisfies the same equation. We get

$$F_n = \frac{\phi^n - (-1/\phi)^n}{\sqrt{5}}$$

.

```
Section Fibonacci.

Lemma fibr_prop1 a x y: realp x -> realp a -> realp y ->
   BRsquare y = a  -> x = (\1r +r y) /r \2r ->
  BRsquare x = (a-r \1r)/r \4r +r x.
Lemma fibr_prop2 x y: realp x -> realp y ->
   BRsquare y = \5r  -> x = (\1r +r y) /r \2r ->
  BRsquare x = \1r +r x.
Lemma BRps_phi: inc BR_phi BRps.
Lemma RS_phi: realp BR_phi.
Lemma fibr_prop3:  BRsquare BR_phi = \1r +r BR_phi.
Lemma fibr_prop4: BRinv BR_phi = BR_phi -r \1r.
Lemma fibr_prop5: BR_phi +r BRinv BR_phi = BRsqrt \5r.
Lemma fibr_prop6 (x := (BRopp (BRinv BR_phi))):  BRsquare x = \1r +r x.
Lemma fibr_prop7 x: realp x -> BRsquare x = \1r +r x ->
  forall n, natp n -> x ^r (csucc (csucc n)) = x ^r (csucc n) +r x ^r n.
```


```
Lemma fibr_prop7 x: realp x -> BRsquare x = \1r +r x ->
  forall n, natp n -> x ^r (csucc (csucc n)) = x ^r (csucc n) +r x ^r n.
```

Chapter 11

Ordinal numbers

What follows is not part of the main text of Bourbaki, but comes from exercises and other sources. Ordinal sums are defined in Ex1.3 (page 519), order-types in in Ex2.13 (page 581) and ordinal numbers in Ex2.14 (page 583). Ex2.20 (page 591) defines "pseudo-ordinals", and an identification between pseudo-ordinals and ordinals. These pseudo-ordinals are the von Neumann ordinals introduced in section 4, page 66. The Cantor Normal Form and properties of multiplication comes from Cantor [7]. The idea of enumerating sequences of ordinals comes from Veblen [22].

11.1 Notations and vocabulary

Recall that an order r is the graph of an order relation $\leq$ on a set E, and an ordered set E is an abuse of language for E together with $\leq$. The triple $\Gamma = (\mathrm{E}, \mathrm{E}, r)$ is called an ordering by Bourbaki. One similarly defines a well-order, a well-order relation, a well-ordered set and a well-ordering. For simplicity, we shall use r instead of Γ in every case.

In Exercise 2.13 (see page 581), Bourbaki says: « Let $\mathrm{Is}(\Gamma, \Gamma')$ be the relation "Γ is an ordering (on E) and Γ' is an ordering (on E$'$), and there exists an isomorphism of E, ordered by Γ, onto E$'$, ordered by Γ'". [...] The term $\tau_\Delta(\mathrm{Is}(\Gamma, \Delta))$ is an ordering called the *order-type* of Γ and is denoted by $\mathrm{Ord}(\Gamma)$, or Ord(E) by abuse of notations. » It follows that $\mathrm{Ord}(r)$ is an order whenever r is an order and $\mathrm{Ord}(r) = \mathrm{Ord}(r')$ if and only if $\mathrm{Is}(r, r')$. We shall admit the existence of an order-type. Since this is rarely used, we shall introduce this axiom in a separate file, see Section 11.28.

```
(*
Parameter order_type: Set -> Set.
Axiom order_type_exists:
  forall x, order x -> x \Is (order_type x).
Axiom order_type_unique:
  forall x y, x \Is y -> (order_type x = order_type y).
*)
```

Bourbaki defines an ordinal as the order-type of a well-ordered set; it is hence a well-ordering. In order to avoid confusions, we use here B-ordinal for the Bourbaki definition and v-ordinal for the von Neumann definition. Recall that, if r is a well-order, then $\mathrm{ord}(r)$ is the v-ordinal of r; it is a v-ordinal, and $o(\mathrm{ord}(r))$ is order isomorphic to r. We have $\mathrm{ord}(r) =$

ord(Ord(r)), whenever r is a well-order. This means that ord is a bijection between the collection of B-ordinals and v-ordinals (these collections are not sets).

If we have an order r on some set I, and a family $(s_i)_{i\in I}$ of orders indexed by I, we can define the ordinal sum $\sum s_i$ and (if I is well ordered) the lexicographic product $\prod s_i$; these operations yield an order and are compatible with order isomorphisms, so that we can define the sum and product of order-types. The result is called the ordinal sum and ordinal product (this gives two different definitions of "ordinal sum", and what they define is in general not a B-ordinal). Given a family of v-ordinals x_i, one can consider ord($\sum o(x_i)$). This quantity could be a v-ordinal. So, we introduce the following sums and products:

- $\sum_s T_i$, $\prod_s T_i$, `disjointU`, `productb`, is the disjoint union or the cartesian product of sets;
- $\sum_c T_i$, $\prod_c T_i$, `csum`, `cprod`, is the cardinal sum or cardinal product of cardinal numbers;
- $\sum_r T_i$, $\prod_r T_i$, `order_sum`, `order_prod`, is the ordinal sum or lexicographic product of ordered sets;
- $\sum_t T_i$, $\prod_t T_i$, `OT_sum`, `OT_prod`, is the sum and product of order-types;
- $\sum_o T_i$, $\prod_o T_i$, `osum`, `oprod`, is the sum and product of ordinal numbers.

Let's recall the definitions of $\sum_r X_i$ and $\prod_r X_i$. We consider an index set I and a family of sets X_i (for $i \in I$). By definition $\sum_s X_i$ is the set of all x which are pairs (x_1, x_2) with $x_2 \in I$ and $x_1 \in X_{x_2}$, and $\prod_s X_i$ is the set of all functional graphs x such that $x_i \in X_i$ for $i \in I$. Assume now that I is an ordered set (well-ordered in the case of a product). Formally, we have a graph r, a substrate I and a relation $\le_I$. Similarly, each E_i is ordered by $\le_i$. Now $\sum_r X_i$ is $\sum_s X_i$ ordered by $x \le y$ whenever either $x_2 <_I y_2$ or $x_2 = y_2 = i$ and $x_1 \le_i y_1$, and $\prod_r X_i$ is $\prod_s X_i$ ordered by by $x < y$ whenever $x_j <_j y_j$, where j is the least (for $\le_I$) index such that $x_j \ne y_j$.

Let `C2` be the canonical doubleton; this is some set with two distinct elements `C0` and `C1`; it can be well-ordered by saying that `C0` is less than `C1`. We shall sometimes write 0, 1 and 2 instead of `C0`, `C1`, and `C2`.

Consider two sets E and F; we can form the family X indexed by 2, such that $X_0 = E$ and $X_1 = F$. An element of the disjoint union $E +_s F$ is a pair $(u, 0)$ with $u \in E$ or a pair $(v, 1)$ with $v \in F$. We denote by $E +_r F$ this set ordered by: $(u, 0) \le (v, 1)$, $(u, 0) \le (u', 0)$, $(v, 1) \le (v', 1)$, whenever $u \le_E u'$ and $v \le_F v'$. The product $E \times_s F$ is the set of all functions f defined on 2 such that $f(0) \in E$ and $f(1) \in F$, it is canonically isomorphic to $E \times F$. We denote by $E \times_r F$ the lexicographic ordering on this set. If (x, i) and (y, j) are in $E \times F$ we say $(x, i) \le (y, j)$ if either $i <_F j$ or $i = j$ and $x \le_E y$.

The Bourbaki definition is «We denote by $\lambda + \mu$ (resp. $\mu\lambda$) the ordinal sum (resp. ordinal product) of the family $(\xi_\iota)_{\iota\in J}$ where $J = \{\alpha, \beta\}$ is a set with two distinct elements, ordered by the relation whose graph is $\{(\alpha, \alpha), (\alpha, \beta), (\beta, \beta)\}$, and where $\xi_\alpha = \lambda$ and $\xi_\beta = \mu$.»

It is a long established tradition to use Greek letters for ordinals, since these objects are a bit "weird". Bourbaki uses upper case letters like E, I, J for sets, and lower case letters x, y, z for elements, Greek letters like ι, κ for indices. However, there is no difference between a set, an element and an index. Thus, instead of $(\xi_\iota)_{\iota\in J}$, one could write $(x_i)_{i\in I}$. The definition above makes the ordinal sum depend on three constants, α, β and J. In particular, associativity cannot be written as $\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$, other letters have to be used. (Bourbaki soon forgets this and writes $x = \alpha + 0 = \alpha.1 = \alpha$.) It has to be noted that the sum $\lambda +_r \mu$ (this is an order) depends on J, but if J is order-isomorphic to $K = \{\gamma, \delta\}$, if x is defined by $x_\gamma = \xi_\alpha$, $x_\delta = \xi_\beta$, then the sums of ξ and x (considered as orders) are order-isomorphic, thus (considered as order-types) are equal (we have shown this property for cardinals, we do not need it for ordinals). Thus: the sum of two order-types does not depend on these constants.

11.2 Basic Properties

If I and each E_i are well-ordered, so is the ordinal sum (this has been proved above); if moreover I is finite, then the product is also well-ordered (for the converse, see Exercises 2.9 (page 577) and 2.11 (page 578)). (Proof by induction on the number of elements of I. Let i be the least element of I, X a non-empty set, u an element of X whose i-th component is minimal, X' the set of all $x \in X$ such that $x_i = u_i$, X'' the set of restrictions of elements of X' to $I - \{i\}$. By induction X'' has a least element, that is the restriction of the least element of X).

In particular the ordinal sum or product of two well-ordered sets are well-ordered.

```
Lemma orprod_wor r g:
  worder_on r (domain g) ->  worder_fam g -> finite_set (substrate r) ->
  finite_set (substrate r) ->
  worder (order_prod r g). (* 97 *)
Lemma orprod2_wor r r':
  worder r -> worder r' -> worder (order_prod2 r r').
Lemma orsum2_wor r r':
  worder r -> worder r' -> worder (order_sum2 r r').
```

Exercise 2.14(c) (page 583) says «Let α be an ordinal. Show that the relation "ξ is an ordinal and $\xi \leq \alpha$" is collectivizing in ξ, and that the set O_α of ordinals $< \alpha$ is a well-ordered set such that $\mathrm{Ord}(O_\alpha) = \alpha$. We shall often identify O_α with α.» The set of ordinals $\leq \alpha$ is denoted by O'_α. With Bourbaki's notations, the relation "ξ is an ordinal and $\xi \leq \alpha$" is equivalent to "ξ is the order-type of a segment of α", while $\xi < \alpha$ is equivalent to "ξ is the order-type of a segment S_x of α". These two relations are thus collectivizing. Moreover, the mapping $\xi \mapsto x$ is an order-isomorphism $O_\alpha \to \alpha$. As a consequence O_α is well-ordered and its ordinal is α. (See page 686 for the initial implementation using Bourbaki ordinals.)

In the case of von Neumann ordinals, we have the trivial result $\xi <_{\mathrm{ord}} \alpha \iff \xi \in \alpha$ and $\xi \leq_{\mathrm{ord}} \alpha \iff \xi \in \alpha^+$ (where $\alpha^+ = \alpha \cup \{\alpha\}$). We thus have $O_\alpha = \alpha$ and $O'_\alpha = \alpha^+$. Note that the restriction of $\leq_{\mathrm{ord}}$ to α is $o(\alpha)$, thus is a well-ordering and its ordinal is α.

```
Lemma set_ord_lt_prop3a a: ordinalp a ->  ole_on a = ordinal_o a.
Lemma set_ord_lt_prop3 a: ordinalp a -> ordinal (ole_on a) = a.
```

We have shown that a total ordering of a finite set is a well-ordering, and that two totally ordered finite sets are isomorphic if they have the same cardinal. Thus, to each finite cardinal is associated a unique ordinal. Let n be an integer, and consider the interval $[0, n[$ (with the ordering induced by that of $\mathbf{N}$). It has n elements, and its ordinal is called the n-th ordinal. If we define a cardinal as the least ordinal equipotent to it, it follows that $n = \mathrm{ord}([0, n[)$.

```
Lemma finite_ordinal1 n: natp n -> ordinal (Nint_co n) = n.
```

Consider the set of ordinals $< \omega$, ordered by $\leq_{\mathrm{ord}}$; this is the set of all integers, ordered by $\leq_{\mathbf{N}}$. The ordinal of this set is ω.

```
Lemma ord_omega_pr: ordinal Nat_order = omega0.
Lemma olt_omegaP x: x <o omega0 <-> natp x.
Lemma clt_omegaP x: x <c omega0 <-> natp x.
Lemma omega_nz: omega0 <> \0o.
Lemma olt_0omega: \0o <o omega0.
Lemma olt_1omega: \1o <o omega0.
Lemma ole_2omega: \2o <=o omega0.
Lemma olt_2omega: \2o <o omega0.
```

We have $o(\emptyset) = \emptyset$ (because the only order on the empty set is empty). From $\mathrm{ord}(o(x)) = x$ we get $\mathrm{ord}(\emptyset) = 0$ since 0 is an ordinal. More generally $\mathrm{ord}(x) = 0$ if x is an order with empty substrate. Conversely if $\mathrm{ord}(x) = 0$ there is a bijection between the substrate of x and the empty set, so the substrate is empty. A successor is never zero.

```
Lemma ordinal_o_set0: ordinal_o emptyset = emptyset.
Lemma ordinal0_pr: ordinal emptyset = \0o.
Lemma ordinal0_pr1 r: substrate r = emptyset -> ordinal r = \0o.
Lemma ordinal0_pr2 r: worder r -> ordinal r = \0o -> substrate r = emptyset.
Lemma osucc_nz x : osucc x <> \0o.
```

All singletons are well-ordered, their ordinal is 1.

```
Lemma ordinal1_pr x: ordinal (singleton (J x x)) = \1o.
Lemma set1_ordinal r: order r -> singletonp (substrate r) ->
  ordinal r = \1o.
```

11.3 Operations on ordinals

The ordinal of the order sum (resp. order product) of the natural orderings of a family of ordinals will be called the *ordinal sum* (resp. *ordinal product*) of the family. It is defined if the index set is well-ordered, and finite in the case of a product.

```
Definition osum r g :=
   ordinal (order_sum r (Lg (domain g) (fun z => (ordinal_o (V g z))))).
Definition oprod r g :=
   ordinal (order_prod r (Lg (domain g) (fun z => (ordinal_o (V g z))))).
Definition osum2 a b := ordinal (order_sum2 (ordinal_o a) (ordinal_o b)).
Definition oprod2 a b := ordinal (order_prod2 (ordinal_o a) (ordinal_o b)).

Notation "a +o b" := (osum2 a b) (at level 50).
Notation "a *o b" := (oprod2 a b) (at level 40).

Lemma OS_sum r g: worder_on r (domain g) -> ordinal_fam g ->
  ordinalp (osum r g).

Lemma OS_prod r g: worder_on r (domain g) -> ordinal_fam g  ->
  finite_set (substrate r)  -> ordinalp (oprod r g).
```

We state some properties of the sum or product of two ordinals.

```
Lemma osum2_rw a b:
  a +o b = osum canonical_doubleton_order (variantLc a b).
Lemma opro2_rw a b:
```

```
  a *o b = oprod canonical_doubleton_order (variantLc b a).
Lemma OS_sum2  a b: ordinalp a -> ordinalp b -> ordinalp (a +o b).
Lemma OS_prod2 a b: ordinalp a -> ordinalp b -> ordinalp (a *o b).
```

In a sum or a product, one can replace an item by an item which is order isomorphic to it.. We give some variants of this property[1] for instance, if E_i is a well-order for every index i of a well-ordered set we have

$$\mathrm{ord}(\sum_r E_i) = \sum_o \mathrm{ord}(E_i)$$

```
Lemma orsum_invariant1 r r' f g g':
  order_on r (domain g) ->
  order_on r' (domain g') ->
  order_isomorphism f r r' ->
  (forall i, inc i (substrate r) -> (Vg g i) \Is (Vg g' (Vf f i))) ->
  (order_sum r g) \Is (order_sum r' g'). (* 58 *)
Lemma orprod_invariant1 r r' f g g':
  worder_on r (domain g) ->
  order_on r' (domain g') ->
  order_isomorphism f r r' ->
  (forall i, inc i (substrate r) -> (Vg g i) \Is (Vg g' (Vf f i))) ->
  (order_prod r g) \Is (order_prod r' g').   (* 94 *)

Lemma orsum_invariant2 r g g':
  order r -> substrate r = domain g ->
  substrate r = domain g' ->
  (forall i, inc i (substrate r) -> (Vg g i) \Is (Vg g' i)) ->
  (order_sum r g) \Is (order_sum r g').
Lemma orprod_invariant2 r g g':
  worder r -> substrate r = domain g -> fgraph g ->
  substrate r = domain g' -> fgraph g' ->
  (forall i, inc i (substrate r) -> (Vg g i) \Is (Vg g' i)) ->
  (order_prod r g) \Is (order_prod r g').

Lemma orsum_invariant3 r g:
  worder_on r (domain g) -> worder_fam g ->
  ordinal (order_sum r g) =
    osum r (Lg (substrate r) (fun i => ordinal (Vg g i))).
Lemma orprod_invariant3 r g:
  worder_on r (domain g) -> worder_fam g ->
  finite_set (substrate r) ->
  ordinal (order_prod r g) =
    oprod r (Lg (substrate r) (fun i => ordinal (Vg g i))).

Lemma orsum_invariant4 r1 r2 r3 r4:
  r1 \Is r3 -> r2 \Is r4 ->
  (order_sum2 r1 r2) \Is (order_sum2 r3 r4).
Lemma orprod_invariant4 r1 r2 r3 r4:
  r1 \Is r3 -> r2 \Is r4 ->
  (order_prod2 r1 r2) \Is (order_prod2 r3 r4).

Lemma orsum_invariant5 r1 r2 r3: worder r1 -> worder r2 ->
  (order_sum2 r1 r2) \Is r3 ->
```

[1] The axiom of choice is used to select an isomorphism for each i; we could avoid it by passing the family of isomorphisms as an argument. In the case of well-orderings, this use of AC could be avoided, as there is a unique isomorphism.

```
  (ordinal r1) +o (ordinal r2) = ordinal r3.
Lemma orprod_invariant5 r1 r2 r3: worder r1 -> worder r2 ->
  (order_prod2 r1 r2) \Is r3 ->
  (ordinal r1) *o (ordinal r2) = ordinal r3.
```

Assume that a and b are two ordinals. We have

$$\sum_{i\in I} a_i = a\cdot b \tag{11.1}$$

where the left hand side is an ordinal sum over a well-ordered set I, order-isomorphic to $o(b)$, of a constant family of ordinals, all equal to a.

```
Lemma oprod_pr1 a b r:
  ordinalp a -> ordinalp b -> (ordinal_o b) \Is r ->
  a *o b = osum r (cst_graph (substrate r) a).
```

We show here

$$\sum_{\emptyset} x_i = 0, \quad \prod_{\emptyset} x_i = 1, \quad \sum_{i\in\{a\}} x_i = x_a, \quad \prod_{i\in\{a\}} x_i = x_a. \tag{11.2}$$

```
Lemma osum_set0 r g:  domain g = emptyset ->  osum r g = \0o.
Lemma oprod_set0 r g:  domain g = emptyset ->  oprod r g = \1o.

Lemma osum_set1 r g i:
  order_on r (domain g) ->
  substrate r = singleton i -> ordinalp (Vg g i) ->
  osum r g = Vg g i.
Lemma oprod_set1 r g i:
  order_on r (domain g) ->
  substrate r = singleton i ->  ordinalp (Vg x i) ->
  oprod r g = Vg g i. (* 56 *)
```

We show here that an ordinal sum remains unchanged if zero terms are removed. In a similar fashion, one can remove ones in a product. In fact, consider $\prod E_i$, assume E_j is one (or an ordering whose substrate is a singleton) for $j \in I-J$; we know that the restriction to J is a bijection from $\prod_I E_i$ to $\prod_J E_i$. It is clearly an order isomorphism.

```
Lemma osum_unit1 r g I:
  orsum_ax r g -> sub I (domain g) ->
  (forall i, inc i ((domain g) -s I) -> Vg g i = emptyset) ->
  (order_sum r g) \IS (order_sum (induced_order r I) (restr g j)).
Lemma oprod_unit1 r g I:
  orprod_ax r g -> sub I (domain g) ->
  (forall i, inc i ((domain g) -s I) ->  singletonp (substrate (Vg g i))) ->
  (order_prod r g) \Is (order_prod (induced_order r I) (restr g I)). (* 80 *)
Lemma osum_unit2 r g I:
  worder_on r (domain g) -> ordinal_fam g ->
  sub I (domain g) ->
  (forall i, inc i ((domain g) -s I) -> Vg g i = \0o) ->
  osum r g = osum (induced_order r I) (restr g I).
Lemma oprod_unit2 r g jI:
  worder_on r (domain g) -> ordinal_fam g ->
  sub I (domain g) ->
  finite_set (substrate r) ->
  (forall i, inc i ((domain g) -s I) -> Vg g i = \1o) ->
  oprod r g = oprod (induced_order r I) (restr g I).
```

We show associativity of the ordinal sum and product:

$$\sum_{\iota\in I} E_\iota = \sum_{\lambda\in L}\Big(\sum_{\iota\in J_\lambda} E_\iota\Big), \qquad \prod_{\iota\in I} E_\iota = \prod_{\lambda\in L}\Big(\prod_{\iota\in J_\lambda} E_\iota\Big), \qquad \text{where } I = \sum_{\lambda\in L} J_\lambda. \tag{11.3}$$

We give two proofs; in the first proof we consider order-types (thus show that two ordered sets are isomorphic by providing the isomorphism), and in the second proof we show that some ordinals are the same. In both cases, I is the ordinal sum of the family J_λ. In the second proof, this set has to be well-ordered (and finite in the case of a product); thus we assume that L and J_λ are well-ordered (and finite in the case of a product).

```
Lemma orsum_assoc_iso r g r' g':
  orsum_ax r g ->  orsum_ax r' g' ->
  r = order_sum r' g' ->
  let order_sum_assoc_aux :=
    fun l =>
    order_sum (Vg g' l) (Lg (substrate (Vg g' l)) (fun i => Vg g (J i l))) in
  let  order_sum_assoc :=
    order_sum r' (Lg (domain g')  order_sum_assoc_aux)
  in order_isomorphism (Lf (fun x=> J (J (P x) (P (Q x))) (Q (Q x)))
      (sum_of_substrates g) (substrate (order_sum_assoc)))
  (order_sum r g) (order_sum_assoc).    (* 92 *)

Lemma orprod_assoc_iso r g r' g':
  orprod_ax r g ->  orsum_ax r' g' ->
  r = order_sum r' g' ->
  worder r' ->
  (forall i, inc i (domain g') -> worder (Vg g' i)) ->
  let order_sum_assoc_aux :=
    fun l =>
    order_prod (Vg g' l) (Lg (substrate (Vg g' l)) (fun i => Vg g (J i l))) in
  let  ordinal_prod_assoc :=
    order_prod r' (Lg (domain g') order_sum_assoc_aux)
  in order_isomorphism (Lf
     (fun z  => Lg (domain g') (fun l  =>
       Lg (substrate (Vg g' l)) (fun j => Vg z (J j l))))
      (prod_of_substrates g) (substrate (ordinal_prod_assoc)))
  (order_prod r g) (ordinal_prod_assoc). (* 112 *)

Lemma osum_assoc1  r g r' g':
  worder_on r (domain g) ->
  worder_on r' (domain g') ->
  ordinal_fam g -> worder_fam g' ->
  r = order_sum r' g' ->
  let order_sum_assoc_aux :=
    fun l =>
    osum (Vg g' l) (Lg (substrate (Vg g' l)) (fun i => Vg g (J i l))) in
  osum r g = osum r' (Lg (domain g') (order_sum_assoc_aux)).
Lemma oprod_assoc1 r g r' g':
  worder_on r (domain g)  ->
  worder_on r' (domain g') ->
  ordinal_fam g -> worder_fam g' ->
  r = order_sum r' g' ->
  finite_set (substrate r) ->finite_set (substrate r') ->
  (forall i, inc i (domain g') ->  finite_set (substrate (Vg g' i))) ->
  let order_prod_assoc_aux :=
```

```
    fun l =>
    oprod (Vg g' l) (Lg (substrate (Vg g' l)) (fun i => Vg g (J i l))) in
  oprod r g = oprod r' (Lg (domain g') order_prod_assoc_aux).
```

Recursive definitions. Let a and b be two ordinals. Assume $x < a + b$. There are two cases: $x < a$ or $a \le x$. In the second case, $x = a + c$, where $c = x - a$ and $c < b$. Assume now $x < a \cdot b$. We can write $x = b \cdot q + r$, with $r < b$. Obviously $q < a$. We can replace $<$ by $\in$ everywhere, and get:

$$a + b = a \cup \{a + t, t \in b\}, \qquad a \cdot b = \{a \cdot i + j, i \in b, j \in a\}. \tag{11.4}$$

It happens that proving these relations is actually easier than defining and studying subtraction and division. By induction on b, for fixed a, we may assume that this relation holds, whenever $b < c$, and we show it for $b = c$. The left hand side of the equality is the ordinal of an ordered set E (one of $a +_s c$ or $a \times_s c$). We denote by f the order isomorphism of E into its ordinal; the characteristic property of this function is (see page 56) that, if for any $x \in \mathrm{E}$, $f'(x)$ is the set of all $f'(y)$ for $y < x$, then $f = f'$.

Consider first the case of the sum. Here $\mathrm{E} = a +_s c$. An element x of E is a pair, either $(u, 0)$ with $u \in a$ or $(v, 1)$ with $v \in c$. If $x = (u, 0)$ then $y < x$ is equivalent to $y = (u', 0)$ with $u' < u$. In the second case, it is equivalent to $y = (u', 0)$, with u' arbitrary in a, or $y = (v', 1)$ with $v' < u'$. We define $f'(x)$ to be u in the first case, $a + v$ in the second case. We have $f'(y) = u'$ or $f'(y) = a + v'$. Since $a \cup \{a + t, t \in c\}$ is the image of f', it suffices to show $f = f'$. The relation $y < x \iff f'(y) \in f'(x)$ is nearly obvious. If $x = (v, 1)$ and $y = (u', 0)$ we use minimality of c, i.e., we rewrite $f'(x) = a + v$ as a union, and $u' = f'(y)$ is a member of this union.

In the case of a product, the set E is (a variant of) the cartesian product $a \times c$, with the lexicographic order. Assume $x = (u, v)$. We define $f''(x) = a \cdot v + u$. The image of this function is the RHS of (11.4), so that it suffices to show that if $x \in \mathrm{E}$, $f''(x)$ is the set of all $f''(y)$ for $y < x$. Assume $z \in f''(x)$. The LHS of (11.4) says $z \in a \cdot v$ or $z = a \cdot v + u'$ with $u' < u$. In the second case $z = f''(u', v)$, and (u', v) is less than (u, v). In the first case, we use minimality of c so that $z = f''(u', v')$ with $u' < a$ and $v' < v$. Here again (u', v') is less than (u, v).

In the code that follows, we write $a \cdot b = \{a \cdot \mathrm{pr}_2 z + \mathrm{pr}_1 z, z \in a \times b\}$, so that z is the pair (j, i).

```
Lemma osum_rec_def a b: ordinalp a -> ordinalp b ->  (* 75 *)
  a +o b = a \cup fun_image b (osum2 a).
Lemma oprod_rec_def a b: ordinalp a -> ordinalp b ->(* 63 *)
  a *o b = fun_image (a\times b) (fun z => a *o (Q z) +o (P z)).
```

One deduces $0 \cdot x = x \cdot 0$ (since the product $a \times b$ is empty). This relation is however true for any x (ordinal or not), so we give an alternate proof. We have

$$0 + x = x + 0 = x; \qquad 1 \cdot x = x \cdot 1 = x; \qquad x^+ = x + 1. \tag{11.5}$$

```
Lemma oprod_zero r g:
  (exists2 i, inc i (domain g) & substrate (Vg g i) = emptyset) ->
   order_prod r g = emptyset.
Lemma oprod0r x: x *o \0o = \0o.
Lemma oprod0l x: \0o *o x = \0o.

Lemma osum0r a: ordinalp a -> a +o \0o = a.
Lemma osum0l a: ordinalp a -> \0o +o a = a.
Lemma oprod1r a: ordinalp a -> a *o \1o  = a.
Lemma oprod1l a: ordinalp a -> \1o *o a  = a.
Lemma osucc_pr a: ordinalp a -> a +o \1o = osucc a.
```

The following relations follow from (11.4). By induction, we may assume the relation true for any $c' < c$. We rewrite (11.4) [four times in case (a) and (c), five times in case (b)]. In case (b) and (c) we rewrite (a); and in case (c) we rewrite (b). The result is then obvious.

$$a+(b+c)=(a+b)+c; \qquad a\cdot(b+c)=a\cdot b+a\cdot c; \qquad a\cdot(b\cdot c)=(a\cdot b)\cdot c. \tag{11.6}$$

```
Lemma osumA a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  a +o (b +o c) = (a +o b) +o c.
Lemma oprodD a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  c *o (a +o b) = (c *o a) +o (c *o b).
Lemma oprodA a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
   a *o (b *o c) = (a *o b) *o c.
```

Utilities for the Cantor Normal Form. If X is a functional term, we say that X is an ordinal below n if $i < n$ implies that X_i is an ordinal; we say that X and Y are equal below n if $i < n$ implies $X_i = Y_i$ (here i and n are natural numbers). We define the finite sum $\sum X_i$ by induction via $S_0 = 0$, $S_{n+1} = X_n + S_n$, and the finite product $\prod X_i$ via $P_0 = 1$ and $P_{n+1} = P_n \cdot X_n$. Note the ordering: X_n is the first term of the sum and the last factor of the product.

```
Definition osumf (f;fterm) :=
  induction_term (fun n v => f n +o v) \0o.
Definition oprodf (f:fterm) :=
  induction_term (fun n v => v *o f n) \1o.
Definition ord_below (f:fterm) n  := (forall k, k<c n -> ordinalp (f k)).
Definition same_below (e e': fterm) n:= (forall i, i <c n -> e i = e' i).
```

We start with trivial properties (for instance, if p holds below $n+1$ it holds below n and for n).

```
Lemma true_below_rec (P:property) n: natp n ->
  (forall i, i <c (csucc n) -> P i) -> (forall i, i<c n -> P i) /\ P n.
Lemma osum_f0 f: osumf f \0c = \0o.
Lemma oprod_f0 f: oprodf f \0c = \1o.
Lemma osum_fS  n f: natp n ->
  osumf f (csucc n) = f n +o (osumf f n).
Lemma oprod_fS n f: natp n ->
  oprodf f (csucc n) = (oprodf f n) *o (f n).
Lemma osum_f1 f:  ordinalp (f \0c) -> osumf f \1c = f \0c.
Lemma oprod_f1 f: ordinalp (f \0c) -> oprodf f \1c = f \0c.
Lemma osumf_exten f g n: natp n -> same_below f g n ->
  osumf f n = osumf g n.
Lemma oprodf_exten f g n:  natp n -> same_below f g n ->
  oprodf f n = oprodf g n.
```

If X is an ordinal below n, then S_n and P_n are ordinals, and the associativity theorem holds trivially.

```
Lemma OS_osumf n f: natp n ->  ord_below f n ->
  ordinalp (osumf f n).
```

```
Lemma OS_oprodf n f: natp n ->  ord_below f n ->
  ordinalp (oprodf f n).
Lemma osum_fA n m f:
  natp n -> natp m -> ord_below f (n +c m) ->
  osumf f (n +c m) =  (osumf (fun z => f (z +c n)) m) +o (osumf f n).
Lemma oprod_fA n m f:
  natp n -> natp m -> ord_below f (n  +c m) ->
  oprodf f (n +c m) =  (oprodf f n) *o (oprodf (fun z => f (z +c n)) m).
Lemma oprod_f1r p n: ord_below p (csucc n) -> natp n ->
    oprodf p (csucc n) = p \0c *o  oprodf (fun i => p (csucc i)) n.
Lemma osum_f1r p n: natp n -> ord_below p (csucc n) ->
    osumf p (csucc n) =  osumf (fun i => p (csucc i)) n +o p \0c.
```

11.4 Operations and ordering

We have already shown that, when α and β are ordinals, then

$$\alpha < \beta \iff \alpha^+ \le \beta \text{ and } \alpha < \beta^+ \iff \alpha \le \beta. \tag{11.7}$$

Some consequences.

```
Lemma olt_12: \1o <o \2o.
Lemma olt_01: \0o <o \1o.
Lemma olt_02: \0o <o \2o.

Lemma olt0S a: ordinalp a -> \0o <o (osucc a).
Lemma orge1P x: \2a <=o a <-> \1o <o a.
Lemma ord2_trichotomy a: ordinalp a ->
  [\/ a = \0o, a = \1o | \2o <=o a].
Lemma ord2_trichotomy1 a: \2o <=o a -> (a <> \0o /\ a <> \1o).
```

If $a \in b$ then $c + a \in c + b$ by (11.4). If moreover $0 \in c$ we get $c \cdot a \in c \cdot b$. We deduce that the product of two non-zero ordinals is non-zero and

$$a < b \implies c + a < c + b \text{ and } c \cdot a < c \cdot b \text{ (when } c \ne 0), \tag{11.8}$$

$$c + a \le c + b \implies a \le b \text{ and } c \cdot a \le c \cdot b \implies a \le b \text{ (when } c \ne 0). \tag{11.9}$$

```
Lemma osum_Meqlt a b c:
  a <o b -> ordinalp c -> (c +o a) <o (c +o b).
Lemma osum_Meqle a b c:
  a <=o b -> ordinalp c -> (c +o a) <=o (c +o b).

Lemma oprod_Meqlt a b c:
  a <o b -> \0o <o c -> (c *o a) <o (c *o b).
Lemma oprod_Meqle a b c:
  a <=o b -> ordinalp c -> (c *o a) <=o (c *o b).
Lemma oprod2_pos a b: \0o <o a -> \0o <o b -> \0o <o a *o b.
Lemma oprod2_nz a b: ordinalp a -> ordinalp b ->
   a <> \0o -> b <> \0o -> a *o b <> \0o.
Lemma osum_Meqler a b c: ordinalp a -> ordinalp b -> ordinalp c ->
   c +o a <=o c +o b -> a <=o b.
Lemma oprod_Meqler a b c: ordinalp a -> ordinalp b -> \0o <o c ->
   c *o a <=o c *o b -> a <=o b.
```

If $a \le b$ then $a+c \le b+c$ (assume this holds for any $c' < c$ and consider (11.4)).

(11.10) $$a \le b \text{ and } a' \le b' \implies a+a' \le b+b', \qquad a \le a+b, \qquad b \le a+b.$$

```
Lemma osum_Mleeq a b c:
  a <=o b -> ordinalp c -> (a +o c) <=o (b +o c).
Lemma osum_Mle0 a b:
  ordinalp a -> ordinalp b -> a <=o (a +o b).
Lemma osum_M0le a b:
  ordinalp a -> ordinalp b -> b <=o (a +o b).
Lemma osum_Mlele a b c d:
  a <=o b -> c <=o d -> (a +o c) <=o (b +o d).
```

The case of a product is similar (some quantities must be non-zero).

(11.11) $$a \le b \text{ and } a' \le b' \implies a\cdot a' \le b\cdot b', \qquad a \le a\cdot b, \qquad b \le a\cdot b.$$

```
Lemma oprod_Mleeq a b c:
  a <=o b -> ordinalp c -> (a *o c) <=o (b *o c).
Lemma oprod_Mlele a b c d:
  a <=o b -> c <=o d -> (a *o c) <=o (b *o d).
Lemma oprod_Mle1 a b:
  ordinalp a -> \0o <o b -> a <=o (a *o b).
Lemma oprod_M1le a b:
  \0o <o a -> ordinalp b-> b <=o (a *o b).
Lemma oprod_Meq1lt b c:
  \1o <o b -> \0o <o c -> c <o (c *o b).
```

Since the order is total we have

(11.12) $$(\mu+\alpha<\mu+\beta \text{ or } \alpha+\mu<\beta+\mu \text{ or } \mu\cdot\alpha<\mu\cdot\beta \text{ or } \alpha\cdot\mu<\beta\cdot\mu) \implies \alpha<\beta.$$

```
Lemma osum_Meqltr a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  (c +o a) <o (c +o b) -> a <o b.
Lemma osum_Mlteqr a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  (a +o c) <o (b +o c) -> a <o b.
Lemma oprod_Meqltr a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  (c *o a) <o (c *o b) -> a <o b.
Lemma oprod_Mlteqr a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  (a *o c) <o (b *o c) -> a <o b.
```

We may simplify on the left.

(11.13) $$c+a=c+b \implies a=b, \text{ and } c\cdot a=c\cdot b \implies a=b, \text{ when } c\neq 0.$$

We give other similar lemmas. For instance $x+y=0$ says $x=y=0$, $x\cdot y=1$ says $x=y=1$. If $a\cdot b=a$, then $b=1$; if $x\cdot y=2$, then one of x and y is one (so that the other is two).

```
Lemma osum2_simpl a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
```

```
  c +o a = c +o b -> a = b.
Lemma oprod2_simpl a b c:
  ordinalp a -> ordinalp b -> \0o <o c ->
  c *o a = c *o b -> a = b.
Lemma oprod2_simpl1 a c: ordinalp a -> \0o <o c -> c *o a = c  -> a = \1o.
Lemma osum2_a_ab a b:
  ordinalp a -> ordinalp b -> a +o b = a -> b = \0o.
Lemma osum2_zero a b: ordinalp a -> ordinalp b ->
  a +o b = \0o -> (a = \0o /\ b = \0o).
Lemma oprod2_a_ab a b: \0o <o a -> ordinalp b ->
  a *o b = a -> b = \1o.
Lemma oprod2_one a b: ordinalp a -> ordinalp b ->
  a *o b = \1o -> (a = \1o /\ b = \1o).
Lemma oprod2_two a b: ordinalp a -> ordinalp b ->
  a *o b = \2o -> (a = \1o \/ b = \1o).
```

Other useful relations.

$$(x+y)^+ = x+y^+, \qquad x+x\cdot y = x\cdot(1+y), \qquad x\cdot y+x = x\cdot(y+1). \tag{11.14}$$

```
Lemma osum2_succ a b: ordinalp a -> ordinalp b ->
   osucc (a +o b) = a +o (osucc b).
Lemma oprod2_nsucc a b: ordinalp a -> ordinalp b ->
   a +o (a *o b) = a *o (\1o +o b).
Lemma oprod2_succ a b:  ordinalp a -> ordinalp b ->
   a *o (osucc b) = (a *o b) +o a.
Lemma osum_fnz n X: natp n ->  ord_below ax X n ->
   (exists2 i, i <c n & X i <> \0o) -> \0o <o osumf X n.
Lemma osum_fnz n X: natp n ->  ord_below X n ->
   (exists2 i, i <c n & X i <> \0o) -> \0o <o osumf X n.
```

It follows from (11.14), by induction, that the cardinal sum and product of two integers are the ordinal sum and product. Since $n <_{\text{ord}} \omega$ is equivalent to $n \in \mathbf{N}$, it follows that the sum and product of two ordinals $< \omega$ is $< \omega$. We have $a\cdot 2 = a+a$.

```
Lemma osum2_2int a b:
  natp a -> natp b -> a +o b = a +c b.
Lemma oprod2_2int a b:
  natp a -> natp b -> a *o b = a *c b.
Lemma osum2_lt_omega0 a b:
  a <o omega0 -> b <o omega0 -> (a +o b) <o omega0.
Lemma oprod2_lt_omega a b:
  a <o omega0 -> b <o omega0 -> (a *o b) <o omega0.
Lemma osum_11_2: \1o +o \1o = \2o.
Lemma ord_double a: ordinalp a -> a *o \2o = a +o a.
```

It follows by (11.4) that, if a is an integer, then $a+\omega = \omega$ and $a\cdot\omega = \omega$ if a is non-zero.

```
Lemma osum_int_omega n:
  n <o omega0 -> n +o omega0 = omega0.
Lemma oprod_int_omega n:
  n <o omega0 -> \0c <o n -> n *o omega0 = omega0.
```

We deduce an example of $a+b \neq b+a$, of $a\cdot b \neq b\cdot a$ and of $(a+b)\cdot c \neq a\cdot c+b\cdot c$. In particular (11.8) and (11.13) are wrong with c on the other side of the operator.

```
Lemma osum2_nc (a := \1o) (b := omega0):
   a +o b <> b +o a.
Lemma oprod2_nc (a := \2o) (b := omega0):
  a *o b <> b *o a.
Lemma osum2_nD (a:= \1o) (b:= \1o) (c:= omega0):
  (a +o b) *o c <> (a *o c) +o (b *o c).
```

Consider the equation $a+b=c$, where c is a fixed ordinal. If this equation holds then $a \le c$ and $b \le c$, so that a and b belong to the set c^+. We shall that for every a in c^+ there is a solution (obviously unique). So, the number of a for which there is some b may be infinite, for instance $n+\omega=\omega$ for every integer n. If b is fixed, the solution is no more unique, and in general does not exist. More precisely, let E_b to be the subset of c^+ formed of all a such that $a+b=c$ and E the subset of c^+ formed of all b such that E_b is non-empty. Then E_b is finite (we shall compute the cardinal later on). Proof. Let $f(b)$ be the intersection of E_b. If $b \in \mathrm{E}$, then E_b is a nonempty ordinal set, so $f(b) \in \mathrm{E}_b$ and $f(b)+b=c$. Now $b<b'$ implies $f(b')<f(b)$. Let F be the image of f. Now f is a strictly decreasing function between two well-ordered sets, so E is finite.

```
Lemma finite_rems c: ordinalp c ->
  finite_set (Zo (osucc c) (fun b => exists2 a, ordinalp a & c = a +o b)).
```

The cardinal of the ordinal sum (respectively product) of two ordinals is the cardinal sum (respectively product) of the cardinals of the arguments. First proof: $a+b$ and $a\cdot b$ are order isomorphic to some sets that have the desired cardinal. Second proof. By (11.4), $a+b$ is the disjoint union of a and a set equipotent to b, while $a\cdot b$ is equipotent to the cartesian product (the non-trivial point is to show injectivity of $(i,j)\mapsto a\cdot i+j$).

```
Lemma cardinal_of_ordinal r:
  worder r ->  (ordinal r) =c (substrate r).
Lemma osum_cardinal a b:
  ordinalp a -> ordinalp b ->
  cardinal (a +o b) = (cardinal a) +c (cardinal b).
Lemma oprod_cardinal a b:
  ordinalp a -> ordinalp b ->
  cardinal (a *o b) = (cardinal a) *c (cardinal b).
Lemma osum_cardinal_gen r X:
  worder_on r (domain X) -> ordinal_fam X ->
  cardinal (osum r X) = (csumb (domain X) (fun z => cardinal (Vg X z))).
```

Cardinal successor. Let C be an infinite cardinal and E its cardinal successor (E is the set of ordinals x such that $\mathrm{card}(x)\le \mathrm{C}$). Properties of infinite cardinals say that E is stable by addition and multiplication. Moreover, let x_i be a family of elements of E, indexed by a set I, whose cardinal is $\le$ C, and let $x=\sup x_i$. If F is the set of all x_i, then $x=\sup \mathrm{F}$, and $\mathrm{card}(\mathrm{F})\le \mathrm{C}$. Now $\mathrm{card}(x)\le \sum \mathrm{card}(x_i)\le \mathrm{C}^2=\mathrm{C}$, thus $x\in \mathrm{E}$.

```
Section InfiniteNormal.
Variable C: Set.
Hypothesis iC: infinite_c C.
Let E := (cnext C).

Lemma cnext_sum x y: inc x E -> inc y E -> inc (x +o y) E.
Lemma cnext_prod x y: inc x E -> inc y E -> inc (x *o y) E.
```

```
Lemma cnext_leomega x: x <=o omega0 ->  inc x E.
Lemma cnext_zero: inc \0o E.
Lemma cnext_succ x: inc x E -> inc (osucc x) E.
Lemma cnext_sup F: cardinal F <=c C -> sub F E -> inc (\osup F) E.
End InfiniteNormal.
```

Assume that C is a cardinal (possibly finite). Then C is the greatest cardinal in E. We restate this as: assume that E is an infinite cardinal successor. Then, let C be the supremum of the set of cardinals that belong to E, it holds that C is an infinite cardinal and E is the successor of C.

```
Lemma cnext_pred c: cardinalp c -> c = \csup (Zo (cnext c) cardinalp).

Lemma cnext_pred_more E (c:= \csup (Zo E cardinalp)):
  (exists2 C, infinite_c C & E = cnext C) ->
  infinite_c c /\ E = cnext c.
```

We say that an ordinal is *countable* if it is a countable set. This is the same as $x \in \aleph_1$, where $\aleph_1$ is the cardinal successor of ω. It follows that the sum and product of two countable ordinals is countable. The supremum of a countable set of countable ordinals is a countable ordinal. Cantor says that an ordinal is of the *first class* if it is finite, of the *second class* if it is countable and infinite.

```
Definition countable_ordinal a := ordinalp a /\ countable_set a.
Definition aleph_one : cnext omega0.

Lemma aleph_oneP a: inc a aleph_one <-> countable_ordinal a.
Lemma osum2_countable a b:
   countable_ordinal a -> countable_ordinal b -> countable_ordinal (a +o b).
Lemma oprod2_countable a b:
   countable_ordinal b -> countable_ordinal b -> countable_ordinal (a *o b).
Lemma countable_ordinal_leomega a:
  a <=o omega0 ->  countable_ordinal a.
Lemma cardinal_omega2: cardinal omega0 = cardinal (omega0 +o omega0).
Lemma countable_one: countable_ordinal \1o.
Lemma countable_succ a:
   countable_ordinal a -> countable_ordinal (osucc a).
Lemma countable_ordinal_sup E:
  countable_set E -> (alls E countable_ordinal) ->
  countable_ordinal (\osup E).
```

Ordinal supremum. Assume that X is a set of ordinals, $x = \sup(X)$. Recall that x is the least ordinal such that $t \le x$ whenever $t \in X$. We say that Y is cofinal in X if Y is a subset of X and every element of X is bounded above by an an element of Y; we say that X and Y are mutually cofinal if each element of one set is bounded by an element of the other set. We state here additional properties.

- If X is an ordinal, then $X = x$ or X is the successor of x.
- If $x \notin X$, then x is a strict upper bound of X.
- X is a subset of an ordinal, namely x^+; hence either $x \in X$ or $X \subset x$.
- Either X is empty, or $x \in X$, or x is a limit ordinal.

- If $Y \subset X$, then $\sup Y \le x$.
- If X and Y are mutually cofinal , then sup(X) = sup(Y).
- if Y is cofinal in X then sup(X) = sup(Y).
- If $t < x$, then t is bounded above by an element of X (the same holds for cardinal supremum).

```
Definition ord_cofinal x y :=
  sub x y /\ forall a, inc a y -> exists2 b, inc b x & a <=o b.

Definition mutually_cofinal x y :=
   (forall a, inc a x -> exists2 b, inc b y & a <=o b) /\
   (forall a, inc a y -> exists2 b, inc b x & a <=o b).

Lemma ord_ub_sup1 a X: ordinalp a -> sub X a -> \osup X <=o a.
Lemma ord_sup_ordinal a (b:= \osup a): ordinalp a ->
  a = b \/ a = osucc b.
Lemma ord_sup_sub X:
  ordinal_set X ->  ~(inc (\osup X) X) ->
  forall x, inc x X -> x <o (\osup X).
Lemma oset_sub_ordinal X: ordinal_set X -> sub X (osucc (\osup X)).
Lemma ord_sup_sub' X:
  ordinal_set X ->  (inc (\osup X) X) \/ sub X (\osup X).
Lemma ord_sup_inVlimit X:
  ordinal_set X -> nonempty X ->
  inc (\osup X) X \/ limit_ordinal (\osup X).
Lemma ord_sup_M  x y:
  sub x y ->  ordinal_set y ->
  (\osup x) <=o (\osup y).
Lemma ord_sup_2cofinal x y:
  mutually_cofinal x y -> \osup x = \osup y.
Lemma ord_sup_2funI X f g:
  {inc X, f =1 g} ->
  \osup (fun_image X f) = \osup (fun_image X g).
Lemma ord_sup_1cofinal x y:
  ord_cofinal x y  ->  ordinal_set y -> \osup x = \osup y.
Lemma olt_sup A x: ordinal_set A -> x <o (\osup A) ->
  exists2 z, inc z A & x <o z.
Lemma clt_sup A x:
  cardinal_set A -> x <c \csup A -> exists2 z, inc z A & x <=c z.
```

Assume A cofinal in B, where B is a set of ordinals. This really means that A is cofinal for the order induced by $\le_{\text{ord}}$ on B. In this case A and B have the same supremum; we show here that the converse may be false, unless B is a limit ordinal.

Assume moreover that B is the cardinal successor of an infinite cardinal C. Then the ordinal of A (i.e., of $\le_{\text{ord}}$ restricted to A) is B (in fact, if α is the ordinal of A, we have card(α) = B, since α is equipotent to A; and card(A) < B implies that the supremum of A is in B, thus cannot be B; the conclusion follows as $\alpha \le_{\text{ord}}$ B and B is a cardinal).

```
Lemma ord_cofinal_p1 A B: ordinal_set B ->
   (ord_cofinal A B <-> cofinal (ole_on B) A).
Lemma ord_cofinal_p2 A B: limit_ordinal B -> sub A B ->
```

```
   (ord_cofinal A B <-> \osup A = \osup B).
Lemma ord_cofinal_p3 a: limit_ordinal a ->
  (\osup a = \osup  (osucc a) /\  ~(ord_cofinal a (osucc a))).
Lemma ord_cofinal_p4 A C (E:= cnect C): infinite_c C ->
  ord_cofinal A E -> cardinal A = E.
Lemma ord_cofinal_p5 A C (E:= cnext C): infinite_c C ->
  ord_cofinal A E -> ordinal (ole_on A) = E.
```

11.5 Ordinal subtraction and division

Consider two ordinals a and b. In Exercise 2.15 page 584, Bourbaki asks to show that $a \le b$ is equivalent to the existence of (a unique) c such that $a + c = b$, written $(-a) + b$. (see page 514 for an implementation). We shall write $b - a$ instead of the curious $(-a) + b$. In the case of von Neumann ordinals, c is the ordinal of the complement of a in b.

By normality of addition (see below), there is c such that $a + c \le b < a + c^+$. The last expression is also $(a + c)^+$, and we get $b \le a + c$, thus $b = a + c$. Since $x < c$ is equivalent to $a + x < a + c$, we deduce that $b - a$ is the set of all x such that $a + x \in b$ (the proof is by induction on b). We take this as the definition of subtraction (note that $b - a$ is transitive and its elements are ordinals, so is an ordinal). This definition gives zero if $a \ge b$ and a non-zero ordinal if $a < b$.

```
Definition odiff b a := Zo b (fun z => inc (a +o z) b).
Notation "x -o y" := (odiff x y) (at level 50).

Lemma OS_diff a b: ordinalp a -> ordinalp b -> ordinalp (b -o a).
Lemma odiff_wrong a b: b <=o a -> b -o a = \0o.
Lemma odiff_Mle a b: ordinalp a -> ordinalp b ->  (b -o a) <=o b.
Lemma odiff_pr a b: a <=o b ->
  (ordinalp (b -o a) /\ b = a +o (b -o a)).
Lemma odiff_pr2 a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  (a +o c = b) -> c = b -o a.
Lemma odiff_pos a b: a <o b -> \0o <o (b -o a).
Lemma odiff_pr1 a b: ordinalp a -> ordinalp b -> (a +o b) -o a = b.
```

We deduce: $x = 1 + x$ if and only if x is infinite. In this case, $x - 1 = x$ and $(x + 1) - 1$ is not x. However, if x is finite non-zero, we have $x = (x - 1) + 1 = (x + 1) - 1$. Note that $x \mapsto x - 1$ is strictly increasing.

```
Lemma osum_1inf a: omega0 <=o a -> \1o +o a = a.
Lemma oadd1_fix a: ordinalp a -> (\1o +o a = a <-> omega0 <=o a).
Lemma odiff_1inf a: omega0 <=o a -> a -o \1o = a.
Lemma odiff_pr1_wrong a: omega0 <=o a -> (a +o \1o) -o \1o <> a.
Lemma oprec_nat n: \0o <o n -> n <o omega0 ->  n = osucc (n -o \1o).
Lemma oprec_nat2 n: \0o <o n -> n <o omega0 ->
  (osucc n -o \1o) = osucc (n -o \1o).
Lemma oprec_Mlt a b: \0o <o a -> a <o b -> (a -o \1o) <o (b -o \1o).
```

Assume that f is a strictly increasing function. For any b, there is at most one y satisfying $f(y) \le b < f(y + 1)$.

```
Definition sincr_ofs (f: fterm) :=
  (forall x y, x <o y -> (f x) <o (f y)).
Lemma sincr_bounded_unique h y y' a:
   sincr_ofs h -> ordinalp y -> ordinalp y' ->
   (h y) <=o a -> a <o h (osucc y)  ->
   (h y') <=o a -> a <o (h (osucc y'))  ->
   y = y'.
```

Let a and b be two ordinals; consider the function $g(q) = b \cdot q$ and the relation

$$a = b \cdot q + r, \qquad r < b$$

The first relation is $r = a - g(q)$, and the second becomes $g(q) \le a < g(q+1)$. There is at most one solution q, thus at most one solution (q, r).

```
Definition odiv_pr0 a b q r :=
  [/\ ordinalp q, ordinalp r, a = (b *o q) +o r & r <o b].

Lemma odivision_unique a b q r q' r':
  ordinalp a -> ordinalp b  ->
  odiv_pr0 a b q r -> odiv_pr0 a b q' r' ->
  (q = q' /\ r = r').
```

The two quantities q and r are called the *quotient* and *remainder* in the division of a by b. Existence follows by normality of multiplication (see below). The Bourbaki argument is the following: there is c such that $a < b \cdot c$ (for instance $a+1$). Consider the cartesian product $b \times_s c$, lexicographically ordered, and its ordinal $b \cdot c$. There is an order isomorphism f, and $f^{-1}(a')$ is a pair (r', q') (here a' is a considered as an element of $b \cdot c$, r' is r considered as an element of b and q' is q considered as an element of c). One gets trivially $q < c$ (this is rarely used) and (this is more complicated) $a = b \cdot q + r$, see page 514 for details. The study of the function f has already been done and leads to (11.4). This makes existence of division trivial. On can replace "$a < b \cdot c$" by "b is non-zero".

```
Definition odiv_pr1 a b c q r :=
  odiv_pr0 a b q r /\ q <o c.
Definition oquorem a b :=
   select (fun z => a = b *o (P z) +o (Q z)) ((osucc a) \times b).
Definition oquo a b := P (oquorem a b).
Definition orem a b := Q (oquorem a b).

Lemma odivision_exists a b c:
  ordinalp b -> ordinalp c ->  a <o (b *o c) ->
  odiv_pr1 a b c (oquo a b) (orem a b).
Lemma oquoremP a b: ordinalp a  -> \0o <o b ->
  odiv_pr0 a b (oquo a b) (orem a b).
Lemma oquoremP2 a b q r: ordinalp a  -> \0o <o b ->
  odiv_pr0 a b q r -> q = (oquo a b) /\ r = (orem a b).
```

11.6 Normal ordinal functional symbols

We study here some properties of the interval $[a, b[$, where a and b are ordinals. In case $a = 0$, the interval is b. Assume $a < b$ so that the interval is non-empty. Then the supremum of the interval is $\sup b$; if b is a limit ordinal, this is b. Consider a property p, false for some non-zero ordinal. Then there exist y, such that $p(y)$ is false, y is non-zero, but p holds in the interval $[1, y[$.

```
Definition ordinal_interval a b := Zo b (fun z => a <=o  z).

Lemma ointvP b: ordinalp b -> forall a z,
  (inc z (ordinal_interval a b) <-> (a <=o z /\ z <o b)).
Lemma ointv_P0 b: ordinalp b -> forall z,
  (inc z (ordinal_interval \0o b) <-> z <o b).
Lemma ointv1 b: ordinalp b -> forall a,
  inc a (ordinal_interval \1o b) <-> (\0o <o a /\ a <o b).
Lemma ointv_pr1 b: ordinalp b ->
  ordinal_interval \0o b = b.
Lemma ointv_pr2 a b z:
  inc z (ordinal_interval a b) -> ordinalp z.
Lemma ointv_sup a b: a <o b ->
  \osup (ordinal_interval a b) =  \osup b.
Lemma ointv_sup1 a b: a <o b -> limit_ordinal b ->
  \osup (ordinal_interval a b) = b.
Lemma least_ordinal5  x (p: property):
 \0o <o x -> ~(p x) ->
 let y := least_ordinal (fun z => (~ (\0o <o z -> p z))) x in
  [/\ ordinalp y, (\0o <o y), ~(p y) &
    (forall z, inc z (ordinal_interval \1o y) -> p z)].
```

We say that f is an *ordinal functional symbol*, in short, OFS, if f is a functional term that maps ordinals onto ordinals. We also consider the case where $f(x)$ is an ordinal for $u \le x$. We consider the case where f is strictly increasing above u. Finally, if f is a function, we consider the case when f is strictly increasing in its source x. Two strictly increasing OFS that have the same range are functionally equal.

```
Definition ofs (f:fterm) := forall a, ordinalp a -> ordinalp (f a).
Definition ofsu (f:fterm) u := forall a, u <=o a -> ordinalp (f a).

Definition sincr_ofsu (f: fterm) u :=
  forall a b, u <=o a -> a <o b -> f a <o f b.
Definition sincr_ofn f x :=
  forall a b, inc a x -> inc b x ->
      a <o b -> (Vf f a) <o (Vf f b).

Lemma ofs_sincru f u: sincr_ofsu f u -> ofsu f u.
Lemma ofs_sincr f: sincr_ofs f -> ofs f.

Lemma sincr_incr f: sincr_ofs f  ->
   (forall a b, a <=o b -> f a <=o f b).
Lemma sincr_ofs_exten f1 f2:
  sincr_ofs f1 -> sincr_ofs f2 ->
  (forall x, ordinalp x -> exists2 y, ordinalp y & f1 x = f2 y) ->
  (forall x, ordinalp x -> exists2 y, ordinalp y & f2 x = f1 y) ->
  f1 =1o f2.
```

Consider the following three properties:

$$u \le a < b \implies f(a) < f(b); \tag{11.15}$$

(11.16) $$\sup_{u\le x<a}(f(x)) = f(a) \qquad (a \text{ limit ordinal});$$

(11.17) $$(\forall a \in \mathrm{X}, u \le a),\ \mathrm{X} \ne \emptyset \implies \sup_{x\in \mathrm{X}}(f(x)) = f(\sup_{x\in \mathrm{X}} x).$$

We say that f is a *normal ordinal functional symbol* if it is an OFS that satisfies (11.15) and (11.16); we can also consider the stronger condition (11.17). We say that f is a *normal function* if it is a function that satisfies the first two properties, with $u = 0$, and a in the source of the function.

```
Definition cont_ofn f x :=
   (forall a, inc a x -> limit_ordinal a ->
      Vf f a = \osup (Vfs f a)).
Definition normal_function f x y:=
  [/\ function_prop f x y, sincr_ofn f x & cont_ofn f x].

Definition normal_ofu_aux (f:fterm) u:=
  forall a, limit_ordinal a -> u <o a ->
    f a = \osup (fun_image (ordinal_interval u a) f).
Definition normal_of_aux (f:fterm) :=
  forall a, limit_ordinal a -> f a = \osup (fun_image a f).

Definition normal_ofs1 (f: fterm) u:=
  sincr_ofsu f u /\
  (forall X, (forall x, inc x X -> u <=o x) -> nonempty X ->
    \osup (fun_image X f) = f (\osup X)).
Definition normal_ofs2 (f:fterm) u:=
  sincr_ofsu f u /\ normal_ofu_aux f u.
Definition normal_ofs (f:fterm):=
  sincr_ofs f /\  normal_of_aux f.
```

According to (11.4), $x \mapsto a + x$ and $x \mapsto a \cdot x$ are normal (when $a > 0$ for the product).

```
Lemma osum_normal a: ordinalp a -> normal_ofs (fun z => a +o z).
Lemma oprod_normal a: \0o <o a -> normal_ofs (fun z => a *o z).
```

If (11.16) holds and $f(x) < f(x+1)$ for all x, then f is strictly increasing, thus is normal (consider the least y not satisfying (11.15); if y is $z+1$ we have $f(x) < f(z) < f(z+1)$, and if y is limit, $f(x) < f(x+1) \le f(y)$). The same holds for a function (provided that the source is a limit ordinal and the target an ordinal).

```
Lemma ord_sincr_cont_propu f u:
    (forall x, u <=o x -> f x <o f (osucc x)) ->
    normal_ofu_aux f u ->
    sincr_ofsu f u.
Lemma ord_sincr_cont_prop f:
    (forall x, ordinalp x -> f x <o f (osucc x)) ->
    (forall x, limit_ordinal x -> f x = \osup (fun_image x f)) ->
    sincr_ofs f.
Lemma ord_sincr_cont_propv f x y: limit_ordinal x -> ordinalp y ->
    function_prop f x y ->
    (forall a, inc a x -> Vf f a <o Vf f (osucc a)) ->
    cont_ofn f x ->
    normal_function f x y.
```

We show equivalence of (11.17) and (11.16), under (11.15). In fact, if f is strictly increasing, then (11.17) is trivial when X has a greatest element. Otherwise, sup X is a limit ordinal, so that (11.16) implies (11.17). Conversely, if x is limit, then x is the supremum of all $t < x$. In the case of a function, defined on an ordinal a, it can happen that $\sup X = a$, case where $f(\sup X)$ is undefined.

```
Lemma normal_ofs_equiv f u:
  normal_ofs1 f u <-> normal_ofs2 f u.
Lemma normal_ofs_equiv1 f:
  normal_ofs1 f \0o <-> normal_ofs f.
Lemma normal_ofs_equiv2  f a:
  ordinalp a -> normal_ofs f-> normal_ofs1 f a.
Lemma normal_function_incr f a b:
  ordinalp a -> ordinalp b -> normal_function f a b ->
  (forall u v, u <=o v -> v <o a -> Vf f u <=o Vf f v).
Lemma normal_function_equiv f a b X:
  ordinalp a -> ordinalp b -> normal_function f a b ->
  sub X a -> nonempty X ->
  (\osup X = a \/ Vf f (\osup X) = \osup (Vfs f X)).
```

Theorem 2 of [22] says: given a term P, a value a, there is a unique normal OFS f, such that $f(0) = a$ and $f(x+1) = \mathrm{P}(f(x))$. Uniqueness is obvious (if f is limit, $f(x)$ is given by equation (11.16)). We define f by induction via by $f(x) = p(f_{(x)})$, where $f_{(x)}$ is the restriction (as a functional graph) of f on x, and $p(\mathrm{F}) = a$ when F is empty, and $p(\mathrm{F})$ is the supremum of all $\mathrm{P}(\mathrm{F}(t))$, for t in the domain of F. In order for this function to be increasing, some hypotheses on P are needed; in particular, we need $\mathrm{P}(y) > y$, whenever y has the form $f(x)$. Since we do not know a priori the range of f, we shall assume that this holds for any y. We shall also assume P increasing.

```
Lemma normal_ofs_uniqueness1 f g (p:fterm) u:
   normal_ofs1 f u -> normal_ofs1 g u -> ordinalp u ->
   (forall x, u <=o x -> f (osucc x) = p (f x)) ->
   (forall x, u <=o x -> g (osucc x) = p (g x)) ->
   (f u = g u) ->
  (forall x, u <=o x -> f x = g x).
Lemma normal_ofs_uniqueness f g (p:fterm):
   normal_ofs f -> normal_ofs g ->
   (forall x,  ordinalp x -> f (osucc x) = p (f x)) ->
   (forall x,  ordinalp x -> g (osucc x) = p (g x)) ->
   (f \0o = g \0o) ->
   f =1o g.
Lemma normal_ofs_existence (p:fterm) a
  (osup := fun f => \osup (fun_image (domain f) (fun z => (p (Vg f z)))))
  (osupp:= fun f => Yo (domain f = \0o) a (osup f))
  (f:= transdef_ord osupp):
  (forall x, ordinalp x -> x <o p x) ->
  (forall x y, x <=o y -> p x <=o p y) ->
  ordinalp a ->
  [/\ normal_ofs f, f \0o = a &
   (forall x,  ordinalp x -> f (osucc x) = p (f x)) ].
```

If f is an OFS, then $f(x)$ is limit if x is limit. Composition of normal OFSs is normal.

```
Lemma normal_ofs_limit1 f u x: normal_ofs1 f u -> u <o x -> limit_ordinal x ->
```

```
  limit_ordinal (f x).
Lemma normal_ofs_limit f x: normal_ofs f ->  limit_ordinal x ->
  limit_ordinal (f x).
Lemma normal_ofs_compose1 f fb g gb:
 ordinalp fb -> ordinalp  gb -> fb <=o g gb ->
 normal_ofs1 f fb -> normal_ofs1 g gb -> normal_ofs1 (f \o g) gb.
Lemma normal_ofs_compose f g:
 normal_ofs f -> normal_ofs g -> normal_ofs (f \o g).
```

Theorem 3 of [22] states that if f is normal, then $x \le f(x)$. In fact it suffices for f to be strictly increasing: there is no least y such that $f(y) < y$ since, if $t = f(y)$, we have $t < y$ and $f(t) < t$. There is however a least y such that $x \le f(y)$. Assume f defined and strictly increasing above u. If for some t we have $t \le f(t)$, then whenever $t \le x$ we have $x \le f(x)$.

```
Lemma osi_gex x f: sincr_ofs f -> ordinalp x -> x <=o (f x).
Lemma normal_fn_unbounded f a x:
  normal_function f a a -> x<o a -> x <=o (Vf f x) /\ Vf f x <o a.
Lemma osi_gex1 x f:
   sincr_ofs f -> ordinalp x -> exists y,
      [/\ ordinalp y, x <=o (f y) &
        forall z, ordinalp z -> x <=o (f z) -> y <=o z].
Lemma osi_gexu f u t x:
  sincr_ofsu f u -> u <=o t -> t <=o f t -> t <=o x ->
   x <=o (f x).
```

We deduce: if x is a limit ordinal, then $a+x$ and $a\cdot x$ are limit (we assume $a \neq 0$ in the case of a product). Moreover $x \mapsto x - a$ is a normal OFS (for $x \ge a$). We deduce by composition that $x \mapsto f(a+x) - a$ is a normal OFS.

```
Lemma osum_limit x y: ordinalp x -> limit_ordinal y ->
  limit_ordinal (x +o y).
Lemma oprod_limit x y: \0o <o x -> limit_ordinal y ->
  limit_ordinal (x *o y).
Lemma odiff_normal a: ordinalp a -> normal_ofs1 (odiff ^~ a) a.
Lemma normal_shift f a: normal_ofs f -> ordinalp a ->
  normal_ofs (fun z => (f(a +o z) -o a)).
```

We extend Theorem 2 of [22] as: given a term P, a value a, a bound u, there is a unique normal OFS f defined for $x \ge u$, such that $f(u) = a$ and $f(x+1) = \mathrm{P}(f(x))$. Proof. Let g be the OFS satisfying the same recurrence with $g(0) = a$; take $f(x) = g(x-u)$. The non-trivial point is to show that f is normal.

```
Lemma normal_ofs_existence1 (p:fterm) a u:
  (forall x, ordinalp x -> x <o p x) ->
  (forall x y, x <=o y -> p x <=o p y) ->
  ordinalp a -> ordinalp u ->
  exists f,
  [/\ normal_ofs1 f u, f u = a &
   (forall x,  u <=o x -> f (osucc x) = p (f x)) ].
```

If a and b are ordinals, then $a+b$ is a successor if and only if either $b = 0$ (case where a has to be a successor) or b is a successor. Hence, the product of two successors is a successor. The converse holds; assume $a \cdot b = c+1$. Then a is non-zero. We have seen above that b is a successor. Write $a \cdot q + r = c$, so that $a \cdot q + (r+1) = c+1$; if a is limit then $r+1 < a$, so that $r+1$ is the remainder in the division of $c+1$ by a; but the remainder is zero since $a \cdot b = c+1$.

```
Lemma osum_succP a b: ordinalp a -> ordinalp b ->
  (osuccp (a +o b) <-> ((b = \0c /\ osuccp a) \/ osuccp b)).
Lemma oprod_succP a b: ordinalp a -> ordinalp b ->
  (osuccp  (a *o b) <-> osuccp a /\ osuccp b).Proof.
```

Assume f strictly increasing above u. Then $f(x)+y \le f(x+y)$ if $u \le x$ (consider the least y not satisfying this inequality, and use normality of addition when y is limit). We have in particular $y \le f(x+y)$. Set $b = u+1$, take $x = b$ and $y = b \cdot n$. Take the supremum for all finite n. Since multiplication is normal, the supremum of y and $x+y$ will be $c = b \cdot \omega$. Thus $c \le f(c)$. It follows $x \le f(x)$ whenever $c \le x$. This is Exercise 6.13(a) (page 657).

```
Lemma osum_increasing5 u f: ordinalp u ->
  (sincr_ofsu f u) ->
  ((forall x y, u <=o x -> ordinalp y -> f(x) +o y <=o f (x +o y))
  /\ (exists2 c, u <=o c & forall x, c <=o x -> x <=o f (x))).
```

Assume f normal, $f(0) \le x$. Let z be the least solution of $x \le f(z)$. Assume $x < f(z)$. Then z is non-zero, and is not limit (by normality), it is thus the successor of some z and $f(y) \le x < f(y+1)$ (note: in the other case, this holds with $y = z$). Such a y is obviously unique. If $f(z) = a+z$, then y is the difference, if $f(z) = a \cdot z$, then y is the quotient.

```
Lemma normal_ofs_bounded x f: ordinalp x -> normal_ofs f->
  x <o f \0o \/ exists y, [/\ ordinalp y, f y <=o x & x <o f (osucc y)].
```

We define the *next fix-point of f after x*, denoted $\mathrm{N}_f(x)$, as the supremum y of the sequence x_i, defined by $x_0 = x$ and $x_{n+1} = f(x_n)$. Assume first that f is a normal OFS. Obviously, if y is equal to one of the x_i, we have $y = f(y)$. Otherwise y is a limit ordinal and $f(y) = \sup f(x_i) = \sup x_i = y$. It is clear that $x \le y$ and that, if $x \le z$ and $f(z) = z$ then $y \le z$.

If f is defined only for $x \ge u$, there is v such that if $v \le x$ then $x \le f(x)$. If we take $x \ge v$, then the sequence x_i is well-defined; and we get the same result. If f is a function, say of type $\mathrm{E} \to \mathrm{E}$, it may happen that $y = \mathrm{E}$; but otherwise we have $f(y) = y$.

Assume now that E is the cardinal successor of an infinite cardinal C, and E is stable by f; if $x \in \mathrm{E}$, then each x_i is in E, so that $y \in \mathrm{E}$ and $f(y) = y$. This can be restated as: if f is a normal function $\mathrm{E} \to \mathrm{E}$, then the set of fix-points of f is cofinal in E.

```
Definition least_fixedpoint_ge f x y:=
 [/\ x <=o y, f y = y & (forall z, x <=o z -> f z = z -> y <=o z)].
Definition the_least_fixedpoint_ge f x :=
  (\osup (target (induction_defined f x))).
Definition fixpoints f := Zo (source f) (fun z => Vf f z = z).

Lemma normal_ofs_fix1 f u x:
  normal_ofs1 f u -> u <=o x -> x <=o f x ->
  least_fixedpoint_ge f x (the_least_fixedpoint_ge f x).
Lemma normal_ofs_fix x f:
  normal_ofs f -> ordinalp x ->
  least_fixedpoint_ge f x (the_least_fixedpoint_ge f x).
Lemma normal_function_fix f a x
  (y:= the_least_fixedpoint_ge (Vf f) x):
  normal_function f a a ->  x <o a ->
  (y = a \/
  [/\ x <=o y, y <o a, Vf f y = y &
    (forall z, x <=o z -> z <o a -> Vf f z = z -> y <=o z)]).
```

```
Lemma next_fix_point_small f C (E:= cnext C): infinite_c C ->
  normal_ofs f ->
  (forall x, inc x E -> inc (f x) E) ->
  (forall x, inc x E -> inc (the_least_fixedpoint_ge f x) E).
Lemma next_fix_point_small1 f C x (E:= cnext C)
  (y:= the_least_fixedpoint_ge (Vf f) x):
  infinite_c C -> normal_function f E E -> inc x E ->
   [/\ x <=o y, inc y E, Vf f y = y &
    (forall z, x <=o z -> inc z E -> Vf f z = z -> y <=o z)].
Lemma normal_fix_cofinal C (E := cnext C) f:
   infinite_c C -> normal_function f E E ->
   ord_cofinal (fixpoints f) E.
```

Let f be a normal OFS. Assume f maps E to E, so that we can consider f as a function $\mathrm{E} \to \mathrm{E}$. This function is normal, and we can apply the previous theorems.

We construct an example where, whatever C and its cardinal successor E, f does not map E to E. If x is an ordinal, we define x^+ as its ordinal successor, and x_+ as the cardinal successor of the cardinal of x. Let f be the OFS defined by $f(x^+) = f(x)_+$. Let C be an infinite cardinal, E its cardinal successor. Note that C^+ has the same cardinal as C (since C is infinite), so that $\mathrm{C}^+ \in \mathrm{E}$. We have $f(\mathrm{C}^+) = f(\mathrm{C})_+ \geq \mathrm{C}_+ = \mathrm{E}$, so that $f(\mathrm{C}^+) \not\in \mathrm{E}$.

```
Lemma normal_ofs_restriction f C (E := cnext C):
  infinite_c  C-> normal_ofs f ->
  (forall x, inc x E -> inc (f x) E) ->
  normal_function (Lf f E E) E E.
Lemma big_ofs
  (p:= fun z => cnext (cardinal z))
  (osup := fun f => \osup (fun_image (domain f) (fun z => (p (Vg f z)))))
  (osupp:= fun f => Yo (domain f = \0o) \0o (osup f))
  (f:= transdef_ord osupp):
  [/\ normal_ofs f, f \0o = \0o,
   (forall x,  ordinalp x -> f (osucc x) = p (f x))&
   forall C, infinite_c C ->
       exists x, inc x (cnext C) /\ ~ (inc (f x) (cnext C)) ].
```

Let $\bar{x}$ the supremum of x and ω. Then $\mathrm{card}(\bar{x})$ is an infinite cardinal, and $x \in \bar{x}_+$, where x_+ denotes the successor cardinal of the cardinal of x.

```
Lemma omax_p2 y (c:= cardinal (omax y omega0)):
    ordinalp y -> (infinite_c c /\ inc y (cnext c)).
Lemma omax_p3 x y (c:= cardinal (omax (omax x y) omega0)):
   ordinalp x ->  ordinalp y ->
   [/\ infinite_c c, inc x (cnext c) & inc y (cnext c)].
```

Let $f(x)$ be $a+x$, $a\cdot x$ or a^x (the exponential will be defined later on). If we take for C the cardinal of the maximum of a, x and ω, then f maps E to E. Moreover $x \in \mathrm{E}$ (as well as all ordinals less than x). This means that we can define a normal function f_{C}, such that $f_{\mathrm{C}}(x) = f(x)$.

```
Definition card_max x y:= cardinal (omax (omax x y) omega0).

Lemma ofs_add_restr a y (c:= card_max a y) (E := cnext c)
  (f:= Lf (fun z => a +o z) E E) :
```

```
  ordinalp a -> ordinalp y ->
  [/\ (forall x, inc x E -> inc (a +o x) E ),
      (forall x, x <=o y -> inc x E),
      normal_function f E E &
      forall x, inc x E -> Vf f x = a +o x].
Lemma ofs_mul_restr a y (c:= card_max a y) (E := cnext c)
  (f:= Lf (fun z => a *o z) E E) :
  \0o <o a -> ordinalp y ->
  [/\ (forall x, inc x E -> inc (a *o x) E ),
      (forall x, x <=o y -> inc x E),
      normal_function f E E &
      forall x, inc x E -> Vf f x = a *o x].
```

Following [22], we say that a set E of ordinals is *internally closed* if it «includes all its limit values with the possible exception of its least upper bound». We translate this: whenever F is a subset of E, and $x = \sup F$, then $x \in E$ or $x = \sup E$. This condition holds trivially if $x \in F$; otherwise x is a limit ordinal. We say that a property D is an *ordinal property* if $D(x)$ implies that x is an ordinal; we say that it is "internally closed" if moreover whenever F is a non-empty set whose elements satisfy D, then the supremum of F satisfies D. We say that D is closed and proper if is is non-collectivizing (i.e., unbounded).

Examples: an ordinal is a closed set; the collection of limit ordinals is closed and proper; the collection of successors is not closed.

```
Definition ordinal_prop (p:property) := forall x, p x -> ordinalp x.
Definition iclosed_set E :=
   (ordinal_set E) /\
   (forall F, sub F E -> nonempty F -> (\osup F = \osup E \/ inc (\osup F) E)).
Definition iclosed_collection (E:property) :=
   (ordinal_prop E) /\
   (forall F, (forall x, inc x F -> E x) -> nonempty F -> E (\osup F)).
Definition iclosed_proper (E: property) :=
  iclosed_collection  E /\ non_coll E.

Lemma iclosed_ord x: ordinalp x -> iclosed_set x.
Lemma iclosed_lim: iclosed_proper limit_ordinal.
Lemma iclosed_nonlim:
   ~(iclosed_collection (fun z => ordinalp z /\ ~limit_ordinal z)).
```

If f is a normal function, its range is closed as well as the set of its fix-points. If f is a normal OFS, its range and fix-points are closed and proper.

```
Lemma iclosed_normal_fun f x y:
  ordinalp x -> ordinalp y -> normal_function f x y ->
  iclosed_set (Imf f).
Lemma iclosed_fixpoints_fun f a:
  ordinalp a -> normal_function f a a -> iclosed_set (fixpoints f).

Lemma iclosed_normal_ofs1 f u:
  ordinalp u -> normal_ofs1 f u ->
  iclosed_proper (fun z => exists2 x, u <=o x & z = f x).
Lemma iclosed_normal_ofs (f:fterm):  normal_ofs f  ->
  iclosed_proper (fun z => exists2 x, ordinalp x & z = f x).
Lemma iclosed_fixpoints (f:fterm): normal_ofs f ->
    iclosed_proper (fun z => ordinalp z /\ f z = z).
```

We cite theorem 4 of [22]: «there exist solutions of the equation $f(x) = x$ and these solutions $\{\xi\}$ form a closed set similar to $\{x\}$», under the assumption «that $X > \omega$ shall be the first ordinal of a certain number class, and that the values of $f(x)$ as well as x shall be less than X».

Let's write E, instead of $\{x\}$, for the domain of f and Ξ, instead of $\{\xi\}$, for the fix-points of f in E. The assumption on E is that it is an infinite uncountable cardinal X (in fact, a cardinal successor is better suited), and f maps E into E. Veblen notes that E is stable by f if $f(1) < X$ and $f(x+1) - f(x) < X$ whatever $x \in E$; in fact, if $f(x)$ and $f(x+1) - f(x)$ are $< X$ then $f(x+1) < X$, as X is an infinite cardinal. Moreover, if x is limit, $f(y) < X$ for $y < x$, then $\sup f(y) < X$ as the cardinal of the set of ordinals y such that $y < x$ is less than X; by normality of f, this says $f(x) < X$.

What Veblen proves is that Ξ is cofinal in E (if $y = N_f(x)$ is the next fix-point of f after x; then $x \in E$ implies $x \le y$ and $y \in \Xi$). We have seen above that this says that the ordinal of Ξ is E (this is the formal interpretation of $\{\xi\}$ is similar to $\{x\}$), in particular E and Ξ have the same cardinal. Let $A \subset \Xi$ be a nonempty set with supremum x. We have $f(x) = \sup f\langle A\rangle$. As A is invariant by f we get $f(x) = x$. This shows that Ξ is internally closed.

```
Lemma normal_fix_points_similar f C (E:= cnext C)
      (Xi := Zo E (fun z => f z = z)):
  infinite_c C ->
  normal_ofs f ->
  (forall x, inc x E -> inc (f x) E) ->
  (ord_cofinal Xi E) /\ cardinal Xi = E.
```

11.7 Enumerating a collection of ordinals

We consider here a well-order relation R. We shall write $x \le_R y$ instead of $R(x, y)$ and $x <_R y$ instead of "$R(x, y)$ and $x \neq y$". The notation $a \le b$ always means $a \le_{\text{ord}} b$. The domain D of R is defined by $D(x) = R(x, x)$. Reflexivity of R means that $x \le_R y$ implies "$D(x)$ and $D(y)$". We denote by D_R the relation "$D(x)$ and $D(y)$ and $R(x, y)$". If R is a well-order, then D_R is a well-order relation whose domain is the intersection of D and the domain of R. In particular, if $D(x)$ implies that x is an ordinal, then $D_\le$ is a well-order relation with domain D.

To say that R has a graph is the same as D is collectivizing, i.e., there is a set E such that $D(x)$ is equivalent to $x \in E$. In this case, E is well-ordered by R, and isomorphic to a unique ordinal. In the other case, D is a proper class. We say that R is a well-ordered proper class. Example: if D is the collection of non-successor ordinals, then $D_\le$ is a well-ordered proper class. Other example: D'_R where $D'(x)$ is "$x = \Omega$ or x is an ordinal" (where Ω is a non-ordinal), and $R(x, y)$ is "$x \le y$ or $y = \Omega$". This is a well-ordering, obtained by adjoining Ω as greatest element to the collection of all ordinals.

Let $S_R(x)$ be the collection of all y such that $y <_R x$. It is called the *initial segment with endpoint* x. We shall assume that this is a set for any x in the domain of R (this holds whenever R has the form $D_\le$, it fails in the example of D'_R above with $x = \Omega$). [Note: if x is not in the domain, then $S_R(x)$ is empty, this is obviously a set]. As R is a well-order relation, the set $S_R(x)$ is well-ordered by R, and we may consider its ordinal $\alpha = J_R(x)$. We are interested in the inverse function $x = K_R(\alpha)$ that maps some ordinal to an element of D. We shall call it an *enumeration* of R (and, by abuse of language, an enumeration of D, when $R = D_\le$).

The main result of this section is the following. Assume that D is a closed proper subclass of the ordinals. Then the enumeration of D is normal (this is the converse of: the range of a normal OFS is closed and proper).

Note. The axiom of choice will be use twice. We could avoid the first use, but not the second one. The assumption is that for any x, there is E such that $y \leq_R x$ is equivalent to $y \in E$. In this case E is unique and the axiom of choice provides the function $x \mapsto E(x)$. Then $S_R(x) = \{y \in E(x), y <_R x\}$. Assume $y <_R x$ implies $y \in F(x)$, so that $S_R(x) = \{y \in F(x), y <_R x\}$. Note that F is not unique. We can avoid the axiom of choice by passing F as argument. In the case where R has the form $D_{\leq}$, then $F(x) = x$ is a possibility.

```
Definition collectivising_srel (r: relation) :=
   forall x, r x x -> exists E, (forall y, inc y E <-> r y x).
Definition worder_rc r := worder_r r /\ collectivising_srel r.
Definition segmentr (r: relation) x :=
   choose (fun E => (forall y, inc y E <-> (r y x /\ y <> x))).
Definition ordinalr r x := ordinal (graph_on r (segmentr r x)).
```

We shall denote by $J_R(x)$ the ordinal of $S_R(x)$. If $x <_R y$ then $J_R(x) < J_R(y)$. Moreover, if $\alpha \leq J_R(y)$, then α is of the form $J_R(x)$.

```
Lemma collectivising_srel_alt (r: relation): reflexive_rr r ->
  (collectivising_srel r <-> forall x,exists E, (forall y, inc y E <-> r y x)).
Lemma segmentrP r x: collectivising_srel r -> r x x ->
   (forall y, (inc y (segmentr r x) <-> r y x /\ y <> x)).
Lemma worder_rc_seg (r: relation) x:
  worder_rc r -> r x x ->
  worder_on (graph_on r (segmentr r x)) (segmentr r x).
Lemma OS_ordinalr r x:
  worder_rc r -> r x x -> ordinalp (ordinalr r x).
Lemma ordinalr_Mle r x y: worder_rc r -> r x y ->
   (ordinalr r x) <=o (ordinalr r y).
Lemma ordinalr_Mlt r x y: worder_rc r -> r x y -> x <> y ->
   (ordinalr r x) <o (ordinalr r y).
Lemma ordinalr_segment r a x (b:=ordinalr r x): (* 58 *)
   worder_rc r -> r x x -> a <=o b ->
   (exists2 y, r y y & a = ordinalr r y).
```

We define the inverse $x = K_R(a)$ of $J_R(x) = a$ via the axiom of choice. Since J is strictly increasing, it is injective, so that we get: $J_R(K_R(a)) = a$ (if a is in the range of J_R) and $K_R(J_R(x)) = x$ (if x is in D).

Assume that there is some ordinal not of the form $J_R(x)$ and take the least, say a. Then, for any x we have $J_R(x) < a$. Thus, every x has the form $K_R(b)$ for some $b < a$. This means that there is a set containing all x. On the other hand, if D is a proper class, then $K_R(a)$ is defined for every ordinal a. This functional is strictly increasing.

```
Definition ordinals r a := choose (fun x => r x x /\ a = ordinalr r x).

Lemma ordinalsP r a:  worder_rc r ->
  (exists2 x, r x x & a = ordinalr r x) ->
  (r (ordinals r a) (ordinals r a) /\ ordinalr r (ordinals r a) = a).
Lemma ordinalsrP r x:  worder_rc r -> r x x ->
  ordinals r (ordinalr r x) = x.
Lemma ordinals_non_coll1 r:  worder_rc r ->
   (non_coll (fun x => r x x)) ->
   (forall a, ordinalp a -> exists2 x, r x x & a = ordinalr r x).
```

```
Lemma ordinalS_Mle r a b:  worder_rc r ->
   (non_coll (fun x => r x x)) ->
   a <=o b -> r (ordinals r a) (ordinals r b).
Lemma ordinals_Mlt r a b:  worder_rc r ->
   (non_coll (fun x => r x x)) ->
   a <o b ->
  (r (ordinals r a) (ordinals r b) /\  (ordinals r a) <>  (ordinals r b)).
```

We assume now that $x \leq_R y$ implies $x \leq y$. In this case x and y are ordinal numbers, $S_R(x)$ is the set of ordinals less than x such that $y <_R x$. If R is total, then it is a well-ordering (in fact, if D is the domain of R, we have $R = D_{\leq}$)

```
Lemma collectivising_srel_ord (r:relation) :
  (forall x y, r x y -> x <=o y) -> collectivising_srel r.
Lemma collectivising_srel_ord_seg (r:relation) :
  (forall x y, r x y -> x <=o y) ->
  forall x, r x x -> segmentr r x = Zo x (fun z => r z x).
Lemma worder_rc_ord (r:relation) :
  (forall x y, r x y -> x <=o y) ->
  (order_r r) -> (forall x y, r x x -> r y y -> (r x y \/ r y x)) ->
  worder_rc r.
```

Assume that E is a set of ordinal numbers. The relation "$x \in E$ and $y \in E$ and $x \leq y$", denoted by R makes E a well-ordered set; let A be its ordinal. Then K_R is an order isomorphism $A \to E$ (the non-trivial point is that every element of A is of the form $J_R(x)$). If E is a closed set, then K_R is a normal function.

```
Lemma worder_rc_op (p:property) :
  worder_rc (fun x y => [/\ p x, p y & x <=o y]).
Lemma ordinalrsP (p: property) (r := fun x y => [/\ p x, p y & x <=o y])
   x (y := ordinalr r x):
   ordinal_prop p -> p x -> (ordinalp y) /\ ordinals r y = x.
Lemma ordinals_set_iso E (p := inc ^~E)     (* 85 *)
      (r:= fun x y => [/\ p x, p y & x <=o y])
      (A:= ordinal (olen_on E)):
  (ordinal_set E) ->
  (lf_axiom (ordinals r) A E) /\
  order_isomorphism (Lf (ordinals r) A E) (ordinal_o A) (ole_on E).
Lemma ordinals_set_normal E (p := inc ^~E)
      (r:= fun x y => [/\ p x, p y & x <=o y])
      (A:= ordinal (ole_on E))
      (f:= Lf (ordinals r) A E):
  (iclosed_set E) ->
  normal_function f A E.
```

Let E be the cardinal successor of some infinite cardinal, and B a closed cofinal subset of E. The function K_R defined above is an isomorphism $A \to B$, where $A = E$. Since $B \subset E$, we may consider it a function $E \to E$, with range B. We shall write it K_B and call it the *enumeration* of B. This is a normal function $E \to E$.

Assume now $B_1 \subset B_2$. We assume B_1 cofinal in E_1, the cardinal successor of C_1, and B_2 cofinal in E_2, the cardinal successor of C_2 (note that if C_1 exists, it is uniquely defined by B_1). The assumption $B_1 \subset B_2$ says $E_1 \subset E_2$, this is equivalent to $C_1 \leq C_2$. We pretend that the enumeration function of B_2 coincides in E_1 with the enumeration function of B_1. This works only if every element of B_2 that is in E_1 is also in B_1. This can be restated as $B_1 = B_2 \cap E_1$ or as: B_1 is an initial segment of B_2 (for the induced ordering). Consider the following: we take an element x of E_1, use K_{B_1} to get an element of B_1, this is in B_2, thus is the image of K_{B_2} of some x' in E_2. We pretend that $x = x'$. In fact, the mapping $x \mapsto x'$ is strictly increasing; our assumption says that its range is an initial segment of E_2; by uniqueness of morphisms whose range are initial segments, this is the identity function.

```
Definition ordinalsE E B :=
  Lf (ordinals (fun x y => [/\ inc x B, inc y B & x <=o y])) E E.

Lemma ordinals_set_normal1 C (E:= cnext C) B (f:= ordinalsE E B):
  infinite_c C -> iclosed_set B -> ord_cofinal B E ->
  [/\ lf_axiom (ordinals (fun x y=> [/\ inc x B, inc y B & x <=o y])) E E ,
   normal_function f E E & Imf f = B].
Lemma ordinals2_extc C1 C2 B1 B2
    (E1 :=cnext C1)(E2 :=cnext C2) :
  infinite_c C1 -> infinite_c C2 ->
  iclosed_set B1 -> iclosed_set B2 ->
  ord_cofinal B1 E1 -> ord_cofinal B2 E2 ->
  C1 <=c C2 -> B1 = B2 \cap E1 ->
  agrees_on E1 (ordinalsE E1 B1) (ordinalsE E2 B2).  (* 100 *)
```

Consider now a proper class D, and the relation $R = D_{\le}$. We assume that $D(x)$ implies that x is an ordinal. We write J_D and K_D instead of J_R and K_R. Then K_D is a strictly increasing OFS with range D. If the collection D is closed then K_D is normal. This is Theorem 1 of [22].

```
Definition ordinalsf (p: property) :=
   ordinals (fun x y  => [/\ p x,  p y  & x <=o y]).
Lemma ordinals_col_p1 (p: property) (f := ordinalsf p):
   (forall x, p x -> ordinalp x) ->  (non_coll p) ->
 [/\
    forall a, ordinalp a -> p (f a),
    forall x, p x -> exists2 a, ordinalp a & x = f a,
    forall a b, a<=o b -> (f a) <=o (f b),
    forall a b, a<o b -> (f a) <o (f b) &
    (forall a b, ordinalp a -> ordinalp b -> (a<=o b <-> (f a) <=o (f b))) /\
    forall a b, ordinalp a -> ordinalp b -> (a<o b <-> (f a) <o (f b))].
Lemma ordinals_col_p2 (p: property) (f := ordinalsf p):
   (iclosed_proper p) ->
   normal_ofs f.
```

We have: $K_D(0)$ is the least ordinal satisfying D, and $K_D(x+1)$ is the least ordinal $> K_D(x)$ satisfying D. Since K_D is normal, these two properties uniquely characterize K_D.

```
Lemma iclosed_col_f0  (p: property) (f := ordinalsf p) (x:= f \0o):
  (iclosed_propze p) ->
  (p x /\ (forall z, p z -> x <=o z))
Lemma iclosed_col_fs  (p: property) (f := ordinalsf p) a
   (x:= f a) (y := f (osucc a)) :
  (iclosed_proper p) ->  ordinalp a ->
  [/\ x <o y, p x, p y &  (forall z, p z -> x <o z -> y <=o z)].
```

For any normal f, the collection D of its fix-points is closed, thus has an enumeration K_D. This is called the *first derivation*[2] of f. Denote it by f'. The range of f' is the collection

[2]veblen says "first derived function", but this is obviously not a function.

of fix-points of f. We have $f'(0) = \mathrm{N}_f(0)$ and $f'(x+1) = \mathrm{N}_f(x+1)$, where $\mathrm{N}_f(x)$ is the next fix-point of f after x. If $f(0) \neq 0$ we have also $f'(0) = \mathrm{N}_f(1)$.

```
Definition first_derivation (f: fterm) :=
  (ordinalsf (fun z => ordinalp z /\ f z = z)).

Lemma first_derivation_p f (fp := first_derivation f): normal_ofs f ->
  ( (forall x, ordinalp x -> f (fp x) = fp x)  /\
    (forall y, ordinalp y  -> f y = y -> exists2 x, ordinalp x & y = fp x)).
Lemma first_derivation_p0 f: normal_ofs f ->
  normal_ofs (first_derivation f).
Lemma first_derivation_p1 f: normal_ofs f ->
  (first_derivation f \0o) = the_least_fixedpoint_ge f \0o.
Lemma first_derivation_p2 f: normal_ofs f -> f \0o <> \0o ->
  (first_derivation f \0o) = the_least_fixedpoint_ge f \1o.
Lemma first_derivation_p3 f x: normal_ofs f ->  ordinalp x ->
  (first_derivation f (osucc x)) =
    the_least_fixedpoint_ge f (osucc (first_derivation f x)).
Lemma normal_ofs_from_exten f g : f =1o g -> normal_ofs f ->  normal_ofs g.
Lemma first_derivation_exten f g : f =1o g -> normal_ofs f ->
  first_derivation f =1o  first_derivation g.
```

Let E be the cardinal successor of an infinite cardinal C. Asume that f is a normal OFS that maps E into E; let F be the set of fixpoints points of f in E. We have shown above that E and F are order isomorphic. In fact, the isomorphism is the first derivation of f.

```
Lemma first_derivation_p4 f C   (* 57 *)
   (E:= cnext C) (f' := first_derivation f)
   (F := Zo E (fun z => f z = z)):
   infinite_c C ->
   normal_ofs f ->
   (forall x, inc x E -> inc (f x) E) ->
   order_isomorphism (Lf f' E F) (ordinal_o E) (ole_on F).
```

Every non-zero ordinal x is $1+(x-1)$, so that $x \mapsto 1+x$ is the enumeration of non-zero ordinals.

```
Lemma ord_rev_pred x (y:= x -o \1o) : \0o <o x  ->
   (ordinalp y /\ x = \1o +o y).
Lemma rev_pred_prop (f := osum2 \1o):
  normal_ofs f /\ (forall y, \0o <o y -> exists2 x, ordinalp x & y = f x).
Lemma non_zero_ord_enum:
  (osum2 \1o) =1o ordinalsf (fun x => \0o <o x).
```

We show here Corollary 2 of Theorem 4 of [22]. If a is an ordinal, then the first derivation of $x \mapsto a+x$ is $x \mapsto a\cdot\omega+x$. Proof. Let $p(x)$ be $a+x=x$. If $x \geq u$ then $x = u+(u-x)$, so that $p(u)$ implies $p(x)$. The first derivation is hence $x \mapsto b+x$ where b is the least ordinal satisfying p. Now $b = \sup b_i$, where $b_0 = 0$ and $b_{i+1} = a+b_i$, hence $b_i = a\cdot i$, and $b = a\cdot\omega$. Note: if $a = 0$, we get: the first derivation of the identity is the identity. [Note: Veblen says $f'(x) = a\cdot\omega+(x-1)$, because zero is not always considered as an ordinal; he also mentions in a footnote that $x-1 = x$ when x is infinite].

```
Lemma add_fix_enumeration a: ordinalp a ->
```

```
  first_derivation (osum2 a) =1o (osum2 (a *o omega0)).
Lemma add1_fix_enumeration:
  first_derivation (osum2 \1o) =1o (osum2 omega0).
```

An ordinal x is limit if and only if it is of the form $y\cdot\omega$, with non-zero y (write $x=\omega\cdot q+r$ by division; we know that if $r=0$, then x is not a successor; otherwise r isa natural number, hence a successor).

It follows that $x\mapsto\omega\cdot x$ is the enumeration of non-successor ordinals. We leave it as an exercise to the reader to show that the enumeration of the limit ordinals is $x\mapsto\omega\cdot(1+x)$ (Hint: $x\mapsto 1+x$ enumerates all non-zero ordinals).

```
Lemma omega_div x: ordinalp x ->
  exists a b, [/\ ordinalp a, b<o omega0, x = omega0 *o a +o b &
     (osuccp x <-> b <> \0o)].
Lemma limit_ordinal_P4 x: ordinalp x ->
  (limit_ordinal x <-> exists2 y, \0o <o y & x = omega0 *o y).
Lemma non_succ_ord_enum:
  (oprod2 omega0) =1o ordinalsf (fun x => ordinalp x /\ ~ (osuccp x)).
```

11.8 Indecomposable ordinals

An ordinal c is called *indecomposable* if it is non-zero and never the sum of two ordinals a and b such that $a<c$ and $b<c$. We shall see later on that indecomposable ordinals are the powers of ω.

Assume $c=a+b$. Then $a\le c$ and $b\le c$. If c is indecomposable, then either $c=a$ or $c=b$. If c is a successor, then c is indecomposable if and only if $c=1$. The ordinal ω is indecomposable since $x<\omega$ implies x finite, and ω is not the sum of two finite ordinals. Any indecomposable ordinal is thus 1, or at least ω. Note that an infinite cardinal is indecomposable: $\mathrm{card}(a+b)=\mathrm{card}(a)+_{\mathrm{card}}\mathrm{card}(b)$ says that if $a+b$ is an infinite cardinal c, then c is the greatest of $\mathrm{card}(a)$ and $\mathrm{card}(b)$; now $a<c$ implies $\mathrm{card}(x)<_{\mathrm{card}}\mathrm{card}(c)$.

```
Definition indecomposable c :=
  \0o <o c /\ (forall a b, a <o c -> b <o c -> a +o b <> c).
Lemma OS_indecomposable xa: indecomposable a -> ordinalp a.
Lemma indecomp_pr c a b:
  ordinalp a -> ordinalp b ->
  indecomposable c -> a +o b = c -> (a = c \/ b = c).
Lemma indecomp_one: indecomposable \1o.
Lemma indecomp_omega: indecomposable omega0.
Lemma indecomp_example x: \0o <o x ->
  ~ (indecomposable (osucc x)).
Lemma indecomp_limit a: indecomposable a ->
  a = \1o \/ limit_ordinal a.
Lemma indecomp_omega1 a: indecomposable a ->
  a = \1o \/ omega0 <=o a.
Lemma cardinal_indecomposable x: infinite_c x ->
   indecomposable x.
```

We prove here the following claims:

(a) A non-zero ordinal c is indecomposable if and only if $a<c$ implies $a+c=c$;

(b) If c is indecomposable, $a < c$ and $b < c$, then $a + b < c$;

(c) If $y > 0$, and $x \neq 1$, then x indecomposable if and only if $y \cdot x$ is indecomposable;

(d) If x is indecomposable and $y < x$ then y divides x and the quotient is indecomposable.

Proof. Assume c indecomposable, and $a < c$. If $b = c - a$ we have $b \leq c$ and $a + b = c$. Now $b < c$ is impossible, so that $b = c$, thus $a + c = c$. Assume moreover $b < c$. By associativity $a + b + c = c$. This implies $a + b \leq c$, and since equality is forbidden $a + b < c$. Assume now that $a < c$ implies $a + c = c$, whatever a. If $a < c$ and $a + b = c$, we deduce $a + b = a + c$, and after simplification $b = c$.

Assume now $z < y \cdot x$, x indecomposable, $x > 1$. By division we have $z = y \cdot q + r$, with $r < y$ and $q < x$. Thus $z + y \cdot x \leq y \cdot q + r + y \cdot x \leq y \cdot q + y + y \cdot x = y \cdot (q + 1 + x)$. If $q + 1 < x$, this is $y \cdot x$. We deduce $z + y \cdot x = y \cdot x$. Otherwise, we have $q + 1 = x$; this is absurd since $1 < x$. Finally, assume $y < x$ and x is indecomposable, write $x = y \cdot q + r$ with $r < y$. From $r < x$ it follows $x = y \cdot q$. From (c) it follows that q is indecomposable.

Note: let c be an infinite cardinal and $a < c$. The complement B of a in c is well-ordered and its ordinal b satisfies $a + b = c$ (This is one way to define the difference $b = c - a$, see page 514). Since c is indecomposable, it follows $c - a = c$; moreover B and c have same cardinal.

```
Lemma indecomp_prop1 c a: indecomposable c ->
  a <o c -> a +o c = c.
Lemma indecomp_prop2 a b c: a <o c -> b <o c -> indecomposable c ->
    a +o b <o c.
Lemma indecompP c: \0o <o c  ->
  (indecomposable c <-> (forall a, a <o c -> a +o c = c)).
Lemma cardinal_indecomposable1 c a : infinite_c c -> a <o c ->
  ((c -o a) = c /\ cardinal (c -s a) = c).
Lemma indecomp_prodP x y: \1o <o x -> \0o <o y ->
  (indecomposable x <-> indecomposable (y *o x)).
Lemma indecomp_div x y: indecomposable x ->
  y <> \0o -> y <o x ->
  exists z, [/\ indecomposable z, ordinalp z & x = y *o z].
```

Veblen [22, Example 4] says « If x_α is any element of a well-ordered set $\{x\}$, then the set of all elements preceding x_α is called a *section* of $\{x\}$ and the set consisting of x_α and all following elements is called a *residue* of $\{x\}$.» So a section of a well-ordered set X is a segment; and a residual is its complement; these are two well-ordered sets. « A well-ordered set is never similar to a section of itself, but some well-ordered set are similar to all of their residues.» Let β be the ordinal of X, $x \in$ X, and α the ordinal of the section; then $\beta - \alpha$ is the ordinal of the residue. Veblen says: $\alpha \neq \beta$, but for some X, one can have $\beta - \alpha = \beta$, whatever x. « Such a set is called *self-residual* and its type or ordinal number is also called self-residual. An equivalent definition is that a self-residual number β satisfies the equation $\alpha + \beta = \beta$ for every α less than β.»

```
Lemma indecomp_diff1 c a: indecomposable c -> a <o c -> c -o a = c.
Lemma indecomp_diff2 c: \0o <o c  ->
  (forall a,  a <o c -> c -o a = c) ->indecomposable c.
```

Let a be a non-zero ordinal. Then $a \cdot \omega$ is an indecomposable ordinal $> a$; if b is indecomposable and $a < b$, then $b = a \cdot c$ for some indecomposable ordinal c, that cannot be 1, hence must be $\geq \omega$. Thus $a \cdot \omega$ is the least indecomposable ordinal $> a$. We deduce $(a + 1) \cdot \omega = a \cdot \omega$

(these quantities are the least indecomposable $> a$ or $> a+1$ (note that $a\cdot\omega$ cannot be equal to $a+1$).

```
Lemma indecomp_prod2 a (b:= a *o omega0): \0o <o a ->
  [/\ indecomposable b, a <o b &
  forall c, indecomposable c -> a <o c -> b <=o c].
Lemma indecomp_prod3 a: \0o <o a ->
  (osucc a) *o omega0 = a *o omega0.
```

If E is a non-empty set of indecomposable ordinals, then $c = \sup \mathrm{E}$ is indecomposable (if $a < c$ and $b < c$, there is $d \in \mathrm{E}$ such that $a < d$ and $b < d$, thus $a + b < d \le c$). If x is a non-zero ordinal, there is a greatest indecomposable $\le x$ (the supremum of the set of indecomposable ordinals $\le x$). It follows that the collection of indecomposable ordinals is a closed proper class. The enumeration function will be given below.

```
Lemma indecomp_sup E:
  (forall x, inc x E -> indecomposable x) ->
  (nonempty E) ->
  indecomposable (\osup E).
Lemma indecomp_sup1 x: \0o <o x ->
  exists y, [/\ indecomposable y, y <=o x &
    forall z, indecomposable z -> z <=o x -> z <=o y].
Lemma indecomp_closed_noncoll: iclosed_proper indecomposable.
```

11.9 Definition by transfinite induction

Bourbaki defines ordinal exponentiation x^y by transfinite induction for $x \ge 2$ and $y > 0$. It is a bit easier to define it for every y. The code that follows corresponds to exercises 2.17 and 2.18 (page 585), with the following modifications: instead of w we use w_1, this is the value at $y = 1$; we denote by w_0 the value at $y = 0$. We shall denote by u the minimal value of x.

Let $f(x, y)$ be the term to be defined, and $f(x, y+1) = g(f(x, y), x)$ the recurrence relation. We have already seen that there is a unique normal OFS f satisfying this relation and $f(x, 0) = w_0(x)$. There are some necessary conditions; we shall chose (11.18), where $w_1(x) = g(w_0(x), x)$:

$$x \le w_1(x), \qquad w_0(x) < w_1(x) \qquad t < g(t, x) \qquad (\text{if } x, t \ge u) \tag{11.18}$$

and define f for $x \ge u$ and all $y > 0$ by

$$f(x, 0) = w_0(x), \quad f(x, y) = \sup_{z<y} g(f(x, z), x). \tag{11.19}$$

Uniqueness is obvious.

```
Definition ord_induction_sup (g: fterm2) x y (fx: fterm) :=
  \osup (fun_image y (fun z => g (fx z) x)).

Lemma ord_induction_unique  (w0: fterm)  (g:fterm2) u (f f':fterm2)
  (P := fun f => forall x, u <=o x ->
   ( f x \0o = w0 x /\
    (forall y, \0o <o y -> f x y = ord_induction_sup g x y (f x)))):
  P f -> P f' ->  forall x, u <=o x -> f x =1o f' x.
```

Our principle of transfinite induction says that there is a unique F such that $F(y) = p(F_{(y)})$; where $F_{(y)}$ is the restriction of F (as a functional graph) to ordinals less than y. Here $F(y) = f(x, y)$. We consider $p(G)$ such that, if the domain of G is empty, then the value is $w_0(x)$, otherwise we take the supremum over the domain of G of all $g(G(z), x)$.

```
Definition ord_induction_p  (w0:fterm) (g:fterm2) x F :=
  (Yo (domain F = \0o) (w0 x) (ord_induction_sup g x (domain F) (Vg F))).

Definition ord_induction_defined (w0:fterm) (g:fterm2) :=
   transdef_ord (ord_induction_p w0 g x).

Lemma ord_induction_y0 w0 g x:
  ord_induction_defined w0 g x \0o = w0 x.
Lemma ord_induction_yp w0 g x (f:= (ord_induction_defined w0 g)):
  (forall y, \0o <o y -> f x y = ord_induction_sup g x y (f x)).
Lemma ord_induction_y1 w0 g x:
  ord_induction_defined w0 g x \1o = g (w0 x) x.
```

We shall consider quite a lot of assumptions. For simplicity, we shall put them is a section, and consider three groups. The first group is (11.18). If we assume g increasing (in its second argument), then f will satisfy a nice induction principle. The second group will ensure that f is increasing in both its arguments. The last group will be used in showing relation (11.23).

```
Section OrdinalInduction.
Variables (u: Set) (w0: fterm) (f g : fterm2).
Hypothesis fv: f = ord_induction_defined w0 g.

Let w1 := fun x => (g (w0 x) x).
Definition OIax_w0 := forall x, u <=o x -> w0 x <o w1 x.
Definition OIax_w1 := forall x, u <=o x -> x <=o w1 x.
Definition OIax_g1 := forall x y, u <=o x -> u <=o y -> x <o (g x y).
Definition OIax_g2:= forall a b a' b',
    u <=o a -> u <=o b ->  a <=o a' -> b <=o b' ->
   (g a b) <=o (g a' b').
Definition OIax_w2w:= forall a a', u <=o a -> a <=o a' -> (w0 a) <=o (w0 a').
Definition OIax_w2:= forall a a', u <=o a -> a <o a' -> w1 a <o w1 a'.
Definition OIax_w3:= forall a, u <=o a -> w1 a = a.
Definition OIax_g3:= forall a, u <=o a -> normal_ofs1 (fun b => g a b) u.
Definition OIax_g4:= forall a b c, u <=o a -> u <=o b -> u <=o c ->
     g (g a b) c = g a (g b c).

Definition OIax1 := [/\ OIax_w0, OIax_w1 & OIax_g1].
Definition OIax1b := OIax1 /\ OIax_g2.
Definition OIax2 := [/\ OIax1,  OIax_g2, OIax_w2w & OIax_w2].
Definition OIax3 := [/\ OIax2, OIax_w3, OIax_g3 & OIax_g4].
```

Assume $u \leq x$. We first show $x \leq f(x, y)$ for non-zero y. Assume this fails for some z, take the least one; since $x \leq f(x, 1)$ we get $1 < z$. Let T be the set of all $g(f(x, t), x)$ for $t < z$. Elements of T are ordinals (this is obvious for $t = 0$, follows from the definition of z otherwise). From $1 < z$, we get $f(x, 2) \in T$, thus $f(x, 2) \leq z$. We conclude with $f(x, 1) < f(x, 2)$.

It follows that $f(x, y)$ is an ordinal, other various relations, and $y \mapsto f(x, y)$ is a normal OFS.

```
Lemma ord_induction_p01 x: OIax1 u <=o x -> f x \0o <o f x \1o.
```

```
Lemma ord_induction_p4 x y:  OIax1 ->
  u <=o x -> \0o <o y -> x <=o (f x y).
Lemma ord_induction_p41 x y:  OIax1 ->
  u <=o x -> \0o <o y -> u <=o (f x y).
Lemma ord_induction_p5 x y:  OIax1 ->
  u <=o x -> ordinalp y -> ordinalp (f x y).
Lemma ord_induction_p6 x y:  OIax1 ->
  u <=o x -> ordinalp y -> ordinalp (g (f x y) x).
Lemma ord_induction_p7 x y y':  OIax1 ->
  u <=o x -> y <o y' -> g (f x y) x <=o (f x y').
Lemma ord_induction_p8 x y y':  OIax1 ->
  u <=o x ->  y <o y' ->  (f x y) <o (f x y').
Lemma ord_induction_p9 x y:  OIax1 ->
  u <=o x -> ordinalp y -> y <=o (f x y).
Lemma ord_induction_p10 x:  OIax1 -> u <=o x -> normal_ofs (f x).
```

Assume now that a and b are ordinals and $f(a,0) \le b$. There exists a unique ordinal y such that

$$f(a,y) \le b < f(a,y+1). \tag{11.20}$$

This holds by normality of f. Note that $y \le b$.

```
Lemma ord_induction_p11 x b y y': OIax1 ->
  u <=o x -> ordinalp y -> ordinalp y' ->
  (f x y) <=o b -> b <o (f x (osucc y))  ->
  (f x y') <=o b -> b <o (f x (osucc y'))  ->
  y = y'.
Lemma ord_induction_p12 x b: OIax1 ->
  u <=o x -> (w0 x) <=o b ->
  exists y,  [/\  y <=o b, (f x y) <=o b & b <o (f x (osucc y))].
```

If we assume $g(x,y) \le g(x,y')$ for $y \le y'$, it follows

$$f(x,y+1) = g(f(x,y),x). \tag{11.21}$$

```
Lemma ord_induction_p13 x y: OIax1b ->
   u <=o x -> ordinalp y -> f x (osucc y) = g (f x y) x.
```

For simplicity, we shall prove relations (11.23) below only in the case where y and z are non-zero, $u \le x$. We shall consider here the special cases. Let $a = f(x,y)$. We have to show that $g(a,w_0(x)) = a$, $g(w_0(x),a) = a$, $w_0(a) = w_0(x)$ and $f(w_0(x),z) = w_0(x)$. For these relations to hold, it is necessary that $w_0(x)$ be independent of x, let's call it c. A necessary condition is then $g(a,c) = g(c,a) = c$. Since g is normal in its second argument for $u \le y$, it follows $c < u$. Thus, we also assume $w_0(c) = c$ and $g(c,c) = c$. [Note: we could as well assume that $g(a,c) = g(c,a) = w_0(a) = c$ hold, whatever a]. These assumptions are sufficient (the relation $f(c,z) = c$ follows by induction from (11.19)).

```
Definition ord_induction_g_unit c :=
  [/\ ordinalp c, g c c = c, c = w0 c &
    forall x, u <=o x -> [/\ g x c = x, g c x = x & w0 x = c]].

Lemma ord_induction_zv c: ord_induction_g_unit c ->  OIax1 ->
```

```
  [/\  (forall a, u <=o a -> (g (w0 a) a) = a),
      (forall a b, u <=o a -> ordinalp  b ->
       f a (b +o \0o) = g (f a b) (f a \0o)),
   (forall a b, u <=o a -> ordinalp b ->
      f a (\0o +o b) = g (f a \0o) (f a b)),
   forall a b, u <=o a -> ordinalp b ->  f a (b *o \0o) = f (f a b) \0o &
   forall a b, u <=o a -> ordinalp b ->  f a (\0o *o b) = f (f a \0o) b].

End OrdinalInduction.
```

There is a unique normal OFS satisfying (11.21) and $f(x,0) = w_0(x)$. Take for instance $g(x,y) = x^+$ and $w_0(x) = x$. Then we get addition. We have $f(f(a,b),c) = f(a,f(b,c))$, since these are two normal OFS (for fixed a and b) satisfying the same induction principle and initial value. Thus addition is associative. If we take $w_0(x) = 0$ and $g(x,y) = x+y$ we get multiplication (we assume $u = 1$, and define $x\cdot y$ to be zero when x is zero). Existence of subtraction and division follows from (11.20).

```
Lemma ord_induction_p14
  (f:= ord_induction_defined id (fun u v:Set => osucc u)):
  (forall a b, ordinalp a -> ordinalp b -> f a b  = a +o b)
  /\ (forall a b c, ordinalp a -> ordinalp b -> ordinalp c ->
     f (f a b) c = f a (f b c)).
Lemma ord_induction_p15 a b:
  \0o <o a -> ordinalp b ->
  ord_induction_defined (fun z:Set=> \0o) osum2 a b = a *o b.
```

Consider now

(11.22) $\quad x < x' \implies w_1(x) < w_1(x'), \qquad x \le x', y \le y' \implies w_0(x) \le w_0(x'),\ g(x,y) \le g(x',y').$

where all variables are $\ge u$. Then f is increasing. Asume that $g(x,y) < g(x,y')$ for $u \le y < y'$; then $f(x,y+1) < f(x',y+1)$. The examples of addition and multiplication show that f is in general not strictly increasing in its first argument.

```
Section OrdinalInduction2.
Variables (u: Set) (w0: fterm) (f g : fterm2).
Hypothesis fv: f = ord_induction_defined w0 g.

Lemma ord_induction_p16 x y x' y': OIax2 u w0 g ->
  u <=o x -> x <=o x' -> y <=o y' -> f x y <=o f x' y'.
Lemma ord_induction_p17 x x' y: OIax2 u w0 g ->
   (forall a b b', u <=o a -> u <=o b -> b <o b' -> g a b <o g a b') ->
   u <=o x -> x <o x' -> ordinalp y ->
   f x (osucc y) <o f x' (osucc y).
```

Let's show that

(11.23) $\qquad g(f(x,y), f(x,z)) = f(x, y+z), \qquad f(f(x,y),z) = f(x, y\cdot z).$

As mentioned above, these relations are in general not valid for $y = 0$ or $z = 0$, and we assume $u \le x$. The expressions to be compared are normal OFS, take the same value at one and satisfy the same recurrence principle, thus must be equal. The second follows from the first. The relation holds if $w_1(x) = x$ and if g is associative, i.e., $g(a, g(b,x)) = g(g(a,b),x)$.

Here $a+b$ could be addition defined by induction. Let g be addition so that f is multiplication. Then $(a \cdot b)+(a \cdot c) = a \cdot (b+c)$. In the second relation, we can use as multiplication the one defined by induction. It follows that multiplication is associative (note that if one of a, b or x are zero, then $f(a, f(b,x)) = f(f(a,b),x)$ holds, as it is zero).

```
Lemma ord_induction_p18 a b c: OIax3 u w0 g ->
  u <=o a -> \0o <o b -> \0o <o c ->
    f a (b +o c) = g (f a b) (f a c).
Lemma ord_induction_p19 a b c: OIax3 u w0 g ->
  u <=o a -> \0o <o b -> \0o <o c -> f a (b *o c) = f (f a b) c.
```

We have $f(x,1)+y \le f(x,y+1)$ (for finite y, by induction). Bourbaki hints in Exercise 6.13(e) that this is true for every y. We shall show a weaker statement; first notice that $x+y \le f(x,y+1)$ for finite y, so that $f(x,\omega) \ge x+\omega$ by taking limits and $x+y \le f(x,y)$ for infinite y.

```
Lemma ord_induction_p21a x y: OIax1b u w0 g ->
    u <=o x -> y <o omega0 ->
   (w1 x) +o y  <=o (f x (osucc y)).
Lemma ord_induction_p21b x: (OIax2 u w0 g) ->
   u <=o x -> x +o omega0  <=o (f x omega0).
Lemma ord_induction_p21c x y: OIax1b u w0 g ->
   u <=o x  -> omega0 <=o y ->
   x +o y  <=o (f x y).
Lemma ord_induction_p21d x y: OIax1b u w0 g ->
   u <=o x -> \2o <=o  y ->  x <o (f x y).
```

We say that y is *critical* if $f(x,y) = y$ for all x such that $x < y$. For instance, if f is addition, then y is critical if, and only if, it is indecomposable. In order to avoid trivial cases, we assume $u < y$. Assume $y = z+1$. Take $x = z$. We get $z+1 = g(f(z,z),z)$. This implies $f(z,z) = z$, which is false for $z \ge 2$, so that y has to be one or two. We shall exclude these cases by assuming y infinite. It follows: all critical points are limit ordinals.

From now on, we shall assume f increasing. If y is infinite, then $y \le f(x,y)$ for all x. Thus, if $x \le x'$ and $f(x',y) = y$ it follows $f(x,y) = y$.

Assume $f(x,y) = y$ whenever $x \in \mathrm{A}$ where A is a set whose supremum is y. By the previous argument, if y infinite, then y is critical.

Consider the sequence defined by $z_{n+1} = f(z_n, z_n)$, with $z_0 \ge u+2$. It is obvious by induction that this is a sequence of ordinals and is strictly increasing. Let y be the supremum. This is a critical ordinal $\ge z_0$. In fact, y is infinite, $> u$ and a limit ordinal. It thus remains to show that $x < y$ and $z < y$ imply $f(x,z) \le y$. There is n such that $x \le x_n$, $z \le x_n$, so that $f(x,z) \le f(x_n,x_n) = x_{n+1} \le y$.

Let A be a set of critical elements and y its supremum. Then y is critical. The case $y \in \mathrm{A}$ is obvious. Otherwise y is a limit ordinal. All we have to do is to show that $f(x,z) \le y$ whenever $x < y$ and $z < y$. Let t be the supremum of x and z. We have $t+1 < y$ since y is limit. Thus, there exists $u \in \mathrm{A}$ such that $x \le t < u$ and $z \le u$. We have $f(x,z) \le f(t,u) = u \le y$. One deduces: the collection of critical elements is a closed and proper class.

Finally, a critical ordinal is indecomposable, for if $x < y$ we have $x+y \le f(x,y)$ (remember that y is infinite).

```
Definition critical_ordinal y :=
```

```
 [/\ omega0 <=o y, u <o y &
   forall x, u <=o x -> x <o y  -> f x y = y].

Lemma critical_limit y: OIax1b u w0 g ->
   critical_ordinal y -> limit_ordinal y.
Lemma is_critical_pr y: OIax1b u w0 g ->
  omega0 <=o y -> u <o y ->
  (forall x, u <=o x ->  x <o y -> f x y <=o y) ->
  critical_ordinal y.
Lemma sup_critical A y: OIax2 u w0 g ->
   omega0 <=o y -> ordinal_set A -> \osup A = y ->
   (forall x, inc x A -> u <=o x) ->
   (forall x, inc x A -> f x y = y) ->
   critical_ordinal y.
Lemma sup_critical2 v: OIax2 u w0 g -> ordinalp u -> u +o \2o <=o v ->
  let A:= target (induction_defined0 (fun _ z => f z z) v) in
  critical_ordinal (\osup A) /\ v <=o (\osup A).
Lemma sup_critical3 A:  OIax2 u w0 g -> nonempty A ->
   (forall x, inc x A -> critical_ordinal x) ->
   critical_ordinal (\osup A).
Lemma critical_closed_proper: OIax2 u w0 g ->  ordinalp u ->
  iclosed_proper critical_ordinal.
Lemma critical_indecomposable y: OIax2 u w0 g ->
  critical_ordinal y -> indecomposable y.
End OrdinalInduction2.
```

11.10 Ordinal power

There are two possible definitions of the ordinal power of two ordinals x and y. Given two well-ordered sets X and Y, one can consider the ordered set of functions Y → X. This is well-ordered only if Y is finite, so let's consider the subset of functions f such that $f(t)$ is the least element of X for all but a finite number of values of t. The ordinal of this set depends only on the ordinals of X and Y, and satisfies the properties of exponentiation; this is the subject of Exercise 2.19 (page 587). A much easier solution is to define exponentiation by induction, following Cantor [7, §18, Theorem A] or Bourbaki, Exercise 2.18 (page 586).

We define a function (for $x \geq 2$) by ordinal induction with $w_0(x) = 1$ and $g(x, y) = x \cdot y$. All assumptions of the previous section are fulfilled. We add the following specifications: $1^x = 1$, $0^0 = 1$ and $0^y = 0$ for non-zero y.

```
Definition opow' := ord_induction_defined (fun z:Set => \1o) oprod2.
Definition opow a b :=
  Yo (a = \0o)
     (Yo (b = \0o) \1o \0o)
     (Yo (a = \1o) \1o (opow' a b)).
Notation "x ^o y" := (opow x y) (at level 30).

Lemma ord_pow_axioms:  OIax3 \2o (fun z:Set => \1o) oprod2.
```

We start with trivial properties.

```
Lemma opow00: \0o ^o \0o = \1o.
Lemma opow0x x: x <> \0o -> \0o ^o x = \0o.
Lemma opow0x' x: \0o <o x ->  \0o ^o x = \0o.
```

```
Lemma opow1x x: \1o ^o x = \1o.
Lemma opowx0 x: x ^o \0o = \1o.
Lemma opowx1 x:  ordinalp x -> x ^o \1o = x.
Lemma opow2x x y: \2o <=o x -> x ^o y = opow' x y.
```

We apply the previous results. The quantity x^y is an ordinal. It is a normal ordinal functional (as a function of y) for $x \geq 2$. The function is strictly increasing in y for $x \geq 2$; it is increasing (in both arguments) if $x > 0$. We shall see below that n^ω is independent of n when n is an integer ≥ 2, so that the function is not strictly increasing in x for fixed y. Note that $x^y = 0$ only when x is zero and y is non-zero, that $x^y = 1$ only when $x = 1$ or $y = 0$. We have, for all ordinals (compare with (11.23)):

$$a^{b+c} = a^b \cdot a^c \qquad a^{b\cdot c} = (a^b)^c. \tag{11.24}$$

```
Lemma OS_pow x y: ordinalp x -> ordinalp y ->
  ordinalp (x ^o y).
Lemma opow_normal  x: \2o <=o x ->
  normal_ofs (opow x).
Lemma opow_Meqle1 x y: \0o <o x -> \0o <o y  -> x <=o (x ^o y).
Lemma opow_Mspec x y: \2o <=o x ->
  ordinalp y  -> y <=o (x ^o y).
Lemma opow_Meqlt x y y': \2o <=o  x ->
  y <o y' -> (x ^o y) <o (x ^o y').
Lemma opow_Meqltr a b c: \2o <=o a ->
  ordinalp b -> ordinalp c ->
  ((b <o c) <-> ( (a ^o b) <o (a ^o c))).
Lemma opow_regular a b c: \2o <=o a ->
  ordinalp b -> ordinalp c -> a ^o b = a ^o c -> b = c.
Lemma opow_nz0 x y:
  ordinalp x -> ordinalp y -> x ^o y = \0o ->
  (x = \0o /\ y <> \0o).

Lemma opow_npos a b: \0o <o a -> ordinalp b -> \0o <o (a ^o b).
Lemma opow2_pos  a b: \2o <=o a -> ordinalp b -> \0o <o (a ^o b).
Lemma opow_Mlele x x' y y':
  x <> \0o ->  x <=o x' -> y <=o y' ->
 (x ^o y) <=o (x' ^o y').
Lemma opow_Mleeq x x' y:
  x <> \0o ->  x <=o x' -> ordinalp y ->
  (x ^o y) <=o (x' ^o y).
Lemma opow_Meqle x y y':
  \0o <o x -> y <=o y' -> (x ^o y) <=o (x ^o y').
Lemma opow_eq_one x y : ordinalp x -> ordinalp y -> x ^o  y = \1o ->
    (x = \1o \/ y = \0o).
Lemma opow_sum a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
    a ^o (b +o c) = (a ^o b) *o  (a ^o c).
Lemma opow_prod a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  a ^o (b *o c) = (a ^o b) ^o c.
Lemma opow_succ x y: ordinalp x -> ordinalp y ->
  x ^o (osucc y) = (x ^o y) *o x.
```

If a and b are integers, then the ordinal power a^b is equal to the cardinal power and is an integer (by induction on b, since $a^{c+1} = a^c \cdot a$). Thus, if both arguments are $< \omega$ so is the power. If both arguments are of cardinal ≤ C, for some infinite cardinal C, so is the power (by ordinal induction; let b be the least for which the power is is not ≤ C; then b has to be a limit ordinal; by normality a^b is then the supremum of ordinals with cardinal ≤ C indexed by a set with cardinal ≤ C). One can replace "x is of cardinal ≤ C" by "x is countable".

```
Lemma opow_2int a b: natp a -> natp b ->  a ^o b = a ^c b.
Lemma opow_2int1 a b: a <o omega0 -> b <o omega0 -> (a ^o b) <o omega0.
Lemma cnext_pow C x y:
  infinite_c C -> inc x (cnext C) -> inc y (cnext C) ->
  inc (x ^o y) (cnext C).
Lemma opow_countable x y:
   countable_ordinal x -> countable_ordinal y -> countable_ordinal (x ^o y).
```

Assume $a \geq 2$. Then $a^b \geq ab$. The result is true for $b = 0$, $b = 1$, and also if $b = c+1$ since $a^{c+1} = a^c \cdot a \geq a^c + a$, since $a \geq 2$ and $a^c \geq 1$. The result is true when b is a limit ordinal by normality of multiplication. By (11.20), there exists a greatest ordinal y such that $\omega^y \leq x$, provided that x is non-zero.

We give a name to the important function $x \mapsto \omega^x$.

```
Definition oopow x := omega0 ^o x.

Lemma opow_Mspec2 a b: ordinalp b ->
  \2o <=o a -> (a *o b) <=o (a ^o b).
Lemma opow_pos x: ordinalp x -> \0o <o oopow x.
Lemma opow_nz x: ordinalp x -> oopow x <> \0o.
Lemma opow_Mo_le a b: a <=o b -> (oopow a) <=o (oopow b).
Lemma opow_Mo_lt a b:  a <o b -> (oopow a) <o (oopow b).
Lemma opow_Mo_leP a b: ordinalp a -> ordinalp b ->
  (a <=o b <-> (oopow a) <=o (oopow b)).
Lemma opow_Mo_ltP a b: ordinalp a -> ordinalp b ->
   (a <o b <-> (oopow a) <o (oopow b)).
Lemma omega_log_p1 x: \0o <o x ->
   exists y, [/\ ordinalp y, oopow y <=o x & x <o oopow (osucc y)].
```

Moreover

$$\exists!(x,y,z), \quad b = a^x \cdot y + z, \qquad z < a^x, \quad 0 < y < a \tag{11.25}$$

(since we require y non-zero, this implies $b > 0$). In the case $b < a$, the triple is $(0,b,0)$. Otherwise, existence and uniqueness of x follows from (11.20). Obviously y and z are the quotient and remainder in the division of b by a^x.

```
Definition ord_ext_div_pr a b x y z :=
 [/\ ordinalp x, ordinalp y, ordinalp z &
  [/\ b =  ((a ^o x) *o y) +o z,
   z <o (a ^o x), y <o a & \0o <o y]].

Lemma ord_ext_div_unique a b x y z x' y' z':
  \2o <=o a -> ordinalp b ->
  ord_ext_div_pr a b x y z ->  ord_ext_div_pr a b x' y' z' ->
  (/\ x=x', y=y' & z=z'].
Lemma ord_ext_div_exists a b:
  \2o <=o a -> \0o <o b ->
  exists x y z,  ord_ext_div_pr a b x y z.
```

We have the equivalent of (11.4), namely that a^b is the set of all $a^x \cdot y + z$. More precisely,

$$a^b = \{0\} \cup \bigcup_{x \in b} \{t, \exists y \in a, \exists z \in a^x, t = a^x \cdot y + z, y \neq 0\}. \tag{11.26}$$

```
Lemma opow_rec_def a b: \2o <=o a -> ordinalp b ->
  a ^o b = unionb (Lg b (fun x =>
    fun_image( (a-s1 \0o)\times a ^o x) (fun p => (a^o x) *o (P p) +o (Q p))))
    +s1 \0o.
```

We state: an ordinal x is indecomposable if and only if it is of the form ω^y for some y. In particular, ω^y is limit if $y > 0$. In fact, write $x = \omega^a \cdot b + c$. If x is indecomposable, it follows $c = 0$. Since $b < \omega$, it is finite, hence a successor. Thus $x = \omega^a \cdot b' + \omega^a$. Note that x cannot be equal to the first term, as the second is non-zero. Thus x is equal to the second. On the other hand, consider the least y such that ω^y is not indecomposable. This cannot be zero, neither a successor (as $t \cdot \omega$ is indecomposable). Thus y is limit, and ω^y is the supremum of a family of indecomposable ordinals, thus is indecomposable. We deduce an enumeration of the indecomposable ordinals. (This is Corollary 2 of Theorem 2 of [22]).

```
Lemma indecomp_prop3 x:
  indecomposable x -> exists2 y, ordinalp y & x = oopow y.
Lemma indecomp_prop4 y: ordinalp y ->
  indecomposable (oopow y).
Lemma indecomp_limit2 n: \0o <o n ->  limit_ordinal (oopow n).
Lemma indecomp_enum:
  oopow =1o ordinalsf indecomposable.
```

Let's show (Corollary 3 of Theorem 4 of [22]) that the first derivation of the OFS $x \mapsto a \cdot x$ is $y \mapsto a^\omega \cdot y$, assuming $a > 0$. First, if $x = a^\omega \cdot y$ then $a \cdot x = x$ (essentially, because $1 + \omega = \omega$). On the other hand, write $x = a^\omega \cdot y + r$ by division. If $a \cdot x = x$, it follows $a \cdot r = r$, thus $a^n \cdot r = r$ for any integer n. We pretend $r = 0$. This is true if $a = 1$, as $r < a^\omega$. If r were non-zero, we would have $a^n \leq r$, thus $a^\omega \leq r$, by normality, absurd. We have shown: the fix-points of $x \mapsto a \cdot x$ is the image $a^\omega \cdot y$; the conclusion follows easily.

```
Lemma oprod_fix1 a y (x := a ^o omega0 *o y):
  \0o <o a  -> ordinalp y -> a *o x = x.
Lemma oprod_fix2 a x: \0o <o a -> ordinalp x -> a *o x = x ->
  exists2 y, ordinalp y & x =  a ^o omega0 *o y.
Lemma mult_fix_enumeration a: \0o <o a ->
  first_derivation (oprod2 a) =1o (oprod2 (a ^o omega0)).
```

11.11 Repeated derivations

We shall consider in follows a normal OFS f, and its n-th derivation. This is written $f(x,n)$ by Veblen, but we prefer $g(x,n)$ or $g_n(x)$. In the case where $f(x) = \omega^x$ we shall write this as $\phi_n(x)$ or $\phi(n,x)$ (note the order of arguments). The quantity $\phi_1(x)$ is what Cantor denotes ϵ_x, and will be studied in a moment.

We shall define g_n when n is an arbitrary non-zero ordinal as the enumeration of Z_n, where Z_n is the collection of all x such $g_i(x) = x$ whatever $i < n$. Let F_i be the collection of fix-points of g_i, so that Z_n is the intersection the $(F_i)_{i<n}$. Note that F_{i+1} is a subset of F_i so that Z_{n+1} is F_n, and g_{n+1} is the first derivation of g_n. However, if n is not a successor, it is

unclear whether Z_n has an enumeration. For this reason, we first assume that f is a function $E \to E$, as well as g_i, so that F_i and Z_n are now subsets of E (thus are sets, not proper classes).

Assume: E is the cardinal successor of some infinite cardinal C; if $i \le j$ then $F_j \subset F_i$; each F_i is closed and cofinal in E; the cardinal of the ordinal I is at most C. Then $\bigcap_{i \in I} F_i$ is closed and cofinal in E (in particular the intersection is non-empty). This is Theorem 5 of [22].

```
Lemma closed_cofinal_inter C S (E := cnext C) (T:= intersectionb S):
  infinite_c C -> fgraph S -> ordinalp (domain S) ->
  cardinal (domain S) <=c C ->
  nonempty (domain S) ->
  (forall i, inc i (domain S) -> iclosed_set (Vg S i)) ->
  (forall i, inc i (domain S) -> ord_cofinal (Vg S i) E) ->
  (forall i j, i<o j -> inc i (domain S) -> inc j (domain S) ->
       sub (Vg S j) (Vg  S i)) ->
  iclosed_set T /\ ord_cofinal T E.  (* 80 *)
```

Let now f be a normal function defined on E. We define by transfinite induction on E a function g_i, the enumeration function of the fix-points of $(g_j)_{j<i}$, where g_0 is f. Theorem 6 of [22] states that such a function exists. Assume $E \subset E'$ (i.e. $C \le C'$) and that f' extends f. Then for $i \in E$, each g'_i extends g_i.

```
Definition many_der_aux f E g :=
  Yo (source g = \0o) f
     (ordinalsE E (intersectionf (source g) (fun z => fixpoints (Vf g z)))).

Definition many_der f E :=
  transfinite_defined (ordinal_o E) (many_der_aux f E).

Lemma many_der_ex C (E := cnext C) f (g:= many_der f E)
  (ii:= fun i => (intersectionf i (fun z => fixpoints (Vf g z)))):
  infinite_c C -> normal_function f E E ->
  [/\ function g, source g = E, Vf g \0o = f,
      (forall i, inc i E ->  i <> \0o -> Vf g i = ordinalsE E (ii i)) &
       [/\ forall i, inc i E -> i <> \0o ->
          lf_axiom (ordinals (fun x y => [/\ inc x  (ii i), inc y  (ii i)
        & x <=o y])) E E,
       (forall i, inc i E ->  i <> \0o ->
          iclosed_set (ii i) /\ ord_cofinal (ii i) E),
       (forall i, inc i E -> normal_function (Vf g i) E E) &
       (forall i, inc i E -> i <> \0o ->
         Imf (Vf g i) =  (ii i))]]. (* 97 *)
Lemma many_der_unique C1 C2 f1 f2 (E1:= cnext C1) (E2:=c cnext C2)
   (g1:= many_der f1 E1) (g2:= many_der f2 E2):
  C1 <=c C2 -> infinite_c C1 -> infinite_c C2 ->
  agrees_on E1 f1 f2 ->
  normal_function f1 E1 E1 -> normal_function f2 E2 E2 ->
  forall i, inc i E1 -> agrees_on E1 (Vf g1 i) (Vf g2 i).  (* 51 *)
```

We have constructed above a normal OFS, such that for no infinite cardinal successor E the restriction f_E can be considered as a function $E \to E$. For this reason, we assume that there is some $b(x,i)$ satisfying: if x and i are ordinals, then $E = b(x,i)$ satisfies

(H) $$x \in E; \quad i \in E; \quad \forall t, t \in E \implies f(t) \in E; \quad E = C_+$$

where $E = C_+$ means: E is the cardinal successor of some infinite cardinal C (which is the greatest cardinal that belongs to E).

```
Definition all_der_bound (f: fterm) (b: fterm2) :=
  forall x i, ordinalp x -> ordinalp i ->
    [/\ inc x (b x i), inc i (b x i),
      (exists2 C, infinite_c C & b x i = cnext C) &
      forall t, inc t (b x i) -> inc (f t) (b x i)].

Lemma all_der_bound_prop f b x i
   (E:= b x i)  (C := (\csup (Zo E cardinalp))):
   ordinalp x -> ordinalp i -> all_der_bound f b ->
   [/\ infinite_c C, inc x E, inc i E, E = cnext C &
    forall t, inc t E -> inc (f t) E].
```

We shall assume here that f is a normal OFS, and that b satisfies the properties described above. For any x and i, if $E = b(x,i)$, we may apply the previous theorem to the normal function F_E and get a function g_E; we define $g_i(x)$ as $g_E(i,x)$.

By uniqueness, if $H(E, f, x, i)$ holds for some E, then $g(x,i) = g_E(x,i)$. By construction $g_0 = f$ and g_i is a normal OFS (in particular is strictly increasing). If $j < i$, every $y = g_i(x)$ is a fix-point of g_j. Conversely, if y a fix-point of g_j for all $j < i$, then y has the form $y = g_i(x)$. One deduces an equivalent formula for $g_i(x) = g_j(y)$ and $g_i(x) < g_j(y)$ (see (11.68) and (11.69), where we write ϕ instead of g).

We state[3] (corollary 1 of Theorem 6 of [22]): if $f(0) \neq 0$, then $g(0)$ is a normal OFS. Proof. Write h instead of $g(0)$. This is the mapping $x \mapsto g(0,x)$, and should not be confused with g_0, the mapping $x \mapsto g(x,0)$. Assume $i < j$. Then $h(i) < h(j)$ is equivalent to $0 < h(j)$ by (11.69). Since $0 < j$ we get $g(h(j),0) = h(j)$. By assumption, $h(j)$ cannot be zero, so that h is strictly increasing. Let j be a limit ordinal, T the set of all $h(i)$ for $i \in j$ and $w = \sup T$. We pretend that w is a fix-point of all g_k for $k < j$. As a consequence, w is in the range of g_j; let's say $w = g(u,j)$. Since $h(j) = g(0,j)$ is an upper bound of T we get $g(u,j) = w \leq g(0,j)$. Since g_j is increasing, this says $u = 0$ and $w = h(j)$. Consider now the set $E = b(0,j)$, and the function g_E. As T is a subset of E whose cardinal is small, we have $w \in E$. Assume now $k < j$. Let T_k be the set of all $h(i)$ for $k < i < j$. As h is increasing, we have $\sup T = \sup T_k$. If $x \in T_k$, then $x = g(x,k)$. Thus T_k is $g_E(k)\langle T_k\rangle$. As $g_E(k)$ is normal, the supremum w of this set is the value $g_E(k)$ at the supremum of T_k. Thus $w = g(w,k)$.

```
Definition all_der_aux (f:fterm) E x i :=
   Vf (Vf (many_der (Lf f E E) E) i) x.

Definition all_der (f:fterm) (b:fterm2) x i := all_der_aux f (b x i) x i.

Section All_derivatives.
Variable f: fterm.
Variable b: fterm2.
Hypothesis nf: normal_ofs f.
Hypothesis bf: all_der_bound f b.
Let g:= all_der f b.

Lemma all_der_bound_prop2 x i:
   ordinalp x -> ordinalp i ->
```

[3]Since Veblen considers only non-zero ordinals, his statement is: if $f(1) > 1$, then g_1 is normal

```
   normal_function (Lf f (b x i) (b x i)) (b x i) (b x i).
Lemma all_der_p1 x i C (E:= cnext C):
  infinite_c C ->
  inc x E -> inc i E -> (forall t, inc t E -> inc (f t) E) ->
  g x i =  all_der_aux f E x i.
Lemma all_der_p2 x:  ordinalp x -> g x \0o = f x.
Lemma all_der_p3 x i: ordinalp x -> ordinalp i -> inc (g x i) (b x i).
Lemma OS_all_der x i: ordinalp x -> ordinalp i -> ordinalp (g x i).
Lemma all_der_p4 x y i: ordinalp i -> x <o y  ->
  [/\  inc x (cnext (union (Zo (b y i) cardinalp))),
       g x i = all_der_aux f (b y i) x i &
       g y i = all_der_aux f (b y i) y i].
Lemma all_der_p5 x y i: ordinalp i -> x <o y  -> g x i <o g y i.
Lemma all_der_p5' x y i: ordinalp i -> ordinalp x -> ordinalp y  ->
  g x i = g y i ->  x = y.
Lemma all_der_p5'' x y i: ordinalp i -> ordinalp x -> ordinalp y  ->
  g x i <o g y i ->  x<o y.
Lemma all_der_p6 i: ordinalp i -> normal_ofs (g ^~i).
Lemma all_der_p7 x i j: ordinalp x -> i <o j ->  g (g x j) i = g x j.
Lemma all_der_p8 y j: ordinalp y -> \0o <o j ->
   (forall i, i <o j ->  g y i = y) ->
   (exists2 x, ordinalp x & y = g x j).
Lemma all_der_p9 x y i j:
  ordinalp x -> ordinalp y -> ordinalp i -> ordinalp j ->
  (g x i = g y j  <->
   [/\ i <o j -> x = g y j, i = j -> x = y & j <o i -> y = g x i]).
Lemma all_der_p10 x y i j:
  ordinalp x -> ordinalp y -> ordinalp i -> ordinalp j ->
  (g x i <o g y j  <->
   [/\ i <o j -> x <o g y j, i = j -> x <o y & j <o i -> g x i <o y]).
Lemma all_der_p10' x y i j:
  ordinalp x -> ordinalp y -> ordinalp i -> ordinalp j ->
  (g x i <=o g y j  <->
   [/\ i <o j -> x <=o g y j, i = j -> x <=o y & j <o i -> g x i <=o y]).
Lemma all_der_p11 x i j : ordinalp x -> i <=o j -> g x i <=o g x j.
Lemma all_der_p12 i : ordinalp i -> f \0o = \0o -> g \0o i = \0o.
Lemma all_der_p13 : f \0o <> \0o -> normal_ofs (g \0o).   (* 93 *)
Lemma all_der_p14 i: ordinalp i ->
  first_derivation (g ^~i) =1o g ^~(osucc i).

End All_derivatives.
```

Assume $f(x) = f'(x)$ for any ordinal x, and that f is a normal OFS; then f' is a normal OFS. Assume that we have a found b for f and a bound b' for f'. We can then construct g and g'. These two functions are identical (we cannot prove $g = g'$, we just pretend $g(x,i) = g'(x,i)$ for any pair of ordinals).

Assume $f(0) = 0$. We have shown that $g(0,i) = 0$ whatever i. More generally, assume $f(x) = x$ for $x < z$. Then, for $x < z$, $g(x,i) = x$ whatever i. (proof by induction on i; this is obvious for $i = 0$; if for all $j < i$ we have $g(x, j) = x$, we deduce that x has the form $g(t,i)$, for some t; obviously $t \leq x$, by induction $t < x$ is impossible).

Let $f'(x) = f(z+x)-z$. We know that this is a normal OFS, and we have an obvious bound, so that there is a associated OFS g'. By induction on i, we have $z + g'(x,i) = g(z + x,i)$. It follows $g(z,i) = z + g'(0,i)$. Assume now $f(z) \neq z$. This is $f'(0) \neq 0$, and says that $g'(0)$ is normal. We deduce: $i \mapsto g(z,i)$ is a normal OFS.

In particular, if $f(0)=0$ and $f(1)\neq 1$, then $i\mapsto g(1,i)$ is a normal OFS.

```
Lemma all_der_unique f1 f2 b1 b2
   (g1:=  all_der f1 b1) (g2:=  all_der f2 b2):
   normal_ofs f1 -> all_der_bound f1 b1 -> all_der_bound f2 b2 ->
   (f1 =1o f2) ->
   (forall x i, ordinalp x -> ordinalp i -> g1 x i = g2 x i).
Lemma all_der_p12_bis f b (g:= all_der f b) z:
  normal_ofs f ->all_der_bound f b ->
  ordinalp z -> (forall x, x <o z -> f x = x) ->
  (forall x i, x <o z -> ordinalp i -> g x i = x).
Lemma all_der_p13_bis f b (g:= all_der f b) z: (* 72 *)
  normal_ofs f ->all_der_bound f b ->
  ordinalp z -> (forall x, x <o z -> f x = x) ->
  f z <> z -> normal_ofs (g z).
Lemma all_der_p13_ter f b (g:= all_der f b):
  normal_ofs f -> all_der_bound f b ->
  f \0o = \0o -> f \1o <> \1o -> normal_ofs (g \1o).
```

11.12 Cantor Normal Form

In Exercise 6.12 (page 656) Bourbaki says that any ordinal α can be written uniquely as

$$\alpha=\gamma^{\lambda_1}\cdot\mu_1+\gamma^{\lambda_2}\cdot\mu_2+\ldots+\gamma^{\lambda_k}\cdot\mu_k, \tag{11.27}$$

where $0<\mu_i<\gamma$ and the sequence λ_i is strictly decreasing, provided that $\gamma\geq 2$. This is a generalization of the fact that every integer has an expansion to base b;

A case of interest, considered by Cantor, is when $\gamma=\omega$,

$$\alpha=\omega^{\lambda_1}\cdot\mu_1+\omega^{\lambda_2}\cdot\mu_2+\ldots+\omega^{\lambda_k}\cdot\mu_k. \tag{11.28}$$

Here the condition $0<\mu_i<\omega$ says that μ_i is a non-zero integer.

As every μ_i has the form $1+1+\ldots+1$ we can write

$$\alpha=\omega^{\beta_1}+\omega^{\beta_2}+\cdots+\omega^{\beta_m}, \tag{11.29}$$

where the sequence β_i is decreasing.

11.12.1 The simple normal form

We consider here (11.29). By a CNFr we mean a functional term f and an integer n; we assume that $f(i)$ is an ordinal for $i<n$ and f increasing, so $f(i)\leq f(i+1)$ whenever $i+1<n$. The greatest term $f(n-1)$ will be called the *degree.*, the sum of the $\omega^{f(i)}$ will be called the *sum* of the CNF and denoted $s_n(f)$. Note that $f(i)$ represents β_{m-i}, and the sum is inductively defined by $s_{n+1}(f)=\omega^{f(n)}+s_n(f)$.

```
Definition CNFrv (f: fterm) n := osumf (fun i => oopow (f i)) n.
Definition CNFr_ax (f:fterm) n :=
  ord_below f n
  /\ (forall i, natp i -> (csucc i) <c n -> (f i) <=o (f (csucc i))).
```

Let d be the degree of a CNFq X. Then

$$\omega^d \le s(\mathrm{X}) < \omega^{d+1}. \tag{11.30}$$

The first relation is trivial; for the second note that ω^{d+1} is indecomposable.

```
Lemma CNFr_ax_s f n: natp n -> CNFr_ax f (csucc n) -> CNFr_ax f n.
Lemma OS_CNFr_t f n i: CNFr_ax f n -> i <c n ->  ordinalp (oopow (f i)).
Lemma CNFr_rec f n: natp n ->
  CNFrv f (csucc n) = oopow (f n) +o CNFrv f n.

Lemma OS_CNFr0 f n: natp n -> CNFr_ax f n -> ordinalp (CNFrv f n).
Lemma CNFr_p2 f n: natp n -> CNFr_ax f (csucc n) ->
  oopow (f n) <=o CNFrv f (csucc n).
Lemma CNFr_p3 f n i:
  natp n -> CNFr_ax f n -> i <c n -> oopow (f i) <=o CNFrv f n.
Lemma CNFr_p4 f n: natp n -> CNFr_ax f (csucc n) ->
  CNFrv f (csucc n) <o oopow (osucc (f n)).
Lemma CNFr_p4' f n m: natp n -> CNFr_ax f (csucc n) ->
  f n <o m -> CNFrv f (csucc n) <o oopow m.
Lemma CNFr_p5 f: CNFrv f \0c = \0o.
Lemma CNFr_p6 f n: natp n -> CNFr_ax f n ->  n  <> \0c ->
    CNFrv0 f n <> \0o.
```

Uniqueness of the CNF follows from (11.30). If x is non-zero, we can write $x = \omega^a \cdot b + r$ by extended division; note that $b < \omega$ says that b is an integer, thus of the form $1 + b'$. In particular $x = \omega^a + r'$, where $r' < x$ and $r' < \omega^{a+1}$. Existence follows by induction.

```
Lemma CNFr_unique f g n m: CNFr_ax f n -> CNFr_ax g m ->
   natp n -> natp m -> CNFrv f n = CNFrv g m ->
   (n = m /\ same_below f g n).
Lemma CNFr_exists_aux x: \0o <o x ->
   exists n y, [/\ ordinalp n, y <o x , y <o oopow n *o omega0  &
                x = oopow n +o y].
Lemma CNFr_exists x: ordinalp x ->
   exists f n, [/\ natp n, CNFr_ax f n & x = CNFrv f n].
```

11.12.2 Indecomposable ordinals

We study here some properties of the relation $x + y = y$. We say that x can be *neglected* before y and write it as $x \ll y$. We shall see that this is equivalent to $d(x) < d(y)$, where d is the degree.

We start with basic facts. If $x \ll y$, then either $x = y = 0$ or $x < y$. It implies $x \ll y + z$, hence $x \ll z$ whenever $y \le z$. If $a \ll y$ and $b \ll y$ then $a + b \ll y$; the converse is true (if $a + b + y = y$ then $b + y \le y$. From $y \le b + y$ we get $b + y = y$, then $a + y = y$). Hence $a \le b$ and $b \ll y$ says $a \ll y$.

```
Definition ord_negl a b := a+o b = b.
Notation "x <<o y" := (ord_negl x y) (at level 60).

Lemma ord_negl_lt a b: ordinalp a -> ordinalp b ->
  a <<o b -> ((a = \0o /\ b = \0o) \/ (a <o b)).
Lemma ord_negl_trans a b c : ordinalp a -> ordinalp b -> ordinalp c ->
```

```
  a <<o b -> b <<o c -> a <<o c.
Lemma ord_negl_sum a b c: ordinalp a -> ordinalp b -> ordinalp c ->
  a <<o b -> a <<o (b +o c).
Lemma ord_negl_sum' a b c:
  ordinalp a -> a <<o b -> b <=o c -> a <<o c.
Lemma ord_negl_sum1 a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  (a <<o c /\ b <<o c <-> (a +o b) <<o c).
Lemma ord_negl_sum3 a b c:
  ordinalp c -> a <=o b -> b <<o c -> a <<o c.
```

Note that $a \ll a+b$ implies $a \ll b$ as we can simplify. If $a \ll b$ and c is non-zero then $a \ll b\cdot c$ since $b\cdot c = b+b\cdot(c-1)$. Note that, if x is finite and y is infinite, then $x < \omega \le y$; from $x \ll \omega$ we get $x \ll y$. We have $a \ll b$ if and only if $a\cdot\omega \le b$. Proof: by induction on the integer n, $a \ll b$ implies $a\cdot n \ll b$, hence $a\cdot n \le b$ and $\sup_n(a\cdot n) = a\cdot\omega \le b$; the converse follows from $a \ll a\cdot\omega$.

```
Lemma ord_neg_sum4 a b : ordinalp a -> ordinalp b ->
  a <<o (a +o b) -> a <<o b.
Lemma ord_negl_prod a b c: ordinalp a -> ordinalp b ->
  \0o <o c -> a <<o b -> a <<o (b *o c).
Lemma ord_negl_prodr a b c: ordinalp a -> ordinalp b -> ordinalp c ->
   a <<o b ->  c *o a <<o (c *o b).
Lemma ofinite_plus_infinite x y:
  finite_o x -> ordinalp y -> infinite_o y -> x +o y = y.
Lemma ord_negl_pint a b n : natp n -> ordinalp a -> ordinalp b ->
  a <<o b -> a *o n <<o b.
Lemma ord_negl_alt a b: ordinalp a -> ordinalp b ->
  (a <<o b <-> a *o omega0 <=o b).
Lemma ord_negl_p6 e f n:
  natp n ->  CNFr_ax f (csucc n) ->
  e <o (f n) -> (oopow e) <<o (CNFrv f (csucc n)).
```

If x is finite, then $x \ll \omega$, thus $x \ll \omega^e$ when $e > 0$. Assume $a < c$; we have $\omega^a \ll \omega^c$ and $\omega^a\cdot b \ll \omega^c\cdot d$, when b is finite, d non-zero.

```
Lemma ord_negl_p1 b:  b <o omega0 -> b <<o omega0.
Lemma ord_negl_opow a b: a <o omega0 ^o b -> ordinalp b -> a <<o  omega0 ^o b.
Lemma ord_negl_p2 b e:
  b <o omega0 -> \0o <o e -> b <<o (oopow e).
Lemma ord_negl_p4 e e':  e <o e' -> (oopow e) <<o (oopow e').
Lemma ord_negl_p5 e c e' c':
  c <o omega0 -> e <o e' -> \0o <o c' ->
  ((oopow e) *o c) <<o ((oopow e') *o c').
```

Let X and Y be two CNFr, with degree d and e, and sum x and y. We have $e < d$ if and only if $y \ll x$. Assume first $e < d$; let $\mathrm{T} = \omega^d$ be the leading term of X. Every term in Y is $\ll \mathrm{T}$ so $y \ll \mathrm{T}$, so $\mathrm{T} \le x$ says $y \ll x$. Conversely, $\mathrm{T}' \ll x$, where T' is the leading term of Y; we deduce $\mathrm{T}'\cdot\omega \le x$. Now $\mathrm{T}'\cdot\omega = \omega^{e+1}$ and $x \le \omega^{d+1}$; it follows $e < d$.

```
Lemma ord_negl_pg f n g m:
  natp n ->  CNFr_ax f (csucc n) ->
  natp m ->  CNFr_ax g (csucc m) ->
  (g m <o f n <->  CNFrv g (csucc m) <<o CNFrv f (csucc n)).
```

¶ Application. If c is a limit ordinal, we can write $c = a + \omega^n$. If a is non-zero, then a is a limit ordinal, is at least ω^n, thus has the same cardinal as c.

```
Lemma cantor_of_limit x: limit_ordinal x ->   (* 68 *)
 exists a n, [/\ ordinalp a, ordinalp n, n <> \0o,
   x = a +o (oopow n)
   & (a = \0o \/
      [/\ limit_ordinal a, (oopow n) <=o a
       & cardinal x = cardinal a])].
```

¶ Application. Let c be an ordinal, E_b the set of all ordinals a such that $a+b=c$, and E the set of all b for which E_b is non-empty. We have shown above that E is finite. We describe these sets in detail, assuming that c is given by equation (11.29). For each k, we denote by a_k and b_k the sums of the ω^{β_i} for $i<k$ and $i \geq k$ respectively. We have $a_k + b_k = c$. (in the code that follows, $\beta_i = f_{m-i}$, the exponents are in increasing order, and k is the number of terms in b_k) .

Let T be the leading term of c, $c = T + c'$. Note that T is indecomposable. Assume $a < T$ and $c = a+b$. We have $a \ll T$, hence $a \ll c$, i.e. $a \ll a+b$ hence $a \ll b$; i.e., $b = c$. In the other case, $a = T + a'$ for some a', and $a' + b = c'$ after simplification. We deduce that E is the set of all b_k: from $a_k + b_k = c$ it follows that E_b is non-empty when b has the form b_k. The converse holds (by induction on the size of c): assume $a+b=c$; if $a<T$, then $b = c_1$, otherwise we conclude by $a'+b=c'$. Since $k \mapsto b_k$ is injective it follows that E has cardinal $m+1$.

Assume $b = b_k$, so that $a_k + b = c$. If $b = 0$, then E_b is the singleton $\{c\}$. Otherwise E_b is the set of all $a_k + d$ where $d < T'$, T' being the leading term of b. Note that $d < T'$ says $d \ll b$ so $a_k+d+b = a_k+b = c$, and $a_k+d \in E_b$. Conversely, assume $a+b=c$. If $a<T$ then $b=c$, $a_k=0$, and $T = T'$. In the other case, $a = T + a'$ for some a', $a'+b=c'$. We proceed by induction as above, except in the special case $b = c$. Here $a'+b=c'$ says $a'+T+c'=c'$. It follows $T+c'=c'$; this is $c=c'$, absurd.

```
Lemma finite_rems_aux f n a b:  natp n ->  CNFr_ax f (csucc n) ->
   ordinalp a -> ordinalp b -> CNFrv f (csucc n) = a +o b ->
   (CNFrv f (csucc n) = b /\ a <o oopow (f n))
   \/ (exists a', [/\  ordinalp a', a = oopow (f n) +o a' &
                       CNFrv f n = a' +o b ]).

Lemma finite_rems2 f n b:  natp n ->  CNFr_ax f n -> ordinalp b ->
  ((exists2 a, ordinalp a & (CNFrv f n)= a +o b) <->
   (exists2 k, k<=c n & b = CNFrv f k)).
Lemma finite_rems3 f n (c := (CNFrv f n)):  natp n ->  CNFr_ax f n ->
  cardinal (Zo (osucc c) (fun b => exists2 a, ordinalp a & c = a +o b)) =
  csucc n.
Lemma finite_rems4 f n k a (b := CNFrv f k) (c := CNFrv f n)  (* 53 *)
  (a1:= CNFrv (fun i => f (i +c k)) (n -c k)):
  natp n ->  CNFr_ax f n ->  k <=c n -> k <> \0c-> ordinalp a ->
  (c = a +o b <->
   exists d, [/\ ordinalp d, d <o (oopow (f(cpred k))) & a = a1 +o d]).
```

11.12.3 The general normal form

We introduce here some definitions for the operations on CNFs; we assume that f is a functional term and F a functional graph whose domain is an integer n (i.e., the set of integers $< n$); other quantities are integers. The first operation is restriction: $\mathrm{O}_r(\mathrm{F}, k)$ is the restriction of F to k, note that $\mathrm{O}_r(f)$ is trivially f. The second operation is shift: $\mathrm{O}_s(f, k)$ is the function that maps i to $f(i+k)$; the domain of $\mathrm{O}_s(\mathrm{F}, k)$ is $n-k$. The third operation is merge $\mathrm{O}_m(f, f', k)$ is the function that maps i to $f(i)$ for $i < k$, ro $f'(i-k)$ otherwise, $\mathrm{O}_m(\mathrm{F}, \mathrm{F}')$ is the functional graph with domain $n+n'$. The last operation is change: $\mathrm{O}_c(f, k, c)$ is the function that maps k to c, and otherwise i to $f(i)$; $\mathrm{O}_c(\mathrm{F}, c')$ will be defined later on.

```
Definition cnf_s (f: fterm) k := (fun z  => f (z +c k)).
Definition cnf_m (f1 f2:fterm) k :=
   (fun z => Yo (z <c k) (f1 z) (f2 (z -c k))).
Definition cnf_c (f:fterm) n x := (fun z => Yo (z=n) x (f z)).
Definition cnf_nr x k := restr x k.
Definition cnf_ns x k := Lg ((domain x) -c k) (cnf_s (Vg x) k).
Definition cnf_nm x y :=
  Lg ((domain x) +c (domain y)) (cnf_m (Vg x) (Vg y) (domain x)).
```

In the case equation (11.28), each μ_i is an ordinal c such that $0 < c < \omega$. This ia a positive integer. We state some properties of such numbers.

```
Definition posnatp x := natp x /\ \0c <c x.

Lemma PN1: posnatp \1c.
Lemma PN2: posnatp \2c.
Lemma posnat_prop n: natp n -> n <> \0c -> posnatp n.
Lemma posnat_ordinalp x : posnatp x <-> (\0o <o x /\ x <o omega0).
Lemma posnat_add m n: natp m -> posnatp n -> posnatp (m +c n).
Lemma posnat_csum2 m n: posnatp m -> posnatp n -> posnatp (m +c n)
Lemma PN_prod a b: posnatp a -> posnatp b -> posnatp (a *c b).
```

We consider now (11.27). This is characterized by four quantities: the base γ, the sequence of exponents λ_i, the sequence of coefficients μ_i, and the number of terms k. Let's write n instead of k, $e(i)$ instead of λ_{k-i}, and $c(i)$ instead of μ_{k-i}. As above, e and c are functional graphs. In what follows, a CNFq is a triple n, e, c satisfying $\mathrm{C}_q(\gamma, n, e, c)$. This condition says that $\gamma \geq 2$, n is an integer, e is an ordinal below n, and strictly increasing so $e(i) < e(i+1)$ when $i+1 < n$; finally $c(i) < \gamma$ for $i < n$. We say that is a CNFb and denote it $\mathrm{C}(\gamma, n, e, c)$ if it is a CNFq with $0 < c_i$ for $i < n$.

The definitions and lemmas that follow belong to a section where γ is fixed. For each i we consider $\gamma^{e(i)} \cdot c(i)$; the sum of these quantities will be the sum of the CNF, denoted $s(\mathrm{X})$.

```
Section CNFQ.
Variable gamma: Set

Definition cantor_mon (e c: fterm) i := (gamma ^o (e i)) *o (c i).
Definition CNFbv (e c:fterm) n := osumf (cantor_mon  e c) n.

Definition CNFq_ax (e c: fterm) n :=
  [/\ \2o <=o gamma,
    ord_below e n,
    (forall i, i<c n -> c i <o gamma) &
    (forall i, natp i -> (csucc i) <c n -> (e i) <o (e (csucc i)))].

Definition CNFb_ax (e c: fterm) n :=
  CNFq_ax e c n /\ (forall i, i<c n -> \0o <o c i).
```

The conditions $C_q(\gamma, e, c, n)$ and $C(\gamma, e, c, n)$ depend only on the value of $e(i)$ and $c(i)$ for $i < n$. The sum $s_n(e, c)$ shares the same property.

```
Lemma CNFq_ax_exten e1 c1 e2 c2 n:
  same_below e1 e2 n -> same_below c1 c2 n ->
  CNFq_ax e1 c1 n -> CNFq_ax e2 c2 n.
Lemma CNFb_ax_exten e1 c1 e2 c2 n:
  same_below e1 e2 n -> same_below c1 c2 n ->
  CNFb_ax e1 c1 n -> CNFb_ax e2 c2 n.

Lemma CNFq_exten e1 c1 e2 c2 n:
  same_below e1 e2 n -> same_below c1 c2 n ->
  CNFq_ax e1 c1 n -> natp n ->
  (CNFq_ax e2 c2 n /\ CNFbv e1 c1 n = CNFbv e2 c2 n).
Lemma CNFb_exten e1 c1 e2 c2 n:
  same_below e1 e2 n -> same_below c1 c2 n ->
  CNFb_ax e1 c1 n -> natp n ->
  (CNFb_ax e2 c2 n /\ CNFbv e1 c1 n = CNFbv e2 c2 n).
```

We start with some trivial lemmas. In particular, if $i < j < n$, then $e(i) < e(j)$, so $i < n-1$ implies $e(i) < e(n-1)$. The quantity $e(n-1)$ is called the degree, as above.

```
Lemma CNFq_p0 e c n i:  CNFq_ax e c n  -> i <c n ->
  ordinalp (cantor_mon e c i).
Lemma OS_CNFq e c n : natp n -> CNFq_ax e c n ->
  ordinalp (CNFbv e c n).
Lemma OS_CNFb e c n : natp n -> CNFb_ax e c n ->
  ordinalp (CNFbv e c n).
Lemma CNFq_p1 e c n: natp n ->
  CNFbv e c (csucc n) = cantor_mon e c n +o (CNFbv e c n).
Lemma CNFq_p2 e c: CNFbv e c \0c = \0o.
Lemma CNFq_p3 e c n : CNFq_ax e c n -> \0o <c n ->
   CNFbv e c \1c = cantor_mon e c \0c.
Lemma CNFq_p4 e c n i: CNFb_ax e c n -> i <c n ->
  \0o <o  cantor_mon e c i.
Lemma CNFq_r_ax e c n m: CNFq_ax e c n -> m <=c n -> CNFq_ b e c m.
Lemma CNFb_r_ax e c n m: CNFb_ax e c n -> m <=c n -> CNFb_ax e c m.
Lemma CNFq_p5 e c n: natp n -> CNFq_ax e c (csucc n)  ->
  CNFq_ax e c n.
Lemma CNFb_p5 e c n: natp n -> CNFb_ax e c (csucc n) ->
  CNFb_ax e c n.
Lemma CNF_exponents_M e c n: natp n -> CNFq_ax e c n ->
  forall i j, i <=c j -> j <c n -> e i <=o e j.
Lemma CNF_exponents_sM e c n: natp n -> CNFq_ax e c n ->
  forall i j, i <c j -> j <c n -> e i <o e j.
Lemma CNF_exponents_I e c n: natp n -> CNFq_ax e c n ->
  forall i j, i <c n -> j <c n -> e i = e j -> i = j.
Lemma CNFq_p6 e c n: natp n ->  CNFq_ax e c (csucc n) ->
   forall i, i <c n -> e i <o e n.
Lemma CNFq_p7 e c n : CNFq_ax e c n -> \1c <c n ->
   CNFbv b e c \2c = cantor_mon e c \1c +o cantor_mon e c \0c.
Lemma CNFb_p8 e c n : CNFb_ax e c (csucc n) -> natp n ->
  ordinalp (e n).
```

We consider now some operations defined above

```
Lemma CNFq_s_ax e c n k m:
  CNFq_ax e c m -> (n +c k) <=c m -> natp k -> natp n ->
  CNFq_ax (cnf_s e k) (cnf_s c k) n.
Lemma CNFb_s_ax e c n k m:
  CNFb_ax e c m -> (n +c k) <=c m -> natp k -> natp n ->
  CNFb_ax (cnf_s e k) (cnf_s c k) n.
Lemma CNFq_m_ax e1 c1 e2 c2 k m:
   natp k -> natp m ->
   CNFq_ax e1 c1 k -> CNFq_ax e2 c2 m ->
   (m = \0c \/ e1 (cpred k) <o e2 \0o) ->
   (CNFq_ax (cnf_m e1 e2 k)  (cnf_m c1 c2 k) (k +c m)).
Lemma CNFb_m_ax e1 c1 e2 c2 k m:
   natp k -> natp m ->
   CNFb_ax e1 c1 k -> CNFb_ax b e2 c2 m ->
   (m = \0c \/ e1 (cpred k) <o e2 \0o) ->
   (CNFb_ax (cnf_m e1 e2 k)  (cnf_m c1 c2 k) (k +c m)).
Lemma CNF_c_ax e c n c0 k:
   \0o <o c0 -> c0 <o gamma ->
   CNFb_ax e c n -> CNFb_ax  e (cnf_c c k c0) n.
```

We show the main associativity theorem, and a variant.

```
Lemma CNFq_A  e c n m:
   natp n -> natp m ->
   CNFq_ax  e c (n +c m) ->
   [/\  CNFq_ax e c n,  CNFq_ax (cnf_s e n) (cnf_s c n) m &
    CNFbv e c (n +c m) = CNFbv (cnf_s e n) (cnf_s c n) m +o CNFbv e c n].
Lemma CNFq_Al e c n (e1 := fun z => e (csucc z)) (c1 := fun z => c (csucc z)):
   natp n ->
   CNFq_ax e c (csucc n) ->
   [/\  ordinalp(cantor_mon e c \0c),  CNFq_ax e1 c1 n &
    CNFbv e c (csucc n) = CNFbv e1 c1 n +o (cantor_mon e c \0c)].
```

By induction, if X is a CNFq with degree $< e$, then $s(\mathrm{X}) < \gamma^e$ (this holds also if all exponents of X are $< e$).

```
Lemma CNFq_pg0 e c n:  CNFq_ax e c n  -> \0o <o gamma.
Lemma CNFq_pg1 e c n: natp n -> CNFq_ax e c (csucc n) ->
  CNFbv b e c (csucc n) <o (gamma ^o (osucc (e n))).
Lemma CNFq_pg2 e c n a: natp n -> CNFq_ax e c (csucc n) ->
  (e n) <o a -> CNFbv e c (csucc n) <o (gamma ^o a).
Lemma CNFq_pg3 e c n a: natp n -> CNFq_ax e c n ->  ordinalp a ->
  (forall i, i <c n ->  e i <o a) ->
  CNFbv e c n <o (gamma ^o a).
Lemma CNFq_pg b e c n: natp n -> CNFq_ax b e c (csucc n) ->
  CNFbv b e c n <o (b ^o (e n)).
```

Any ordinal x can uniquely be written as $x = \gamma^e \cdot c + r$, with $r < x$. By induction, every x has a CNF (assume $r = s_k(\mathrm{X})$, let X' be the CNF of length $k+1$ with e and c as exponents and coefficients at position k; then $x = s_{k+1}(\mathrm{X}')$). This CNF is unique: write $s_{k+1}(\mathrm{X}') = b^e \cdot c + s_k(\mathrm{X})$. Assume $s_{k+1}(\mathrm{X}') = s_{m+1}(\mathrm{Y}')$. Uniqueness of division gives $s_k(\mathrm{X}) = s_m(\mathrm{Y})$, and by induction, we get $\mathrm{X} = \mathrm{Y}$ and $k = m$; the relation $\mathrm{X}' = \mathrm{Y}'$ follows.

```
Lemma CNFq_pg4 e c n: natp n -> CNFb_ax b e c (csucc n) ->
```

```
  (gamma ^o (e n)) <=o CNFbv e c (csucc n).
Lemma CNFq_pg5 e c n: natp n -> CNFb_ax e c (csucc n) ->
  \0o <o CNFbv e c (csucc n).
Lemma CNF_singleton c0 e0 (c:= fun _: Set => c0)(e:= fun _: Set => e0):
  ordinalp e0 -> c0 <o gamma -> \2c <=o gamma ->
  [/\ CNFq_ax e c \1c, CNFbv b e c \1c = gamma ^o e0 *o c0 &
    (\0o <o c0 -> CNFb_ax e c \1c)].
Lemma CNF_exp_bnd e c n e0:
  \2c <=o gamma -> ordinalp e0 -> natp n ->
  CNFb_ax e c n -> CNFbv e c n <o gamma ^o e0 ->
  (forall i, i <c n -> e i <o e0).
Lemma CNFb_unique e1 c1 n1 e2 c2 n2:
  natp n1 -> natp n2 -> CNFb_ax e1 c1 n1 ->  CNFb_ax e2 c2 n2 ->
  CNFbv e1 c1 n1 = CNFbv e2 c2 n2 ->
  (n1 = n2 /\ forall i, i <c n1 -> e1 i = e2 i /\ c1 i = c2 i).
Lemma CNFb_exists b:
  ordinalp a -> \2c <=o  b ->
  exists e c n, [/\ CNFb_ax b e c n, natp n & a = CNFbv b e c n].

End CNFQ.
```

We consider now the special case where the base is ω. We restate the associativity property. The last lemma says; let $X = (e, c, n+1)$ be a CNFb, u an ordinal $< \omega$, and $c' = O_c(c, n, u)$. Then $Y = (e, c', n+1)$ is a CNFb and $s(Y) = \omega^{e(n)} \cdot u + s(X)$. This holds as the sum of two ordinals $< \omega$ is $< \omega$.

```
Definition CNFb_axo := CNFb_ax omega0.
Definition CNFbvo := CNFbv omega0.

Lemma CNFb_A e c n m:
   natp n -> natp m ->
   CNFb_axo e c (n +c m) ->
   [/\  CNFb_axo e c n,  CNFb_axo (cnf_s e n) (cnf_s c n) m &
    CNFbvo e c (n +c m) = CNFbvo (cnf_s e n) (cnf_s c n) m +o CNFbvo e c n].
Lemma CNFb_Al e c n (e1 := fun z => e (csucc z)) (c1 := fun z => c (csucc z)):
   natp n ->
   CNFb_axo e c (csucc n) ->
   [/\  ordinalp(cantor_mon omega0 e c \0o),  CNFb_axo e1 c1 n &
    CNFbvo e c (csucc n) = CNFbvo e1 c1 n +o (cantor_mon omega0 e c \0o)].
Lemma CNFb_change_nv e c n m (c':= (cnf_c c n (m +o (c n)))):
  natp n ->  CNFb_axo e c (csucc n) -> m <o omega0 ->
  (CNFb_axo e c' (csucc n) /\
    CNFbvo e c' (csucc n)  = omega0 ^o (e n) *o m +o CNFbvo e c (csucc n)).
```

¶ We define here *the* CNF of an ordinal x as an object of the theory of Bourbaki (i.e., a set, and not an aggregate formed of an integer and a functional term). In the previous version of the software, this was a triple (n, E, C) where E and C where functional graphs, with domain **N** corresponding to e and c. We simplify this as a functional graph X whose domain is n, such that $X_i = (e(i), c(i)$ for $i < n$. [The only disadvantage of this solution is that the degree of zero is no more defined].

Note that $e \le \gamma^e \le \gamma^e \cdot c \le x$. Hence (e, c) belongs to $A = x^+ \times \gamma$. It follows that the CNF belongs to $\mathfrak{P}(\mathbf{N} \times A)$.

```
Definition ocoef x i := Q (Vg x i).
```

```
Definition oexp x i :=  P (Vg x i).
Definition cnf_size x := (cpred (domain x)).

Lemma fgraph_bd x A: fgraph x -> natp (domain x) ->
  (forall i, inc i (domain x) -> inc (Vg x i) A) ->
  inc x (\Po (Nat \times A)).
```

We start with the general case; we fix $\gamma \geq 2$.

```
Section CNF_baseb.
Variable (gamma: Set).
Hypothesis bg2: \2o <=o gamma.

Definition CNFB_ax x :=
  [/\ fgraph x, natp (domain x),
   forall i, inc i (domain x) -> pairp (Vg x i) &
   CNFb_ax gamma (oexp x) (ocoef x) (domain x)].
Definition CNFBv X := CNFbv gamma (oexp X) (ocoef X) (domain X).
Definition CNFB_bound x:=
  \Po(Nat \times ((osucc x) \times gamma)).
Definition the_CNFB x :=
  select (fun X => CNFB_ax X /\ CNFBv X = x) (CNFB_bound x).
```

We state here some obvious properties. Note that the CNF of zero is empty, otherwise, it is a functional graph with $n+1$ items.

```
Lemma CNFB_unique X1 X2: CNFB_ax X1 -> CNFB_ax X2 ->
   CNFBv X1 = CNFBv X2 -> X1 = X2.
Lemma CNFB_bound_p X: CNFB_ax X -> inc X (CNFB_bound (CNFBv X)).
Lemma CNFbB_prop n e c (X :=  Lg n (fun i => J (e i) (c i))):
  CNFb_ax gamma e c n -> natp n ->
   CNFB_ax X /\ CNFBv X = CNFbv gamma e c n.
Lemma CNFB_exists x: ordinalp x ->
   exists2 X, CNFB_ax X &  CNFBv X = x.
Lemma the_CNF_p0 x (X:= the_CNFB x): ordinalp x ->  CNFB_ax X /\ CNFBv X = x.
Lemma cnf_val0:  cnf_val emptyset = \0o.
Lemma the_cnf_p0_nz x (y:= the_cnf x): \0o  <o x -> cnfp_nz y /\ cnf_val y = x.
Lemma the_CNF_p1: the_CNFB \0o = emptyset.
Lemma the_CNF_p2 x (m := (cnf_size (the_CNFB x))): \0o <o x ->
  natp m /\ domain (the_CNFB x) = csucc m.
Lemma the_CNF_p3 e c n: CNFb_ax gamma e c n -> natp n ->
  the_CNFB (CNFbv gamma e c n) = Lg n (fun i => J (e i) (c i)).
Lemma OS_degree_aux x:  ordinalp x -> x <> \0o ->
     ordinalp (oexp (the_CNFB x) (cnf_size (the_CNFB x))).
End CNF_baseb.
```

¶ From now on, we shall consider only the case where the base γ is ω. So, a *cnf* will be a functional graph, whose domain is an integer n, such that every X_i is a pair (e_i, c_i) for $i < n$, and such that $C(\omega, n, e, c)$ holds. If we define a monomial as a pair (e, c) such that e is an exponent and c a non-zero integer, we can restate the definition as: a functional graph, whose domain is an integer n, such that each value is a monomial, and the exponent function e is strictly increasing. The sum $s(X)$ is defined as above. If $A(x) = x^+ \times \omega$ then X belongs to $A(s(X))$, so that we can define the CNF X of an ordinal x as the unique cnf in $A(x)$ such that $s(X) = x$.

We define the degree and leading coefficient of an ordinal x or its CNF X as the greatest exponent or the associated coefficient. We define the degree of zero to be zero. We define also the remainder, see below.

```
Definition cnfp x :=
  [/\ fgraph x, natp (domain x),
   forall i, inc i (domain x) -> pairp (Vg x i) &
   CNFb_axo (oexp x) (ocoef x) (domain x)].

Definition cnfp_nz x:= cnfp x /\ x <> emptyset.
Definition cnf_val X := CNFbvo (oexp X) (ocoef X) (domain X).
Definition cnf_bound x:=
  \Po(Nat \times ((osucc x) \times omega0)).
Definition the_cnf x :=
  select (fun X => cnfp X /\ cnf_val X = x) (cnf_bound x).
Definition omonomp m := [/\ pairp m,ordinalp (P m) & posnatp (Q m)].

Definition cnf_degree X := oexp X (cnf_size X).
Definition cnf_lc X := ocoef X (cnf_size X).
Definition cnf_rem X := CNFbvo (oexp X) (ocoef X) (cnf_size X).
Definition odegree x :=
  Yo (x = \0o) \0o (cnf_degree (the_cnf x)).
Definition the_cnf_lc x :=  cnf_lc (the_cnf x).
Definition the_cnf_rem x := cnf_rem (the_cnf x).
```

We deduce the followong results.

```
Lemma cnfpP x:  cnfp x <->
  [/\ fgraph x, natp (domain x),
   forall i, inc i  (domain x) -> omonomp (Vg x i) &
   forall i, natp i -> (csucc i) <c (domain x) ->
             oexp x i <o oexp x (csucc i)].
Lemma cnf_val_inj: {when cnfp &, injective cnf_val}.
Lemma the_cnf_p0 x (X:= the_cnf x): ordinalp x -> cnfp X /\ cnf_val X = x.
Lemma the_cnf_0: the_cnf \0o = emptyset.
Lemma the_cnf_p2 x (m := (cnf_size (the_cnf x))): \0o <o x ->
  natp m /\ domain (the_cnf x) = csucc m.
Lemma the_cnf_p3 e c n: CNFb_axo e c n -> natp n ->
  the_cnf (CNFbvo e c n) = Lg n (fun i => J (e i) (c i)).

Lemma odegree_of_nz x: x <> \0o ->
     odegree x = oexp (the_cnf x) (cnf_size (the_cnf x)).
Lemma odegree_of_pos x: \0o <o x ->
     odegree x = oexp (the_cnf x) (cnf_size (the_cnf x)).
Lemma OS_degree x:  ordinalp x -> ordinalp (odegree x).
```

If x is a cnf, then $s(x)$ is an ordinal; if x is non-empty, then $s(x) > 0$. The remainder of a cnf is an ordinal.

```
Lemma OS_cnf_val x: cnfp  x -> ordinalp(cnf_val x).
Lemma OS_cnf_rem x : cnfp x -> ordinalp (cnf_rem x).
Lemma OS_cnf_valp x: cnfp_nz  x -> \0o <o (cnf_val x).
```

Consider the property $\mathrm{H}(x, y)$ that says: x is a cnf, y is an ordinal, the CNF of y is x, the sum of x is y. Then H is true whenever x is a cnf and y is its sum, or when y is an ordinal and x is its CNF. This allows us for instance to compute the CNF of ω^e.

```
Definition cnf_and_val x y :=
  [/\ cnfp x, ordinalp y, the_cnf y =x & cnf_val x = y].

Lemma cnf_and_val_pa x: ordinalp x -> cnf_and_val (the_cnf x) x.
Lemma cnf_and_val_pb x: cnfp x -> cnf_and_val x (cnf_val x).

Lemma cnf_two_terms n1 c1 n2 c2
      (x := (oopow n1 *o c1) +o (oopow n2 *o c2))
      (X := variantLc (J n2 c2) (J n1 c1)):
  n2 <o n1 ->  posnatp c1 -> posnatp c2  ->
  cnf_and_val X x.
Lemma cnf_one_term e c:
  ordinalp e -> posnatp c ->
  cnf_and_val (Lg \1c (fun _: Set => (J e c)))  ((oopow e) *o c).
Lemma cnf_singleton1 e: ordinalp e ->
    cnf_and_val (Lg \1c (fun _: Set => (J e \1o))) (oopow e).
```

Assume x non-zero with degree d. Then $\omega^d \le x < \omega^{d+1}$. This implies that the degree of one is zero, the degree of ω is one. Let c be the leading coefficient of x, then $x = \omega^d \cdot c + r$, where $r < \omega^d$. We know that such a decomposition is unique; we say here that the remainder r can be obtained from x by taking its CNF X, removing the highest monomial, and summing.

```
Lemma the_cnf_p4 x (n := odegree x): \0o <o x ->
  oopow n <=o x /\ x <o  oopow (osucc n).
Lemma the_cnf_p5 e c n (x:= CNFbvo e c (csucc n)):
  natp n -> CNFb_axo e c (csucc n) ->
  \0o <o x /\ odegree x = e n.
Lemma the_cnf_p6 a e: ordinalp e ->  \0o <o a -> a <o (oopow e) ->
  odegree a <o e.
Lemma odegree_opow n: ordinalp n -> odegree (oopow n) = n.
Lemma odegree_one: odegree \1o = \0o.
Lemma odegree_omega: odegree omega0 = \1o.
Lemma odegree_zero:  odegree \0o = \0c.
Lemma odegree_of_monomial e c (x := oopow e *o c):
  ordinalp e -> posnatp c ->
  \0o  <o x /\ odegree x = e.
Lemma odegree_finite x :  x <o omega0 -> odegree x = \0o.
Lemma odegree_infinite x :  omega0 <=o x -> \0o <o odegree x.
Lemma the_cnf_split x (e:= odegree x)  (c:= the_cnf_lc x) (r := the_cnf_rem x):
  \0o <o x -> [/\ \0o <o c, c <o omega0, r <o oopow e &
    x =  oopow e *o c +o r].
```

Let $x = \omega \cdot j + k$, where j is a non-zero integer and k is an integer. The degree is 1, the leading coefficient is j and the remainder is k (note: if $k = 0$, the CNF has one term, otherwise it has two terms).

```
Lemma the_cnf_omega_kj j k (X := omega0 *o j +o k):
  posnatp j  -> natp k ->
  \0o <o X /\  [/\ odegree X = \1o,  the_cnf_lc X = j & the_cnf_rem X = k].
Lemma the_cnf_omega_k k (X := omega0 +o k):
  natp k ->
  [/\ \0o <o X, odegree X = \1o &  the_cnf_rem X = k].
```

We define the *valuation* of a non-zero ordinal x, and denote it $v(x)$, as the least exponent in the CNF of x. We have $x = x' + \omega^{v(x)}$ for some ordinal x'. It follows that x is a successor if $v(x) = 0$, a limit ordinal if $v(x) > 0$.

```
Definition ovaluation x := oexp (the_CNF x) \0c.

Lemma OS_valuation x:  \0o <o  x -> ordinalp (ovaluation x).
Lemma ovaluationE e c n (x:= CNFbvo e c (csucc n)):
  natp n -> CNFb_axo e c (csucc n) -> ovaluation x = e \0c.
Lemma ovaluation_opow n: ordinalp n ->  ovaluation (oopow n) = n.
Lemma ovaluation1 x: \0o <o x ->
   exists2 y, ordinalp y & x = y +o oopow (ovaluation x).
Lemma ovaluation2 x: \0o <o x -> ovaluation x = \0o -> osuccp x.
Lemma ovaluation3 x: \0o <o x -> \0o <o ovaluation x ->
  limit_ordinal x.
Lemma ovaluation3_rev x: limit_ordinal x ->  \0o <o x /\ \0o <o ovaluation x.
Lemma ovaluation2_rev x: osuccp x -> ovaluation x = \0o  /\ \0c <c domain x.
```

We state here: if $d(\alpha) < d(\beta)$, then $\alpha \ll \beta$.

```
Lemma ord_negl_p8 ep cp p en cn n:
  CNFb_axo ep cp (csucc p) -> CNFb_axo en cn (csucc n) ->
  natp p -> natp n -> ep p <o en n ->
  (CNFbvo ep cp (csucc p)) <<o (CNFbvo en cn (csucc n)).
Lemma ord_negl_p7 x y: \0o <o x -> \0o <o y ->
  odegree x <o odegree y -> x <<o y.
```

11.12.4 Properties of the Cantor Normal Form

We study here some properties of a cnf X. If $i < j$, then $e_i(X) < e_j(X)$. This implies that $i \mapsto e_i(X)$ is injective. We deduce that the range of X, the set of all pairs $(e_i(X), c_i(X))$, is a functional graph. It follows that two cnfs with the same range are equal. Proof: if X and Y have the same range, they have the same domain (since the domain is the cardinal of the range). Assume $X_i = Y_i$ for $i < k$. Now X_k is of the form Y_i, with $i \geq k$,. Since exponents of Y are increasing it follows $e_k(Y) \leq e_k(X)$. By symmetry, we have equality. This implies $X_k = Y_k$.

```
Notation "\0f" := emptyset (only parsing).

Lemma cnf_monomial_inj x: cnfp x ->
  {when inc ^~ (domain x) & , injective (oexp x)}.
Lemma cnf_range_fgraph x: cnfp x -> fgraph (range x).
Lemma cnf_card_range z: cnfp z ->  domain z = cardinal (range z).
Lemma cnf_same_range: {when cnfp &, injective range}.
Lemma cnfp0: cnfp \0f.
```

Note that the range of a cnf is a finite functional graph, formed of monomials. The converse holds, proof by induction. Assume that E is such a set, with n elements. If it is empty, it is the range of $\emptyset$, which is a cnf. Otherwise, compare two elements of E by there first component, what we get is a finite totally ordered set. It thus has a greatest element g; by induction $E - \{g\}$ is the range of a cnf x with domain $n-1$; it suffices to extend x via $x_{n-1} = g$.

```
Definition cnf_rangep E :=
  [/\ finite_set E, fgraph E & forall x, inc x E -> omonomp x].
Definition cnf_sort E :=
  select (fun x =>  cnfp x /\ range x = E) (\Po (Nat \times E)).

Lemma finite_subset_ord X: finite_set X -> nonempty X -> ordinal_set X ->
```

```
   exists2 n, inc n X & forall m, inc m X -> m <=o n.
Lemma cnf_sort_correct E (x := cnf_sort E): cnf_rangep E ->
  cnfp x /\ range x = E.  (* 81 *)
```

If x is a cnf we denote by $\mathrm{E}(x)$ the set of its exponents (the domain of the range of x). This is a finite set of ordinals. We define $x_c(e)$ as the unique c such that the pair (e, c) belongs to the range of x. This is $c_i(x)$ when i is in the domain of x and $e = e_i(x)$. So $e \in \mathrm{E}(x)$ says that $x_c(e) > 0$. We shall use here the following trick: if $e \not\in \mathrm{E}(x)$ then $x_c(e) = 0$. In particular, if $x_c(e) \neq 0$, then e is an ordinal. It follows that If $x_c(e) = y_c(e)$ whatever e, then $x = y$.

```
Definition Vr x e := Vg (range x) e.
Definition cnf_exponents x :=  domain (range x).

Lemma Vr_correct x i: cnfp x -> inc i (domain x) ->
    Vr x (oexp x i) = (ocoef x i).
Lemma Vg_out_of_range f x: ~ inc x (domain f) -> Vg f x = emptyset.
Lemma cnf_coef_or_e_zero d: Vr \0f d = \0c.
Lemma Vr_posnat x e: cnfp x -> inc e (cnf_exponents x) ->
  posnatp (Vr x e).
Lemma cnf_coef_of_lc x: cnfp_nz  x ->
 cnf_lc x = Vr x (cnf_degree x).
Lemma NS_Vr x e: cnfp x -> natp (Vr x e).
Lemma Vr_exten x y: cnfp x -> cnfp y ->
    (forall e, Vr x e = Vr y e) -> x = y.
Lemma cnf_exponents_of x (E := cnf_exponents x):
  cnfp x -> (finite_set E /\ ordinal_set E).
Lemma Vr_ne2 x e: cnfp x -> Vr x e <> \0c ->
   ordinalp e.
```

Some other properties.

```
Lemma cnf_size_nz x (m:= cnf_size x): cnfp_nz x ->
   natp m /\ domain x = csucc m.
Lemma cnf_size_nz_bis x: cnfp_nz x ->
   inc (cnf_size x) (domain x).
Lemma cnf_size_nz_ter x: cnfp_nz x ->   inc \0c (domain x).
Lemma posnat_lc x: cnfp_nz x -> posnatp (cnf_lc x).
Lemma OS_cnf_degree x:  cnfp_nz x -> ordinalp (cnf_degree x).
Lemma cnf_degree_greatest_bis x i: cnfp x ->  i <c (cnf_size x) ->
  oexp x i <o cnf_degree x.
Lemma cnf_degree_greatest x i: cnfp x -> inc i (domain x) ->
  oexp x i <=o cnf_degree x.

Lemma cnf_sum_prop5 x: cnfp_nz x ->
  cnf_val x = (oopow (cnf_degree x)) *o (cnf_lc x) +o (cnf_rem x).
Lemma cnf_rem_prop1 x: cnfp x ->
  cnf_rem x = cnf_val (cnf_nr x (cnf_size x)).
Lemma cnfp_cnf_nr x k: cnfp x -> k <=c domain x -> cnfp (cnf_nr x k).
```

We denote by $x \le_f y$ the relation "x and y are cnfs, and either $x = y$ or there exists d such that $x_c(d) < y_c(d)$, and if $d < e$ then $x_c(e) = y_c(e)$". We have shown above that it is antisymmetric.

```
Definition cnf_lt1 x y := exists2 d,
```

```
   (Vr x d <c Vr y d)
   & forall i, d <o i ->  (Vr x i = Vr y i).
Definition clf_le x y := [/\ cnfp x, cnfp y & (x = y \/ cnf_lt1 x y)].
Definition cnf_lt x y :=  cnf_le x y /\ x <> y.

Notation "x <=f y" := (cnf_le x y) (at level 60.
Notation "x <f y" := (cnf_lt x y) (at level 60).
```

We start with some trivial properties. If x is different from y there is e such that $x_c(e) \neq y_c(e)$ so $e \in \mathrm{E}(x) \cup \mathrm{E}(y)$; there is hence a greatest such exponent; it follows that $\leq_f$ is a total ordering (the compatibility result below shows that it is in fact a well-order).

```
Lemma cnf_lt_prop x y: cnfp x -> cnfp y -> (x <f y <-> cnf_lt1 x y).
Lemma cnf_leR x: cnfp x -> x <=f x.
Lemma cnf_leA x y: x <=f y -> y <=f x -> x = y.
Lemma cnf_leT x y z: x <=f y -> y <=f z -> x <=f z.
Lemma cnf_le_total x y: cnfp x -> cnfp y -> x <=f y \/ y <=f x.
Lemma cnf_lex0 x: x <=f \0f -> x = \0f.
Lemma cnf_le0x x : cnfp x -> \0f <=f x.
```

We show here

$$x \leq_f y \iff s(x) \leq_o s(y).$$

The proof is by transfinite induction on an upper bound of the degrees of x and y. Since the result is true when one argument is zero, we can ignore the fact that zero has no degree. We may assume $d(x) \leq d(y) = d$, and the result true for arguments of degree $< d$. Let c be the leading coefficient of y, y' its remainder, and denote by Z the quantity $s(z)$ so that $\mathrm{Y} = \omega^d \cdot c + \mathrm{Y}'$. If $d(x) < d$, then $x <_f y$ as well as $\mathrm{X} < \mathrm{Y}$ are clear. So assume x of degree d and and write $\mathrm{X} = \omega^d \cdot c' + \mathrm{X}'$. In case $c \neq c'$, then X and Y compare like c and c'. The same holds for x and y. So we may assume $c = c'$. Now $x <_f y$ is equivalrnt to $x' <_f y'$ [note that unless i is the degree of x we have $x_c(i) = x'_c(i)$] and $\mathrm{X} < \mathrm{Y}$ is equivalent to $\mathrm{X}' < \mathrm{Y}'$; the result follows by induction.

```
Lemma cnf_lt_deg x y: cnfp_nz  x -> cnfp_nz y ->
  cnf_degree x <o cnf_degree y ->  x <f y.
Lemma cnf_lt_eq_deg x y: cnfp_nz  x -> cnfp_nz y ->
  cnf_degree x = cnf_degree y -> cnf_lc x <c cnf_lc y ->  x <f y.

Lemma cnf_le_compat x y: cnfp x -> cnfp y ->
   (x <=f y <-> cnf_val x <=o cnf_val y).  (* 196 *)
```

We can define addition $x +_f y$ of two cnfs x and y such that

$$s(x +_f y) = s(x) +_o s(y). \tag{11.31}$$

Let $z = x +_f y$; if $y = 0$ then $z = x$; otherwise, let d be the degree of y, A monomial $m = (e, c)$ is in the range of z if either $e < d$ and m is in the range of y, or $e > d$ and m is in the range of x, or $e = d$, case where $c = x_c(e) + y_c(e)$. Note that $x_c(e)$ could be zero, but $y_c(e)$ is the leading coefficient of y. We obtain the sum by sorting the range.

```
Definition cnf_sum_monp x y d p :=
  [\/ inc p (range y) /\ P p <o d,
      inc p (range x) /\ d <o P p |
```

```
      p = J d ((Vr x d) +c (Vr y d))].
Definition cnf_sum_mons x y d :=
  Zo (range y) (fun p => P p <o d) \cup
  Zo (range x) (fun p => d <o P p) \cup
  singleton (J d  ((Vr x d) +c (Vr y d))).
Definition cnf_sum x y :=
  Yo (y = \0f) x (cnf_sort (cnf_sum_mons x y (cnf_degree y))).

Notation "x +f y" := (cnf_sum x y) (at level 50).

Definition cnf_sum_compat x y :=
  cnf_val (x +f y) = (cnf_val x) +o (cnf_val y).

Lemma cnfp_sum x y: cnfp x -> cnfp y -> cnfp (x +f y).
Lemma cnf_sum_monP x y d:
  (forall p, inc p (cnf_sum_mons x y d) <-> cnf_sum_monp x y d p).
Lemma cnf_sum_mon_range x y  (d := (cnf_degree y)): cnfp x -> cnfp_nz y ->
  cnf_rangep (cnf_sum_mons x y d).
Lemma cnf_sum_range x y: cnfp x -> cnfp_nz y ->
   range (x +f y) = cnf_sum_mons x y (cnf_degree y).
Lemma cnf_sum_rangeP x y:  cnfp x -> cnfp_nz y ->
  forall t, inc t (range (x +f y)) <->  cnf_sum_monp x y (cnf_degree y) t.
Lemma cnf_sum_nz x y: cnfp x -> cnfp_nz y ->
  x +f y <> \0f.
```

Assume x zero or of degree $< d$. Then $x_f y = y$. Obviously, the second case of the definition of the sum is excluded; in the third case, $x_c(e) = 0$, so that $c = y_c(e)$ and (e, c) is a monomial of y.

```
Lemma cnf_sum0l y: cnfp y -> \0f +f y = y.
Lemma cnf_sum0r x: x +f \0f  = x.
Lemma cnf_sum_small_deg x y: cnfp_nz  x ->  cnfp_nz y ->
  cnf_degree x <o cnf_degree y ->
  x +f y = y.
```

We state here some properties of the operation O_r, O_s and O_m introduced above. Let's write that $x \prec y$ if every exponent of x is smaller than every exponent of y. Note that $\mathrm{O}_r(x,k) \prec \mathrm{O}_s(x,k)$ because the exponents of x are strictly increasing. The condition $x \prec y$ holds when one argument is empty; otherwise, it can be restated as: the degree of x is less than the valuation of y. In this case $\mathrm{O}_m(x, y)$ is a cnf, and $\mathrm{O}_m(x, y) = y +_f x$.

In general, the range of $\mathrm{O}_m(x, y)$ is the union of the ranges of x and y. and $v(\mathrm{O}_m(x, y)) = v(y) +_o v(x)$. So, when $y \prec x$ then equation (11.31) holds.

```
Definition cnf_nr x k := restr x k.
Definition cnf_ns x k := Lg ((domain x) -c k) (cnf_s (Vg x) k).
Definition cnf_nm x y :=
  Lg ((domain x) +c (domain y)) (cnf_m (Vg x) (Vg y) (domain x)).
Definition cnf_all_smaller x y :=
  forall i j,  i <c domain x -> j <c domain y -> oexp x i <o oexp y j.

Lemma cnf_nr_degree x k (v := cnf_val (cnf_nr x k)):
  cnfp x -> k <=c domain x -> k <> \0c ->
  \0o <o v /\ odegree v = oexp x (cpred k).
Lemma cnfp_cnf_ns x k: cnfp x -> k <=c domain x -> cnfp (cnf_ns x k).
```

```
Lemma cnf_all_smaller_prop x y: cnfp x -> cnfp y ->
  [\/ x = emptyset, y = emptyset | (cnf_degree x) <o (oexp y \0c) ] ->
  cnf_all_smaller x y.
Lemma cnfp_cnf_nm x y: cnfp x -> cnfp y -> cnf_all_smaller x y ->
  cnfp (cnf_nm x y).
Lemma cnfp_cnf_nm_range x y: cnfp x -> cnfp y ->
  range (cnf_nm x y) = (range x) \cup (range y).
Lemma cnfp_cnf_mergeK x k :  cnfp x -> k <=c domain x ->
  cnf_nm (cnf_nr x k) (cnf_ns x k) = x.
Lemma cnf_nm_sum x y: cnfp x -> cnfp y ->
  cnf_all_smaller x y ->
  (cnf_nm x y) = y +f x.
Lemma cnf_sum_prop1 x y: cnfp x -> cnfp y ->
  cnf_val (cnf_nm x y) = (cnf_val y) +o (cnf_val x).
Lemma cnf_sum_prop2 x y: cnfp x -> cnfp y ->
  cnf_all_smaller x y ->
   cnf_val (cnf_sum y x) = (cnf_val y) +o (cnf_val x).
Lemma cnf_sum_prop3 x k :  cnfp x -> k <=c domain x ->
  cnf_val (cnf_ns x k) +o cnf_val(cnf_nr x k) = cnf_val x.
Lemma cnf_sum_rec x: cnfp_nz x ->
 cnf_val x = cnf_val (cnf_ns x \1c) +o (oopow (oexp x \0c)) *o (ocoef x \0c).
Lemma cnf_sum_prop4 x k
      (x1 := cnf_val (cnf_nr x k)) (x2 := cnf_val (cnf_ns x (csucc k)))
      (x3 := oopow (oexp x k) *o (ocoef x k)):
  cnfp x ->  k <c (domain x) ->
  [/\ ordinalp x1, ordinalp x2, ordinalp x3 &
  cnf_val x = x2 +o (x3 +o x1) ].
```

We consider now the case where the degree d of y is an exponent of x, say $d = e_i(x)$. Let $c = c_i(x)$ and c' be the leading coefficient of y. Let's split x in three parts: x_1 is the part with exponents $< d$, x_2 is the part with exponents $> d$, and x_3 is the part with exponents $= d$ (it is characterized by a single coefficient). Let x_4' be the complement of x_2, i.e., the part with exponents $\le d$. We can apply the decomposition to y and to $z = x +_f y$. Obviously y_2 is empty, and y_4 is y. By definition $z_1 = y_1$, $z_2 = x_2$ and the coefficient of z_3 is $c + c'$. We have $z_4 = \mathrm{O}_c(y, c)$ for some operation O_c and $x +_f y = \mathrm{O}_m(\mathrm{O}_c(y, c), \mathrm{O}_s(x, i + 1))$.

In the special case where x and y have the same degree, the result simplifies to $x +_f y = \mathrm{O}_c(y, c)$.

```
Definition cnf_nc y c :=
  Lg (domain y) (cnf_c (Vg y) (cnf_size y) (J (cnf_degree y) (c +c cnf_lc y))).
Definition cnf_ncms x y k :=
  cnf_nm (cnf_nc y (ocoef x k)) (cnf_ns x (csucc k)).

Lemma cnf_nc_prop y c (z :=  cnf_nc y c) :
   cnfp_nz y -> natp c ->
   [/\ cnfp z, domain z = domain y,
    forall i, i<c domain y -> oexp z i = oexp y i,
    forall i, i<c cnf_size y -> Vg z i = Vg y i &
    cnf_lc z = c +c cnf_lc  y].
Lemma cnf_ncms_prop x y k:
  cnfp x -> cnfp_nz y -> k <c (domain x) ->
  oexp x k = cnf_degree y ->
  x +f y = cnf_ncms x y   (* 00 *)

Lemma cnf_ncms_prop_sd x y :
```

```
  cnfp_nz x  -> cnfp_nz y ->
  cnf_degree x = cnf_degree y ->
  x +f y = cnf_nc y (cnf_lc x).
Lemma cnf_sd_lc x y :
  cnfp_nz x -> cnfp_nz y ->
  cnf_degree x = cnf_degree y ->
  cnf_lc(x +f y) = (cnf_lc x) +c cnf_lc y.
```

Same notations as above. If y has size $n+1$, then $y_1 = \mathrm{O}_r(y,n)$. The quantity $v(y_1)$ is also known as the remainder of y. We have $v(y) = \omega^d \cdot c' + v(y_1)$. We have $v(\mathrm{O}_c(y,c)) = \omega^d \cdot c + v(y)$. We have now $v(x) +_o v(y) = v(x_2) + m + v(x_1) + v(y)$ where n is a monomial of degree d. Note that $v(x_1)$ is zero or of degree $< d$ hence can be neglected before y. Now $m + v(y)$ is $v(\mathrm{O}_c(y,c))$? This shows equation (11.31).

```
Lemma cnf_sum_prop6 x c: cnfp_nz x -> natp c ->
   cnf_val (cnf_nc x c) = oopow (cnf_degree x) *o c +o (cnf_val x).
Lemma ord_negl_p9 x y k:
   cnfp x -> cnfp°nz y -> k <=c (domain x) ->
   (forall i, i <c k -> oexp x i <o cnf_degree y) ->
   cnf_val (cnf_nr x k) <<o cnf_val y.
Lemma cnf_sum_prop7  x y k:
  cnfp x -> cnfp_nz y -> k <c (domain x) ->
  oexp x k = cnf_degree y ->
  cnf_sum_compat x y.
```

Let x be a cnf with exponents e and domain n, and d an ordinal. There is k, given by some formula such that $k \le n$, whenever $i < k$ then $e_i(x) < d$, and, in the case $k < n$, then $d \le e_k(x)$. This means that we can split x in two parts: a first part with exponents $< d$, a second part with exponents $\ge d$. Let $x_s = \mathrm{O}_s(x,k)$ this second part.

Consider now a non-zero cnf y with degree d. We have $x +_f y = x_s +_f y$ because the monomials of x with exponent $< d$ are ignored in the sum, and these are exactly the x_i with $i < k$. Assume that d is not an exponent of x, so that every exponent of x_s is $> d$. By the previous result, $x +_f y = \mathrm{O}_m(y, \mathrm{O}_s(x,k))$. It follows that (11.31) holds in this case, hence in every case. We deduce that addition of cnfs is associative. Note that, if a and b are the CNF of x and y then $a +_f b$ is the CNF of $x +_o y$.

```
Definition position_in_cnf x d :=
  intersection ((Zo (domain x) (fun i => d <=o oexp x i)) +s1 (domain x)).

Lemma position_in_cnf_prop x d (k :=  position_in_cnf x d):
  cnfp x -> ordinalp d ->
  [/\ k <=c domain x,
      forall i, i <c k -> oexp x i <o d &
      k = domain x \/ d <=o oexp x k].
Lemma position_in_cnf_prop2 x i:
  cnfp x -> inc i (domain x) ->  position_in_cnf x (oexp x i) = i.
Lemma cnf_sum_prop8 x y (d := cnf_degree y) (k := position_in_cnf x d):
  cnfp x -> cnfp_nz y ->
  x +f y = cnf_sum (cnf_ns x k) y.  (* 57 *)
Lemma cnf_sum_prop9 x y (d := cnf_degree y) (k := position_in_cnf x d):
  cnfp x -> cnfp_nz y -> ~(inc d (domain (range x))) ->
  x +f y = cnf_nm y (cnf_ns x k).
Lemma cnf_sum_prop10 x y:  cnfp x -> cnfp_nz y  ->
```

```
  ~(inc (cnf_degree y) (domain (range x))) ->
  cnf_sum_compat x y.
Lemma cnf_sum_compat_prop x y: cnfp x -> cnfp y -> cnf_sum_compat x y.
Lemma cnf_osum x y: ordinalp x -> ordinalp y ->
  the_cnf (x +o y) = the_cnf x +f the_cnf y.
Lemma cnf_sumA x y z: cnfp x -> cnfp y -> cnfp z ->
   x +f (y +f z) = (x +f y) +f z.
```

We state: the degree of $x +_o y$ is the maximum of the degrees of x and y; whenever x and y are ordinals. This holds when one argument is zero. The result holds when x has smaller degree than y since the sum is then y. In the general case, we consider the CNF of x and y. In this case, the result is easy.

We deduce $x \ll y$ if and only if $d(x) < d(y)$. One implication is known. Assume $x + y = y$. Comparing degrees says $d(x) \le d(y)$. Assume $d(x) = d(y)$. We know $l(x + y) = l(x) + l(y) > l(y)$, where $l(x)$ is the leading coefficient of x.

We state: $v(x +_o y) = v(y)$ whenever y is a non-zero ordinal, and v is the valuation. We have shown above that we can write $y = z + \omega^e$, for some ordinals z and e. The formula becomes $v((x + z) + \omega^e) = v(y + \omega^e))$. It thus suffices to prove the result when $y = \omega^e$. If we take the CNF; we are reduced to compute the smallest exponent of $x +_f y$, in the case where y has a single monomial with exponent e. The result is obviously e.

```
Lemma odegree_sum a b: ordinalp a -> ordinalp b ->
  odegree (a +o b) = omax (odegree a) (odegree b).
Lemma ord_negl_p7_bis x y: \0o <o x -> \0o <o y ->
  (odegree x <o odegree y <-> x <<o y).

Lemma cnf_valuation_sum_spec x y: cnfp x -> cnfp y -> domain y = \1c ->
   oexp (x +f y) \0c = oexp y \0c.
Lemma ovaluation_4a x e c: ordinalp x -> ordinalp e -> posnatp c ->
    ovaluation (x +o (oopow e) *o c ) = e.
Lemma ovaluation4 y e: ordinalp y -> ordinalp e ->
  ovaluation (y +o oopow e) = e.

Lemma ovaluation_sum a b: ordinalp a -> \0o <o b ->
  ovaluation (a +o b) = ovaluation b.
```

11.12.5 Cantor normal form and operations

Let's consider α and its CNF (11.28) and β with its CNF

(11.32) $$\beta = \omega^{\kappa_1} \cdot \nu_1 + \omega^{\kappa_2} \cdot \nu_2 + \ldots + \omega^{\kappa_m} \cdot \nu_m.$$

The objective of this section is to compute the CNF of $\alpha+\beta$, $\alpha\cdot\beta$ and α^β. The value of the sum depends on how exponents of α compare with the degree κ_1 of β. We know

(11.33) $$\lambda_1 < \kappa_1 \implies \alpha + \beta = \beta.$$

We distinguish two cases; the case where the degree is an exponent or not. We get

(11.34)
$$\lambda_i > \kappa_1 > \lambda_{i+1} \implies \alpha + \beta = \omega^{\lambda_1} \cdot \mu_1 + \omega^{\lambda_2} \cdot \mu_2 + \ldots + \omega^{\lambda_i} \cdot \mu_i + \omega^{\kappa_1} \cdot \nu_1 + \omega^{\kappa_2} \cdot \nu_2 + \ldots + \omega^{\kappa_m} \cdot \nu_m.$$

(11.35)
$$\kappa_1 = \lambda_{i+1} \implies \alpha + \beta = \omega^{\lambda_1} \cdot \mu_1 + \ldots + \omega^{\lambda_i} \cdot \mu_i + \omega^{\kappa_1} \cdot (\mu_{i+1} + \nu_1) + \omega^{\kappa_2} \cdot \nu_2 + \ldots + \omega^{\kappa_m} \cdot \nu_m.$$

The relations follow from (11.31). For the first formula we apply `cnf_sum_prop9`; the non-trivial point is to show that the degree of y is not an exponent of x, and say that it is just before n.

```
Lemma CNF_sum_pr1 x y n: cnfp x -> cnfp_nz y -> n <c domain x ->
   cnf_degree y <o oexp x n ->
   (n = \0c \/  oexp x (cpred n) <o cnf_degree y) ->
   cnf_val x +o cnf_val y = cnf_val (cnf_nm y (cnf_ns  x n)).
Lemma CNF_sum_pr2 x y n: cnfp x -> cnfp_nz y ->
   n <c (domain x) -> oexp x n = cnf_degree y ->
   cnf_val x +o cnf_val y = cnf_val(cnf_ncms x y n).
```

If the two numbers have the same degree, so $\kappa_1 = \lambda_1$, then (11.35), simplifies. We deduce a formula for the product of α by an integer.. We start with an additional definition: we multiply the leading coefficient of α by an integer.

We consider here the special case of the sum of two ordinals with the same degree; this is $\kappa_1 = \lambda_1$ in (11.35). It can be used to compute $\alpha+\alpha$, and by induction $\alpha\cdot\nu$ for any integer ν.

$$\kappa_1 = \lambda_1 \implies \alpha+\beta = \omega^{\kappa_1}\cdot(\mu_1+\nu_1)+\omega^{\kappa_2}\cdot\nu_2+\ldots+\omega^{\kappa_m}\cdot\nu_m. \tag{11.36}$$

$$\alpha\cdot\nu = \omega^{\lambda_1}\cdot(\mu_1\nu)+\omega^{\lambda_2}\cdot\mu_2+\ldots+\omega^{\lambda_k}\cdot\mu_k. \tag{11.37}$$

```
Definition cnf_nck y k :=
  Lg (domain y) (cnf_c (Vg y) (cnf_size y) (J (cnf_degree y)
       ((cnf_lc y) *c k))).

Lemma cnf_nck_prop1 y k : natp k ->
   cnf_nck y (csucc k) = cnf_nc y ((cnf_lc y) *c k).
Lemma cnf_nck_prop2 x: cnfp_nz x -> nf_nck x \1c = x.
Lemma cnf_nck_prop3 x k: cnfp_nz_ne x ->
      cnfp (cnf_nck x (csucc k)).

Lemma CNF_sum_pr3 x y: cnfp_nz x -> cnfp_nz y ->
  cnf_degree x = cnf_degree y ->
  cnf_val x +o cnf_val y = cnf_val(cnf_nc y (cnf_lc x)).
Lemma CNF_prod_pr1 x k : cnfp_nz x -> posnatp k ->
  (cnf_val x) *o k = cnf_val (cnf_nck x k).
```

The CNF of $\omega^i\cdot x$ is obtained by increasing all exponents of the CNF of x by i. Conversely, we can factor out ω^i, provided that $i \le \nu(x)$; if $i = \nu(x)$, the remaining factor becomes a successor. So, if x is a non-zero ordinal, then $x = \omega^{\nu(x)}\cdot z$, where z is a successor. Such a factorisation is unique: assume $\omega^a\cdot b = \omega^c\cdot d$. If b is a successor, say $b = b'+1$, the LHS is $\omega^a\cdot b'+\omega^a$. This says that a is the valuation of the LHS. This implies $a = c$, and after simplification $b = d$.

```
Definition cnf_shift_expo x d :=
  Lg (domain x) (fun i =>  J (d +o (oexp x i)) (ocoef x i)).

Lemma cnfp_shift_expo x d: cnfp x -> ordinalp d -> cnfp(cnf_shift_expo x d).
Lemma CNF_prod_pr0 x d: cnfp x -> ordinalp d ->
  cnf_val (cnf_shift_expo x d) =  oopow d  *o  (cnf_val x).
Lemma ovaluation6 x: \0o <o x ->
  exists2 z, osuccp z & x = oopow (ovaluation x)  *o z.
Lemma ovaluation7 e x e' x' :
```

```
  ordinalp e -> ordinalp e' -> osuccp x -> osuccp x' ->
  oopow e  *o x = oopow e'  *o x' ->
  (e = e' /\ x = x').
```

If we take the supremum (11.37) over all integers ν we find that $\alpha\cdot\omega = \omega^{\lambda_1}\cdot\omega$, so that $\alpha\cdot\beta = \omega^{\lambda_1}\cdot\beta$ for any limit ordinal β, where λ_1 is the degree of α. We deduce: if $y = \omega^{\omega^n}$ then y is a critical point for the product, meaning that if $1 \le x < y$ then $x\cdot y = y$. The converse will be shown later on.

```
Lemma CNF_prod_pr2 x : cnfp_nz x ->
  cnf_val x *o omega0 = oopow (cnf_degree x) *o omega0.

Lemma CNF_prod_pr2bis x: \0o <o x->
  x *o omega0 = oopow (odegree x) *o omega0.
Lemma CNF_prod_pr3 x y: \0o <o x -> limit_ordinal y ->
  x *o y = oopow (odegree x) *o y.

Lemma oprod_crit_aux n x (y := oopow (oopow n)):
  ordinalp n ->  \1o <=o x ->  x <o y -> x *o y = y.
```

Let's write $\beta = \beta_1+\nu$, where β_1 is limit and ν integer. We have then $\alpha\cdot\beta = \alpha\cdot\beta_1+\alpha\cdot\nu$. These terms has the form $s(A)$ and $s(B)$, where all exponents of A are greater than the exponents of B. It follows that the sum is $s(C)$ where C is the merge of A and B.

Thus, we get, in the case where the least exponent is non-zero

$$\kappa_m \neq 0 \implies \alpha\cdot\beta = \omega^{\lambda_1}\cdot\beta = \omega^{\lambda_1+\kappa_1}\cdot\nu_1 + \omega^{\lambda_1+\kappa_2}\cdot\nu_2+\cdots+\omega^{\lambda_1+\kappa_m}\cdot\nu_m, \tag{11.38}$$

and otherwise

$$\kappa_m = 0 \implies \alpha\cdot\beta = \omega^{\lambda_1+\kappa_1}\cdot\nu_1+\ldots+\omega^{\lambda_1+\kappa_{m-1}}\cdot\nu_{m-1}+\omega^{\lambda_1}\cdot\mu_1\cdot\nu_m+\omega^{\lambda_2}\cdot\mu_2+\ldots+\omega^{\lambda_k}\cdot\mu_k. \tag{11.39}$$

In any case, the first exponent is $\lambda_1+\kappa_1$, so that the degree of a product is the sum of the degrees. If the valuation of β is zero (i.e., $\kappa_m = 0$) then $\alpha\cdot\beta$ and α have the same valuation; otherwise $v(\alpha\cdot\beta) = d(\alpha)+v(\beta)$.

```
Definition cnf_prod_gen x y :=
  cnf_nm (cnf_nck x (ocoef y \0c))
         (cnf_shift_expo (cnf_ns y \1c) (cnf_degree x)).

Lemma cnfp_prod_gen x y : cnfp_nz x -> cnfp_nz y -> cnfp( cnf_prod_gen x y).
Lemma CNF_prod_pr4 x y: cnfp_nz x -> cnfp_nz y ->  \0o <o oexp y \0c ->
  cnf_val x *o cnf_val y =  oopow (cnf_degree x) *o cnf_val y.
Lemma CNF_prod_pr5 x y: cnfp_nz x -> cnfp_nz y -> \0o <o oexp y \0c ->
  (cnf_val x) *o (cnf_val y)= cnf_val (cnf_shift_expo y (cnf_degree x)).
Lemma CNF_prod_pr6 x y: cnfp_nz x -> cnfp_nz y ->  oexp y \0c = \0o ->
  (cnf_val x) *o (cnf_val y) = cnf_val (cnf_prod_gen x y).
Lemma odegree_prod a b: \0o <o a -> \0o <o b ->
  odegree (a *o b) =  odegree a +o odegree b.
Lemma ovaluation_prod a b: \0o <o a -> \0o <o b ->
  ovaluation (a *o b) =
  Yo (ovaluation b = \0o) (ovaluation a) (odegree a +o ovaluation b).
```

Let's define the *leading coefficient* as the coefficient associated to the degree. Consider two non-zero ordinals x and y with leading coefficient a and b. Obviously, the leading coefficient of $x+y$ is a (respectively b or $a+b$) if the degree of ix s greater than (respectively less than or equal to) the degree of y. The case of a product is more interesting. If y is an integer, then yjre leading coefficient is ab. Otherwise it is b.

```
Definition oleading_coef x := let y := the_cnf x in ocoef y (cnf_size y).

Lemma oleading_coef_sum x y (a :=oleading_coef x) (b:=oleading_coef y):
  \0o <o x -> \0o <o y ->
  oleading_coef (x +o y) =
     Yo (odegree x <o odegree y) b (Yo (odegree y <o odegree x) a (a +c b)).
Lemma oleading_coef_prod1 x y: \0o <o x -> posnatp y ->
   oleading_coef (x *o y) =(oleading_coef x) *c y.
Lemma oleading_coef_prod2 a b: \0o <o a -> omega0 <=o b ->
  oleading_coef (a *o b) = oleading_coef b.
```

Let's compute α^β. If both arguments are finite, by induction, the value is finite, and equal to the cardinal power. Taking the supremum for $\beta < \omega$ yields $\alpha^\omega = \omega$.

$$2 \le \alpha < \omega \implies \alpha^\omega = \omega. \tag{11.40}$$

If $\lambda_k > 0$, we get $\alpha^2 = \omega^{\lambda_1} \cdot \alpha$ and by induction

$$\lambda_k \neq 0 \implies \alpha^{n+1} = \omega^{\lambda_1 \cdot n} \cdot \alpha. \tag{11.41}$$

Otherwise α is a successor. We can use the Cantor Product Form, defined below, and the formula for α^k becomes clear.; and there is no closed formula. so, we just show here the following formula, in case α is infinite.

$$n > 0, \lambda_1 > 0 \implies \alpha^n = \omega^{\lambda_1 \cdot n} \cdot \mu_1 + \dots \tag{11.42}$$

```
Lemma opow_int_omega n:
  \2o <=o n -> n <o omega0 -> n ^o omega0 = omega0.
Lemma CNF_pow_pr1 x k: cnfp_nz x -> \0o <o (oexp x \0c) -> natp k ->
  (cnf_val x) ^o (osucc k) =  (oopow ((cnf_degree x) *o k)) *o (cnf_val x).
Lemma CNF_pow_pr1_bis x k: limit_ordinal x -> natp k ->
  x ^o (osucc k) =  (oopow ((odegree x) *o k)) *o x.
Lemma CNF_pow_pr2 x k: omega0 <=o x ->  posnatp k ->
  odegree (x ^o k) = (odegree x) *o k  /\
  oleading_coef(x ^o k) = oleading_coef x.
```

Cantor deduces the relation (11.43) below, by taking the supremum for $n < \omega$. There is a simpler proof. We have $\omega^{\lambda_1} \le \alpha \le \omega^{\lambda_1+1}$ so that $\omega^{\lambda_1 \cdot \omega} \le \alpha^\omega \le \omega^{(\lambda_1+1)\cdot\omega}$. We have $(\lambda_1 + 1) \cdot \omega = \lambda_1 \cdot \omega$, so that we get

$$\lambda_1 > 0 \implies \alpha^\omega = \omega^{\lambda_1 \cdot \omega}. \tag{11.43}$$

If $\kappa_m \neq 0$ then $\beta = \omega \cdot \beta'$, for some β'; it follows

$$\kappa_m \neq 0, 2 \le n < \omega,\ \lambda_1 > 0 \implies \alpha^\beta = \omega^{\lambda_1 \cdot \beta}, \qquad n^\beta = \omega^{\beta'}. \tag{11.44}$$

If $\kappa_m = 0$, then $\beta = \beta_1 + \mu_m$, and $\alpha^\beta = \alpha^{\beta_1} \cdot \alpha^{\mu_m}$. For the first factor we can apply (11.43). For the second factor, we apply (11.42), unless α is finite. The exact expression is much too complicated, and we shall not give it.

```
Lemma CNF_pow_pr3 x: omega0 <=o x ->
  x ^o omega0  = oopow ((odegree x) *o omega0).
Lemma CNF_pow_pr4 x y: omega0 <=o x ->  limit_ordinal y ->
  x ^o y  =  oopow ((odegree x) *o y).
Lemma CNF_pow_pr5 x y:
  \2o <=o x -> x <o omega0 -> limit_ordinal y ->
  exists z,
   [/\ ordinalp z, y = omega0 *o z & x ^o y = oopow z].
```

Write $\beta = \omega \cdot z + n$, where n is finite. The degree of α^β is $d((\alpha^\omega)^z) \cdot d(\alpha^n)$. By induction $d(\alpha^n) = d(\alpha)^n$. So, if α is finite, and at least 2, then $d(\alpha^\beta) = z$; and if α is infinite (has non-zero degree) then $d(\alpha^\beta) = d(\alpha) \cdot \beta$.

```
Lemma CNF_pow_pr5_deg x y: \2o <=o x -> x <o omega0 -> ordinalp y ->
   odegree (x ^o y) = oquo y omega0.
Lemma CNF_pow_pr4_deg x y: omega0 <=o x -> ordinalp y ->
    odegree (x ^o y) = odegree x *o y.
```

11.12.6 The product form

Assume that e is a non-zero ordinal and c a non-zero integer. We have $c \cdot (\omega^e + 1) = \omega^e + c$ and

$$(\omega^c + 1) \cdot c = \omega^e \cdot c + 1. \tag{11.45}$$

The first relation is obvious (and will not be needed), the second is a consequence of (11.37).

```
Definition CNFp_value1 e c := osucc ((oopow e) *o c).
Definition CNFp_value2 e c := (osucc (oopow e)) *o c.

Lemma odegree_succ_pow n : \0o <o n -> odegree (osucc (oopow n)) = n.
Lemma CNFp_pr1 e c:
  \0o <o e -> \0o <o c -> c <o omega0 ->
  CNFp_value1 e c = CNFp_value2 e c.
```

The CNF of $\omega^e \cdot c + 1$ is trivial for non-zero e.

```
Lemma CNF_succ_pow1 n c (x := (osucc ((oopow n) *o c)))
      (X := variantLc (J \0o \1o) (J n c)):
  \0o <o n -> posnatp c-> cnf_and_val X x.
Lemma CNF_succ_pow n (x:= osucc (oopow n))
      (X := variantLc (J \0o \1o) (J n \1c)):
  \0o <o n ->  cnf_and_val X x.
```

We consider now a sequence of pairs (e_i, c_i) of exponents and coefficients. We assume $0 < c_i < \omega$ for $i \le n$ and $0 < e_i$ for $i < n$. The quantity e_n may be zero, other exponents are > 0. coefficients are non-ero integers. To these ordinals, we associate $\omega^{e_n} \cdot c_n \cdot p$, where p is the product of the $(\omega^{e_i} + 1) \cdot c_i$ for $i < n$.

```
Definition pmonomp m := [/\ pairp m, \0o <o P m &  posnatp (Q m)].
Definition CNFp_ax (p: fterm) n:=
```

```
   (forall i, i <c n -> pmonomp (p i)) /\ omonomp (p n).
Definition CNFp_ax1 (p: fterm) n:=
  forall i, i <c n -> pmonomp (p i).

Definition cantor_pmon (p: fterm) i :=  CNFp_value1 (P (p i)) (Q (p i)).

Definition CNFpv1 (p: fterm) n :=  oprodf (cantor_pmon p) n.

Definition CNFpv (p:fterm) n :=
  ((oopow (P (p n))) *o (Q (p n))) *o (CNFpv1 p n).
```

We start with trivial properties of the product p. In particular $(\omega^{e_{n-1}}+1)\cdot c_{n-1}$ is the rightmost factor.

```
Lemma OS_CNFp0 p n: CNFp_ax1 p n ->
   ord_below (cantor_pmon p) n.
Lemma OS_CNFp1 p n: CNFp_ax1 p n -> natp n -> ordinalp (CNFpv1 p n).
Lemma OS_CNFp1r p n m: CNFp_ax1 p n -> natp n -> m <=c n ->
   ordinalp (CNFpv1 p m).
Lemma OS_CNFp p n:  CNFp_ax p n -> natp n -> ordinalp (CNFpv p n).
Lemma CNFp_0 p: CNFpv1 p \0c = \1o.
Lemma CNFp_1 p n: CNFp_ax1 p n -> \0c <c n ->
  CNFpv1 p \1c = cantor_pmon p \0c.
Lemma CNFp_A p n m :
  natp n -> natp m -> CNFp_ax1 p (n +c m) ->
  CNFpv1 p (n +c m) = (CNFpv1 p n) *o
    CNFpv1 (fun z => p (z +c n)) m.
Lemma CNFp_r p n: natp n ->
  CNFpv1 p (csucc n) = CNFpv1 p n  *o (cantor_pmon p n).
```

Assume $k\geq 2$ in (11.28); we can write $\lambda_1=\lambda_2+p$ with $p>0$. Write $\alpha=\omega^{\lambda_2+p}\cdot c+r$. Then $r\cdot(\omega^p\cdot c+1)=(r\cdot\omega^p)\cdot c+r=\alpha$ according to (11.38). By induction,

$$\alpha=\omega^{\lambda_k}\cdot\mu_k\cdot(\omega^{\lambda_{k-1}-\lambda_k}\mu_{k-1}+1)\cdots(\omega^{\lambda_2-\lambda_3}\mu_2+1)\cdot(\omega^{\lambda_1-\lambda_2}\mu_1+1)\tag{11.46}$$

or

$$\alpha=\omega^{\lambda_k}\cdot\mu_k\cdot(\omega^{\lambda_{k-1}-\lambda_k}+1)\cdot\mu_{k-1}\cdots(\omega^{\lambda_2-\lambda_3}+1)\cdot\mu_2\cdot(\omega^{\lambda_1-\lambda_2}+1)\cdot\mu_1.\tag{11.47}$$

Renaming the exponents yields

$$\alpha=\omega^{\nu_k}\cdot\mu_k\cdot(\omega^{\nu_{k-1}}+1)\cdot\mu_{k-1}\cdots(\omega^{\nu_2}+1)\cdot\mu_2\cdot(\omega^{\nu_1}+1)\cdot\mu_1.\tag{11.48}$$

This form exists (if α is non-zero), and is unique provided all exponents are non-zero (with the possible exception of ν_k), since (11.28) and (11.48) are equivalent.

```
Lemma CNFp_p2 e c n (a:= e n -o (e (cpred n))) (b := c n):
   CNFb_axo e c (csucc n) -> natp n -> n <> \0c ->
   [/\ \0o <o a, posnatp b &
    CNFbvo e c (csucc n) = (CNFbvo e c n) *o (CNFp_value1 a b)].
Lemma CNFp_p3 e c n:  CNFb_axo e c (csucc n) -> natp n ->
  exists p,
   [/\ CNFp_ax p n, CNFbvo e c (csucc n) = CNFpv p n,
     (forall i, i <c n -> (p  i) = J (e (csucc i) -o (e i) )  (c (csucc i))) &
       p n = J (e \0c) (c \0c) ]. (* 53 *)
```

```
Lemma CNFp_exists x: \0o <o x ->
  exists p n, [/\ CNFp_ax p n, natp n & x = CNFpv p n].
Lemma CNFp_p4 p n:
   CNFp_ax p n -> natp n ->
   exists e c,
   [/\ CNFb_axo e c (csucc n),
       CNFbvo e c (csucc n) = CNFpv p n,
       forall i, i <c n -> p i = J (e (csucc i) -o (e i)) (c (csucc i)) &
       p n = J (e \0c)(c \0c)]. (* 52 *)
Lemma CNFp_unique p n p' n' :
   CNFp_ax p n  ->  CNFp_ax p' n' -> natp n -> natp n' ->
   CNFpv p n = CNFpv p' n' ->
   n = n' /\  same_below p p' (csucc n).
```

We now study the question when (S): $\alpha+\beta=\beta+\alpha$ or (P): $\alpha\cdot\beta=\beta\cdot\alpha$. Let ($S_1$) (respectively ($P_1$)) be the assumption that there exist an ordinal γ and two integers n and m (i.e., two finite ordinals) such that $\alpha=\gamma\cdot n$, $\beta=\gamma\cdot m$ (respectively: $\alpha=\gamma^n$ and $\beta=\gamma^m$). Then (S_1) implies (S) and (P_1) implies (P). Condition (P_1) has been studied by Cantor [7, §19K].

Conversely, assume (S). Consider the CNF of these numbers. If $\lambda_1<\kappa_1$, then $\alpha+\beta=\beta$. It follows $\beta=\beta+\alpha$, which implies $\alpha=0$. In this case, we can chose $\gamma=\beta$, $n=0$ and $m=1$. In the case $\lambda_1=\kappa_1$ relation (11.36) shows that the expansions are the same, with the possible exception of the leading coefficient. In other terms, $\alpha=\omega^k\cdot n+r$ and $\beta=\omega^k\cdot m+r$. Let $\gamma=\omega^k+r$; relation (11.37) says $\gamma\cdot n=a$ and $\gamma\cdot m=b$.

```
Lemma osum2_commutes a b: (* 66 *)
  ordinalp a -> ordinalp b ->
 ((a +o b = b +o a) <-> (exists c n m,
  [/\ ordinalp c, natp n, natp m, a = c *o n & b = c *o m])). (* 85 *)
```

The case of a product is a bit more complicated. There are two obvious sufficient conditions: (P_2) that says that one factor is zero, and (P_3) that says both arguments are integers. There are some other cases, for instance $\alpha=\omega^2+\omega$ and $\beta=\omega\cdot\alpha$ satisfy (P). This pair does not satisfy (P_1) for, if $\alpha=\gamma^n$, then either $n=1$, γ is of degree two, and β is of degree $3=2m$, absurd, or $n=2$ and α is the square of an ordinal of degree one, which is equally absurd, as $(\omega+c)^2=\omega^2+\omega\cdot c+c$.

Let (P_4) be the assumption that the pair (α,β) is such that α is a limit ordinal (i.e., $\lambda_k\neq 0$), there exists an ordinal γ, two integers n and m such that the degree of α is $\gamma\cdot n$, while $\beta=\omega^{\gamma\cdot m}\cdot\alpha$. The example above satisfies this condition.

Assume (P_4) holds. The pair $(\omega^{\gamma\cdot n},\omega^{\gamma\cdot m})$ satisfies (P), and, by (11.38), it follows that (P) holds for (α,β). Conversely, assume that α and β are limit ordinals satisfying (P) so that we can apply (11.38) twice. We get $k=m$, $\nu_i=\mu_i$ and $\lambda_1+\kappa_i=\kappa_1+\lambda_i$. Taking $i=1$ shows that the degrees of α and β satisfy (S). Write $\lambda_1=\gamma\cdot n$ and $\mu_1=\gamma\cdot m$. This shows that the pair (α,β) satisfies (P_4).

Assume $\kappa_m\neq 0$ and $\lambda_k=0$. We may consider (11.38) and (11.39). Counting the number of terms yields $m=1$. This means that β is an integer. Looking at the coefficients, we see that $\beta=1$, so that (P_1) holds with $\gamma=\alpha$, $n=1$ and $m=0$.

Assume finally that α and β are successors. If they are finite, they commute. Assume β finite; so that $\alpha\cdot\beta$ is given by (11.37). Using (11.39) for $\beta\cdot\alpha$ and considering trailing coefficients shows $\beta=1$.

```
Definition oprod2_comm_P4 x y :=
```

```
  exists z gamma c1 c2,
  [/\ cnfp_nz z, \0o <ooexp z \0c, ordinalp gamma &
  [/\ natp c1, natp c2,
  x = cnf_val z,
  cnf_degree z = gamma *o c1 &
  y = oopow (gamma *o c2) *o x]].

Lemma oprod2_comm1  a b (x := cnf_val a) (y:= cnf_val b):
  cnfp_nz a -> cnfp_nz b ->
  oexp b \0c = \0o -> \0o <o oexp a \0c ->
  oprod_comm x y -> y = \1o.
Lemma oprod2_comm2 a mu
  (x:= cnf_val a) (y := (oopow mu) *o x):
  cnfp_nz a -> \0o <o oexp a \0c -> ordinalp mu ->
  (cnf_degree a) +o mu = mu +o (cnf_degree a) -> oprod_comm x y.
Lemma oprod2_comm3 x y:  oprod2_comm_P4 x y ->  oprod_comm x y.
Lemma oprod2_comm4  a b
  (x := cnf_val a) (y:= cnf_val b):
  cnfp_nz a -> cnfp_nz b ->
  \0o <o (oexp a \0c) -> \0o <o oexp b \0c ->
  oprod_comm x y ->
  (oprod2_comm_P4 x y \/ oprod2_comm_P4 y x).
Lemma oprod2_comm5 a b
  (x := cnf_val a) (y := cnf_val b):
  cnfp_nz a -> cnfp_nz b ->
  oexp a \0c = \0o -> oexp b \0c = \0o -> cnf_size a = \0c -> oprod_comm x y ->
  (x = \1o \/ (x <o omega0 /\ y <o omega0)).
```

Assume now our numbers infinite and successors. We can apply (11.39), and use uniqueness of the CNF. This yields $m+k-1$ pairs of equations. Let $p = m+k-2$. This is the number of non-zero exponents, and the number of equations concerning non-zero exponents. Concerning coefficients, we have $p+1$ equations and $p+2$ unknowns. Thus, there is generically one free parameters, a coefficient. Consider for example $m = 5$ and $k = 3$. Solving the equations is trivial; there are 3 redundant equations for coefficients and 2 for the exponents. In fact, any β is a solution, and the equations are equivalent to $\alpha = \beta^2$. The case $m = 6$ and $k = 3$ is a bit more difficult. If $c = \lambda_5$ and $d = \lambda_4$, then one equation is $d+d+d = d+d+c+c$. After simplification, this gives $d = c+c$. In this case, we have 3 free parameters, two coefficients a and b, and one exponent c. The ordinals α and β satisfy the following properties: the leading coefficient is a, the trailing coefficient is b, other coefficients are ab. The i-th exponent (in increasing order, starting with zero) is $c\cdot i$. Let $\gamma = \omega^c \cdot a + b$. It is easy to check that $\beta = \gamma^2$, and one could verify that $\alpha = \gamma^5$.

In the general case we shall use the product form (11.47). The important property we use here is that $\lambda_k = 0$. Thus, we may write, with new notations:

$$\alpha = \mu_{k+1}\cdot(\omega^{\lambda_k}+1)\cdot\mu_k\cdots(\omega^{\lambda_2}+1)\cdot\mu_2\cdot(\omega^{\lambda_1}+1)\cdot\mu_1.$$

$$\beta = \nu_{m+1}\cdot(\omega^{\kappa_m}+1)\cdot\nu_m\cdots(\omega^{\kappa_2}+1)\cdot\nu_2\cdot(\omega^{\kappa_1}+1)\cdot\nu_1.$$

The product of two such products is such a product. The exponents are the exponents of α or β; and the coefficients are the coefficients of α or β, with an exception: there is $\mu_1\cdot\nu_{m+1}$ or $\nu_1\cdot\mu_{k+1}$. Since k and m are non-zero, it follows that $\mu_1 = \nu_1$. We also have $\lambda_1 = \kappa_1$, $\mu_2 = \nu_2$, etc. In fact, if $k = m$, we get $\alpha = \beta$. More generally, if $k > m$, there exists γ that has the same form, such that $\alpha = \gamma\cdot\beta$. Moreover, $\beta\cdot\gamma = \gamma\cdot\beta$. The proof is straightforward, the relation (P_1) follows by induction on $k+m$.

```
Definition oprod2_comm_P1 x y :=
  exists c n m,
    [/\ ordinalp c, natp n, natp m, x = c ^o n & y = c ^o m].
Definition CNFp_ax4 (p:fterm) n x :=
  [/\ CNFp_ax p (csucc n), natp n, (P (p (csucc n))) = \0c &
    x = (Q (p (csucc n))) *o CNFpv1 p (csucc n)].
Definition CNF_npec (px py: fterm) n m :=
  fun i => Yo (i = (csucc n +c m)) (px (csucc n))
              (Yo (i = n) (J (P (px n)) ((Q (px n) *o (Q (py m)))))
                  (Yo (i <c n) (px i) (py (i -c (csucc n))))).

Lemma CNFp_pg  px n py m
  (pz := CNF_npec px py n m)
  (lx := Q (px (csucc n))) (ly := Q (py m)):
  CNFp_ax px (csucc n) -> CNFp_ax py m -> natp n -> natp m ->
  (CNFp_ax pz (csucc n +c m) /\
  (lx *o (CNFpv1 px (csucc n))) *o (ly *o (CNFpv1 py m))
   = lx *o CNFpv1 pz (csucc n +c m)). (* 66 *)
Lemma CNFp_ph px n py m  x y
  (pz1 := CNF_npec px py n (csucc m)) (pz2 := CNF_npec py px m (csucc n)):
  CNFp_ax4  px n x  -> CNFp_ax4 py m y ->
  oprod_comm x y ->
   [/\ (same_below pz1 pz2 (csucc n +c csucc m)),
    ( n = m -> x = y) &
    ( m <c n -> exists pz p z,
       [/\ CNFp_ax4 pz p z, x = z *o y, z *o y = y *o z & p <c n])]. (* 119 *)
Lemma oprod2_comm6 px n x py m y:
  CNFp_ax4 px n x -> CNFp_ax4 py  m y ->
  oprod_comm x y -> oprod2_comm_P1 x y.
```

Thus (P) is equivalent to (P_1) or (P_2) or (P_3) or (P_4).

```
Theorem oprod2_comm x y: ordinalp x -> ordinalp y ->
   ((oprod_comm x y) <->
   (x = \0o \/ y = \0o \/ (oprod2_comm_P4 x y \/  oprod2_comm_P4 y x) \/
   (finite_o x /\ finite_o y) \/ oprod2_comm_P1 x y)).  (* 74 *)
```

11.12.7 Factorisation into prime numbers

We shall show in this section that an ordinal number can be uniquely factored into prime numbers. The key relation is (11.48).

We first solve the equation $\alpha\cdot\beta = \beta$. (note that $\alpha\cdot\beta = \alpha$ says $\alpha = 0$ or $\beta = 1$ so has only trivial solutions). By considering degrees, we find that $d(\alpha) \ll d(\beta)$. This condition is of course not sufficient.

Assume that β is a successor. Consider equation (11.39); by counting the number of terms, we get $k = 1$, then $\lambda_1 = \kappa_m = 0$ and $\mu_1\nu_m = \nu_m$, so $\mu_1 = 1$. This means $\alpha = 1$, there is only the trivial solution. Assume β limit, let d be the degree of α, and v the valuation of β. We have $\beta = \omega^v \cdot z$ for some successor z. Moreover $\alpha\cdot\beta = \omega^d\cdot\beta = \omega^{d+v}\cdot z$. By uniqueness of the factorisation of β as a power of ω and a successor, the equation is equivalent to $d + v = v$, hence to $d \ll v$.

```
Lemma oprod_neg_p1 x y: ordinalp x -> ordinalp y ->
   x *o y = y -> osuccp y -> x = \1o.
```

```
Lemma oprod_neg_p2 x y : \0o <o x -> limit_ordinal y ->
   (x *o y = y  <-> odegree x <<o ovaluation y).
```

We say that x is a *strong prime* if $x = a \cdot b$ implies that at least one factor is equal to 1; we say that it is a *prime* when at least one factor is x. A strong prime is prime. The example of $2 \cdot \omega = \omega$ and $\omega \cdot \omega^{\omega} = \omega^{\omega}$. shows that some prime are not strong.

```
Definition ord_sprime a :=
  [/\ ordinalp a, \1o <o a &
     forall b c, ordinalp b -> ordinalp c -> a = b *o c -> b = \1o \/ c = \1o].
Definition ord_prime a :=
  [/\ ordinalp a, \1o <o a &
     forall b c, ordinalp b -> ordinalp c -> a = b *o c -> a = b \/ a = c].
Lemma ord_prime_prop1 a: ord_sprime a -> ord_prime a.
```

As noted by Cantor [7, §19G], $\omega^p + 1$ is a strong prime. Assume $\alpha \cdot \beta = \omega^p + 1$. One could use the CNF and the two formulas (11.38) and (11.39). Another approach is the following: we have already noted that if a product is a successor, so are both factors. So, let's assume $(a+1) \cdot (b+1) = \omega^p + 1$. After simplification we get $(a+1) \cdot b + a = \omega^p$. This has the form $x + a = \omega^n$; since ω^n is indecomposable we get $x = \omega^n$ or $a = \omega^n$. In the first case, we also have $a = 0$; in the second case, if b were non-zero, we would have $a < x \le x + a = a$, absurd.

```
Lemma succ_pow_omega_irred p: ordinalp p -> ord_sprime (osucc (oopow p)).
Lemma succ_pow_omega_prime p: ordinalp p -> ord_prime (osucc (oopow p)).
```

Let's consider $\alpha \cdot \beta = \omega^n$. Consider first the case where β is a successor. This says that the valuation of α is n, so that $\alpha = \omega^n \cdot z$, for some z. It follows $z = \beta = 1$. Assume now β limit. Let p and q be the degrees of α and β. We have $p + q = n$ and $\alpha \cdot \beta = \omega^p \cdot \beta = \omega^{p+q}$. After simplification $\beta = \omega^q$.

Obviously, ω^n prime says that n is indecomposable. Conversely, n indecomposable and $p + q = n$ says $p = n$ or $q = n$. If $q = n$, then $\beta = \omega^n$. From $\omega^p \le \alpha \le \omega^n$, if $p = n$, then $\alpha = \omega^n$. So ω^n is prime.

```
Lemma ord_prime_prop2 a b c: \0o <o a -> \0o <o b -> osuccp b -> ordinalp c ->
   a *o b = oopow c -> a =  oopow  c /\ b = \1o.
Lemma ord_prime_prop3 a b c (d := odegree a):
  \0o <o a -> limit_ordinal b -> ordinalp c ->
  a *o b = oopow c -> d <=o c /\ b = oopow (c -o d).
Lemma ord_prime_prop4 n: ordinalp n ->  ord_prime (oopow n) ->
  indecomposable n.
Lemma ord_prime_prop5 n: indecomposable n ->  ord_prime (oopow n).
Lemma ord_prime_prop5' n: ordinalp n -> ord_prime (oopow (oopow n)).
```

Let's say that a natural number is prime if it has no trivial factorisation on **N**. In such a case it is a strong prime ordinal (if $a = b \cdot c$ and a is finite, so are both factors). Conversely, a finite prime ordinal is a natural prime. We say that a is composite if there is b such that $1 < b < a$ and b divides a. A composite is a non-prime. Every number p has a prime factor (if p is non-prime, it is composite, has a non-trivial fact, by induction this factor has a prime factor).

```
Definition nat_prime a :=
  [/\ natp a, \1c <c a &
     forall b c, natp  b -> natp c -> a = b *c c -> b = \1c \/ c = \1c].

Lemma nat_prime_p1 a : nat_prime a -> ord_prime a.
Lemma nat_prime_P a : nat_prime a <-> ord_prime a /\ a <o omega0.
Lemma nat_prime_p2 a: natp a -> \1c <c a ->
   (nat_prime a <-> ~ composite a).
Lemma nat_prime_p3 a: natp a -> \1c <c a ->
  exists2 p, nat_prime p & p %|c a.
```

The existence of a factorisation follows. We consider here a decreasing list p_i and the product $P = \prod p_i$. We shall consider the ordinal product (so that the largest factor comes first). Every p_i divides the product P. Let n be an integer. If it is prime or equal to 1, the factorisation is trivial. Otherwise n has a non-trivial factor, hence a least prime factor a. Write $n = ab$, and (by induction) consider the factorisation $b = \prod p_i$. Since p_i divides n; we have $a \le p_i$. So we obtain a factorisation of n by adding a as least factor.

If a prime q divides a product of prime $\prod p_i$, it is one of them. In effect, either $q = p_0$ or is coprime with p_0. In the second case, it divides the product of all p_i (with $i > 0$), hence is equal to one of them by induction. We deduce uniqueness: assume $\prod p_i = \prod q_i$; the smallest of the p_i is one of the q_j and vice-versa; so the smallest of the p_i is the smallest of the q_j, the result follows by induction.

```
Definition nat_factor_list (p: fterm) n :=
  [/\ ord_below p n, forall i, i<c n -> nat_prime (p i) &
    forall i, natp i -> csucc i <c n -> (p (csucc i)) <=o p i].

Lemma nat_factor_list_rec p n: natp n ->
  nat_factor_list p (csucc n) -> nat_factor_list p n.
Lemma NS_nat_prime_factor p n: natp n -> (nat_factor_list p n) ->
  natp (oprodf p n).
Lemma nat_prime_factor_vS p n: natp n -> nat_factor_list p (csucc n) ->
  [/\ natp  (oprodf p n), natp (p n) &
  oprodf p (csucc n) = (oprodf p n) *c (p n)].
Lemma nat_prime_p4 p n i: natp n -> (nat_factor_list p n) -> i <c n->
  p i %|c (oprodf p n).
Lemma nat_prime_coprime a b: nat_prime a -> nat_prime b ->
   a = b \/ coprime a b.

Lemma nat_prime_p5 p n q: natp n -> (nat_factor_list p n) ->
   nat_prime q -> q %|c (oprodf p n) ->
   exists2 i, i <c n & q = p i.
Lemma nat_prime_p6 p p' n n': natp n -> natp n' ->
   nat_factor_list p n  ->  nat_factor_list p' n' ->
   oprodf p n = oprodf p' n' ->
   n = n' /\ same_below p p' n.
Lemma nat_prime_p7 a: natp a -> \1c <=c a -> exists p n,
   [/\ nat_factor_list p n, natp n & oprodf p n = a ].
```

We pretend that a prime ordinal is either (F), a natural prime number (hence a finite successor), or (L) a power of a power of ω (hence an infinite limit ordinal), or (I) the successor of a power of ω (hence an infinite successor). Note that $\omega^0 + 1$ is equal to 2, hence of the form (F). Obviously, an ordinal belongs to at most one category. These numbers have been proved prime. There is no other prime: Let α be a prime ordinal, and consider equation (11.48). Now α is one of the factors. If it is a μ_i, it is a (F), if it is a $\omega^\nu+1$, it is (I), otherwise it is (L).

```
Definition ord_ptypeF a := nat_prime a.
Definition ord_ptypeI a := exists2 p, \0o <o p & a = osucc (oopow p).
Definition ord_ptypeL a := exists2 p, ordinalp p & a = oopow (oopow p).

Lemma ord_ptypeF_prop a: ord_ptypeF a -> osuccp a /\ a <o omega0.
Lemma ord_ptypeI_prop a: ord_ptypeI a -> osuccp a /\ omega0 <=o a.
Lemma ord_ptypeL_prop a: ord_ptypeL a -> limit_ordinal a /\ omega0 <=o a.
Lemma ord_prime_p1 a:
  ord_prime a <-> [\/ ord_ptypeF a,  ord_ptypeI a | ord_ptypeL a].
```

We consider now a list formed of elements of type (F), (L) or (I), subject to the following conditions; if a is followed by b then: If b has type (L), then a has type (L) and $a \le b$, if a and b have type F then $b \le a$. We call this an ordinal factor list, in short OFL. We pretend that every non-zero ordinal is uniquely the product of an OFL.

Note that in an OFL, limit ordinals come before successors. So there is a unique boundary k, such that if our list L is L_1 followed by L_2, then both lists are OFLs, the first has only limit ordinals, the second has only successor ordinals. Every element of L_1 has the form ω^{ω^e}, and e is the degree of the degree of the element. Let L_3 be the list of these degrees in reverse order. Let p_1 be the product of the elements of L_1, and s the sum of elements of L_3. Because of our conventions on the order of terms in a sum or a product, we get $p_1 = \omega^s$. Moreover, the elements of L_3 are in increasing order. Hence $p_1 = p'_1$ implies $L_3 = L'_3$, hence $L_1 = L'_1$.

Let p and p_2 be the products of L and L_2. Then $p = p_1 \cdot p_2$, and p_2 is a successor.. Assume that we have a second OFL, with the same product; hence $p = p'_1 \cdot p'_2$. By uniqueness of the factorisation of an ordinal as a power of a prime and a successor, we get $p_1 = p'_1$ and $p_2 = p'_2$. it follows $L_1 = L'_1$.

```
Definition ord_prime_le a b:=
  (a <o omega0 -> b <o omega0 -> b <=o a)
  /\ (forall p, ordinalp p -> b = oopow (oopow p) ->
     exists2 q, p <=o q & a = oopow (oopow q)).
Definition ord_factor_list (p: fterm) n :=
  [/\ natp n, (forall i, i<c n ->ord_prime (p i)) &
   forall i, natp i -> csucc i <c n -> ord_prime_le (p i) (p (csucc i))].

Lemma ord_factor1 p n: ord_factor_list p n -> ord_below p n.
Lemma ord_factor2 p n:  ord_factor_list p n ->
   exists k,  ord_factor_boundary p n k.
Lemma ord_factor3 p n k k':  ord_factor_list p n ->
  ord_factor_boundary p n k -> ord_factor_boundary p n k' ->
  k = k'.

Lemma ord_factor4 p n
      (q := fun i => odegree (odegree (p i)))
      (r := fun i => q (cpred (n -c i))):
  ord_factor_list p n ->
  (forall i, i <c n -> limit_ordinal (p i)) ->
 [/\ forall i, i <c n -> p i = oopow (oopow (q i)),
     CNFr_ax r n & oprodf p n = oopow (CNFrv r n)].
Lemma ord_factor5 p n p' n':
  ord_factor_list p n ->
  ord_factor_list p' n' ->
```

```
  (forall i, i <c n -> limit_ordinal (p i)) ->
  (forall i, i <c n' -> limit_ordinal (p' i)) ->
  oprodf p n =  oprodf p' n' ->
  (n = n' /\ same_below p p' n).  (* 58 *)

Lemma ord_factor6 p n k (p':= fun i => p (i +c k)):
  ord_factor_list p n ->
  ord_factor_boundary p n k ->
  [/\ ord_factor_list p k, ord_factor_list p' (n -c k),
     forall i, i<c k -> limit_ordinal (p i),
     forall i, i<c (n -c k) -> osuccp (p' i) &
     [/\ oprodf p n = (oprodf p k) *o (oprodf p' (n -c k)),
      exists2 e, ordinalp e & oprodf p k = oopow e &
      osuccp (oprodf p' (n -c k)) ] ].
Lemma ord_factor7 p n p' n' k k' (p1:= fun i => p (i +c k))
      (p1':= fun i => p' (i +c k')):
  ord_factor_list p n ->
  ord_factor_list p' n' ->
  ord_factor_boundary p n k ->   ord_factor_boundary p' n' k' ->
  oprodf p n = oprodf p' n' ->
  [/\ k = k', same_below p p' k,
    forall i, i <c n -c k -> osuccp (p1 i),
    forall i, i <c n' -c k -> osuccp (p1' i) &
    oprodf p1 (n -c k) = oprodf p1' (n' -c k)].
```

In order to prove uniqueness of the factorisation it suffices to consider the case where all factors are successors. We consider in this description a list L; in the code, this is given by the enumeration function and the length.

Every element in L is an integer or has the form ω^e+1 where e is the degree of this term. We denote by L_- the list L without its first, by $p(L)$ the product of the elements of L, by $i(L)$ the index of the first non-integer in L, by F(L) the product of these integers, and by $r(L)$ what remains in L after these integers. Assume L starts with a non-integer, say ω^e+1. Then $p(L) = (\omega^e+1)\cdot F(L_-)\cdot p(r(L_-))$. By induction there are integers c_i and exponents e_I, such that if $P(e,c) = \prod(\omega^{e_i}+1)\cdot c_i$ then $p(L) = P(e,c)$. The first exponent e_0 is the degree of the first element of L. In the general case, $p(L) = F(L)\cdot p(r(L))$, so $p(L) = n_0\cdot P(e,c)$, where $e_0 = F(L)$. In both cases $p(L)$ has the form (11.48), where the initial power of ω is trivial.

Assume now $p(L) = p(L')$. Let's show $L = L'$ by induction on the length of L. If one list is empty, then the product is 1, if it is non-empty, then the product > 1. So, we may assume L and L' non-empty. By uniqueness of (11.48), the initial factors n_0 are the same. This means $F(L) = F(L')$. Uniqueness of the factorisation of integers now says $i(L) = i(L')$ and that the first $i(L)$ elements are the same. If $i(L)$ is non-zero, the result follows by induction. Otherwise L and L' start with a non-integer. Write $p(L) = P(e,c)$. Uniqueness of (11.48) says that the e_0 are the same, so that L and L' sart with the same element; the result follows by induction.

```
Definition ofact_list_succ (p: fterm) n :=
   ord_factor_list p n /\ forall i, i <c n -> osuccp (p i).
Definition first_non_int (p: fterm) n :=
  intersection (Zo (csucc n) (fun z => z = n \/ ~ (natp (p z)))).

Lemma ord_factor0_aux k p: natp k ->
   (forall j, j <c k -> natp (p j)) ->
   (forall j, j <c k -> osuccp (p j)) ->
  posnatp (oprodf p k).
```

```
Lemma first_non_int_p1 p n (i := first_non_int p n):
  natp n ->
  (i = n \/ (i <c n /\ ~ natp (p i)))
  /\ forall j, j <c i -> natp (p j).
Lemma ord_factor8 p n i: ofact_list_succ p n -> i <c n ->
  nat_prime (p i)  \/
  [/\ ordinalp (p i), ~(natp (p i))&
       p i = osucc (oopow (odegree (p i)))].
Lemma ord_factor9 p n: ofact_list_succ p (csucc n) -> natp n ->
   ~ (natp (p \0c)) ->
   exists ec m, [/\ natp m, CNFp_ax1 ec (csucc m),
                  oprodf p (csucc n) = CNFpv1 ec (csucc m),
        let p' := p \o csucc in  Q(ec \0c) = oprodf p' (first_non_int p' n)&
        P(ec \0c) = odegree (p \0c)]. (* 116 *)
Lemma ord_factor10 p n: ofact_list_succ p (csucc n) -> natp n ->
   ~ (natp (p \0c)) ->
   exists ec m, [/\ natp m, CNFp_ax ec (csucc m),
                  oprodf p (csucc n) = CNFpv ec (csucc m) &
    [/\ ec (csucc m)  = J \0o \1o,
     let p' := p \o csucc in  Q(ec \0c) =  oprodf p' (first_non_int p' n) &
     P(ec \0c) = odegree (p \0c) ]].
Lemma ord_factor11 p n: ofact_list_succ p n -> natp n ->
   exists ec m, [/\ natp m, CNFp_ax ec m,
                  oprodf p n = CNFpv ec m &
    ec m  = J \0o (oprodf p (first_non_int p n)) ].
Lemma ord_factor12 p n p' n':
  ofact_list_succ p n ->
  ofact_list_succ p' n' ->
  oprodf p n = oprodf p' n' ->
  (n = n' /\ same_below p p' n). (* 144 *)
```

Uniqueness follows. Existence is tedious but straightforward.

```
Lemma ord_factor_unique p n p' n':
  ord_factor_list p n ->
  ord_factor_list p' n' ->
  oprodf p n = oprodf p' n' ->
  (n = n' /\ same_below p p' n).
Lemma ord_factor_exits x: \0o <o x ->
  exists p n, [/\ ord_factor_list p n, natp n & x = oprodf p n]. (* 265 *)
```

11.13 An exercise of Bourbaki

The objective of this section is to solve Exercise 6.12, part (b), page 656

(b) For each integer n let $f(n) \le n!$ be the greatest number of elements in the set of ordinals of the form $\alpha_{\sigma(1)}+\alpha_{\sigma(2)}+\cdots+\alpha_{\sigma(n)}$, where $(\alpha_i)_{1\le i\le n}$ is an arbitrary sequence of n ordinals and σ runs through the set of permutations of the interval $[1,n]$. Show that

$$f(n) = \sup_{1\le k\le n-1} (k.2^{k-1}+1)f(n-k). \tag{1}$$

(Consider first the case where all the $\phi(\alpha_i)$ are equal and show that the largest possible number of distinct ordinals of the desired form is equal to n, by using Exercise 16 (a) of § 2. Then use induction on the number of ordinals α_i for which $\phi(\alpha_i)$ takes the least possible value among the set of ordinals $\phi(\alpha_j)$ $(1 \le j \le n)$.) Deduce from (1) that for $n \ge 20$ we have $f(n) = 81 f(n-5)$.

The idea consists first in solving equation (1) unsing Coq integers, then solve a variant of the problem.

11.13.1 Study of the equation

Let's introduce

$$\mathrm{C}_n = 1 + n2^{n-1} \tag{11.49}$$

and notice that the first values are $\mathrm{C}_0 = 1$, $\mathrm{C}_1 = 2$, $\mathrm{C}_2 = 5$, $\mathrm{C}_3 = 13$, $\mathrm{C}_4 = 33$, $\mathrm{C}_5 = 81$, $\mathrm{C}_6 = 193$ and $\mathrm{C}_7 = 449$. We are interested in the following fixed point equation

$$f(n) = \sup_{0<k<n} \mathrm{C}_k f(n-k) \quad (n \ge 2); \qquad f(0) = f(1) = 1. \tag{11.50}$$

Note that the supremum is taken over a non-empty finite set, so that $f(n)$ is the greatest among $\mathrm{C}_1 f(n-1), \dots, \mathrm{C}_{n-1} f(1)$. In particular $f(2) = \mathrm{C}_1 f(1)$, and for $n \ge 2$, $f(n)$ is independent of $f(0)$. An equivalent version of the equation is the following, where $n \ge 2$, and $0 < k < n$:

$$\forall n, \exists k, f(n) = \mathrm{C}_k f(n-k); \qquad \forall nk, f(n) \ge \mathrm{C}_k f(n-k). \tag{11.51}$$

Erdös [10] notices that $f(n+1) = \mathrm{C}_n$ for $1 \le n \le 7$, and that the next values are 1089, 2673, 6561, 15633, 37249, 88209, 216153. The last value is a misprint as $f(15) = \mathrm{C}_4\mathrm{C}_5^2 = 216513$. If $x \ge 3$, then $f(5x+1) = \mathrm{C}_5^x$, $f(5x+2) = \mathrm{C}_6\mathrm{C}_5^{x-1}$, $f(5x+3) = \mathrm{C}_6^2\mathrm{C}_5^{x-2}$, $f(5x+4) = \mathrm{C}_6^3\mathrm{C}_5^{x-3}$ and $f(5x+5) = \mathrm{C}_4\mathrm{C}_5^x$. Finally, he says $f(n) = 81(f(n-5)$ for $n \ge 21$ [He says: "We obtain from (1) by a simple computation that for $n \le 20$ the value of $f(n)$ is given by Theorem 1. The rest of Theorem 1 is easily proved by induction, we have to use that $(k2^{k-1}+1)^{1/k}$ (k integer) increases for $k \le 5$ and decreases for $k \ge 5$. We suppress the details since they can easily be given and depend only on numerical estimates. Thus the proof of Theorem 1 is complete."

Proving monotonocity of the function $\mathrm{C}_k^{1/k}$ is non-trivial and it is hard to see how one can use it to prove the theorem. We explain here the idea of [24, 25]; in the first paper, Wakulicz says $f(n) = \mathrm{C}_5^a\mathrm{C}_6^b$ (for some a and b when n is large) because he wrongly assumes $\mathrm{C}_4\mathrm{C}_5^4 < \mathrm{C}_6^4$; the second paper corrects the mistake. Note that $\mathrm{C}_4\mathrm{C}_5^4 = 1420541793$ and $\mathrm{C}_6^4 = 1387488001$.

In our approach, all inequalities are proven by Coq. Since the numbers involved are huge, we use their binary versions; for instance, $\mathrm{C}_4\mathrm{C}_5^4$ requires about thirty digits. Let c_i be the binary version of of C_i. Thanks to the compatibility lemmas (we give here those not provided by the standard library), $\mathrm{C}_4\mathrm{C}_6 < \mathrm{C}_5^2$ becomes equivalent to $c_4 \cdot c_6 < c_5 \cdot c_5$, where $a \cdot b$ denotes the binary multiplication. We lock the definition of C; this means that, when we ask Coq no simplify $f(9)$, with the definition given below, we get C_4^2 rather than 1089. This also means that when we go from $\mathrm{C}_4\mathrm{C}_6 < \mathrm{C}_5^2$ $c_4 \cdot c_6 < c_5 \cdot c_5$, no C_I is evaluated, and no big integer is created. When we unlock, then in each c_i, C_I is evaluated as a natural number, then converted into a binary number. Thus, the greatest natural number used in the computations is $\mathrm{C}_7 = 449$., everything else uses binary numbers.

```
Lemma NoB_add a b : N.add(bin_of_nat a) (bin_of_nat b) = bin_of_nat (a +b).
Lemma NoB_mul a b : N.mul (bin_of_nat a) (bin_of_nat b) = bin_of_nat (a  * b).
Lemma NoB_ler a b : N.le (bin_of_nat a) (bin_of_nat b) -> a <= b.
Lemma NoB_ltr a b:  N.lt(bin_of_nat a) (bin_of_nat b) -> a < b.
```

The following relations hold: $C_6^4 < C_4C_5^5$, $C_4^2C_5^2 < C_6^3$, $C_4^3 < C_6^2$, $C_7^2 < C_4C_5^2$, $C_6C_7 < C_4^2C_5$, $C_5C_7 < C_6^2$, $C_4C_7 < C_5C_6$, $C_3C_7 < C_5^2$, $C_2C_7 < C_4C_5$, $C_2C_6 < C_3C_5$, $C_2C_5 < C_3C_4$, $C_2C_4 < C_3^2$, $C_3C_6 < C_4C_5$, $C_3C_5 < C_4^2$, $C_4C_6 < C_5^2$, $C_2C_3 < C_5$, $C_2^2 < C_4$, $C_3C_4 < C_7$ and $C_3^2 < C_6$. We deduce other relations, such as $C_2C_5^2 \le C_3C_4C_5 \le C_4^3 \le C_6^2$. We could also deduce $C_i^{i+1} < C_{i+1}^i$ for $i \le 4$, as well as $C_6^5 < C_5^6$. So, if $d(i) = C_i^{1/i}$, then $d(6) < d(5) > d(4)$. **Note**. For $i \le 7$, C_I is a prime number, except that $C_5 = 3^4$ and $C_4 = 11^3$. It follows that all inequalities are strict. In our code, only weak inequalities are needed.

One has $C_1C_n < C_{n+1}$ for $n > 0$. This is equivalent to $2+n2^n < 2+2^n+n2^n$, hence to $1 < 2^n$. Similarly, if $4 \le q$, then $C_q^2 > C_{2q}$ and $C_qC_{q+1} > C_{2q+1}$. In fact, let $t = 2^{n-1}$. The first inequality (after subtracting 1 and factoring out qt) is equivalent to $4t < 2 + qt$. The second inequality is $1 + at^2 < 1 + bt + ct^2$, where $a = 4(2q+1)$ and $c = 2q(q+1)$. Now $a \le c$ follows from $2 \le q$ and $4 \le q^2$. The result follows from $b > 0$.

The proof of Wakulicz is the following. Assume that f is a solution to (11.51), and $n > 1$. The first equality says that $f(n)$ is a product of some C_i, hence there is a list L such that $f(n) = \prod_{i\in L} C_i$ and $n - 1 = \sum_{i\in L} i$; the second relation says that L is "optimal" in that, if M is any list such that $\sum_{i\in L} i = \sum_{i\in M} i$ then $\prod_{i\in L} C_i \ge \prod_{i\in M} C_i$.

Every sublist of an optional list is optional. In particular, if k is an element of an optional list, the list with a single element k is optional. We know that this is false when $k = 2q$ or $k = 2q + 1$ for $q \ge 4$. Hence: every element of an optional list is ≤ 7. Relation $C_4C_6 < C_5^2$ says that an optional list cannot contain both 4 and 6. A careful application of all the inequalities given above shows that an optimal list satisfies some condition (O), and clearly, (O) has a unique solution. So, f satisfies the conditions stated above.

Consider the following code:

```
Definition ndsCb n := (n * (2 ^ (n.-1))) .+1.
Definition ndsC  := locked  ndsCb.

Definition nds_k_of n :=
  if (n <=8) then n.-1 else if (n== 9) then 4 else if (n== 13) then 6 else
  if (n==19) then 6 else 5.

Definition nds_value n c := let k := n %/5 in let i := n %% 5 in
      if n <= 7 then c n else
       if (i== 0)  then (c 5) ^ k
       else if (i==1) then (c 6) * (c 5) ^ k.-1
       else if (i==2) then (c 6) ^ 2  * (c 5) ^ k.-2
       else if (i==3) then
         if(k== 1) then  (c 4) ^ 2 else if(k==2) then (c 4) ^ 2 * (c 5)
         else (c 6)^3  * (c 5) ^ (k-3)
       else (c 4)  * (c 5) ^ k.

Definition nds_sol n := if n is m.+1 then nds_value m ndsC else 1.

Lemma ndsCE n : ndsC n = ndsCb n.
Lemma ndsC0 : ndsC 0 =  1.
Lemma nds_sol0: nds_sol 0 = 1.
Lemma nds_sol1: nds_sol 1 = 1.
```

This defines C (locked and unlocked), a function $k(n)$ and a function $f_e(n)$. There are some trivial lemmas such as $C_0 = 1$, $f_e(0) = f_e(1) = 1$. The result of Wakulicz is: whenever f is solution of (11.51), then $f = f_e$. Consider now the following code

```
(*
Definition nds_prods L n :=
   [seq ndsC i.+1 * nth 0 L i | i <- iota 0 n.+1].
Fixpoint nds_rec (n : nat) : seq nat :=
  if n is n1.+1 then (\max_(i <- nds_prods (nds_rec n1) n1) i) ::  (nds_rec n1)
  else [::1].
Definition nds_sol n := if n is m.+1 then head 0 (nds_rec m) else 1.
*)
```

The last function f is clearly a solution of (11.50), hence of (11.51). It follows: $f = f_e$. In particular, f_e is a solution to (11.51).

11.13.2 Solution of the equation

In this section we show that f_e is a solution to (11.51). The first claim is that $0 < k(n) < n$ whenever $n > 1$. We then say that f_e satisfies

$$f(n) = \mathrm{C}_{k(n)} f(n - k(n)). \tag{11.52}$$

Obviously, this equation has a unique solution. As $k(n) = 5$ for large n, we get

$$f_e(n+5) = \mathrm{C}_5 f_e(n) \qquad (n \geq 15) \tag{11.53}$$

It follows that f_e satisfies the first part of equation (11.51).

```
Lemma nds_k_of_bd n: 1 <n -> 0 < nds_k_of n < n.
Lemma nds_sol_with_k n: 1 < n ->
  nds_sol n = (ndsC (nds_k_of n)) * nds_sol (n - (nds_k_of n)).  (* 64 *)

Lemma nds_value_prop f:
  f 0 = 1 -> f 1 = 1 ->
  (forall n, 1 < n ->  f n = (ndsC (nds_k_of n)) * f (n - (nds_k_of n))) ->
  (forall n,  f n = nds_sol n).
Lemma nds_sol_max n: 1 < n ->
  exists2 k, 0 < k < n & (nds_sol n = (ndsC k) * (nds_sol (n - k))).
Lemma nds_sol_rec5 n: 15 <= n -> nds_sol (n + 5) = ndsC 5 * nds_sol n.
```

The results shown here are all trivial.

```
Lemma nds_soll7 n: n <= 7 -> nds_sol n.+1 = ndsC n.
Lemma nds_sol2: nds_sol 2 = ndsC 1.
Lemma nds_sol3: nds_sol 3 = ndsC 2.
Lemma nds_sol4: nds_sol 4 = ndsC 3.
Lemma nds_sol5: nds_sol 5 = ndsC 4.
Lemma nds_sol6: nds_sol 6 = ndsC 5.
Lemma nds_sol7: nds_sol 7 = ndsC 6.
Lemma nds_sol8: nds_sol 8 = ndsC 7.
Lemma nds_sol9: nds_sol 9 = ndsC 4 ^2.
Lemma nds_sol10: nds_sol 10 = ndsC 4 * ndsC 5.
Lemma nds_sol11: nds_sol 11 = ndsC 5 ^2.
Lemma nds_sol12: nds_sol 12 = ndsC 5 * ndsC 6.
Lemma nds_sol13: nds_sol 13 = ndsC 6 ^2.
Lemma nds_sol14: nds_sol 14 = ndsC 4 ^2  * ndsC 5.
Lemma nds_sol15: nds_sol 15 = ndsC 4 * ndsC 5 ^2.
Lemma nds_sol16: nds_sol 16 = ndsC 5 ^3.
```

```
Lemma nds_sol17: nds_sol 17 = ndsC 5 ^2  * ndsC 6.
Lemma nds_sol18: nds_sol 18 = ndsC 5 * ndsC 6 ^ 2.
Lemma nds_sol19: nds_sol 19 = ndsC 6 ^ 3.
Lemma nds_sol20: nds_sol 20 =  ndsC 4 * ndsC 5 ^3.
Lemma nds_sol21: nds_sol 21 =  ndsC 5 ^4.
Lemma nds_sol22: nds_sol 22 =  ndsC 5 ^3 * ndsC 6.
Lemma nds_sol23: nds_sol 23 =  ndsC 5 ^2 * ndsC 6 ^ 2.
Lemma nds_sol24: nds_sol 24 =  ndsC 5 * ndsC 6 ^ 3.
Lemma nds_sol25: nds_sol 25 =  ndsC 4 * ndsC 5 ^ 4.
Lemma nds_sol26: nds_sol 26 =  ndsC 5 ^ 5.
```

We have $\mathrm{C}_k f_e(n) \le f_e(n+k)$ when $k \le 7$ and $n < 20$. Proof; There is a finite number of cases, In each case, we use the values of f given above, and the inequalities involving the C_i.

```
Lemma nds_sol_aux k n:   (* 422 *)
   k <= 7 -> 0 < n < 20 -> ndsC k * nds_sol n <= nds_sol (n + k).
```

We have $\mathrm{C}_k f_e(n-k) \le f_e(n)$. This is equivalent to $\mathrm{C}_k f_e(n) \le f_e(n+k)$. Assume first $k \le 7$. If $n < 20$, this is the previous lemma; otherwise $f_e(n) = \mathrm{C}_5 f_e(n-5)$ and $f_e(n+k) = \mathrm{C}_5 f_e(n-5+k)$ and the result holds by induction on n. We finish with an induction on k. Assume $k > 7$ even, say $k = 2q$ with $q \ge 4$. We know $\mathrm{C}_k \le \mathrm{C}_q^2$. Hence $\mathrm{C}_k f_e(n) \le \mathrm{C}_q^2 f_e(n) \le \mathrm{C}_q f_e(n+q) \le f_e(n+2q) = f_e(n+k)$. The case k odd is similar. This shows that f_e is a solution.

```
Lemma nds_sol_bd n k: 2 <= n -> 0 < k < n ->
    (ndsC k) * (nds_sol (n - k)) <= nds_sol n.
```

We state now a variant. Assume $f(0 = f(1) = 1$ and

$$\forall n > 1, \exists k, 0 < k < n \text{ and } f(n) \le \mathrm{C}_k \cdot f(n-k) \qquad \forall n > 1, f(n) \ge f_e(n) \tag{11.54}$$

Then $f = f_e$. Proof by induction on n. From $f(n) \le \mathrm{C}_k f(n-k)$ we get $f(n) \le \mathrm{C}_k f_e(n-k) \le f_e(n) \le f(n)$.

```
Lemma nds_alt (f: nat -> nat) (c:= ndsC) :
  f 0 = 1 -> f 1 = 1 ->
  (forall n,   2 <= n ->
             exists2 k, 0 < k < n & f n <= (c k) * (f (n - k))) ->
  (forall n,  nds_sol n.+1 <= f n.+1) ->
  (forall n,  nds_sol n = f n).
```

11.13.3 Bourbaki integers

We restate the previous result n, the Theory of Sets of Bourbaki. first need to introduce some integers

```
Definition card_8 := nat_to_B 8.
Definition card_15 := nat_to_B 15.
Definition card_13 := nat_to_B 13.
Definition card_19 := nat_to_B 19.

Lemma nat_to_B1: nat_to_B 1 = \1c.
Lemma nat_to_B2: nat_to_B 2 = \2c.
```

```
Lemma nat_to_B3: nat_to_B 3 = \3c.
Lemma nat_to_B4: nat_to_B 4 = \4c.
Lemma nat_to_B5: nat_to_B 5 = \5c.
Lemma nat_to_B6: nat_to_B 6 = \6c.
```

We can now define $k(n)$. $\mathrm{C}(n)$ $f(n)$

```
Definition nds_k_of n :=
  Yo (n <=c card_8) (cpred n)
     (Yo (n = \9c) \4c (Yo (n = card_13) \6c (Yo (n = card_19) \6c \5c))).
Definition ndsC n := csucc (n *c (\2c ^c (cpred n))).
Definition nds_explicit C4 C5 C6 k i :=
  Yo (i = \0c) (C5 ^c k)
    (Yo (i = \1c) (C6 *c C5 ^c (k -c \1c))
       (Yo (i = \2c) (C6 ^c \2c *c C5 ^c (k -c \2c))
          (Yo (i = \3c)
              (Yo (k = \1c) (C4 ^c \2c)
                  (Yo (k = \2c) (C4 ^c \2c  *c C5)
                      (C6 ^c \3c *c C5 ^c (k -c \3c))))
              (C4 *c C5 ^c k)))).

Definition nds_sol' n :=
  Yo (n <=c \7c) (ndsC n)
     (nds_explicit (ndsC \4c) (ndsC \5c)(ndsC \6c) (n %/c \5c) (n %%c \5c)).
Definition nds_sol n := Yo (n = \0c) \1c (nds_sol' (cpred n)).
```

We prove here some compatibility relations, and show that the functions map integers to integers.

```
Lemma nds_k_of_v n:
  nat_to_B (Nds.nds_k_of n) = nds_k_of (nat_to_B n).
Lemma ndsC_v n: nat_to_B (Nds.ndsC n) = ndsC (nat_to_B n).
Lemma nds_sol_nz n: natp n -> nds_sol (csucc n) = nds_sol' n.
Lemma NS_ndsC n: natp n -> natp (ndsC n).
Lemma nds_k_of_bd n: natp n -> \1c <c n ->
  \0c <c nds_k_of n /\ nds_k_of n <c n.
Lemma NS_nds_sol n: natp n -> natp (nds_sol n).
```

We introduce now three equations; the first two equations correspond to (11.51), the third one to (11.50).

```
Definition nds_fbd f:= forall n k, natp n ->  \2c <=c n ->
     \0c <c k -> k <c n ->
      (ndsC k) *c (f (n -c k)) <=c (f n).
Definition nds_fmax f:= forall n, natp n ->  \2c <=c n ->
   (exists k,  [/\ \0c <c k, k <c n &
      (ndsC k) *c (f (n -c k)) = (f n)]).

Definition nds_fixpt_eq f := forall n, natp n -> \2c <=c n ->
    f n = \csup (fun_image (Zo Nat (fun k => \0c <c k /\ k <c n ))
       (fun k => (ndsC k) *c (f (n -c k)))).
```

If f is a solution of these equation, and n is an integer, then $f(n)$ is an integer. The proof is by induction; we assume that $f(m)$ is an integer whenever $m < n$ and deduce that $f(n)$ is an integer. In the first case we use the relation $f(n) = C_k f(n-k)$. In the second case, $f(n)$ is the supremum of some set E, which is finite, nonempty, formed of integers '(by induction). It is obvious that the equations have the same solution. As f_e is a solution (with natural numbers), then f_b is a solution (with Bourbaki integers). We deduces that f_b satisfies equation (11.53), since this is the Bourbaki conclusion.

```
Lemma nds_bdmax_nat f:
   [/\ f \0c = \1c, f \1c = \1c, nds_fbd f &  nds_fmax f] ->
   forall n, natp n -> natp (f n).
Lemma nds_bdmax_nat' f:
   [/\ f \0c = \1c, f \1c = \1c &  nds_fixpt_eq f] ->
   forall n, natp n -> natp (f n).
Lemma nds_max_bd_prop f:
   [/\ f \0c = \1c, f \1c = \1c, nds_fbd f &  nds_fmax f] <->
   [/\ f \0c = \1c, f \1c = \1c & nds_fixpt_eq f].
Lemma nds_sol_fix_pt (f := nds_sol):
   [/\ f \0c = \1c, f \1c = \1c & nds_fixpt_eq f].

Lemma nds_sol_Bourbaki_conc (f := nds_sol):
  forall n, natp n -> card_15 <=c n ->
     f(n +c \5c) = (ndsC \5c) *c f n.
```

We can now state the COQ results in Bourbaki.

```
Lemma nds_k_of_prop f:
   f \0c = \1c -> f \1c = \1c ->
   (forall n, natp n -> \1c <c n ->
             f n = ndsC (nds_k_of n) *c f (n -cnds_k_of n)) ->
   forall n, natp n ->  f n = nds_sol n.

Lemma nds_alt f:
  f \0c = \1c -> f \1c = \1c ->
  (forall n, natp n -> \1c <c n ->
     exists k, [/\ \0c <c k, k <c  n & f n <=c (ndsC k) *c f (n -c k)]) ->
  (forall n, natp n -> nds_sol (csucc n) <=c f (csucc n)) ->
  (forall n, natp n -> f n = nds_sol n).
```

11.13.4 Solution of the problem.

By abuse of language, a sequence of n ordinals is a functional term X such that $X(i)$ is an ordinal for $i < n$. If $\sigma \in \mathfrak{S}_n$ (i.e., is a permutation of I_n), we denote by X_σ the sequence $i \mapsto X(\sigma(i))$.

We denote by $\sum X_\sigma$ the sum $\sum_i X_{\sigma(i)} = X(\sigma(n-1)) + \ldots + X(\sigma(1)) + X(\sigma(0))$, by S_X the set of all these quantities when $\sigma \in \mathfrak{S}_n$, by N(X) the cardinal of S_X, and by $f(n)$ the supremum of all N(X).

```
Definition nds_sc (X:fterm) n g := osumf (fun z => (X (Vf g z))) n.
Definition nds_sums (X:fterm) n := fun_image (permutations n) (nds_sc n X).
Definition nds_card  (X: fterm) n := cardinal (nds_sums n X).
Definition nds_ax := ord_below_n.
```

We start with some easy facts: $1 \le N(X) \le n!$, and $N(X) = 1$ when $n \le 1$. Note that $N(X) = N(X_\sigma)$ for any permutation σ.

```
Lemma nds_card_bd X n: natp n ->  nds_card n X <=c (factorial n).
Lemma nds_ax_perm X n f: natp n -> nds_ax X n ->
   inc f (permutations n) ->
   (nds_ax (fun z => X (Vf f z)) n  /\  ordinalp (nds_sc n X f)).
Lemma nds_sums_exten m Y1 Y2: natp m -> same_below Y1 Y2 m ->
   nds_sums m Y1 = nds_sums m Y2.
Lemma nds_sc_exten X X' n f:
  natp n -> (same_below X X' n) ->
  inc f (permutations n) ->
  nds_sc n X f = nds_sc n X' f.
Lemma nds_card_exten Y1 Y2 m:
   natp m -> same_below Y1 Y2 m ->
   nds_card m Y1 = nds_card m Y2.

Lemma nds_card_perm_inv X n g:
  natp n -> nds_ax X n -> inc g (permutations n) ->
  nds_card n X = nds_card n (fun z => X (Vf g z)).
Lemma NS_nds_card X n: natp n -> natp (nds_card n X).
Lemma nds_card_0 X: nds_card X \0c = \1c.
Lemma nds_card_1 X: nds_card X \1c = \1c.
```

Step 1. if X is sequence of ordinals of length $n+1$ with different degrees, then $\mathrm{N}(\mathrm{X}) = 2^n$. The proof is rather long.

Denote by d_i the degree of $\mathrm{X}(i)$. We may assume $d_i < d_j$ whenever $i < j$. In this case: $\mathrm{X}(i)+\mathrm{X}(j) = \mathrm{X}(j)$. If $f \in \mathfrak{S}_{n+1}$ we denote by $i = i(f)$ the index such that $f(i) = n$. We have $\sum \mathrm{X}_f = \sum_{j\le i} \mathrm{X}(f(j))$, call this (E). Denote by $k = k(f)$ the greatest index such that f is increasing up to k. We have $k = n$ or $f(k+1) < f(k)$. Note that for $j \le k$ we have $j \le f(j)$ so if $k = n$, we get $f(n) = n$. Assume $f(k) < n$, so that $k < i$. Let g the element of $\mathfrak{S}_{n+1}$ defined by: if $k < j < i$, then $g(j) = f(j+1)$, $g(i) = f(k+1)$, and $g(j) = f(j)$ otherwise. Note that $g(i-1) = f(i) = n$ so $i(g) < i(f)$. We pretend $\sum \mathrm{X}_f = \sum \mathrm{X}_g$. We have $\sum \mathrm{X}_g = \mathrm{C}+\mathrm{X}(f(k+1))+\mathrm{B}+\mathrm{A}$ and $\sum \mathrm{X}_f = \mathrm{C}+\mathrm{B}+\mathrm{X}(f(k+1))+\mathrm{A}$, where A contains terms with index $\le k$, C contains terms with index $\ge i$, and B contains the other terms. By equation (E), we may discard some terms, so $\sum \mathrm{X}_g = \mathrm{B}+\mathrm{A}$ and $\sum \mathrm{X}_f = \mathrm{B}+\mathrm{X}(f(k+1))+\mathrm{A}$. Note that $\mathrm{A} = \mathrm{X}(f(k))+\mathrm{A}'$ and $\mathrm{X}(f(k+1))+\mathrm{X}(f(k)) = \mathrm{X}(f(k))$ since $f(k+1) < f(k)$. It follows $\mathrm{X}(f(k+1))+\mathrm{A} = \mathrm{A}$. Let now $\mathrm{T}(f)$ be the set of all permutations g such that $\sum \mathrm{X}_f = \sum \mathrm{X}_g$. Choose in this set a function g that minimises i. Then $g(k(g)) < n$ is absurd. Hence $g(k(g)) = n$, so that $k(g) = i(g)$. Let's denote by $\mathrm{P}(g)$ the property that g is strictly increasing below $g^{-1}(n)$. We have shown: $\mathrm{T}(f)$ contains a function that satisfies P.

By the axiom of choice we can define a function $\mathrm{F} : \mathrm{S}_\mathrm{X} \to \mathfrak{S}_{n+1}$ such for every x, $\mathrm{F}(x)$ satisfies P and $x = \sum \mathrm{X}_{\mathrm{F}(x)}$. If f is a permutation satisfying P, we denote by $\mathrm{K}(f)$ the set all $f(j)$ for $j < i(f)$. This set contains $i(f)$ elements, but not n, hence $\mathrm{K}(f) \subset \mathrm{I}_n$. Let $\phi : \mathrm{S}_\mathrm{X} \to \mathfrak{P}(\mathrm{I}_n)$ be defined by $\phi(x) = \mathrm{K}(\mathrm{F}(x))$. It suffices to show that ϕ is a bijection.

Injectivity. Assume x and x' in S_X, $f = \mathrm{F}(x)$, $f' = \mathrm{F}(x')$, $\mathrm{K} = \mathrm{K}(f)$ and $\mathrm{K}' = \mathrm{K}(f')$. We have: if $\mathrm{K} = \mathrm{K}'$ then $x = x'$. By equation (E), $x = \sum_{j\le i} \mathrm{X}(f(j))$ and $x' = \sum_{j\le i} \mathrm{X}(f'(j))$. Here i is the cardinal of K (and K′). Now f and f' (restricted to indices $\le i$) are two strictly increasing functions, with the same image, so are equal; it follows that the sums are the same. Surjectivity. If K is a subset of I_n, there is a permutation f satisfying P such that $\mathrm{K} = \mathrm{K}(f)$ (see the additional lemma). Let $x = \sum \mathrm{X}_f$. We have $\phi(x) = \mathrm{K}(\mathrm{F}(x)) = \mathrm{K}(f) = \mathrm{K}$. because if f and g satisfy P and $\sum \mathrm{X}_f = \sum \mathrm{X}_g$, then $\mathrm{K}(f) = \mathrm{K}(g)$. The proof is as follows: By (E) $\sum \mathrm{X}_f = \sum_{j<i} \mathrm{X}(f(j)) = \mathrm{S}_i$ for some i. The assumption is now that $\mathrm{S}_i = \mathrm{S}'_j$ (where S′ is the sum associated to g), f and g are strictly increasing below i and j respectively. We have to show that $i = j$, and that f and

g agree up to j. The proof is by induction. The result is obvious if one of i or j is zero. Now $S_{i+1} = X(f(i)) + S_i$, $S'_{j+1} = X(g(j)) + S'_j$. Clearly, the degree of the sum is the degree of the first term (because the terms are in strictly decreasing degree). Since $X(f(i))$ and $X(g(j))$ have the same degree it follows $f(i) = g(j)$. Now $X(f(i)) + S_i = X(g(j)) + S'_i$ implies $S_i = S'_j$; the result follows by induction.

```
Definition opos_below (f : fterm) n := forall k, k <c n -> \0o <o (f k).

Lemma osum_negl_recd X n: natp n -> opos_below X (csucc n) ->
  (forall i, i<c n -> odegree (X (csucc i)) <o odegree (X \0c)) ->
  osumf X (csucc n) = X \0c.
Lemma nth_elt_dbl_prop_ter n K (k := cardinal K)
  (s:= (Lf (nth_elt_dbl (K +s1 n) (csucc n)) (csucc n) (csucc  n))):
 natp n -> inc K (\Po n) ->
     [/\ inc s (permutations (csucc n)),
      k<=c n, K = fun_image k (Vf s),
    Vf s k = n &
    (forall i j, i<c j -> j <c k -> (Vf s i) <c (Vf s j))].

Lemma nds_card_different_deg_gen X n: natp n ->  (* 462 *)
  opos_below X (csucc n) ->
  (forall i j, i <c j -> j <=c n -> odegree (X i) <o odegree (X j)) ->
  nds_ax X (csucc n) /\ nds_card X (csucc n) =  \2c ^c n.
Lemma nds_card_different_deg n: natp n ->
  nds_ax oopow (csucc n) /\ nds_card  oopow (csucc n) = \2c ^c n.
```

Step 2. If X is a sequence with at least two terms such that $X_i = 0$ for some i, there is a sequence Y of the same length such that $N(X) < N(Y)$.

We start with a preliminary remark: if all X_i are integers; then $\sum X_i$ is the cardinal sum, which is commutative, so that $N(X) = 1$. This means that we may assume not all X_i zero, since replacing a single zero by a one does not change $N(X)$. Hence no $\sum X_\sigma$ is zero; if $A = S_X \cup \{0\}$, then $N(X) < \mathrm{card}(A)$. We may assume X of length $n+1$, and $X(n) = 0$. There is $u = \omega^e$ such that $X(i) < u$ for every i. Let Y be the sequence obtained by replacing $X(n) = 0$ by u. We have $u \in S_Y$ (take any permutation σ with $\sigma(0) = n$, and note that $X(i) + u = u$ whatever i). We have $B \subset S_Y$: assume $u + v \in B$; the case $v = 0$ has been considered above, so $v = \sum X_\sigma$, for some σ. Let $k = \sigma^{-1}(n)$. Let $\tau(i)$ be $\sigma(i)$ for $i < k$, $\sigma(i+1)$ for $k < i < n$; and n for $i = n$. Then $v = \sum X_\tau$ and $u + v = \sum Y_\tau$. Note that $v \mapsto u + v$ is injective so A and B have the same cardinal, $\mathrm{card}(B) \le N(Y)$.

```
Lemma osum_of_nat n X: natp n -> (forall i, i <c n -> natp (X i)) ->
  osumf X n = csumb n X.
Lemma osum_of_nat_bis n X f: natp n -> (forall i, i <c n -> natp (X i)) ->
  inc f (permutations n) ->
  nds_sc X n f  =  osumf X n.
Lemma nds_of_nat n X:  natp n -> (forall i, i <c n -> natp (X i)) ->
  nds_card X n = \1c.
Lemma nds_type0 n X:
  natp n -> \2c <=c n -> nds_ax X n -> (exists2 i, i<c n & X i = \0o) ->
  exists2 Y, nds_ax Y n & nds_card X n <c nds_card Y n.  (* 141 *)
```

Step 3. If X is a sequence of constant degree, then $N(X) \le n$. In fact, let e be the common degree, write $X_i = \omega^e \cdot c_i + r_i$, where c_i is a non-zero integer and $r_i < \omega^e$. We have $X_1 + X_2 =$

$\omega^e \cdot (c_1 + c_2) + r_2$. By induction, for every permutation σ, we get

$$\sum X_\sigma = \omega^e \cdot \sum c_i + r_{\sigma(0)}. \tag{11.55}$$

The result follows as the first term of the sum is independent og σ/

```
Lemma nds_same_deg_s2 e c1 c2 r1 r2:
  ordinalp e -> ordinalp c1 -> r1 <o oopow e ->
  \0o <o c2 -> ordinalp r2 ->
  (oopow e *o c1 +o r1) +o (oopow e *o c2 +o r2) =
  (oopow e *o (c1 +o c2) +o r2).
Lemma nds_same_deg_sn e (c r: fterm) n:
  natp n -> \0c <c n -> ordinalp e ->
  (forall i, i<c n -> \0o <o c i /\ r i <o oopow e) ->
  osumf (fun i => oopow e *o (c i) +o r i) n =
  oopow e *o (osumf c n) +o (r \0c).
Lemma nds_same_deg_s_perm e (c r: fterm) n f
      (X := (fun i => oopow e *o (c i) +o r i)):
  natp n -> \0c <c n -> ordinalp e ->
  (forall i, i<c n -> \0o <o c i /\ r i <o oopow e) ->
  (forall i, i<c n ->  c i <o omega0)  ->
  inc f (permutations n) ->
  nds_sc X n f =
  oopow e *o (osumf c n) +o (r (Vf f \0c)).
Lemma nds_card_same_deg_bd e X n: natp n -> \0c <c n ->
  (opos_below X n) ->
  (forall i, i <c n -> odegree (X i) = e) ->
  nds_card X n <=c n.
```

Step 4. Definition and first properties of f.

We introduce here the function f of Bourbaki. Fo every integer n, there exists a sequence X of n ordinals such that $N(X) = f(n)$, and for every such sequence we have $N(X) \le f(n)$. Obviously $f(n) \le n!$ and $f(n)$ is an is an integer. By step 1, $f(n) \ge 2^{n-1}$, hence $f(2) = 2$, and $f(n) > n$ for $n > 2$.

```
Definition nds_F n :=
 \osup (Zo Nat (fun z => exists2 X, nds_ax X n & nds_card n X = z)).

Lemma nds_f_def n (f := nds_F n): natp n ->
  [/\ natp f,
   f <=c factorial n,
   (exists2 X, nds_ax X n & nds_card n X = f) &
   (forall X, nds_ax X n -> nds_card n X <=c f)].

Lemma nds_sd1 n: natp n -> \2 ^c n <=c nds_F (csucc n).
Lemma nds_f2: nds_F \2c = \2c.
Lemma nds_sd2 n: natp n -> \2c <c n -> n <c (nds_F n).
```

Step 5. The function f_k.

We say that X is of type k, if there is an ordinal d such that the sequence contains k elements of degree d, all other elements being of degree $> d$. We define $f_k(n)$ to be the supremum of the N(X) where the sequence has length n and type k.

```
Definition nds_type X n k :=
  (opos_below X n) /\
  (exists m, [/\ ordinalp m,
     (forall i,  i<c n -> m <=o odegree (X i)) &
     cardinal (Zo n (fun i => odegree (X i) = m)) = k]).
Definition nds_FA n k:=
 \osup (Zo Nat (fun z => exists X, [/\ nds_ax X n, nds_card n X = z
     & nds_type X n k])).
```

We show for $n \geq 2$ that

$$f(n) = \sup_{0<k<n} f_k(n). \tag{11.56}$$

First, if $0 < k \leq n$, there is a sequence of type k (take $X(i) = 1$ for $i < k$, ω otherwise). This allows us to characterize $f_k(n)$ as the maximal value of all $N(X)$ where X is of type k. Then $f_k(n) \leq f(n)$ is obvious; so let's show that for every $n \geq 2$, there is k such that $0 < k < n$ with $f_k(n) = f(n)$. In the case $n = 2$, this follows from $f_1(2) = f(2) = 2$. Take X such that $f(n) = N(X)$. If there is i such that $X_i = 0$, then X is not optimal by step 2, so that we can ignore this case. Otherwise each X_i has a degree, the set of degrees has a least element m and if k is the number of elements of degree m, then $k > 0$ and X is of type k. Hence $f_k(n) = f(n)$. Now $k = n$ is absurd: this means that every $X(i)$ has degree m, hence $N(X) \leq n$ by step 3. This contradicts $n < f(n) = N(X)$.

```
Lemma nds_type_exists_bd n k:
   natp n -> \0c <c k -> k <=c n -> exists X,
    nds_ax X n /\ nds_type X n k.
Lemma nds_FA_def n k (v := nds_FA n k):
  natp n -> \0c <c k-> k <=c n ->
  [/\ natp v,
   v <=c  (nds_F n),
   (forall X,  nds_ax X n -> nds_type X n k -> nds_card n X <=c v) &
   (exists  X, [/\ nds_ax X n, nds_type X n k & nds_card n X = v])].
Lemma nds_FA_21: nds_FA \2c \1c = \2c.
Lemma nds_F_FA_def n (g := nds_FA n) (f:= nds_F n):
  natp n -> \2c <=c n ->
  (forall k,  \0c <c k -> k <c n  -> (g k) <=c f) /\
  (exists k, [/\ \0c <c k,  k <c n & (g k) = f]).  (* 57 *)
```

Step 6. We introduce here a function C, and show that it satisfies $C_k = 1 + k2^{k-1}$.

We consider here the maximum number of partial sums of X. If q is the number of terms in the partial sum, K the set of indices of the partial sum, then a partial sum has the form $\sum_i X(e_K(\tau(i)))$ for some permutation τ of I_q. If each $X(i)$ is an integer, the sum is independent of τ and the partial sum is $\sum_{i\in K} X(i)$.

```
Definition nds_sc_K X K := csumb (cardinal K)(fun z => X (nth_elt K z)).

Lemma nds_sc_Kval n c K: natp n ->  inc K (\Po n) ->
   nds_sc_K c K = csumb K c.
Lemma nds_C_prop n a: natp n -> inc a n ->
  cardinal (\Po (n -s1 a)) =  \2c ^c (cpred n).
```

We introduce now the partial sum $\sum X_{K\tau} := \sum_i X(e_K(\tau(i)))$, the set S_{Xn} of all partial sums and its cardinal $C_{X,n}$. We introduce the condition that X is a sequence of non-zero ordinals with constant degree, then C_n, the maximum of $C_{X,n}$ over all those X. We then prove: $C_{X,n} \leq n!2^n$ (this implies that C_n is well-defined and is an integer). We have $C_0 = 1$ since the only partial sum is the empty sum.

```
Definition nds_sc_Ktau X K tau :=
  nds_sc (fun z => X (nth_elt K z))  (cardinal K) tau.
Definition nds_tn_S X n:=
   unionf (\Po n) (fun K =>
      fun_image (permutations (cardinal K)) (nds_sc_Ktau X K)).
Definition nds_tn_C X n :=  cardinal (nds_tn_S X n).
Definition nds_tn_ax X n :=
    (forall i, i<c n -> \0o <o (X i)) /\
    exists2 e, ordinalp e & forall i, i<c n -> odegree (X i)  = e.
Definition nds_tn_sup n :=
  \csup(Zo Nat (fun z => exists2 X, nds_tn_ax X n & nds_tn_C X n = z)).

Lemma nds_tn_small X n:
   natp n -> nds_ax X n ->
   nds_tn_C X n <=c (\2c ^c n) *c (factorial n).
Lemma nds_tn_def n (f:=  nds_tn_sup n): natp n ->
  [/\ natp f, \0c <c f,
  (exists2 X, nds_tn_ax X n & nds_tn_C X n = f) &
  (forall X, nds_tn_ax X n -> nds_tn_C X n <=c f)].
Lemma nds_tn_zero: nds_tn_sup \0c = \1c.
```

Assume $X(i) = \omega^e \cdot c_i + r_i$. According to (11.55), we have

$$\sum X_{K\sigma} = \omega^e \cdot \sum c_K + r(e_K(\sigma(0)))$$

whenever K is non-empty and σ is a permutation. If K is empty, then the sum is zero. We can eliminate σ and state

$$s \in S_{Xn} - \{0\} \iff \exists K, \exists a \in K, \quad s = \omega^e \cdot \sum c_K + r(a).$$

We prefer the following form

$$S_{Xn} = \{0\} \cup \bigcup_{a \in I_n} \{s, \exists K \subset I_n - \{a\}, s = \omega^e \cdot (\sum_{i \in K \cup \{a\}} c(i)) + r(a)\}. \tag{11.57}$$

An easy computation says $\mathrm{card}(S_{Xn}) \leq 1 + n2^{n-1}$.

```
Lemma nds_tn_prop2 n X:  natp n -> \0c <c n -> nds_tn_ax X n -> (* 67 *)
  (nds_tn_S X n) =
  unionf n (fun a => fun_image (\Po (n -s1 a))
       (fun K => oopow  (odegree (X \0c)) *o (csumb (K +s1 a) (CNF_lc \o X))
           +o CNF_rem (X a)))
  +s1 \0c.

Lemma nds_tn_max X n: natp n -> \0c <c n -> nds_tn_ax X n ->
  (nds_tn_C X n) <=c ndsC n.
```

Let $G_K = \sum_{i \in K} 2^i$. Assume K is a finite set of integers; then G_K is finite. Assume all elements of K are $< n$. Then $G_K < 2^n$. By induction on n, $K \mapsto G_K$ is injective. Also $(a, b) \mapsto \omega \cdot a + b$ is injective when a and b are integers (uniqueness of the CNF in case arguments are non-zero).

```
Definition sumpow2 K := csumb K (fun z => \2c ^c z).
Lemma sumpow2_N1 K: finite_set K -> sub K Nat -> natp (sumpow2 K).
Lemma sumpow2_N2 n a K:
  natp n -> inc K (\Po (n -s1 a)) -> natp (sumpow2 K).
Lemma sumpow2_rec a K: inc a K ->
  sumpow2 K = sumpow2 (K -s1 a) +c (\2c ^c a).
Lemma sumpow2_inj n K1 K2 : natp n ->
  (forall i, inc i K1 -> i<c n) -> (forall i, inc i K2 -> i<c n) ->
   sumpow2 K1 = sumpow2 K2 -> K1 = K2.
Lemma omega_monom_inj a b c d:
  natp a -> natp b -> natp c -> natp d ->
  omega0 *o a +o b = omega0 *o c +o d -> (a = c /\ b = d).
```

We introduce now a sequence $B_i = \omega \cdot 2^i + i$. Every element in the sequence has degree one. The s in (11.57) simplifies to $\omega \cdot G_{K\cup\{a\}} + a$. We show here C_n satisfies equation (11.49). We have already shown that $\mathrm{card}(S_{Xn}) \le 1 + n2^{n-1}$, and it suffices to show that equality holds for the case of B.

The proof is the following. We have $S_{Xn} = \{0\}/cup\bigcup_{a\in n} T_a$. We have that the T_a are mutually disjoint, of cardinal 2^{n-1} and do not contain zero. This says that the cardinal is $1 + n2^{n-1}$. Note that an element x of T_a has the form $\omega \cdot G_{K\cup\{a\}} + a$, so is not zero, since the coefficient in G of ω is non-zero. An element y of T_b has the form $\omega \cdot G_{L\cup\{b\}} + b$. Now $x = y$ implies $a = b$ (so that the sets are mutually disjoint), and that the G are the same, hence $G_K = G_L$. This implies $K = L$ (by induction on n).

```
Definition nds_base := fun i => omega0 *o (\2c ^c i) +o i.
Definition sumpow2 K := csumb K (fun z => \2c ^c z).

Lemma nds_base_prop i (u := nds_base i) : natp i ->
  [/\ odegree u = \1c, the_cnf_lc u = (\2c ^c i) & the_cnf_rem u = i].
Lemma nds_tn_ex_ax n:  natp n -> nds_tn_ax nds_base n.

Lemma nds_tnmax_ex n: natp n ->
  (nds_tn_S nds_base n) =
  unionf n (fun a => fun_image (\Po (n -s1 a))
    (fun K => omega0 *o (sumpow2 (K +s1 a)) +o a)) +s1 \0c.
Lemma nds_tnmax_ex n (X := fun i => omega0 *o (\2c ^c i) +o i):
  natp n ->
  (nds_tn_S X n) =
  unionf n (fun a => fun_image (\Po (n -s1 a))
    (fun K => omega0 *o (sumpow2 (K +s1 a)) +o a)) +s1 \0c.
Lemma nds_tn_ex_val n: natp n ->  nds_tn_C nds_base n = ndsC n.
Lemma nds_tn_value n: natp n -> nds_tn_sup n = ndstnC n.
```

Step 7. We consider here a sequence X of type k, with $0 < k < n$. There is a permutation σ of I_n such that, if $X' = X_\sigma$, then $N(X) = N(X')$, and (for some e) the first k elements of X' are of degree e, others are of degree $> e$.

```
Definition nds_type_nor X n k e:=
 [/\ nds_type X n k, ordinalp e,
    forall i, i<c k -> odegree (X i) = e  &
    forall i, k <=c i -> i <c n -> e <o odegree (X i)].

Lemma nds_type_nor_ex X n k:
```

```
   natp n -> \0c <c k -> k <=c n -> nds_type X n k ->
   exists Y e,  [/\ nds_type_nor Y n k e &
                    nds_card Y n = nds_card X n].
Lemma nds_tn_C_exten Z Y n: natp n -> same_below Z Y n ->
  nds_tn_C Z n = nds_tn_C Y n.
```

Thanks to the previous result, we may assume here that $0 < k < n$, $\mathrm{X}(i) > 0$, $\mathrm{X}(i)$ has degree e if $i < k$ and $\mathrm{X}(i)$ has degree $> e$ otherwise. Let σ be a permutation of I_n; we say that i is small [resp. large) if $\sigma(i) < k$ (resp. $\sigma(i) \geq k$. Note that small means $\mathrm{X}(\sigma(i))$ has degree e, and large means that $\mathrm{X}(\sigma(i))$ has degree $> e$. We denote by α_σ the smallest large index.

```
Section NdsStudy.
Variables (X: fterm) (n k e: Set).
Hypothesis nN: natp n.
Hypothesis knz: \0c <c k.
Hypothesis kln: k <c n.
Hypothesis Xax: nds_type_nor X n k e.

Let pL f i := k <=c Vf f i.
Let pS f i := Vf f i <c k.
Let fL f := intersection (Zo Nat (fun i => i <c n /\ pL f i)).
```

Consider a permutation σ. Let H_m be the condition that i is large for $\alpha_\sigma \leq i < \alpha_\sigma + m$. By definition α_σ is large, elements i with $i < \alpha_\sigma$ are small, and since there are k small elements, $\alpha_\sigma \leq k$. Let $q = \alpha_\sigma + m$. Let E be the set of large indices $\geq q$. This set has $n - k - m$ elements. If it's empty then $m = n - k$, and indices $\geq q$ are small. It has otherwise a least element, that is either q (so that a is large) or $q + p + 1$, so that $q - 1$ and $q + p + 1$ are large, and elements between are small.

Assume now i and $i + l + 2$ are large, and everything between them is small. Le's compose σ with the transposition that exchanges $i + 1$ and $i + l + 2$. This gives a permutation τ such that $i + 1$ is large; moreover it agrees with σ up to i, and $\sum \mathrm{X}_\sigma = \sum \mathrm{X}_\tau$. Let $a = \sigma(i)$, $b = \sigma(i+1)$ and $c = \sigma(i+2)$. The first sum is $s_1 + \mathrm{X}_c + s_2 + \mathrm{X}_b + \mathrm{X}_a + s_3$, the second sum is similar, with b and c exchanged, the s_i being the same. Now, a and c are large, while b and indices of s_2 are small. This means that $s_2 + \mathrm{X}_a = \mathrm{X}_a$, $s_2 + \mathrm{X}_c = \mathrm{X}_c$, $\mathrm{X}_b + \mathrm{X}_a = \mathrm{X}_a$ and $s_2 + \mathrm{X}_c = \mathrm{X}_c$. In both cases, the sums simplify to $s_1 + \mathrm{X}_c + \mathrm{X}_a + s_3$,

We combine now these two results: there is τ such that $\sum \mathrm{X}_\sigma = \sum \mathrm{X}_\tau$, $\alpha_\sigma = \alpha_\tau$, and the large elements are all grouped together. This means that H_m holds for $m = n - k$, proof by induction on m. Now $\sum \mathrm{X}_\tau = s_1 + s_2 + s_3$, s_1 is formed of small terms with index $\geq \alpha_\tau + (n-k)$, s_3 is formed of small terms with index $< \alpha_\tau$, and s_2 is formed of the big terms. Since there is at least one big term, we have term $s_1 + s_2 = s_2$ hence $\sum \mathrm{X}_\sigma = s_2 + s_3$. Recall that for the identity permutation we have first the small then the large elements. This means that there is a permutation f of I_{n-k} that puts them in the same order as τ, hence if $\mathrm{Y}(i) = \mathrm{X}(k+i)$ we have $s_2 = \sum \mathrm{Y}_f$. Let K be the set of all $\tau(i)$ for $i < \alpha_\sigma$. We have $s_2 = \sum_{i \in \mathrm{K}} \mathrm{X}_i$, for some ordering of K.

```
Lemma nds_fL_prop f: inc f (permutations n) ->
  [/\ natp (fL f), (fL f) <=c k,   (fL f) <c n,  pL f (fL f) &
      forall j, j <c (fL f) -> pS f j].
Lemma nds_fL_prop2 f m (q:= (fL f) +c m) :
  inc f (permutations n) -> natp m -> q <c n ->
  (forall j, j<c m -> pL f ((fL f) +c j)) ->
```

```
  [\/ (m = n -c k /\ forall j, j <c n -> q <=c j -> pS f j),
     pL f q |
    q <> \0c /\
    exists p, [/\ natp p, csucc (q +c p) <c n,
       pL f (csucc (q +c p)),
       pL f (cpred q) &
       forall z, z <=c p -> pS f (q +c z) ]].   (* 93 *)
Lemma nds_fL_prop3 i l f:
  inc f (permutations n) -> natp i -> natp l ->
  (csucc (csucc (i +c l))) <c n ->
  pL f i ->
  (forall j, j <=c l -> pS f (csucc (i+c j))) ->
  pL f (csucc (csucc (i +c l))) ->
  exists g,
     [/\ inc g (permutations n), nds_sc n X f = nds_sc n X g,
        forall j, j<c (csucc i) -> Vf f j = Vf g j & pL g (csucc i)]. (* 151 *)
Lemma nds_fL_prop4 f:
  inc f (permutations n) -> exists g,
  [/\ inc g (permutations n), nds_sc X n f = nds_sc X n g,
      fL f  = fL g &
      forall i, i<c n -c k -> pL g ((fL g) +c i) ].
```

Assume $\mathrm{H}(\sigma, n-k)$. Let $\alpha = \alpha_\sigma$ and $\beta = \alpha_\sigma + (n-k)$. Let's partition the interval I_n into three parts, E_1 corresponds to indices $< \alpha$, E_2 to indices i such that $\alpha \le i < \beta$, and E_3 to indices $> \alpha$. Let K_1, K_2 and K_3 be the images by σ of these sets. They are of cardinal α, $n-k$ and $k-\alpha$ respectively. We have $\mathrm{L}_\sigma(i)$ for $i \in \mathrm{E}_2$; this means $\mathrm{K}_2 \subset \mathrm{I}_n - \mathrm{I}_k$. Since these two sets have the same cardinal, it follows $\mathrm{K}_2 = \mathrm{I}_n - \mathrm{I}_k$, thus $\mathrm{K}_1 \cup \mathrm{K}_3 = \mathrm{I}_k$. In particular, $\mathrm{S}_\sigma(i)$ holds when $i < \alpha$ (this is true by definition of α and also when $i > \beta$). Write $\sum \mathrm{X}_\sigma = \mathrm{A} + \mathrm{B} + \mathrm{C}$, according to the partition of I_n. Since B is non-empty (i.e., $k < n$), it has a greatest element, and $\mathrm{B} = b + \mathrm{B}'$. Now, elements of A have degree e, while b has degree $> e$; the same computation as above shows $\mathrm{A} + b = b$. Thus, $\sum \mathrm{X}_\sigma = \mathrm{B} + \mathrm{C}$.

Let $\sigma_2(i) = \sigma(i+\alpha) - k$. If $i < n-k$ then $i + k \in \mathrm{E}_2$. Thus, σ_2 is a permutation of I_{n-k}. Moreover $\mathrm{B} = \sum \mathrm{X}'_{\sigma_2}$ where $\mathrm{X}'(i) = \mathrm{X}(i+k)$ (note: $\mathrm{X}'_{\sigma_2}(i) = \mathrm{X}(\sigma_2(i)+k) = \sigma(i+\alpha)$). Let e be the enumeration of K_1. This is a bijection $\mathrm{I}_\alpha \to \mathrm{K}_1$. Let $\sigma_1(i) = e^{-1}(\sigma(i))$. If $i \in \mathrm{I}_\alpha$, then $\sigma(i) \in \mathrm{K}_1$, so that $\sigma_1(i) \in \mathrm{I}_\alpha$ and σ_1 is a permutation of I_α. Moreover $\mathrm{C} = \sum \mathrm{X}''_{\sigma_1}$ where $\mathrm{X}''(i) = \mathrm{X}(e(i)))$ (since $\mathrm{X}''_{\sigma_1}(i) = \mathrm{X}(e(e^{-1}(\sigma(i)))) = \mathrm{X}(\sigma(i)))$.

```
Lemma nds_fL_prop5 f:  (* 188 *)
  inc f (permutations n) -> exists K s1 s2,
  [/\ sub K k, inc s1 (permutations (cardinal K)),
     inc s2 (permutations (n -c k)) &
     nds_sc X n f = nds_sc (fun z => X (k +c z)) (n-c k)  s2 +o
        nds_sc_Ktau X K s1].
```

We have shown: for every permutation σ, there exists K, σ_1, σ_2 such that $\sum \mathrm{X}_\sigma = \sum \mathrm{X}'_{\sigma_2} + \sum \mathrm{X}''_{\sigma_1}$. The converse holds: given K, σ_1, σ_2, we can construct σ as follows: let α be the cardinal of K. If $i \le \alpha$, then $\sigma(i) = e_\mathrm{K}(\sigma_1(i))$, if $i < n-k$, then $\sigma(i+\alpha) = \sigma_2(i) + k$ and if $i \ge \alpha + (n-k)$ then $\sigma(i) = e_{\mathrm{K}'}(\sigma_1(i - (\alpha + (n-k)))$, where $e_{\mathrm{K}'}$ is the enumeration of the complement of K in I_k.

It follows that S_X, the set of all $\sum \mathrm{X}_\sigma$ is the set of all $a + b$, where a is in $\mathrm{S}_{\mathrm{X}'}$ and b is in $\mathrm{S}_{\mathrm{X}k}$. We deduce that the cardinal of S_X is at most the product of the cardinals of these two sets. The first cardinal is at most $f(n-k)$, and the second cardinal is at most C_k (since the first k elements of X are of degree k).

```
Lemma nds_fL_prop6 v (X2:= (fun z => X (k +c z))):
  inc v (nds_sums n X) ->
  exists v1 v2, [/\ inc v1 (nds_tn_S X k), inc v2 (nds_sums (n -c k) X2) &
   v = v2 +o v1].
Lemma nds_tg9 (X2:= (fun z => X (k +c z))):  (* 199 *)
  (nds_sums n X) =
     unionf (nds_tn_S X k) (fun v1 => fun_image (nds_sums (n -c k) X2)
         (fun v2 => v2 +o v1)).
Lemma nds_tg10 : nds_card n X <=c (ndstnC k) *c (nds_F (n -c k)).
End NdsStudy.
```

We deduce the first relation of (11.54).

```
Lemma nds_alt_prop1 n: natp n ->  \1c <c n ->
   exists k, [/\ \0c <c k, k <c n & nds_F n <=c ndsC k *c nds_F (n -c k)].
```

Step 8. We prove here the second relation of (11.54). we define the shift of x by $s(x) = \omega^2 \cdot x$. We introduce a operation M_k that produces a sequence Y from a sequence X as follows: if $i < k$ then $Y(i) = B_i$, otherwise $Y_i = s(X(i))$. We say that a sequence is *small* if each term is $< \omega^\omega$. This produces a small sequence from a small sequence. Note that the k first terms of Y are of degree 1, other terms are of degree > 1.

```
Definition nds_shift X := (oopow \2o) *o X.
Definition nds_small_nz x :=  \0o <o x /\ x <o oopow omega0.
Definition nds_small_ax (X:fterm) n := forall k, k <c n -> nds_small_nz (X k).
Definition nds_merge k X :=
  fun i => Yo (i <c k) (nds_base i) (nds_shift (X (i -c k))).

Lemma nds_small_nz_shift x: nds_small_nz x -> nds_small_nz (nds_shift x).
Lemma nds_small_base i: natp i -> nds_small_nz (nds_base i).
Lemma nds_merge_prop1 k X n:
  natp k -> natp n -> opos_below X n ->
  opos_below (nds_merge k X) (k +c n).
Lemma nds_merge_prop2 k X n:
  natp k -> natp n -> opos_below X n ->
  (forall i, i <c k -> odegree (nds_merge k X i) = \1c) /\
  (forall i, k <=c i -> i <c (k +c n) ->
     \1c <o (odegree (nds_merge k X i))).
```

We say that an ordinal is a *binomial* if it is $< \omega^2$, in other terms, if it has the for $\omega \cdot a + b$, where a and b are integers. Being a binomial is invariant by addition so a sum of terms of the form B_i is a binomial. If $x = s(a) + b$ where a is an ordinal and b is a binomial, then a and b are uniquely defined by x (being the quotient and remainder in the division by ω^2). Finally, $\sum s(X(i)) = s(\sum X(i))$, so if $Z(i) = s(X(i))$ then $N(Z) = N(X)$. It follows that $N(M_k(X)) = C_k.N(X)$. Proof. At the end of step 6, we said that S_X was the set of all $a + b$; for $a \in A$ and $b \in B$. Every element of B is a binomial, every element of A is a shift. hence a and b are uniquely defined by the sum $a+b$, and N(X) is the product of the cardinals of A and B. The first cardinal is N(X) and the second cardinal is $C(k)$.

```
Definition nds_binomp x := x <o oopow \2o.

Lemma nds_binomp_base i: natp i -> nds_binomp (nds_base i).
Lemma nds_binomp_add u v: nds_binomp u -> nds_binomp v ->
```

```
  nds_binomp (u +o v).
Lemma nds_add_sb_inj x y u v: ordinalp x -> ordinalp y ->
  nds_binomp u -> nds_binomp v ->
  (nds_shift x) +o u = (nds_shift y) +o v -> x = y /\ u = v.
Lemma osumf_shift n X: natp n -> nds_ax X n ->
  ordinalp (osumf X n) /\
  osumf (fun i => (nds_shift (X i))) n = nds_shift (osumf X n).
Lemma nds_card_shift n X:
  natp n -> nds_ax X n ->
  nds_card X n = nds_card  (fun i => nds_shift (X i)) n.
Lemma nds_card_merge n k Y: natp n ->  k <c n -> (* 84 *)
  opos_below Y (n -c k) ->
  nds_card (nds_merge k Y) n = ndsC k *c nds_card Y (n -c k).
```

We define by transfinite induction a sequence X_n by $X_n = M_k(X_{n-k})$, where $k = k(n)$. For each n, X_n is a sequence of n small ordinals, of type $k(n)$. By the previous properties, if $g(n) = N(X_n)$, then $g(n) = C_{k(n)} g(n - k(n)))$. This says $g = f_e$. Obviously $f(n) \geq g(n)$, this shows the second relation of (11.54). The result follows.

```
Definition nds_induction a b (T: fterm2) :=
  transfinite_defined Nat_order
     (fun u => Yo (source u = \0c) a (Yo (source u = \1c) b
     (T (nds_k_of (source u)) (Vf u ((source u) -c (nds_k_of (source u))))))).
Definition nds_example_set :=
  (nds_induction (Lg Nat (fun i => \1o))
                (Lg Nat (fun i => \1o))
                (fun k t => (Lg Nat (nds_merge k (Vg t))))).
Definition nds_example n := fun i => (Vg (Vf nds_example_set n) i).

Lemma nds_induction_prop a b T (f := nds_induction a b T) :
  [/\ Vf f \0c = a, Vf f \1c = b &
      forall n, natp n -> \1c <c n ->
         Vf f n = T (nds_k_of n) (Vf f (n -c (nds_k_of n)))].
Lemma nds_induction_prop2 (f := nds_example) :
  [/\ forall i, natp i -> f \0c i = \1o,
     forall i, natp i -> f \1c i = \1o &
      forall n, natp n -> \1c <c n -> let k := nds_k_of n in
      same_below (f n) (nds_merge k (f (n -c k)))  n].
Lemma nds_example_small_ax n:
  natp n -> nds_small_ax (nds_example n) n.

Lemma nds_card_merge n k Y: natp n ->  k <c n ->
  nds_small_ax Y (n -c k) ->
   nds_card (nds_merge k Y) n = ndsC k *c nds_card Y (n -c k).
Lemma nds_card_sol (X:= nds_example) n: natp n -> \1c <=c n ->
  nds_ax (X n) n /\ nds_sol n = nds_card (X n) n.
Theorem nds_alt_som n: natp n ->  nds_F n = nds_sol n.
```

11.14 Natural sum and products of ordinals

Consider two ordinals α and β, and two disjoint sets A and B, ordered by $\leq_A$ and $\leq_B$ such that the order-type of $\leq_A$ is α, and the order-type of $\leq_B$ is β. Let $C = A \cup B$, partially ordered by "$x \leq_A y$ or $x \leq_B y$". The set of all order-types of well-orderings that extend this order is non-empty (it contains the order-type of "$x \le_A y$ or $x \le_B y$ or ($x \in A$ and $y \in B$)", which is the ordinal sum of α and β), and is independent of A and B. Its supremum is called the *natural sum* and denoted by $\alpha \oplus \beta$. We have obviously $\alpha + \beta \le \alpha \oplus \beta$. One could prove that the supremum is a greatest element, i.e., there is an ordering on C whose ordinal is $\alpha \oplus \beta$. On the product $D = A \times B$ one can consider the relation $x \le y$ defined by "$\mathrm{pr}_1 x \le_A \mathrm{pr}_1 y$ and $\mathrm{pr}_2 x \le_B \mathrm{pr}_2 y$". This is the usual order-product, and is in general not total. The set of all order-types of well-orderings that extends this order is non-empty (it contains $\alpha \cdot \beta$, the order-type of the lexicographic product), and is independent of A and B. Its supremum is called the *natural product* and denoted by $\alpha \otimes \beta$. We have $\alpha \cdot \beta \le \alpha \otimes \beta$.

One can show that these operations are commutative and associative, and the distributivity law also holds. Moreover $\alpha \oplus 0 = \alpha \otimes 1 = \alpha$. These functions are monotone in the sense that $\delta \oplus \alpha > \delta \oplus \beta$ and $\delta \otimes \alpha > \delta \otimes \beta$ are equivalent to $\alpha > \beta$. Finally, the product of two powers of ω is a power of ω. The natural sum and product are the "smallest" (in some sense) operations that satisfy these properties (see [8]). One has $\omega^a \otimes \omega^b = \omega^c$ where $c = a \oplus b$. Thus,

$$\left(\sum_i \omega^{a_i} \cdot c_i\right) \otimes \left(\sum_j \omega^{b_j} \cdot d_j\right) = \sum_{ij} \omega^{a_i \oplus b_j} \cdot c_i d_j \tag{11.58}$$

provided that c_i and d_i are integers. One could use this formula as the definition of the natural product; one has to be careful as $\sum$ is non-commutative: the RHS has to be understood as the sum of all $\omega^k \cdot e_k$ where k has the form $a_i \oplus b_j$, for some i, j, listed in decreasing order, and e_k is the sum of all corresponding $c_i d_j$ (this is a natural integer).

11.14.1 Natural sum

If c_i and d_i are two sequences of integers, e_i is a strictly increasing sequence of ordinals, then

$$\left(\sum_i \omega^{e_i} \cdot c_i\right) \oplus \left(\sum_j \omega^{e_j} \cdot d_j\right) = \sum_i \omega^{e_i} \cdot (c_i + d_i). \tag{11.59}$$

The first implementation of the natural sum was complicated. We use here a completely different approach; we define the natural sum of cnfs via (11.59), then deduces a formula for ordinal numbers.

If x and y are two cnfs, let $f(e)$ be the quantity $x_c(e) + y_x(e)$; and s the ordinal such that $s_c(e) = f(e)$. To say that $f(e)$ is non-zero is the same as $e \in E(x) \cup E(y)$. So s exists; we call it the *natural sum* of x and y. Clearly addition is associative and commutative, zero being the unit.

```
Definition cnf_nat_sum_mons x y :=
  fun_image (cnf_exponents x \cup cnf_exponents y)
           (fun i => J i (Vr x i +c Vr y i)).
Definition cnf_nat_sum x y := cnf_sort (cnf_nat_sum_mons x y).

Notation "x +#f y" := (cnf_nat_sum x y) (at level 50).

Lemma cnf_nat_sum_range x y: cnfp x -> cnfp y ->
   cnf_rangep (cnf_nat_sum_mons x y).
Lemma cnfp_nat_sum x y: cnfp x -> cnfp y  -> cnfp (x +#f y).
Lemma cnf_nat_sumC x y: x +#f y = y +#f x.
Lemma cnf_nat_sum_p1 x y : cnfp x -> cnfp y ->
```

```
  cnf_exponents (x +#y) = cnf_exponents x \cup cnf_exponents y.
Lemma cnf_nat_sum_p2 x y : cnfp x -> cnfp y ->  forall e,
   Vr (x +#f y) e = (Vr x e) +c  Vr y e.
Lemma cnf_nat_sum_p3 x y z : cnfp x -> cnfp y -> cnfp z ->
  (forall e, Vr z e = (Vr x e) +c  Vr y e) -> z = x +#f y.
Lemma cnf_nat_sum0 x : cnfp x ->x +#f \0f = x.
Lemma cnf_nat_sum0n x : cnfp x -> \0f +#f x = x.
 x +#f (y +#f z) = (x +#f y) +#f z.
```

We define here the natural sum of ordinal numbers.

```
Definition natural_sum x y :=  cnf_val ((the_cnf x) +#f (the_cnf y)).

Notation "x +#o y" := (natural_sum x y) (at level 50).
Lemma OS_natural_sum x y :
  ordinalp x -> ordinalp y -> ordinalp (x +#o y).
Lemma natural_sum0 x : ordinalp x -> (x +#o \0o) = x.
Lemma natural_sumC x y : (x +#o y) = (y +#o x).
Lemma natural_sumA a b c : ordinalp a -> ordinalp b -> ordinalp c ->
Lemma cnf_nat_sumA x y z: cnfp x -> cnfp y -> cnfp z ->
  a +#o (b +#o c) = (a +#o b) +#o c.
Lemma cnf_compare_nat_sum5 x y z: cnfp x -> cnfp y -> cnfp z ->
  (x +#f z <=f y +#f z <->  x <=f y).
```

Let's compare some numbers. Obviously $x \le x \oplus y$. Also $x + y \le x \oplus y$: consider the contribution of an exponent of x or y to the sum; since for every exponent e of x which is at least the degree of y the contributions are the same we get: if $d(y) \le v(x)$ then $x + y = x \oplus y$. Otherwise inequality is strict (since every monomial of x with exponent $< d(y)$ contributes to the natural sum but not the ordinal sum. Note that $x \oplus z \le y \oplus z$ is equivalent to $x \le y$

```
Lemma cnf_le_prop x y: cnfp x -> cnfp y -> (forall e, Vr x e <=c Vr y e) ->
   x <=f y.
Lemma cnf_compare_nat_sum1 x y: cnfp x -> cnfp y -> x <=f x +#f y.
Lemma cnf_compare_nat_sum2 x y: cnfp x -> cnfp y -> x +f y <=f x +#f y.
Lemma cnf_compare_nat_sum3 x y: cnfp_nz x -> cnfp_nz y ->
  cnf_degree y <=o oexp x \0c ->  x +f y = x +#f y.
Lemma cnf_compare_nat_sum4 x y: cnfp_nz x -> cnfp_nz y ->
   oexp x \0c <o cnf_degree y  ->  x +f y <f x +#f y.
Lemma cnf_compare_nat_sum5 x y z: cnfp x -> cnfp y -> cnfp z ->
  (x +#f z <=f y +#f z <->  x <=f y).
```

We restate the previous results for ordinal numbers. If x and y are $< \omega^e$ so is the natural sum (note that the degree is the sum is the maximum of the degrees of the arguments).

```
Lemma ord_compare_nat_sum1 x y: ordinalp x -> ordinalp y -> x <=o x +#o y.
Lemma ord_compare_nat_sum2 x y: ordinalp x -> ordinalp y -> x +o y <=o x +#o y.
Lemma ord_compare_nat_sum3 x y: \0o <o x -> \0o <o y ->
  odegree y <=o ovaluation x ->  x +o y = x +#o y.
Lemma ord_compare_nat_sum4 x y: \0o <o x -> \0o <o y ->
  ovaluation x <o odegree y  ->  x +o y <o x +#o y.
Lemma ord_compare_nat_sum5 x y z: ordinalp x -> ordinalp y -> ordinalp z ->
  (x +#o z <=o y +#o z <->  x <=o y).
Lemma natural_small x y e (v := oopow e):
  ordinalp e -> x <o v -> y <o v ->  (x +#o y) <o v.
```

If E is a finite set, c an integer, we consider the set A of all z obtained by sorting a subset t of $E\times c^+$ that satisfies the property that z will be a cnf. Obviously, A is finite, and its elements are cnfs. If z is a cnf, we consider the set $B(z)$ obtained by taking for E the set of its exponents and for c the greatest coefficient, If x and y are cnfs, we consider the set $B = B(x\oplus y)$. Then B is finite and $x\in B$.

Let x be an ordinal, r and exponent, $E = \omega^e$. Let F be the set of all pairs of ordinal (a,b) such that $a\oplus b = x$. This is a subset of $E\times E$. We pretend that it is finite and non-empty. Obviously $(x,0)\in F$. Let $B = B(x')$ where x' is the CNF of x, and let $C = \{s(t), t\in B\}$. Then B and C are finite sets. If $a\oplus b = x$, then the CNF of a belongs to B; so a belongs to C. By commutativity of addition b belongs to C. So F is a subset of the finite set $C\times C$.

```
Definition cnf_subset E c :=
  fun_image (Zo (\Po (E \times (csucc c))) cnf_rangep) cnf_sort.
Definition cnf_subset1 x :=
  cnf_subset (cnf_exponents x) (\csup (range (range x))).

Lemma cnf_subset_finite E c: finite_set E -> natp c ->
  finite_set (cnf_subset E c).
Lemma cnf_subset_prop E c x: finite_set E -> natp c ->
   inc x (cnf_subset E c) ->
   [/\ cnfp x, sub (cnf_exponents x) E & forall e, Vr x e <=c c].
Lemma cnf_subset_sum_prop x y (F :=  (cnf_subset1 (x +#f y))):
  cnfp x -> cnfp y -> (inc x F /\ finite_set F).
Lemma natural_finite_cover x e (E:= oopow e):
  ordinalp e -> inc x E ->
  let n := cardinal (Zo (coarse E) (fun z => (P z) +#o (Q z) = x)) in
  natp n /\ n <> \0c.
```

One deduces Proposition LXVII of Hessenberg [12]: for two ordinals α and β, if $\aleph_\alpha \ge \aleph_\beta$, then $\aleph_\alpha+\aleph_\beta = \aleph_\alpha\cdot\aleph_\beta = \aleph_\alpha$. In this framework, an aleph is just the cardinal of an infinite well-ordered set. The proof does not require the axiom of choice: the cardinal of a well-ordered set is the least ordinal equipotent to it. So we consider two infinite well-ordered sets M and N, and their cardinals μ and ν.

We shall denote by $A+B$ the ordinal sum, by $A+_s B$ the disjoint union, by $A\cdot B$ the ordinal product, by $A\times B$ the cardinal product, and by $A\sim B$ the property that the sets are equipotent. We have obviously $A+B\sim A+_s B$ and $A\cdot B\sim A\times B$, In particular $A+B\sim B+A$ and $A\cdot B\sim B\cdot A$. Moreover if $A\sim B$ then $X\cdot A\sim X\cdot B$.

(a) If X is an infinite countable set, whose elements are finite and non-empty, then $\bigcup X$ is infinite countable [Proposition XV]. Proof. Let $i\mapsto X_i$ be a bijection $\mathbf{N}\to X$; write each X_i as the set of all $x_i^{(k)}$ where $k\le n_i$. Let m_i be the sum of the n_j+1 for $j<i$. Define the index of $x_i^{(k)}$ to be m_i+k. This is a bijection $\bigcup X\to\mathbf{N}$.

(b) If I is a limit ordinal, α_i a finite non-zero ordinal, then $\sum_{i\in I}\alpha_i = I$ [Proposition LV]. Proof. Since I is limit, it has the form $\omega\cdot\xi$, so is the sum of ξ copies of ω. By associativity, $\sum_{i\in I}\alpha_i$ is a double sum $\sum_{j\in\xi}\sum_{i\in Y_j}\alpha_i$. By (a), the inner sum is ω since each Y_j is isomorphic to ω. We are left with $\sum_{j\in\xi}\omega = \omega\cdot\xi = I$.

(c) Let α be an infinite ordinal, and h the greatest indecomposable ordinal $\le a$. Then $\alpha\sim h$ [proposition LVII]. Write $\alpha = \omega\cdot\beta+n$ where n is finite by Euclidean division. We have $\alpha\sim n+\omega\cdot\beta$, hence $\alpha\sim\omega\cdot\beta$. It follows $\omega\cdot\alpha\sim\omega\cdot(\omega\cdot\beta)$. By associativity of multiplication and the well-know relation $\omega\cdot\omega\sim\omega$ it follows $\omega\cdot\alpha\sim\omega\cdot\beta$, hence $\omega\cdot\alpha\sim\alpha$ and finally $\alpha\cdot\omega\sim\alpha$. The same relation holds for h as well. The conclusion follows from $h\cdot\omega = \alpha\cdot\omega$.

(d) $\mu+\nu \sim \mu \sharp \nu$ [proposition LXV]. Write both arguments as sums of powers of ω. There is a finite sequence of (well-ordered) sets E_i such that $\mu+\nu$ and $\mu \sharp \nu$ are the ordinal sums of these sets (in a possible different order). These two sums are equipotent.

(e) Let m and n be the degree of μ and ν. By (c), $\mu \sim \omega^m$ and $\nu \sim \omega^k$. Let p be the greatest of m and n. This is the degree of both $\mu+\nu$ and $\mu \sharp \nu$. The relation $\mu \ge \nu$ implies $m \ge n$, thus $p = m$. In this case $\mu+\nu \sim \mu$ and $\mu \sharp \nu \sim \mu$. [note: if μ and ν are cardinal, they are indecomposable ordinals, so that $\mu \ge \nu$ is equivalent to $m \ge n$].

(f) If $\alpha < \mu$ and $\beta < \nu$ then $\alpha \sharp \beta < \mu \sharp \nu$. The first relation says that we can write $\alpha = x+\alpha'$ and $\mu = x+\mu'$ where $\deg(\alpha') < \deg(\mu')$ (it may happen that $\alpha' = 0$, case where its degree is zero). Write $\beta = y+\beta'$ and $\nu = y+\nu'$, so that $\alpha \sharp \beta = (x \sharp y) \sharp (\alpha' \sharp \beta')$ and $\mu \sharp \nu = (x \sharp y) \sharp (\mu' \sharp \nu')$. Now $\deg(\alpha' \sharp \beta') = \max(\deg(\alpha'), \deg(\beta')) < \max(\deg(\mu'), \deg(\nu')) = \deg(\mu' \sharp \nu')$.

The proof is now as follows. We assume that μ and ν are two infinite ordinals. We let L_γ be the set of pairs $(\alpha,\beta) \in \mu\times\nu$ with $\alpha \sharp \beta = \gamma$, and S the set of all γ of this form. Each L_γ is finite (the previous lemma, Proposition LXVI), in particular is well-ordered, while S being a set of ordinals is well-ordered (Proposition XXXII). The ordinal $\sum_{\gamma_i n S} L_\gamma$ is denoted $\mu\times\nu$ by Hessenberg (it is not the ordinal product; it is however equipotent to the cartesian product, since it is the ordinal of a well-ordered set E, which is, by definition, the disjoint union of the L_γ; now these sets are mutually disjoint so that E is equipotent to the union, which is, by definition, the cartesian product). In order to avoid any confusion, we write $\mu \times_s \nu$ for the cartesian product and state $\mu\times\nu \sim \mu\times_s \nu$.

Let σ be the ordinal of S; write it as the sum of a limit ordinal and an integer. So $\sum L_\gamma$ becomes the sum of two sums. The second sum is indexed by some integer k; as each term is finite and ≥ 1, the sum is finite and $\ge k$. By (b), the first sum is equal to the index set. All in all, $\sum L_\gamma$ is σ plus an integer, say $\mu\times\nu = \sigma+\kappa$. Now (Proposition XVIII), the cardinal of an infinite set is left unchanged when a finite number of elements is added. This means that $\mu\times\nu \sim \sigma$.

By (f) every element γ of S is less than $\mu \sharp \nu$. It follows $\mathrm{card}(\sigma) \le \mathrm{card}(\mu \sharp \nu)$ and by (d) $\mathrm{card}(\sigma) \le \mathrm{card}(\mu+\nu)$. So $\mathrm{card}(\mu\times\nu) \le \mathrm{card}(\mu+\nu)$. The converse inequality is rather obvious (it suffices that both μ and ν are ≥ 2). So $\mathrm{card}(\mu\times\nu) = \mathrm{card}(\mu+\nu)$. By (e), this is the largest of the two cardinals $\mathrm{card}(\mu)$ and $\mathrm{card}(\nu)$.

11.14.2 Double Induction Principle

We prove in this section that there is a unique solution f to

$$f(a,b) = T(\{(x,y,f(x,y)) \text{ such that } (x,y) < (a,b)\}). \tag{11.60}$$

Here the right hand side is obtained by applying some operator T to the set of values of f on a set E; here a and b are arbitrary ordinal numbers. We first prove that the equation has a solution which is a *function* (whose source is a set of pairs of ordinals), see details below.

Define

$$F_{ab} = \{a\}\times b \cup a\times\{b\}, \qquad E_{ab} = F_{ab} \cup a\times b.$$

In the set of all (x,y) such that $x \le a$ and $y \le b$, E_{ab} (respectively F_{ab}) is the set of all pairs for which at least one (resp. exactly one) inequality is strict. In case $a \in A$, $b \in B$, where A and B are ordinals, then $E_{ab} \subset A\times B$.

```
Definition doubleI_E a b :=
  (singleton a \times b) \cup (a \times singleton b) \cup (a\times b).
Definition doubleI_F a b :=
```

```
  (singleton a \times b) \cup (a \times singleton b).

Lemma doubleI_E_pair a b x: inc x (doubleI_E a b) -> pairp x.
Lemma doubleI_E_P1 a b: ordinalp a -> ordinalp b ->
  forall x y, inc (J x y)  (doubleI_E a b) <->
     [\/ x = a /\ y <o b, x <o a /\ y = b | x <o a /\ y <o b].
Lemma doubleI_E_P2 a b: ordinalp a -> ordinalp b ->
  forall x y, inc (J x y)  (doubleI_E a b) <->
     [/\ x <=o a, y <=o b & (x <> a \/ y <> b)].
Lemma doubleI_E_sub A B a b:
  ordinalp A -> ordinalp B -> inc a A -> inc b B ->
  sub (doubleI_E a b) (A \times B).
```

In order to simplify notations, we restate our principle of transfinite induction on a well-ordered set, in the special case where this set is the well-ordering of an ordinal.

```
Definition otrans_def a (T: fterm) f :=
  [/\ surjection f, source f = a &
                   forall x, x <o a ->  Vf f x = T (restriction1 f x)].
Definition otrans_defined a := transfinite_defined (ordinal_o a).

Lemma otrans_def_prop a T f: ordinalp a ->
  (otrans_def a T f  <-> transfinite_def (ordinal_o a) T f).
Lemma otrans_pr a f T:
   ordinalp a -> otrans_def a T f -> otrans_defined a T = f.
Lemma otrans_defined_pr a T: ordinalp a ->
  otrans_def a T (otrans_defined a T).
```

We shall denote by $\mathrm{R}(f,\mathrm{E})$ the surjective restriction of a function f to a set E (this is `restriction1`). By abuse of notations, if f is a functional term, $\mathrm{R}(f,\mathrm{E})$ will be the surjective function defined on E that maps x to $f(x)$. This will be denoted by `Lfs` in COQ. By another abuse of notations, if f is a functional term of two variables, $\mathrm{R}(f,\mathrm{E})$ will be the surjective function defined on E that maps a pair (a,b) to $f(a,b)$.

```
Definition Lfs (f: fterm) s := Lf f s (fun_image s f).

Lemma Lfs_surjective f s: surjection (Lfs f s).
Lemma Lfs_V f s x : inc x s -> Vf (Lfs f s) x = f x.
Lemma Lfs_source f s :source (Lfs f s) = s.
Lemma Lfs_restriction1 f x: function f -> sub x (source f) ->
   restriction1 f x = Lfs (Vf f) x.
Lemma Lfs_exten f1 f2 x: (forall t, inc t x -> f1 t = f2 t) ->
   Lfs f1 x  = Lfs f2 x.
```

Equation (11.60) has the form $f(a,b) = \mathrm{T}(\mathrm{X})$, where X is a set canonically isomorphic to $\mathrm{R}(f,\mathrm{E}_{ab})$. We allow T to depend on a and b, so that our fix-point equation becomes

$$f(a,b) = \mathrm{T}(a,b,\mathrm{R}(f,\mathrm{E}_{ab})) \qquad \text{fo all } a,b \text{ ordinal.} \tag{11.61}$$

```
Definition fterm3 := Set -> Set -> Set -> Set.
Notation "f =2o g" := (forall x y, ordinalp x -> ordinalp y -> f x  y = g x y)
  (at level 70, format "'[hv' f '/ '  =2o  g ']'", no associativity).

Definition doubleI_restr (f:fterm2) a b :=
```

```
  Lfs (fun p => f (P p) (Q p)) (doubleI_E a b).
Definition doubleI_fix1 (T: fterm3) (f:fterm2) :=
  f =2o fun a b => T a b (doubleI_restr f a b).
```

We consider equation (11.61), where f is a surjective function on $A \times B$, and T takes two arguments (the first being the pair (a, b). Uniqueness is easy (by a double induction, first over a, then over b).

```
Definition doubleI_def A B (T: fterm2 ) f :=
  [/\ surjection f,  source f = A \times B &
   forall p, inc p (A\times B) ->
            Vf f p = T p (restriction1 f (doubleI_E (P p) (Q p)))].

Lemma doubleI_unique A B T f1 f2: ordinalp A -> ordinalp B ->
  doubleI_def A B T f1 ->  doubleI_def A B T f2 ->
  f1 = f2.
```

Existence is easy as well. The trick is to define by transfinite induction a function of one variable whose value is a function defined by transfinite induction, and consider this as a function of two variable, see the definition below.

Let f be the function defined by transfinite induction on A, and f' the associated surjective function with two variables: if $p = (a, b)$ then $f'(p) = f(a)(b)$. Now $f(a)(b)$ is the function obtained by applying some T_5 to the restriction of $f(a)$ to b; this procedure takes as first argument the restriction of f to a. Given these two arguments, the procedure deduces a, b (hence p) and a function h defined on E_{ab} in such a way that h is the restriction of f' to E_{ab}. The procedure returns $T(p, h)$.

```
Definition doubleI_transdef A B (T: fterm2) :=
  let T1 a f g x y := Yo (x <o a) (Vf (Vf f x) y) (Vf g y) in
  let T2 a f g p := T1 a f g (P p) (Q p) in
  let T4 a b f g := Lfs (T2 a f g) (doubleI_E a b)  in
  let T5 fa fb := T (J (source fa) (source fb))
                  (T4 (source fa) (source fb) fa fb) in
  let T6 fa := otrans_defined B (T5 fa) in
  let f := otrans_defined A T6 in
  Lfs (fun p => Vf (Vf f (P p)) (Q p)) (A\times B).

Lemma doubleI_correct A B T : ordinalp A -> ordinalp B ->
  doubleI_def A B T  (doubleI_transdef A B T).
```

By uniqueness, if we have a bigger pair of sets, we get a second function that extends the first.

```
Lemma doubleI_unique2 T A1 A2 B1 B2 f: A1 <=o A2 -> B1 <=o B2 ->
  doubleI_def A2 B2 T f ->
  doubleI_def A1 B1 T (restriction1 f (A1 \times B1)).
Lemma doubleI_unique3 T A1 A2 B1 B2 f g: A1 <=o A2 -> B1 <=o B2 ->
  doubleI_def A2 B2 T f -> doubleI_def A1 B1 T g ->
  g = (restriction1 f (A1 \times B1)).
```

Let f_{AB} be the function defined above. If $a \in A$ and $b \in B$, then $f_{AB}(a, b)$ is independent of A and B. We denoted it by $f(a, b))$ [concretely, if the define of $f(a, b)$ we tale $A = a^+$ and $B = b^+$). By uniqueness we get a solution to (11.61),

```
Definition doubleI_tdef (T: fterm3) a b :=
  let T' := fun p f => T (P p) (Q p) f in
  Vf (doubleI_transdef (osucc a) (osucc b) T') (J a b).

Lemma doubleI_tdef_rec T:  doubleI_fix1 T (doubleI_tdef T).
Lemma doubleI_tdef_unique T f g :
  doubleI_fix1 T f ->  doubleI_fix1 T g ->
  forall  a b,  ordinalp a -> ordinalp b -> f a b = g a b.
```

Variant. We replace E_{ab} by F_{ab}. The trick is that $T(R(f, F_{ab}))$ is some $T'(R(f, E_{ab}))$.

```
Definition doubleI_restr2 (f:fterm2) a b :=
  Lfs  (fun p => f (P p) (Q p)) (doubleI_F a b).
Definition doubleI_tdef2 (T: fterm3) :=
  doubleI_tdef (fun a b F => T a b (restriction1 F (doubleI_F a b))).
Definition doubleI_fix2 (T: fterm3) (f:fterm2) :=
  f =2o fun a b =>  T a b (doubleI_restr2 f a b).

Lemma doubleI_tdef_rec2 T: doubleI_fix2 T (doubleI_tdef2 T).
Lemma doubleI_tdef2_unique T f g :
  doubleI_fix2 T f ->  doubleI_fix2 T g -> f =2o g.
```

11.14.3 More about natural sums

The objective of this section is to give an alternative definition of Hessenberg's natural sum via a double induction. We first introduce the notion of *least strict upper bound.* If E is an ordered set, A a subset of E, the lsub, if it exists, is the least element of the set B of all $x \in E$ such that, whenever $y \in A$ we have $y < x$. Assume that A has a least upper bound z. In case $z \notin A$, then z is the lsub of A. In case $z \in A$, then B is the set of all x such that $z < x$. We get a similar definition when E is replaced by the collection of ordinal numbers; here neither E nor B are sets, but z always exists. If $z \in A$, the lsub is obviously the successor of z.

```
Definition osups X:= let x := \osup X in Yo (inc x X) (osucc x) x.

Lemma OS_sups X: ordinal_set X -> ordinalp (osups X).
Lemma osup_ub X x: ordinal_set X -> inc x X ->  x <o osups X.
Lemma osups_least X x: ordinalp x -> (forall y, inc y X -> y <o x) ->
  osups X <=o x.
Lemma osups_max X x: ordinalp x -> (forall y, inc y X -> y <=o x) -> inc x X ->
  osups X = osucc x.
Lemma osups_set0: osups emptyset = \0o.
Lemma osups_ordinal a : ordinalp a -> osups a = a.
```

According to [8], we say that an operation $\oplus$ is a *natural sum* if, for every ordinals α, β, δ:

(0) $\alpha \oplus \beta$ is an ordinal

(1) $\alpha \oplus \beta = \beta \oplus \alpha$

(2) $(\alpha \oplus \beta) \oplus \delta = \alpha \oplus (\beta \oplus \delta)$

(3) $\alpha \oplus 0 = \alpha$

(4) $\delta \oplus \alpha > \delta \oplus \beta$ if and only if $\alpha > \beta$.

```
Definition is_natural_sum (f: fterm2) :=
 [/\ forall a b, ordinalp a -> ordinalp b -> ordinalp (f a b),
     forall a b, ordinalp a -> ordinalp b ->  (f a b) = f b a,
     forall a b c, ordinalp a -> ordinalp b -> ordinalp c ->
                  f a (f b c) = f (f a b) c,
     forall a, ordinalp a -> f a \0o = a &
     forall a b c, ordinalp a -> ordinalp b -> ordinalp c ->
          (f c a <=o f c b <->  a <=o b)].
```

Let $\sigma(\alpha,\beta)$ denote the natural sum defined by Hessenberg (and studied above). It is the unique natural sum satisfying

$$\omega^\alpha \cdot m + \omega^\beta \cdot n = \sigma(\omega^\alpha \cdot m, \omega^\beta \cdot n), \qquad (\alpha \ge \beta, m \in \mathbf{N}, n \in \mathbf{N}).$$

Uniqueness is not obvious; the remaining part of the claim should be straightforward but is not yet implemented.

```
Lemma is_natural_sum_nat: is_natural_sum natural_sum
```

In what follows, S will denote the term defined by double induction via the least strict upper bound of the values on F_{ab}.

```
Section OsumS.
Let S := doubleI_tdef2 (fun _ _ f => (osups (target f))).
```

So, S is the unique solution of

$$\mathrm{S}(a,b) = \sup{}^+(\mathrm{F}^l_{ab} \cup \mathrm{F}^r_{ab}), \qquad \mathrm{F}^l_{ab} = \{\mathrm{S}(x,b), x \in a\} \quad \mathrm{F}^r_{ab} = \{\mathrm{S}(a,y), y \in b\}.$$

We deduce (by double induction), that $\mathrm{S}(a,b)$ is an ordinal and $\mathrm{S}(a,b) = \mathrm{S}(b,a)$. If $z < \mathrm{S}(a,b)$ the either $z \le \mathrm{S}(x,b)$ for some $x < a$ or $z \le \mathrm{S}(a,y)$ for some $y < b$. .

```
Definition doubleI_rg f a b :=
   fun_image a (f ^~ b) \cup fun_image b (f a).
Lemma doubleI_rgP f a b x:
  inc x (doubleI_rg f a b) <->
     ((exists2 y, inc y b & x = f a y) \/ (exists2 y, inc y a & x = f y b)).


Lemma nsv_rec: S =2o fun a b => osups (doubleI_r S a b).
Lemma nsv_lt_prop z a b: ordinalp a -> ordinalp b -> z <o S a b ->
   (exists2 x, x <o a & z <=o S x b) \/ (exists2 x, x <o b & z <=o S a  x).
Lemma OS_nsv a b: ordinalp a -> ordinalp b -> ordinalp(S a b).
Lemma nsvC a b: ordinalp a -> ordinalp b -> (S a b) = (S b a).
Lemma nsv_lt_prop z a b: ordinalp a -> ordinalp b -> z <o S a b ->
   (exists2 x, x <o a & z <=o S x b) \/ (exists2 x, x <o b & z <=o S a  x).
Lemma nsv_le_prop z a b: ordinalp a -> ordinalp b -> ordinalp z ->
  (forall x, x <o a -> S x b <o z) ->
  (forall x, x <o b -> S a x <o z) ->
  S a b <=o z.
```

We first show that S is a natural sum. Relations (0) and (1) have already been prove, (4) is trivial, it says that S is strictly increasing. Relation (3) says $\mathrm{S}5a,0) = a$. (with the above notations, F^l_{ab} is empty, and by induction $\mathrm{F}^r_{ab} = b$). Associativity holds by a triple induction. Let's show $S(a,S(b,c)) = S(S(a,b),c)$. Let's show $S(a,S(b,c)) \le S(S(a,b),c)$ (the other inequality can be proved in a similar way). Let r be the RHS. Assume $z < a$, Then $S(z,S(b,c)) = S(S(z,b),c)$ by induction and the result follows by monotonicity. It suffices to show that if $z < Sb,c)$ then $S(a,z) < r$. However $z < S(x,c)$ or $z < S(b,y)$ for some x or y, so $S(a,z) \le S(a,S(x,c))$ or $S(a,z) \le S(a,S(b,y))$; By induction so $S(a,z) \le S(S(a,x),c)$ or $S(a,z) \le SS((a,b),y)$; the result follows by monotonicity.

```
Lemma nsv_00: S \0o \0o = \0o.
Lemma nsv_n0 a: ordinalp a -> S a \0o = a.
Lemma nsv_0n a: ordinalp a -> S \0o a = a.
Lemma nsv_ltl a1 a2 b: a1 <o a2 -> ordinalp b -> (S a1 b) <o (S a2 b).
Lemma nsv_ltr a b1 b2: ordinalp a -> b1 <o b2 -> (S a b1) <o (S a b2).
Lemma nsv_lel a1 a2 b: a1 <=o a2 -> ordinalp b -> (S a1 b) <=o (S a2 b).
Lemma nsv_ler a b1 b2: ordinalp a -> b1 <=o b2 -> (S a b1) <=o (S a b2).
Lemma nsvA a b c: ordinalp a -> ordinalp b -> ordinalp c ->
   S a (S b c) = S (S a b) c.
Lemma nsvACA a b c d: ordinalp a -> ordinalp b -> ordinalp c -> ordinalp d ->
  S (S a b) (S c d) = S (S a c) (S b d).
Lemma nsv_natsum: is_natural_sum S.
```

We have $S(a^+,b) = S(a,b)^+$. Proof by induction on b. Let $c = S(a^+,b)$. Then $c > S(a,b)$, hence $c \ge S(a,b)^+$. It suffices to show that $S(a,b)$ is an upper bound of $F^l \cup F^r$ (with the previous notations, with a replaced by a^+) i.e. $S(x,b) \le S(a,b)$ for $x \le a$ (true by monotonicity) and $S(a^+,x) \le S(a,b)$ for $x < b$ (true by induction and monotonicity). It follows $a+b \le S(a,b)$ (by induction on b since addition is normal). By induction on b, we have $S5a,b) = a+b$ when n is an integer.

```
Lemma nsv_Sn a b: ordinalp a -> ordinalp b -> S (osucc a) b = osucc (S a b).
Lemma nsv_nS a b: ordinalp a -> ordinalp b ->  S a (osucc b) = osucc (S a b).
Lemma nsv_nat a b: ordinalp a -> b <o omega0 -> S a b = a +o b.
Lemma nsv_ge_sum a b: ordinalp a -> ordinalp b -> a +o b <=o (S a b).
```

We pretend that

$$S(\omega^p \cdot a + x, \omega^p \cdot b + y) = \omega^p \cdot (a+b) + S(x,y), \qquad (a,b<\omega, \quad x,y<\omega^p) \tag{11.62}$$

and

$$S(x,y) < \omega^p \qquad (x,y<\omega^p). \tag{11.63}$$

Consider four assertions: $H_a(p)$ is (11.62), $H_b(p)$ is (11.63), $H_c(a,y)$ is $H_a(p)$ with $x = 0$ and $b = 0$, and $H_d(a,b)$ is $H_a(p)$ with $x = y = 0$.

Step one: $\forall q, q \le p \implies H_a(q)$ implies $H_b(p)$. One deduces (after proving that H_a holds for every p), that H_b holds also for every p. The proof is by induction on p. We have to show that $S(x,y)$ is small when both arguments are small; this is trivial when one argument is zero, so we may assume them non-zero. In particular each argument has a degree, let d be the greatest of them. We can write $x = \omega^d \cdot a + x'$ and $y = \omega^d \cdot b + y'$, where a and b are integers, x' and y' are $< \omega^d$, and $d < p$. By $H_a(d)$ we have $S(x,y) = \omega^d \cdot (a+b) + S(x',y')$. By H_b we have $S(x',y') < \omega^d$, so $S(x,y) < \omega^d \cdot (a+b+1) < \omega^{d+1}$.

We now prove the result by induction on p. So we assume $\forall q, q \le p \implies H_a(q)$. In particular $H_b(p)$ holds. Both H_c and H_d are of the form $S(u,v) = u+v$. We know that $S(u,v) \ge u+v$, so it suffices to show (l): $S(z,v) < u+v$ for $z < u$ and (r): $S(u,z) < u+v$, for $z < v$.

Step two: $\mathrm{H}_c(a,y)$ holds. Point (r) holds by induction on y. Point (l) by induction on a. It suffices to show that $x < \omega^p \cdot a$ implies $\mathrm{S}(x,y) < \omega^p \cdot a$. The result is obvious for $a = 0$. Set $v = \omega^p \cdot (a-1)$. The case $x < v$ follows from $x = v$. Now we have $x = v + r$, where r is small, and by induction $x = \mathrm{S}(v,r)$, so $\mathrm{S}(x,y) = \mathrm{S}(\mathrm{S}(v,r),y) = \mathrm{S}(v,\mathrm{S}(r,y))$ by associativity. By step one, $\mathrm{S}(r,y)$ is small, so $\mathrm{S}(x,y) = v + \mathrm{S}(r,y) < v + \omega^p = \omega^p \cdot a$.

Step three: $\mathrm{H}_d(a,b)$ holds. Set $u = \omega^p \cdot a$ and $v = \omega^p \cdot b$. Cases (l) and (r) are similar, so let's show (l), by induction on a. We must show that $x < u$ implies $\mathrm{S}(x,v) < u + v$. Write $x = \omega^p \cdot c + r = w + r$, where r is small and $c < a$. We have $\mathrm{S}(x,v) = \mathrm{S}(\mathrm{S}(w,r),v) = \mathrm{S}(\mathrm{S}(v,w),r)$. By induction, this is $\mathrm{S}(v+w;r) = v + w + r < \omega^o \cdot (b+c+1)$

Step four.: H_a holds. Use H_c twice, associativity and commutativity. The LHS becomes $\mathrm{S}(u,v)$ where $u = \mathrm{S}(\omega^p \cdot a, \omega^p \cdot b)$ and $v = \mathrm{S}(x,y)$. Simplify u via H_d, note that v is small, apply H_c again.

```
Lemma nsv_aux u d: \0o <o u -> odegree u <=o d -> exists c r,
  [/\ c <o omega0, r <o oopow d   & u = oopow d *o c +o r].
Lemma nsv_cantor_p1 p: ordinalp p ->
  (forall q, q <o p ->
   (forall a b x y, a <o omega0 -> b <o omega0 ->
    x <o oopow q -> y <o oopow q ->
  S (oopow q *o a +o x) (oopow q *o b +o y) = oopow q *o (a +o b) +o S x y)) ->
  forall x y,  x <o oopow p -> y <o oopow p ->
  S x y <o oopow p.
Lemma nsv_cantor_p2 p: ordinalp p -> (* 81 *)
  forall a b x y, a <o omega0 -> b <o omega0->
  x <o oopow p -> y <o oopow p ->
  S (oopow p *o a +o x) (oopow p *o b +o y) = oopow p *o (a +o b) +o S x y.
Lemma nsv_cantor_p3 p x y : ordinalp p -> x <o oopow p -> y <o oopow p ->
  S x y <o oopow p.
```

More generally

$$\mathrm{S}(\omega^p \cdot a + x, \omega^p \cdot b + y) = \omega^p \cdot \mathrm{S}(a,b) + \mathrm{S}(x,y), \qquad (x,y < \omega^p) \tag{11.64}$$

holds whatever a and b. Same method as above. We first generalize now H_c as $\mathrm{S}(\omega^p \cdot a, y) = \omega^p \cdot a + y$ for any ordinal a; this is easy. We then generalize H_d in the form $\mathrm{S}(\omega^p \cdot a, \omega^p \cdot b) = \omega^p \cdot \mathrm{S}(a,b)$; the result follows easily. The proof of H_d is by induction on the maximum of the degrees of a and b. Write $a = \omega^d \cdot a' + x$ and $b = \omega^d \cdot b' + y$. We have $\omega^p \cdot a' = \omega^{p+d} \cdot a' + \omega^p \cdot x$. By H_c this is $\mathrm{S}(\omega^{p+d} \cdot a', \omega^p \cdot x)$. Now, the LHS is $\mathrm{S}(\mathrm{S}(u,v),\mathrm{S}(u',v')) = \mathrm{S}(\mathrm{S}(u,u'),\mathrm{S}(v,v')) = \mathrm{S}(u'',v'')$. Obviously $\mathrm{S}(u,v) = \omega^{p+d} \cdot (a' + b')$ and $\mathrm{S}(v,v')$ can be reduced by induction to $\omega^p \cdot \mathrm{S}(x,y)$. It follows $\mathrm{S}5u'', v'') = u'' + v''$ and we can factor out ω^p. So the LHS becomes $\omega^p \cdot (\omega^d \cdot (a + a') + \mathrm{S}(x,y))$. The second factor is $\mathrm{S}(a,b)$ (by the same computations, without induction).

```
Lemma nsv_cantor_p4 p a y:  ordinalp p -> ordinalp a -> y <o oopow p ->
  S (oopow p *o a) y =  oopow p *o a +o y.
Lemma nsv_cantor_p5 p a b: ordinalp p -> ordinalp a -> ordinalp b ->
  S (oopow p *o a) (oopow p *o b) = oopow p *o S a b.  (* 59 *)
Lemma nsv_cantor_p6 p a b x y: ordinalp p -> ordinalp a -> ordinalp b ->
   x <o oopow p -> y <o oopow p ->
   S (oopow p *o a +o x) (oopow p *o b +o y) =
   oopow p *o (S a b) +o S x y.
```

Note that $S(a+n, b+m) = S(a,b)+(m+n)$, whenever m and n are integers (by induction on these integers) and

$$S(a,b) = \sup(F^l_{ab} \cup F^r_{ab}), \qquad (a, b \text{ limit})$$

We deduce $S(\omega, n) = \omega + n$ for every integer n, hence $S(\omega,\omega) = \omega+\omega$. More generally $S(a,\omega) = a+\omega$ when a is a limit ordinal. The non-trivial point is to show that $y < a$ implies $S(y,\omega) < a+\omega$. Write $a = \omega \cdot c$ and $y = \omega \cdot d + e$. The proof is by induction on c. We have $d < c$, hence $S(y,\omega) = S(\omega \cdot d, \omega) + e = \omega \cdot (d+1) + e$.

```
Lemma nsv_with_nat a b n m: ordinalp a -> ordinalp b -> natp n -> natp m ->
  S (a +o n) (b +o m) = (S a b) +o (n +o m).
Lemma nsv_limit a b: limit_ordinal a -> limit_ordinal b ->
  S a b = \osup (doubleI_rg S a b).
Lemma nsv_with_nat1 a n: ordinalp a -> natp n -> S a n =  a +o n.
Lemma nsv_oo: S omega0 omega0 = omega0 +o omega0.
Lemma nsv_olim a: limit_ordinal a -> S a omega0 = a +o omega0.
```

[Needs to be completed]

11.15 Numbers equal to their degree

In section §20 of [7], Cantor considers the following question: can an ordinal x of the second class be equal to its degree? Recall that "second class" means infinite and countable; in what follows we consider arbitrary ordinals. Write $x = \omega^a \cdot b + c$, where a is the degree of x, b its associated coefficient, and c the remainder. We have obviously $a \le x$, and if equality holds, we have $c = 0$ then $b = 1$, thus

$$x = \omega^x. \tag{11.65}$$

Such numbers are called ϵ-ordinals by Cantor; they are obviously infinite.

```
Definition epsilonp x := ordinalp x /\ oopow x = x.
Lemma ord_eps_p1 x: ordinalp x -> x <=o (oopow x).
Lemma ord_eps_p2 a b c (x := ((oopow a) *o b) +o c) :
  ordinalp a ->  \0o <o b -> ordinalp c ->
  (a <=o x /\ (x = a -> [/\ c = \0o, b = \1o & epsilonp x])).
Lemma ord_eps_p3 x: epsilonp x -> omega0 <o x.
Lemma expilon_prop x: \0o <o x ->(odegree x = x <-> epsilonp x).
Lemma ord_eps_p4: iclosed_collection epsilonp.
```

Let $f(x) = \omega^x$. Then f is a normal OFS, and an ϵ-number is a fixed-point of f. Let $E(x)$ be the next fix-point of f after x. This is defined by

$$x_0 = x, \quad x_{n+1} = f(x_n), \quad E(x) = \sup_{i\in\mathbf{N}} x_i. \tag{11.66}$$

(Cantor considers the supremum of x_1, x_2, etc, but the supremum is the same, as $x_0 \le x_1$). Any infinite cardinal successor is stable by E; in particular, if x is countable so is $E(x)$. Cantor says [7, §20, Theorem A], that if x is a number of the first or second class, then $E(x)$ is an ϵ-number of the second class (note that $E(x)$ is infinite). He also says that if x is not an ϵ-number, then the sequence x_i is strictly increasing (verification left to the reader). Theorem B says that $E(1)$ is the least ϵ-number.

```
Definition epsilon_fct := the_least_fixedpoint_ge oopow
Definition epsilon_fam := first_derivation oopow.

Lemma epsilon_fct_pr0 x: ordinalp x ->
  least_fixedpoint_ge oopow x (epsilon_fct x).
Lemma epsilon_fct_pr1 x: ordinalp x ->
  epsilon_pr1 (epsilon_fct x).
Lemma epsilon_fct_pr2 C (E:= cnext C): infinite_c C ->
  (forall x, inc x E -> inc (epsilon_fct x) E).
Lemma epsilon_fct_pr3 (e:= epsilon_fct \1o ) :
  epsilonp e /\
  forall x, epsilonp x -> e <=o x.
```

In a previous version of Gaia, we defined ϵ_x by transfinite induction via

$$\epsilon_0 = \mathrm{E}(1), \qquad \epsilon_x = \sup_{y<x} \mathrm{E}(\epsilon_y + 1). \tag{11.67}$$

Here we define ϵ_x as the first derivation of $f(x) = \omega^x$ (this is Corollary 4 of Theorem 4 of [22]). Then we have $\epsilon_0 = \mathrm{E}(0)$ and $\epsilon_{x+1} = \mathrm{E}(\epsilon_x + 1)$ (this is Theorem D of Cantor). Since zero is not an ϵ-number we also have $\epsilon_0 = \mathrm{E}(1)$ (theorem B again). Theorem C says: if ϵ' and ϵ'' are consecutive ϵ-ordinals, and $\epsilon' < \gamma < \epsilon''$, then $\mathrm{E}(\gamma) = \epsilon''$. We restate this as: whatever x, $\epsilon_x < \gamma \le \epsilon_{x+1}$ implies $\mathrm{E}(\gamma) = \epsilon_{x+1}$. Theorem E can be restated as: the collection of ϵ-ordinals is closed and ϵ_x is a normals OFS.

```
Lemma epsilon_fam_pr0 x: ordinalp x ->
  epsilonp (epsilon_fam x).
Lemma epsilon_fam_pr1 y: epsilonp y ->
  exists2 x, ordinalp x & y = epsilon_fam x.
Lemma epsilon_fam_pr2: epsilon_fam \0o = epsilon_fct \1o.
Lemma epsilon_fam_pr2': epsilon_fam \0o = epsilon_fct \0o.
Lemma epsilon_fam_pr2'': epsilon_fam \0o = epsilon_fct (osucc omega0).
Lemma epsilon_fam_pr3 x y: ordinalp x ->
  epsilon_fam x <o y -> y <=o epsilon_fam (osucc_o x) ->
  epsilon_fct y =  epsilon_fam (osucc x).
Lemma epsilon_fam_pr4 x: ordinalp x ->
  epsilon_fam (osucc x) = epsilon_fct (osucc (epsilon_fam x)).
Lemma epsilon_fam_pr5: normal_ofs epsilon_fam.
```

11.15.1 Range of epsilon

Being a normal OFS, the range of ϵ_x is a proper subclass of the ordinals. More precisely, if E is an infinite cardinal successor, F the set of ϵ-ordinals in E, then $x \mapsto \epsilon_x$ is an order-isomorphism $\mathrm{E} \to \mathrm{F}$. In particular, E and F has the same cardinal. If one of x or ϵ_x is countable, so is the other.

```
Lemma epsilon_fam_pr6 C (E := cnext C) (F:= Zo E epsilonp) :
  infinite_c C ->
  order_isomorphism (Lf epsilon_fam E F) (ordinal_o E) (ole_on F).
Lemma epsilon_fam_pr7 C (E := cnext C)
   (F:= Zo E epsilonp) : infinite_c C -> cardinal E = cardinal F.
Lemma countable_epsilon x: ordinalp x ->
  (countable_ordinal x <-> countable_ordinal (epsilon_fam x)).
```

Theorem F of Cantor says: the set T of ϵ-numbers of the second class is well-ordered; its ordinal is Ω, its cardinal is aleph-one.

Let's denote by N the first class of ordinals, by Z the second class, by F the union. Then $x \in \mathrm{N}$ when x is a finite ordinal, $x \in \mathrm{Z}$ when x is an infinite countable ordinal, and $x \in \mathrm{F}$ when x is a countable ordinal. One property of F is that the supremum of a countable subset of F is in F; in particular F is uncountable, and the set of ϵ-numbers of the second class (this is the set of all ϵ numbers in F) is order-isomorphic to F. Let (F) this property; let's explain the connection between (F) and Theorem F.

First, Cantor shows that Z is uncountable: let $(\gamma_i)_i$ be a countable sequence of elements of Z. One can extract a strictly increasing subsequence (there is no greatest element in the sequence). The limit of this sub-sequence is an element γ of Z. If the sub-sequence is well-chosen, then γ is an upper bound of the sequence, so that the range cannot be Z. Then Cantor shows that any infinite cardinal, less than the cardinal of Z, must be $\aleph_0$. He deduces: the cardinal of Z is the next cardinal after aleph-zero, he denotes it by aleph-one (the big question is: is this the cardinal of the powerset of N?). The two sets F and Z have clearly the same cardinal.

Let's denote by ω or ω_0 the ordinal of N, by ω_1 the ordinal of F and by Ω the ordinal of Z. We have

$$\omega + \Omega = \omega_1.$$

Property (F) says: the ordinal of T is ω_1, and theorem F says it is Ω; so it suffices to show $\Omega = \omega_1$. This follows from the previous equation and the fact that ω_1 is an infinite cardinal, thus an indecomposable ordinal. Cantor uses

$$\omega + \Omega = \omega + \omega^2 + (\Omega - \omega^2), \qquad \omega + \omega^2 = \omega^2$$

in order to get $\omega + \Omega = \Omega$.

```
Definition Cantor_Omega := aleph_one -s omega0.
Lemma Cantor_omega_pr x:
  inc x Cantor_Omega <-> (countable_ordinal x /\ omega0 <=o x).
Lemma Cantor_omega_pr2: cardinal Cantor_Omega = aleph_one.
```

11.15.2 Critical points of product

We study now the critical ordinals of the ordinal product (Exercise 14b, page 583). These are the numbers y satisfying (CP): for all x, $1 \le x < y$ implies $x \cdot y = y$. Let (H) be the property that, for all z such that $2 \le z \le y$ there exists an indecomposable ordinal t such that $y = z^t$. Let (H′) be the property that $y = \omega^{\omega^n}$ for some n and (H″) the property that $y = \omega^m$ for some indecomposable m. These properties are equivalent. Since m is indecomposable if, and only if, it has the form ω^n, properties (H′) and (H″) are clearly equivalent. Obviously, (H) implies (H″). Conversely, assume (H″) and fix z. If $z = y$ it suffices to take $t = 1$. If z is finite, then $z^{\omega \cdot m} = \omega^m = y$. Assume z infinite of degree k so that $k < m$. There exists q indecomposable such that $m = k \cdot q$ hence $z^q = \omega^{k \cdot q} = y$. If (H″) holds then (CP) holds also, for if k is the degree of x, relation (11.38) says $x \cdot y = \omega^{k+m}$, and $k + m = m$.

Assume that (CP) holds, consider x with $x < y$. Write $y = x^a \cdot b + c$. Assume first $x^a \cdot b < y$. Then $x^a \cdot b \cdot (x^a \cdot b + c) = x^a \cdot b + c$. Write $x^a \cdot b = 1 + u$, so that $x^a \cdot b \cdot (u + c) = c$. The LHS is at least x^a, contradicting $c < x^a$. Thus $y = x^a \cdot b$. Assume $x^a < y$. Then $x^a \cdot y = y$, i.e., $x^a \cdot x^a \cdot b = x^a \cdot b$ After simplification by x^a, we get $x^a \cdot b = b$. This is $y = b$, contradicting $b < x$. Thus $y = x^b$.

Assume $d < b$, so that $x^d < y$. We have $x^d \cdot y = y$, which is $d + b = b$, and this says that b is indecomposable.

```
Lemma critical_productP: (* 66 *)
   (CP := critical_ordinal \1o oprod2)
   (p1 := fun y => (exists2 n, ordinalp n & y = oopow (oopow n)))
   (p2 := fun y => omega0 <=o y /\
       (forall z, \1o <o z -> z <=o y ->
          exists t, [/\ ordinalp t, indecomposable t & y = z ^o t])):
  forall y, (p1 y <-> p2 y) /\ (CP y <-> p1 y).
```

We show the following properties (see Cantor, [7, §20, Theorem G]): if x is an ϵ-number and $a < x$, then $a + x = x$, $a \cdot x = x$ and $a^x = x$ (we assume $a > 0$ for the product, $a \geq 2$ for the exponentiation). This says that x is critical for the sum, product and exponentiation. The first two properties are obvious. We have $a^x = a^{\omega \cdot x} = (a^\omega)^x$. The case a finite is trivial as $a^\omega = \omega$. Otherwise, let n be the degree of a, so that $a^\omega = \omega^{n \cdot \omega}$. Then $a^x = \omega^{n \cdot x}$. But $n < x$, so that $n \cdot x = x$.

Assume $a^x = x$. This is impossible if $a = 0$, and gives $x = 1$ for $a = 1$. In all other cases, $a \geq 2$ so that $2^x = x$, and $a < x$. It is clear that x cannot be a successor, since 2^{n+1} is at least $n + n$, and this cannot be $n + 1$. In particular, $x = \omega \cdot y$ for some y, and $2^x = (2^\omega)^y = \omega^y = \omega \cdot y$. Note that y cannot be zero, so let's write $y = 1 + t$. We get $\omega^t = 1 + t$ after simplification by ω. Since ω^t is indecomposable, one term is ω^t. Assume it is 1, so that $t = 0$, $y = 1$ and $x = \omega$. Otherwise, $t = \omega^t$ so t is an ϵ-ordinal and $1 + t = t$. It trivially follows $t = x$. In summary: if $2^x = x$, then x is critical for exponentiation and is ω or an ϵ-ordinal. If $a^x = x$ and a is infinite, we have $\omega \leq a < x$, so that x is an ϵ-ordinal. This is [7, §20, Theorem H], the last theorem of Cantor's paper.

```
Lemma ord_epsilon_p9 x a: epsilonp x -> a <o x->
  [/\ a +o x = x,
    (\1o <=o a -> a *o x = x) &
    (\2o <=o a -> a ^o x = x) ].
Lemma ord_epsilon_p10 x:
  epsilonp x -> critical_ordinal \2o opow x.

Lemma ord_epsilon_p11 x: ordinalp x -> (\2o ^o x = x) ->
  x = omega0  \/ epsilonp x.
Lemma ord_epsilon_p12 x: ordinalp x -> (\2o ^o x = x) ->
  critical_ordinal \2o opow x.
Lemma ord_epsilon_p13 a x: ordinalp a -> ordinalp x -> (a ^o x = x) ->
  ( (a <o omega0 -> ((a = \1o /\ x = \1o) \/  (\2o ^o x = x)))
  /\ (omega0 <=o a -> (epsilonp x /\ a <o x))).
```

11.16 The functions phi and psi

11.16.1 First derivation of exponentiation

Consider $f(x) = a^x$. This is a normal OFS for $a \geq 2$, it has thus a first derivation $f'(x)$. If $a = \omega$, we have $f'(x) = \epsilon_x$ by definition. The previous lemma characterizes the fix-points of f. If $a < \omega$, the fix-points of f are all ϵ-numbers, together with ω, so $f'(x) = \epsilon_{x-1}$ for non-zero x and $f'(0) = \omega$; if $a > \omega$ the fix-points of f are all ϵ-numbers that are large enough, so if β is the least ordinal such that $a \leq \epsilon_\beta$ then $f'(x) = \epsilon_{\beta+x}$. Note that Veblen [22, Theorem 4, Corollary 5] has ϵ_{x-2} and $\epsilon_{\beta+(x-1)}$, since he defines f' only for non-zero values of x.

```
Definition inverse_epsilon :=
   ordinalr (fun x y => [/\ epsilonp x, epsilonp y & x <=o y]).

Lemma inverse_epsilon_pr x (y := inverse_epsilon x):
  epsilonp x -> (ordinalp y /\ epsilon_fam y = x).
Lemma least_critical_pow a (b := inverse_epsilon (epsilon_fct (osucc a))):
  omega0 <=o a -> least_fixedpoint_ge (opow a) \0o (epsilon_fam b).

Lemma pow_fix_enumeration1 a: \2o <=o a -> a <o omega0 ->  (* 79 *)
  forall x, ordinalp x -> first_derivation (opow a) x =
    Yo (x = \0o) omega0 (epsilon_fam (x -o \1o)).
Lemma pow_fix_enumeration2 a (b := inverse_epsilon (epsilon_fct (osucc a))):
   omega0 <=o a ->
  forall x, ordinalp x -> first_derivation (opow a) x =
     (epsilon_fam (b +o x)).
```

11.16.2 Many derivations

We shall study in this section the repeated derivations of addition, multiplication and exponentiation.

Let $f(x)$ by $a+x$, $a\cdot x$, a^x (in the last case, we shall assume $a=\omega$). Let $b(x,i)$ be the cardinal successor of the maximum of a, x, and ω. This is an infinite cardinal successor, contains x and i, and is stable by f. This allows us to construct $g(x,i)$, the i-th derivation of f.

```
Lemma all_der_sum_aux c (f:= osum2 c)
   (b:= fun x i => cnext (card_max (omax c x) i)):
  ordinalp c ->  normal_ofs f /\ all_der_bound f b.
Lemma all_der_prod_aux c (f:= oprod2 c)
   (b:= fun x i => cnext (card_max (omax c x) i)):
  \0o <o c ->  normal_ofs f /\ all_der_bound f b.
Lemma all_der_pow_aux
   (b:= fun x i => cnext (card_max x i)):
  normal_ofs oopow /\ all_der_bound oopow b.
```

Corollaries 2 and 3 of theorem 6 of [22]) say: If $f(x) = a+x$, then $g(x,i) = a\cdot\omega^i + x$ and, if $a>0$, $f(x) = a\cdot x$, then $g(x,i) = a^{\omega^i}\cdot x$. In particular, if $f(x)=x$, then $g(x,i)=x$.

Proof. Let h be the RHS. We prove by induction on i that $g(x,i) = h(x,i)$. We distinguish the case where i is zero, a successor, or is limit. In the case $i=0$, it suffices to note that $h(x,0) = f(x)$. If $i = j+1$, both functionals are the first derivation of equal functionals. Finally, consider the case where i is limit. Consider first $h(x,i)$. This is a fix-point of $h(\cdot,j)$ for $j<i$ (if $j<i$ then $\omega^j \ll \omega^i$), thus of $g(\cdot,j)$ thus is in the range of $g(\cdot,i)$. Conversely, $y = g(x,i)$ is a common fix-point. In the case of a sum, this says $y = a\cdot\omega^j + y$. In particular $y \geq a\cdot\omega^j$. As this is a normal function of j, taking the supremum over $j<i$ says $a\cdot\omega^i \leq y$, hence the conclusion. In the case of a product, we write $t_j = a^{\omega^j}$. To say that y is a fix-point becomes $t_j\cdot y = y$, for $j<i$. Write $y = t_i\cdot q + r$ by Euclidean division. Since $t_j\cdot t_i = t_i$, we deduce, after simplification that $t_j\cdot r = r$. This (assuming r non-zero) gives $r \geq \sup t_j = t_i$, contradicting $r < t_i$. It follows $y = t_i\cdot q$, qed.

Corollary: $a \cdot \omega^i$ and a^{ω^i} are normal OFS (as a function of i, when a is fixed; assuming $a > 0$ and $a \geq 2$ respectively). This is clear by composition.

Note: Veblen says: if $f(x) = a + x$, then $f(x,i) = a \cdot \omega^i + (x-1)$. Remember, that for Veblen, x is always non-zero; thus $f(x,1)$, the first derivation of f enumerates the fix-points of f. The first fix-point is $f(1,1) = a \cdot \omega$. The $n+1$-th fixpoint is $f(n+1,1) = a \cdot \omega + n$. Assume also a non-zero, so that $f(1) \neq 1$. Veblen deduces that $a \cdot \omega^i$ is a normal OFS of i.

Veblen says: if $f(x) = a \cdot x$, then $f(x,i) = a^{\omega^i} \cdot x$. He obviously assumes a non-zero. As $f(0) = 0$, we have $g(0,i) = 0$. Thus, $g(1,1)$ is the first non-zero fixed point of f, thus is equal to $f(1,1)$. Assume $a \neq 1$, so that $f(1) \neq 1$; Veblen deduces that a^{ω^i} is normal. We show here: 1 is the least non-fix point of f, so that $g(1)$ is normal.

```
Lemma all_der_sum c x i (f:= osum2 c)
   (b:= fun x i => cnext (card_max (omax c x) i)):
  ordinalp c ->  ordinalp x -> ordinalp i  ->
  all_der f b x i = c *o omega0 ^o i +o x.
Lemma all_der_prod c x i (f:= oprod2 c)   (* 65 *)
   (b:= fun x i => cnext (card_max (omax c x) i)):
  \0o <o c ->  ordinalp x -> ordinalp i  ->
  all_der f b x i = c ^o (omega0 ^o i) *o x.
Lemma all_der_ident x i (f:= @id Set)
   (b:= fun x i => cnext (card_max x i)):
  ordinalp x -> ordinalp i -> all_derivatives f b x i = x.
Lemma all_der_prod_cor c: \2o <=o c ->
   normal_ofs (fun i => c ^o (oopow i)).
```

11.16.3 The function phi

We consider now the function ϕ defined by Schütte in [18, Chapter V, §13]. He considers only countable ordinals (i.e., he considers the cardinal successor E of ω), but everything applies as well to uncountable ordinals. He starts with a function f, the enumeration of the set of *additive principal numbers*. These are non-zero numbers x such that, for all $y < x$, $y + x = x$. We have shown that this is the same as to say that x is indecomposable, and we have shown that $f(x) = \omega^x$. Note that Schütte does not define multiplication nor exponentiation of ordinals. He shows however some properties of $f(x) = \omega^x$, including the existence of the Cantor Normal Form (11.29).

We denote by $\phi(i,x)$ the i-th derivation of ω^x. Note the order of arguments: this is denoted $f(x,i)$ by Veblen. We start with trivial results. Note that any cardinal successor is stable by ϕ (in particular, if x and y are countable, so is $\phi(x,y)$).

Theorem 13.9 of [18] says

$$\phi_i(x) = \phi_j(y) \iff \begin{cases} \text{if } i < j \text{ then } x = \phi_j(y) \\ \text{if } i = j \text{ then } x = y \\ \text{if } j < i \text{ then } \phi_i(x) = y, \end{cases} \tag{11.68}$$

while Theorem 13.10 of [18] says

$$\phi_i(x) < \phi_j(y) \iff \begin{cases} \text{if } i < j \text{ then } x < \phi_j(y) \\ \text{if } i = j \text{ then } x < y \\ \text{if } j < i \text{ then } \phi_i(x) < y. \end{cases} \tag{11.69}$$

```
Definition Sphi x y :=
  all_der oopow (fun x i => cnext (card_max x i)) y x.

Lemma OS_Sphi x y: ordinalp x -> ordinalp y ->
   ordinalp (Sphi x y).
Lemma Sphi_bounded C (E :=cnext C) x y:
   infinite_c C -> inc x E -> inc y E -> inc (Sphi y x) E.
Lemma Sphi_countable x y:
    countable_ordinal x -> countable_ordinal y ->
    countable_ordinal (Sphi x y).
Lemma Sphi_p0 x: ordinalp x ->  Sphi \0o x = oopow x.
Lemma Sphi_p1 x: ordinalp x -> normal_ofs (Sphi x).
Lemma Sphi_p2 x: ordinalp x ->
  first_derivation (Sphi x) =1o (Sphi (osucc x)).
Lemma Sphi_p3  x i j: ordinalp x -> i <o j ->
   Sphi i (Sphi j x) = Sphi j x.
Lemma Sphi_p4 y j: ordinalp y -> \0o <o j ->
   (forall i, i <o j ->  Sphi i y = y) ->
   (exists2 x, ordinalp x & y = Sphi j x).
Lemma Sphi_p5 : normal_ofs (Sphi ^~ \0o).

Lemma Sphi_p6 x y i j:
  ordinalp x -> ordinalp y -> ordinalp i -> ordinalp j ->
  (Sphi i x = Sphi j y <->
   [/\ i <o j -> x = Sphi j y,
       i = j -> x = y &
      j <o i -> y = Sphi i x]).
Lemma Sphi_p7 x y i j:
  ordinalp x -> ordinalp y -> ordinalp i -> ordinalp j ->
  (Sphi i x <o Sphi j y <->
   [/\ i <o j -> x <o Sphi j y,
       i = j -> x <o y &
      j <o i -> Sphi i x <o y]).
Lemma Sphi_p8 x y i j:
  ordinalp x -> ordinalp y -> ordinalp i -> ordinalp j ->
  (Sphi i x <=o Sphi j y <->
   [/\ i <o j -> x <=o Sphi j y,
       i = j -> x <=o y &
      j <o i -> Sphi i x <=o y]).
```

Let $\mathrm{Cr}(\alpha)$ be the range of ϕ_α. Since ϕ_α is a normal OFS, this is a closed proper class. To say that β is in the class is: there is an ordinal x such that $\beta = \phi_\alpha(x)$. We shall write $\mathrm{Cr}(\alpha,\beta)$ in this case. Since $\mathrm{Cr}(\alpha)$ is a closed proper class, it has an enumeration function, which is nothing else then ϕ_α.

For $\alpha = 0$, $\mathrm{Cr}(\alpha)$ is the collection of indecomposable ordinals, and otherwise, it is the collection of common fix-points of $(\phi_\gamma)_{\gamma<\alpha}$. These properties uniquely characterize $\mathrm{Cr}(\alpha)$, so that our $\mathrm{Cr}(\alpha)$ is the same as that of Schütte.

If α is a limit ordinal, then $\mathrm{Cr}(\alpha)$ is the intersection of all $\mathrm{Cr}(\beta)$ for $\beta < \alpha$. If α is a successor, say of α', then $\mathrm{Cr}(\alpha)$ is the predicate: to be a fix-point of $\phi_{\alpha'}$.

```
Definition Schutte_Cr a b := exists2 x, ordinalp x & b = Sphi a x.

Lemma Schutte_Cr_p1 a: ordinalp a ->
  iclosed_proper (Schutte_Cr a).
```

```
Lemma Schutte_Cr_p2 a: ordinalp a ->
   ordinalsf (Schutte_Cr a) =1o phi a.
Lemma Schutte_Cr_p3 b: ordinalp b ->
  (Schutte_Cr \0o b <-> indecomposable b).
emma Schutte_Cr_p4 a b: \0o <o a -> ordinalp b ->
  (Schutte_Cr a b <-> (forall c, c <o a -> Sphi c b = b)).
Lemma Schutte_Cr_p5 a b: limit_ordinal a -> ordinalp b ->
  (Schutte_Cr a b <-> (forall c, c <o a -> Schutte_Cr c b)).
Lemma Schutte_Cr_p6 a b: ordinalp a -> ordinalp b ->
  (Schutte_Cr (osucc a) b <-> Sphi a b = b).
Lemma Schutte_Cr_p7 a a' b:  a <=o a' -> ordinalp b ->
   Schutte_Cr a' b ->  Schutte_Cr a b.
Lemma Schutte_Cr_p8 a  b: ordinalp a  -> ordinalp b ->
   Schutte_Cr a b -> indecomposable b.
Lemma Schutte_Cr_p8' a b: ordinalp a  -> ordinalp b ->
   indecomposable (Sphi a b).
```

Schütte claims $x \le \phi(x,0)$ (Theorem 13.11). This follows trivially from the fact that $\phi(x,0)$ is an OSF. He deduces: if $y \in \mathrm{Cr}(x)$, then $x \le y$. Moreover every indecomposable ordinal x is uniquely $x = \phi(a,b)$ where $a \le x$ and $b < x$. Uniqueness is obvious. We have $x \le \phi(x,0) < \phi(x,x)$. Thus, there is a least a such that $x \ne \phi(a,x)$. Thus if $c < a$, we have $x = \phi(c,x)$. We deduce $x = \phi(a,b)$ for some b (if $a = 0$, we use the fact that x is indecomposable).

An ordinal α is said *strong critical* if $\alpha \in \mathrm{Cr}(\alpha)$. This is equivalent to $\alpha = \phi(\alpha,0)$. (note that $\alpha = \phi(\alpha,\beta)$ says $\beta = 0$). As $\alpha \mapsto \phi(\alpha,0)$ is normal, the collection of strong critical ordinals is a closed proper class (this is Theorem 13.14 of [18]).

```
Definition strong_critical x :=  Schutte_Cr x x.

Lemma Sphi_p9 x: ordinalp x ->  x <=o Sphi x \0o.
Lemma Schutte_Cr_p9 x y: ordinalp x -> ordinalp y ->  Schutte_Cr x y ->
   x <=o y.
Lemma Sphi_p10a a b a' b' x:
   ordinalp a -> b <o x -> x = Sphi a b ->
   ordinalp a' -> b' <o x -> x = Sphi a' b' ->
   (a = a' /\ b = b').
Lemma Schutte_phi_p10b x: indecomposable x ->
  (exists a b, [/\ a <=o x, b <o x & x = Sphi a b]).
Lemma strong_criticalP x: ordinalp x ->
  (strong_critical x  <-> Sphi x \0o = x).
Lemma strong_critical_closed_proper:
  iclosed_proper (fun z => ordinalp z /\ strong_critical z).
```

The least strongly critical ordinal is called the Feferman-Schütte ordinal, and denoted by Γ_0. Thus Γ_0 is the least ordinal x such that $\phi(x,0) = x$. This is a countable limit ordinal. Moreover, if $x < \Gamma_0$ and $y < \Gamma_0$ then $\phi(x,y) < \Gamma_0$.

```
Definition Gamma_0 := the_least_fixedpoint_ge (Sphi ^~ \0o) \0o.

Lemma countable_Gamma_0: countable_ordinal Gamma_0.
Lemma OS_Gamma_0: ordinalp Gamma_0.
Lemma Gamma0_limit: limit_ordinal Gamma_0.
Lemma Gamma0_p :
   Sphi Gamma_0 \0o = Gamma_0 /\
   forall x, ordinalp x -> Sphi x \0o = x -> Gamma_0 <=o x.
```

```
Lemma Gamma0_s x y: x <o Gamma_0 -> y <o Gamma_0 -> Sphi x y <o Gamma_0.
Lemma Gamma0_epsilon: oopow Gamma_0 = Gamma_0.
```

Let ζ be the iteration of $x \mapsto \phi(x,0)$; thus

$$\zeta_0 = \phi(0,0) \qquad \zeta_{n+1} = \phi(\zeta_n,0).$$

Lets' prove theorem 14.16 of [18]: for any ordinal x, there exists an integer n such that $x < \zeta_n$. Schütte implicitly assumes $x < \Gamma_0$, and his statement is equivalent to $\Gamma_0 = \sup \zeta_n$ as $\zeta_n < \Gamma_0$ by induction. With the definition given above, the statement is obvious, as Γ_0 is the supremum of zero and the ζ_n. The proof of Schütte is by induction. He considers the leading term y of the Cantor Normal Form of x. All we need is that $y = \omega^k$ for some k and $\omega^k \le x < \omega^{k+1}$. Assume $y < \zeta_n$. There is m such that $\zeta_n = \omega^m$, so that $k < n$, $k+1 \le n$ and $x < \omega^m = \zeta_n$. Thus, it suffices to prove the property for y. Write it as $\phi(a,b)$. We have $b < y$, and $a \le y$. Since we are below Γ_0, we get $a < y$. By induction, there is n, m such that $a < \zeta_n$, $b < \zeta_m$. We may assume $m = n$, in fact, $b < \phi(\zeta_n,0)$. We deduce $y = \phi(a,b) < \phi(\zeta_n,0) = \zeta_{n+1}$.

```
Definition Szeta :=
   induction_term (fun (n:Set) v => Sphi v \0o) \1o.

Lemma Szeta_0: Szeta \0c = \1o.
Lemma Szeta_1: Szeta \0c = Sphi \0o \0o.
Lemma Szeta_2 n: natp n -> Szeta (csucc n) = Sphi (Szeta n) \0o.
Lemma OS_Szeta n: natp n -> ordinalp (Szeta n).
Lemma Szeta_3 n: natp n -> Szeta n <o Gamma_0.
Lemma Szeta_4 k: natp k -> Szeta k <o Szeta (csucc k).
Lemma Szeta_5 n m: natp n -> natp m -> n <c m ->
  Szeta n <o Szeta m.
Lemma Szeta_6 x: ordinalp x ->
   (forall n, natp n -> Szeta n <=o x) -> Gamma_0 <=o x.
Lemma Schutte_14_16 x:  x <o Gamma_0 ->
   exists2 n,natp n & x <o Szeta n.
```

11.16.4 The function psi

We say that γ is a *maximal* α*-critical* ordinal if γ is in $\mathrm{Cr}(\alpha)$ but not in $\mathrm{Cr}(\xi)$ for $\xi > \alpha$. This predicate is not closed, but has an enumeration function ψ_α (which is not normal).

We first notice that γ is a *maximal* α*-critical* if it has the form $\phi(\alpha,\beta)$ for some $\beta < \gamma$, so that each indecomposable ordinal is maximal α-critical for some α.

```
Definition maximal_critical a x :=
   Schutte_Cr a x /\ forall a', a <o a' -> ~ Schutte_Cr a' x.

Lemma maximal_criticalP a x: ordinalp a -> ordinalp x ->
  (maximal_critical a x <-> exists2 b, b<o x & x = Sphi a b).
Lemma maximal_critical_p1 x: indecomposable x ->
  exists2 a, ordinalp a & maximal_critical a x.
```

Consider the condition: $b = c + n$, where n is an integer, and $\phi(a,c) = c$; in this case we define $\psi(a,b) = \phi(a,b+1)$, otherwise $\psi(a,b) = \phi(a,b)$.

As Γ_0 is a limit ordinal, we deduce

$$a < \Gamma_0, b < \Gamma_0 \implies \psi(a,b) < \Gamma_0. \tag{11.70}$$

Note that $\psi(a,0) = \phi(a,0)$. We have: if $\psi(a,b) = \phi(a,b')$ then $b' < \phi(a,b')$. One deduces that $\psi(a,b)$ is a maximal a-critical ordinal. Taking for b' the quantity that appears in the definition of ψ gives

$$b < \psi(a,b).$$

Note that ψ is increasing (for fixed a) and injective (write $\psi(a,b) = \phi(a,c)$ and $\psi(a',b') = \phi(a',c')$, where $c < \phi(a,c)$ and $c' < \phi(a',c)$. If $\psi(a,b) = \psi(a',b')$, comparing ϕ gives $a = a'$, then $b = b'$). We deduce $a < \psi(a,b) \iff \phi(c,0) \neq c$, where $c = \psi(a,b)$ (i.e., c is not strongly critical). In particular

$$a < \Gamma_0, b < \Gamma_0 \implies a < \psi(a,b).$$

```
Definition Spsi_aux a b :=
  exists n c, [/\ n <o omega0, ordinalp c, b = c +o n & Schutte_phi a c = c].
Definition Spsi a b :=
  Sphi a (Yo (Spsi_aux a b) (osucc b) b).

Lemma Spsi_p0 a: ordinalp a ->  Spsi a \0o = Sphi a \0o.
Lemma OS_psi a b: ordinalp a -> ordinalp b ->  o(Spsi a b).
Lemma Spsi_p0 a: ordinalp a ->
   Spsi a \0o = Schutte_phi a \0o.
Lemma Gamma0_s_psi a b: a <o Gamma_0  -> b <o Gamma_0 ->
  Schutte_psi a b <o Gamma_0.
Lemma Spsi_p1 a b b':
  ordinalp a -> ordinalp b -> ordinalp b' ->
  (Spsi a b = Sphi a b') -> b' <o Sphi a b'.
Lemma Spsi_p1' a b (b' := (Yo (Spsi_aux a b)  (osucc b) b)):
  ordinalp a -> ordinalp b -> b' <o Sphi a b'.
Lemma Spsi_p2 a b: ordinalp a -> ordinalp b ->  b <o Spsi a b.
Lemma Spsi_p3 a b b':
  ordinalp a -> b <o b' -> (Spsi a b <o Spsi a b').
Lemma Spsi_p4 a a' b b':
  ordinalp a -> ordinalp a' -> ordinalp b -> ordinalp b' ->
  (Spsi a b = Spsi a' b') ->
  (a = a' /\ b = b').
Lemma Spsi_p5 a b: ordinalp a -> ordinalp b ->
  (a <o Spsi a b <-> ~ (strong_critical (Spsi a b))).
Lemma Spsi_p6 a b: a <o Gamma_0  -> b <o Gamma_0 ->
  a <o Spsi a b.
```

Every indecomposable ordinal is of the form $\psi(a,b)$, see [18, Theorem 13.15]. We know $x = \phi(a,c)$ for some c; if we cannot take $b = c$, then $c = b_0 + n$, n is non-zero, and we can take $b = b_0 + (n-1)$. We deduce: there is a function $\mathrm{P}(x)$, such that $\mathrm{P}(x)$ is a pair (a,b), and $\omega^x = \psi(a,b)$; moreover $a \le \omega^x$ and $b < \omega^x$. If y is the degree of $\psi(a,b)$, then $\omega^y = \psi(a,b)$.

```
Definition Spsip p := Spsi (P p) (Q p).
Definition inv_psi_omega x:=
  select (fun z => Spsip z = oopow x)
    (coarse (osucc (oopow x))).

Lemma Spsi_p7 x: indecomposable x ->
  (exists a b, [/\ a <=o x, b <o x  & x = Spsi a b]).
Lemma Spsi_p2' a b: ordinalp a -> ordinalp b ->
  a <=o Spsi a b.
Lemma inv_psi_omega_p x (z:= oopow x) (y:= inv_psi_omega x) :
```

```
   ordinalp x ->
  [/\ pairp y, (P y) <=o z, (Q y) <o z & Spsip y = z].
Lemma odegree_psi a b: ordinalp a -> ordinalp b ->
  oopow (odegree (Spsi a b)) = (Spsi a b).
```

A similar argument says that every maximal a-critical ordinal has the form $\psi(a,b)$. Thus $\psi(a)$ enumerates the collection of maximal a-critical ordinals; however, $\psi(a)$ is not an OFS (in other terms, the collection is not closed). Proof: let $x = \phi(a+1,0)$. This is a power of ω. If $x = 1$ we get $\phi(a,1) = 1$, thus $\phi(a,0) = 0$, absurd. Thus x is a limit ordinal. If $t < x$ then $\phi(a,t) < x$; it follows $\phi(a,t)$ is not in $\mathrm{Cr}(a+1)$ thus, $t \neq \phi(a,t)$. One deduces $\phi(a,t) = \psi(a,t)$ thus $\sup_{t<x} \phi(a,t) = \sup_{t<x} \psi(a,t)$. If ψ were normal, we would get: $\phi(a,x) = \psi(a,x)$. But $\phi(a,x) = x < \psi(a,x)$.

The equivalent of (11.69) for ψ is, [18, Theorem 13.18]:

$$\psi_i(x) < \psi_j(y) \iff \begin{cases} \text{if } i < j \text{ then } x < \psi_j(y) \\ \text{if } i = j \text{ then } x < y \\ \text{if } j < i \text{ then } \psi_i(x) \le y. \end{cases} \tag{11.71}$$

Proof: if $x < \psi_j(y))$ then $x+1 < \psi_j(y)$, thus $\psi_i(x) \le \phi_i(x+1) < \phi_i(\psi_j(y))$. Note that $\phi_j(y)$ is the range of ϕ_j thus is a fix-point of ϕ_i when $< j$.

```
Lemma Spsi_p8 a b:
  ordinalp a -> ordinalp b -> (maximal_critical a (Spsi a b)).
Lemma Spsi_p9 a b: ordinalp a -> ordinalp b ->
  (maximal_critical a b <-> (exists2 c, ordinalp c & b = Spsi a c)).
Lemma Spsi_p10 a: ordinalp a -> ~(normal_ofs (Spsi a)).
Lemma Spsi_limit a b: ordinalp a -> ordinalp b ->
   (a <> \0o \/ b <> \0o) -> limit_ordinal (Spsi a b).
Lemma Spsi_p11 x y : ordinalp x -> ordinalp y ->
  Spsi x y <=o Sphi x (osucc y).
Lemma Spsi_p12 i j x y :
  ordinalp i -> ordinalp j -> ordinalp x -> ordinalp y ->
  (Spsi i x <o Spsi j y <->
   [/\ i <o j -> x <o Spsi j y,
       i = j -> x <o y &
      j <o i -> Spsi i x <=o y]).
```

11.16.5 The Cantor Normal Form

The Cantor Normal Form (11.29) can be written as:

$$x = \psi(a_1,b_1) + \ldots + \psi(a_n,b_n), \qquad \psi(a_k,b_k) \ge \psi(a_{k+1},b_{k+1}) \tag{11.72}$$

For each exponent e in (11.29), we consider (a,b) such that $\psi(a,b) = \omega^e$; conversely, for each (a,b) there is an exponent e. Thus existence and uniqueness of (11.29) implies existence and uniqueness of (11.72).

Assume $x < \Gamma_0$. Then $\psi(a,b) < \Gamma_0$.

```
Definition CNF_from_psi (p: fterm) z := odegree (Spsip (p z)).
Definition CNF_psi_ax p n :=
  (forall i, i <c n -> ordinal_pair (p i)) /\
```

```
  (forall i, natp i -> csucc i <c n ->
   Spsip (p i) <=o  Spsip (p (csucc i))).
Definition CNF_psi_ax2 p n :=
  CNF_psi_ax p n /\
     (forall i, i <c n -> P (p i) <o Gamma_0  /\ Q (p i) <o Gamma_0).
Definition CNF_psiv p n :=CNFrv (CNF_from_psi p) n.


Lemma CNF_psi_p0 p n: CNF_psi_ax p n -> natp n ->
  CNF_psiv p n =  osumf (fun i => (Spsip (p i))) n.
Lemma CNF_psi_p1 p n: CNF_psi_ax p n -> CNFr_ax (CNF_from_psi p) n.
Lemma CNF_psi_unique p1 n p2 m:
  natp n ->natp m -> CNF_psi_ax p1 n -> CNF_psi_ax p2 m ->
  CNF_psiv p1 n = CNF_psiv p2 m ->
  n = m /\  same_below p1 p2 n.
Lemma CNF_psi_exists x: ordinalp x ->
  exists p n, [/\ natp n, CNF_psi_ax p n & x = CNF_psiv p n].
Lemma CNF_psi_exists_Gamma0 x: x <o Gamma_0 ->
  exists p n, [/\ natp n, CNF_psi_ax2 p n & x = CNF_psiv p n].
```

Consider the following definition:

```
Inductive T2 : Set :=
  zero : T2
| cons : T2 -> T2  -> nat -> T2 -> T2.
```

It is possible define an order relation on T_2 and a subset T_2' of T_2 on which the ordering is a well-ordering. Let x be an object of type T_2. We define $f(x)$ by: if x is zero, then $f(x) = 0$, and if x is `cons a b n c`, then $f(x) = \psi(f(a), f(b))\cdot(n+1) + f(c)$. We pretend that the restriction of f to T_2' makes it order isomorphism onto the set of all ordinals $< \Gamma_0$.

We consider first a simpler case: here Γ_0 is replaced by ϵ_0. We first define ϵ_0. This is a power of ω, thus is indecomposable. If $x < \epsilon_0$, then $x \cdot n < \epsilon_0$ (for any integer n) and $\omega^x < \epsilon_0$.

```
Definition epsilon0 := epsilon_fam \0o.

Lemma epsilon0p: epsilonp epsilon0.
Lemma sum_lt_eps0 a b: a <o  epsilon0 -> b <o epsilon0 ->
  a +o b <o epsilon0.
Lemma prod_lt_eps0 a n: a <o epsilon0 ->  natp n -> a *o n <o epsilon0.
Lemma pow_lt_eps0 a: a <o epsilon0 -> oopow a <o epsilon0.
```

We can uniquely write $x = \omega^a\cdot(n+1)+b$ where $b < \omega^a$. Note that, if $x < \omega_0$, both quantities a and b are $< x$.

```
Definition CNF_simple_ax a n b x :=
  [/\ ordinalp a, ordinalp b, natp n, x = oopow a *o (succ n) +o b &
      b <o  omega0 ^o a].
Definition the_CNF_simpl x :=
   select (fun z => (CNF_simple_ax (P z) (P (Q z)) (Q (Q z)) x))
     (osucc x \times (Nat \times x)).

Lemma CNF_simple_p1 a n b x: CNF_simple_ax a n b x ->
  ord_ext_div_pr  omega0 x a (csucc n) b.
Lemma CNF_simple_p2 a n b x: CNF_simple_ax a n b x ->
```

```
 [/\ a <=o x, b <o x & oopow a <=o x].
Lemma CNF_simple_bnd a n b x: CNF_simple_ax a n b x ->
  inc (J a (J n b)) (osucc x \times (Nat \times x)).
Lemma CNF_simple_unique a n b a' n' b' x:
  CNF_simple_ax a n b x -> CNF_simple_ax a' n' b' x ->
  [/\ a = a', b = b' & n = n'].
Lemma CNF_simple_exists x: \0o <o x ->
  exists a n b, CNF_simple_ax a n b x.

Lemma ord_eps_p2_bis a b c (x := oopow a *o b +o c):
  ordinalp a -> \0o <o b -> ordinalp c -> x <o epsilon0 -> a <o x.
Lemma CNF_simple_bdn2 x (z := the_CNF_simpl x): \0o <o x ->
  x <o epsilon0 -> ( P z <o x /\ (Q (Q z)) <o x).
```

For any ordinals, a and b, $\psi(a,b)$ is indecomposable (in particular > 0). If $x = \psi(a,b)$, then $u < x$ and $v < x$, n integer imply $u \cdot (n+1) + v < x$. In particular, if a, b, c are $< \Gamma_0$, then $\psi(a,b)\cdot(n+1)+c < \Gamma_0$. From this, it trivially follows that $f(x) < \Gamma_0$, where f is the function mentioned above.

```
Lemma Gamma0_p1: Spsi Gamma_0 \0o = Gamma_0.
Lemma Spsi_indecomposable a b:
  ordinalp a -> ordinalp b -> indecomposable (Spsi a b).
Lemma Spsi_pos a b: ordinalp a -> ordinalp b ->
   \0o <o (Spsi a b).
Lemma Spsi_nz a b: ordinalp a -> ordinalp b ->
  (Spsi a b) <> \0o.
Lemma Spsi_indecomp_rec u v a b n (x := Schutte_psi u v):
  ordinalp u -> ordinalp v ->
  a <o x -> b <o x -> a *o csucc (nat_to_B n) +o b <o x.
Lemma T2_to_bourbaki_small a b c n:
   a <o Gamma_0 -> b <o Gamma_0 -> c <o Gamma_0 ->
   Spsi a b  *o csucc (nat_to_B n) +o c <o Gamma_0.
```

Let x be an ordinal. Write $x = \omega^a \cdot (n+1) + b$ where $b < \omega^a$. We can write $\omega^a = \psi(u,v)$. If $x < \Gamma_0$, then $a < \Gamma_0$, and this says that u and v are $< \omega^a$. In particular, these quantities are $< x$.

```
Lemma inv_psi_omega_p2 x (z: oopow x) (y:= inv_psi_omega x) :
   ordinalp x ->x <o Gamma_0 ->
  [/\ pairp y, (P y) <o z, (Q y) <o z & Spsi (P y) (Q y) = z].

Lemma  CNF_simple_bdn3 x (v := (P (the_CNF_simpl x))) (y:= inv_psi_omega v):
   \0o <o x -> x <o Gamma_0 ->
   [/\ pairp y, (P y) <o x, (Q y) <o x &
     Spsi (P y) (Q y) = omega0 ^o v ].
```

For convenience, we rewrite one implication of (11.71) as three conditions:

```
Lemma Spsi_cpa a a' b b':
  ordinalp b' ->
  a <o a' -> b <o Spsi a' b' -> Spsi a b <o Spsi a' b'.
Lemma Spsi_cpb a a' b b':
  ordinalp a ->
  a = a' -> b <o b' -> Spsi a b <o Spsi a' b'.
```

```
Lemma Spsi_cpc a a' b b':
  ordinalp b ->
  a' <o a -> Spsi a b <=o b' -> Spsi a b <o Spsi a' b'.
```

11.17 Initial ordinals

We start with section with the Bourbaki construction of initial ordinals (Exercice 6.10).

If **N** is the set of all integers, it can be well-ordered by $\leq_{\text{Card}}$; the ordinal of this set is denoted by ω or ω_0. If n is an integer, the set of all integers less than n is also well-ordered by the same relation; let n' be its ordinal; since n' has the same cardinal as n, one may identify the cardinal n and this ordinal n'. More generally, let $\mathfrak{a}$ be a cardinal. By Zermelo's theorem there exists a well-ordering on $\mathfrak{a}$; let α be it ordinal. Denote by $\mathrm{O}(\alpha)$ the set of ordinals $< \alpha$ and by $\mathrm{O}'(\alpha)$ the set of ordinals $\leq \alpha$. Consider the relation "$\mathrm{S}(\xi)$: ξ is an ordinal, and $\mathrm{Card}(\xi) < \mathfrak{a}$". If $\alpha \leq \xi$ there is an injection $\alpha \to \xi$, so that $\mathfrak{a} = \mathrm{Card}(\alpha) \leq \mathrm{Card}(\xi)$. Thus, $\mathrm{S}(\xi)$ implies $\xi < \alpha$, so that $\xi \in \mathrm{O}(\alpha)$. As a consequence there exists a set $\mathrm{W}(\mathfrak{a})$ formed of all ordinals ξ such that $\mathrm{S}(\xi)$. Since $\mathfrak{a} < 2^{\mathfrak{a}}$ there is a set $\mathrm{W}'(\mathfrak{a})$ formed of all all ordinals ξ such that $\mathrm{Card}(\xi) \leq \mathfrak{a}$.

Let α be a non-zero ordinal. One can define by transfinite induction on $\mathrm{O}'(\alpha)$ a function f_α such that $f_\alpha(0) = \omega_0$ and $f_\alpha(\xi) = \sup\bigcup_{\eta<\xi} \mathrm{W}'(\mathrm{Card}(f_\alpha(\eta)))$ when $0 < \xi \leq \alpha$. Note that $f_\alpha(\xi)$ is independent of α and it makes sense to define $\omega_\alpha = f_\alpha(\alpha)$. As $\mathrm{Card}(f_\alpha(\xi))$ is a strictly increasing function of ξ, the quantity $\aleph_\alpha = \mathrm{Card}(\omega_\alpha)$ is strictly increasing. The quantity ω_α is called the initial ordinal of index α, and $\aleph_\alpha$ is called the aleph of index α.

If $\mathfrak{a}$ is an infinite cardinal, the least upper bound of $\mathrm{W}(\mathfrak{a})$ has the form ω_α and $\mathfrak{a} = \aleph_\alpha$. In particular, every infinite cardinal is an aleph. The mapping $\aleph$ is an order isomorphism $\mathrm{O}'(\alpha) \to \mathrm{W}'(\aleph_\alpha)$. The relation (11.75) holds, the next cardinal after $\aleph_\alpha$ is $\aleph_{\alpha+1}$ and ω_ξ is normal.

If we consider von Neumann ordinals, this construction is just the enumeration of the collection of all infinite cardinals. Let's first show that it is closed and unbounded.

```
Lemma iclosed_non_coll_infinite_c:
  (iclosed_proper infinite_c).
```

We let $x \mapsto \omega_x$ be the enumeration of the infinite ordinals. If α is an ordinal, then ω_α is call the *initial ordinal of index* α. Considered as a cardinal, it is denoted $\aleph_\alpha$ and called the *aleph of index* α. We give a name to these quantities when $\alpha = 0$, 1 or 2.

```
Definition omega_fct := ordinalsf infinite_c.

Notation "\omega" := omega_fct.
Notation "\aleph" := omega_fct (only parsing).
Definition omega1 :=  \omega \1o.
Definition aleph1 :=  \aleph \1o.
Definition omega2 :=  \omega \2o.
Definition aleph2 :=  \aleph \2o.
```

We consider now some properties that follow trivially from the properties of enumeration. In particular, ω_0 is the quantity introduced a long time ago. The next cardinal after $\aleph_\alpha$ is $\aleph_{\alpha+1}$. In particular, the quantity denoted aleph-one above is $\aleph_1$.

```
Lemma aleph_normal: normal_ofs \omega.
Lemma aleph0E: \omega \0o = omega0.
```

```
Lemma CIS_aleph x: ordinalp x -> infinite_c (\aleph x).
Lemma CS_aleph x: ordinalp x -> cardinalp (\aleph x).
Lemma OS_aleph x: ordinalp x -> ordinalp (\omega x).
Lemma aleph_pr5 x: ordinalp x -> omega0 <=o (\omega x).
Lemma cnextE n: ordinalp n ->
  cnext (\aleph n) = \aleph (osucc n).
Lemma Cantor_omega_pr3: aleph1 = aleph_one.
```

The continuum hypothesis (CH) asserts $\aleph_1 = 2^{\aleph_0}$; and the generalized continuum hypothesis (GCH) asserts that $\aleph_{\alpha+1} = 2^{\aleph_\alpha}$ for every ordinal α. These statements are undecidable.

```
Definition ContHypothesis:= cnext omega0 = \2c ^c omega0.
Definition GenContHypothesis:= forall x, infinite_c x -> cnext x = \2c ^c x.
```

These lemmas express that $x \mapsto \omega_x$ is strictly increasing.

```
Lemma aleph_lt_lto x y: x <o y -> \omega x <o \omega y.
Lemma aleph_le_leo x y: x <=o y -> \omega x <=o \omega y.
Lemma aleph_leo_le x y: ordinalp x -> ordinalp y ->
  \omega x <=o \omega y -> x <=o y.
Lemma aleph_inj x y: ordinalp x -> ordinalp y ->
  \omega x = \omega y -> x = y.
Lemma aleph_lto_lt x y: ordinalp x -> ordinalp y ->
  \omega x <o \omega y -> x <o y.
Lemma aleph_pr6 x: ordinalp x -> x <=o \omega x.
```

These lemmas express that $x \mapsto \aleph_x$ is strictly increasing.

```
Lemma aleph_le_lec x y: x <=o y -> \aleph x <=c \aleph y.
Lemma aleph_lt_ltc x y: x <o y -> \aleph x <c \aleph y.
Lemma aleph_lec_le x y: ordinalp x -> ordinalp y ->
  \aleph x <=c \aleph y -> x <=o y.
Lemma aleph_ltc_lt x y: ordinalp x -> ordinalp y ->
  \aleph x <c \aleph y -> x <o y.
Lemma aleph_nz x: ordinalp x -> \aleph x <> \0c.
Lemma aleph_nz1 x: ordinalp x -> \0c <c \aleph x.
```

Recall that the enumeration is the inverse functional of some quantity; in the case of infinite cardinals, we call this the *ordinal index* of x. Every infinite cardinal has an index α and $x = \aleph_\alpha$. On the other hand, the index of $\aleph_\alpha$ is α. If x is an infinite cardinal, we have $x < 2^x$, so that 2^x is at least the next cardinal after x (and GCH says there is equality).

```
Definition ord_index:=
    ordinalr (fun x y => [/\ infinite_c x, infinite_c y & x <=o y]).


Lemma ord_index_pr1 x: infinite_c x ->
  (ordinalp (ord_index x) /\ \aleph (ord_index x) = x).
Lemma ord_index_pr n: ordinalp n ->
   ord_index (\aleph n) = n.

Lemma aleph_pr10a x y: ordinalp x ->
   y <c \aleph (osucc x) -> y <=c \aleph x.
Lemma aleph_pr10b x y: ordinalp x ->
```

```
  \aleph x <c y -> \aleph (osucc x) <=c y.
Lemma aleph_pr10c x: ordinalp x ->
  \aleph x <c \aleph (osucc x).
Lemma aleph_limit x: ordinalp x -> limit_ordinal (\omega x).
Lemma aleph_pr12b x z :
  ordinalp x -> ordinalp z -> \aleph x <c cardinal z ->
  \aleph (osucc x) <=o z.
Lemma aleph_pr12c x z:
  ordinalp x -> ordinalp z -> cardinal z <=c \aleph x ->
  z <o \aleph (osucc x).
Lemma aleph_pr12d x z:
  ordinalp x -> z <o (\aleph (osucc x)) ->
  cardinal z <=c (\aleph x).
Lemma aleph_pr12e x: ordinalp x ->
  \aleph (osucc x) <c \2c ^c (\2c ^c (\aleph x)).
Lemma aleph_pr12f a: ordinalp a -> \aleph (osucc a) <=c \2c ^c (\aleph a).
```

11.18 Cardinal Cofinality

Let x be a cardinal. If $X = (X_i)_{i\in I}$ is a family of cardinals, we write $(X) < x$ by abuse of language instead of "for all $i \in I$, $X_i < x$". The least cardinal of the domain I such that $\sum_{i\in I} X_i = x$ for at least one such family, is called the *cofinality* of x.

```
Definition csum_of_small0 x f:=
  fgraph f /\ allf f (fun z => z <=c x).
Definition csum_of_small1 x f:=
  fgraph f /\  allf f (fun z => z <c x).
```

We denote by $s(X)$ the sum $\sum_{i\in I} X_i$ and by $d(X)$ the domain. Let f be a bijection, $Y_i = X_{f(i)}$. Then $s(Y) = s(X)$. Moreover $(X) < x$ implies $(Y) < x$. So, in the definition of cofinality, we may assume that I is a cardinal (take for f the bijection between I and its cardinal). If $(X) \le x$ (in particular, if $(X) < x$), we have $s(X) \le xd(X)$. Assume $(X) < x$, $d(X) < x$ and x is an infinite cardinal successor; then $s(X) < x$. Assume that the supremum t of the X_i is infinite and $d(X) \le t$; in this case $s(X) = t$ (we give a variant of this property).

```
Lemma csum_commutative1 f x:
  csum_of_small1 x f -> exists g,
   [/\ csum_of_small1 x g, domain g = (cardinal (domain f)),
     csum f = csum g & range g = range f].
Lemma csum_of_small_b1 x f: csum_of_small0 x f ->
  csum f <=c (x *c (domain f)).
Lemma csum_of_small_b2 x f: csum_of_small1 x f ->
  csum f <=c (x *c (domain f)).
Lemma csum_of_small_b3 x f:
  csum_of_small1 x f ->
  (exists2 n, ordinalp n & x = \aleph (osucc n)) ->
  (cardinal (domain f)) <c x ->  (csum f) <c x.
Lemma csum_of_small_b4 f (s:= \csup (range f)) :
   fgraph f -> cardinal_fam f -> infinite_c s ->
   (cardinal (domain f) <=c s) -> csum f = s.
Lemma csum_of_small_b5 f (s:= \csup (range f)) :
   fgraph f -> cardinal_fam f ->
   (exists i, [/\ inc i (range f), infinite_c i & cardinal (domain f) <=c i])
   -> csum f = s.
```

We define now cf(x) as the least ordinal z which is the domain of a family X such that $s(\mathrm{X}) = x$ and (X) < x. If $x = 0$, it is obvious zero, if $x = 1$ it does not exists (note; with our definition, if p is always false, the least ordinal such that p is equal to zero). Otherwise we can take $\mathrm{X}_i = 1$, and get cf(x) ≤ x. Removing zeroes from X_i does not change the sum, but reduces the domain, so that we may assume $\mathrm{X}_i \neq 0$. If x is infinite, we may assume $\mathrm{X}_i \geq 2$ (since replacing X_i by $\mathrm{X}_i + 2$ does not change the value of the sum).

```
Definition cofinality_c_ex x z :=
   exists f, [/\ csum_of_small1 x f, domain f = z & csum f = x].
Definition cofinality_c x:=
  least_ordinal (cofinality_c_ex x) x.

Lemma cofinality_c_of0: cofinality_c \0c = \0c.
Lemma cofinality_c_of1: cofinality_c \1c = \0c.
Lemma cofinality_c_rw x (y:= cofinality_c x): \2c <=c x ->
  [/\ y <=c x,
      cofinality_c_ex x y
     & (forall z, ordinalp z -> (cofinality_c_ex x z) -> y <=o z)].
Lemma cofinality_c_pr2 x: \2c <=c x ->
 exists f, [/\ csum_of_small1 x f, domain f = (cofinality_c x), csum f = x
       & (allf f (fun z => z <> \0c)) ].
Lemma cofinality_c_pr3 x: infinite_c x ->
 exists f, [/\ csum_of_small1 x f, domain f = (cofinality_c x), csum f = x
       & (allf f (fun z => \2c <=c z)) ].
```

As mentioned above, we can always replace the domain by its cardinal; this says that a cofinality is a cardinal. Note that $x = (x-1)+1$, so that if x is finite, its cofinality is two, but the cofinality of an infinite cardinal is infinite. An infinite cardinal is *regular* if it is equal to its cofinality (in particular 2 is not considered regular). We characterize regular cardinals by the fact that, for every family X with (X) < x, if $\mathrm{card}(d(\mathrm{X})) < x$ then $\sum \mathrm{X}_i < x$.

Note: assume that x is an infinite regular cardinal, and consider a family of ordinals λ_i indexed by some set I; we assume each λ_i, as well as I, less than x; then the ordinal sum $\sum_{i\in\mathrm{I}} \lambda_i$ is less than x. As x is a cardinal, this is the same as $\mathrm{card}(\sum \lambda_i) <_{\mathrm{card}} x$. This cardinal is the cardinal sum of the cardinals of the λ_i, but each λ_i has cardinal $< x$, and $\mathrm{card}(\mathrm{I}) <_{\mathrm{card}} x$ holds as well. The result holds by regularity of x.

```
Definition regular_cardinal x :=
  infinite_c x /\ cofinality_c x = x.

Lemma cofinality_c_small x: cardinalp x -> (cofinality_c x) <=c x.
Lemma CS_cofinality_c x: cardinalp ->
  cardinalp (cofinality_c x).
Lemma cofinality_c_finite x: \2c <=c x -> finite_c x ->
   cofinality_c x = \2c.
Lemma cofinality_infinite x: infinite_c x ->
  infinite_c (cofinality_c x).
Lemma infinite_regularP x: infinite_c x ->
  (regular_cardinal x <->
   forall f, csum_of_small1 x f -> cardinal(domain f) <c x -> csum f <c x).
Lemma regular_alt_prop X x:
  fgraph X -> allf X (fun z => z <o x) -> (domain X) <o x ->
  regular_cardinal x ->
  osum (ordinal_o (domain X)) X <o x.
```

For instance $\aleph_0$, $\aleph_1$ and $\aleph_{\alpha+1}$ are regular. (a finite sum of finite numbers is finite, a sum of quantities $\le \aleph_\alpha$ indexed by a set with cardinal $\le \aleph_\alpha$ is also $\le \aleph_\alpha$). If x is regular, E a set of cardinals all $< x$, if card(E) $< x$, then sup(E) $< x$. This holds in particular when x is a cardinal successor (and this property has been used a lot).

```
Lemma regular_cardinal_omega: regular_cardinal omega0.
Lemma regular_initial_successor x: ordinalp x ->
   regular_cardinal (\aleph (osucc x)).
Lemma regular_cnext x: infinite_c x ->
   regular_cardinal (cnext x).
Lemma regular_cardinal_aleph1: regular_cardinal aleph1.
Lemma regular_sup X b:
   regular_cardinal b -> cardinal X <c b ->
   (forall i, inc i X -> i <c b) -> \csup X <c b.
```

Let Z_i be the complement of $\aleph_i$ in $\aleph_{i+1}$. Each cardinal is either in ω or in exactly one Z_i. Moreover Z_i has cardinal $\aleph_{i+1}$ (this generalizes the fact that the set of infinite countable ordinals has the same cardinal as the set of all countable ordinals).

```
Definition aleph_succ_comp x :=
   (\aleph (osucc x)) -s (\aleph x).
Lemma aleph_succ_P1 x: ordinalp x ->
  (forall t, inc t  (aleph_succ_comp x) <->
     ((\aleph x) <=o t /\ t <o (\aleph (osucc x)))).
Lemma aleph_succ_pr2 x: ordinalp x ->
  cardinal (aleph_succ_comp x) = \aleph (osucc x).
Lemma aleph_succ_pr3 x y: ordinalp x -> ordinalp y ->
  x = y \/ disjoint (aleph_succ_comp x) (aleph_succ_comp y).
Lemma aleph_sum_pr1 x: ordinalp x ->
  inc x omega0 \/ exists2 y, ordinalp y & inc x (aleph_succ_comp y).
```

Since $\aleph_\alpha$ is the disjoint union of $\aleph_0$ and the Z_i, computing cardinals gives:

$$\aleph_\alpha = \aleph_0 + \sum_{\beta<\alpha} \aleph_{\beta+1}. \tag{11.73}$$

If we replace $\aleph_{\beta+1}$ by $\aleph_\beta$ and omit $\aleph_0$, we get a quantity which is $\le \aleph_\alpha$; so if we add $\aleph_\alpha$ we get $\aleph_\alpha$. Thus

$$\aleph_\alpha = \sum_{\beta\le\alpha} \aleph_\beta. \tag{11.74}$$

Assume that α is a limit ordinal, E is subset of α, whose supremum is α. We have

$$\aleph_\alpha = \sum_{\beta\in E} \aleph_\beta, \qquad (\sup E = \alpha,\ \text{limit ordinal}). \tag{11.75}$$

(note that card(E) $\le \aleph_\alpha = x$, and the supremum of the $\aleph_\beta$ which is some $\aleph_\gamma$ cannot be $< \aleph_\alpha$). In particular

$$\aleph_\alpha = \sum_{\beta<\alpha} \aleph_\beta \qquad (\alpha \text{ limit ordinal}). \tag{11.76}$$

Note that, if α is a successor, say $\gamma+1$, then the sum in (11.76) is $\aleph_\gamma$ by (11.74). Relations (11.75) and (11.76) can be expressed as: $\alpha \mapsto \aleph_\alpha$ is a normal cardinal functional symbol.

```
Lemma aleph_sum_pr2 x: ordinalp x ->
  \aleph x = omega0 +c csumb x (fun z => \aleph(osucc z)).
Lemma aleph_sum_pr3 x: ordinalp x ->
  \aleph x = csumb (osucc x) \aleph.
Lemma aleph_sum_pr4 x E: limit_ordinal x ->
  sub E x -> \csup E = x ->
  \aleph x = csumb E \aleph.
Lemma aleph_sum_pr5 x: limit_ordinal x ->
  \aleph x = csumb x \aleph.
```

We shall consider, in what follows, increasing functional graphs X, indexed by an ordinal I. We assume that $i \le j$ implies $X_i \le X_j$ (in the weak case) and $i < j$ implies $X_i < X_j$ (in the strong case). Here X_i can be a cardinal or an ordinal.

```
Definition fg_Mle_lec X :=
  (forall a b, inc a (domain X) -> inc b (domain X) -> a <=o b ->
    Vg X a <=c Vg X b).
Definition fg_Mle_leo X :=
  (forall a b, inc a (domain X) -> inc b (domain X) -> a <=o b ->
    Vg X a <=o Vg X b).
Definition fg_Mlt_lto X :=
  (forall a b, inc a (domain X) -> inc b (domain X) -> a <o b ->
    Vg X a <o Vg X b).
Definition fg_Mlt_ltc X :=
  (forall a b, inc a (domain X) -> inc b (domain X) -> a <o b ->
    Vg X a <c Vg X b).
Definition ofg_Mlt_lto X := [/\ fgraph X, ordinal_fam X & fg_Mlt_lto X].
Definition ofg_Mle_leo X := [/\ fgraph X, ordinal_fam X & fg_Mle_leo X].
```

We shall often assume, in the strict increasing case, that I is a limit ordinal, so that $i+1 \in I$ whenever $i \in I$. This implies $X_i < X_{i+1}$, so that X_i is an ordinal and X_i is weakly increasing.

```
Lemma ofg_Mle_leo_os X: fgraph X -> ordinal_fam X -> ordinal_set (range X).
Lemma ofg_Mle_leo_p1 X:
  fgraph X -> fg_Mle_leo X -> ordinalp (domain X) ->  ofg_Mle_leo X.
Lemma ofg_Mlt_lto_p1 X:
  fgraph X -> fg_Mlt_lto X -> limit_ordinal (domain X) ->
  forall u, inc u (domain X) ->
    (inc (osucc u) (domain X) /\ Vg X u <o Vg X (osucc u)).
Lemma ofg_Mlt_lto_p2 X:
  fgraph X -> fg_Mlt_lto X -> limit_ordinal (domain X) ->  ofg_Mle_leo X.
Lemma ofg_Mlt_lto_p3 X:
  ofg_Mlt_lto X -> limit_ordinal (domain X) -> ofg_Mle_leo X.
```

Let $\alpha = \sup X_i$. Then α is a limit ordinal if (1) X is weakly increasing and I is non-empty and α is not in the range of X or (2) X is strictly increasing and I is limit. In this case we have $\sup \aleph_{X_i} = \aleph_\alpha$.

```
Lemma increasing_sup_limit1 X (a:= \osup (range X)):
  ofg_Mle_leo X ->nonempty (domain X) ->
  (allf X (fun t => t <> a)) ->
  limit_ordinal a.
Lemma increasing_sup_limit2 X:
  ofg_Mlt_lto X -> limit_ordinal (domain X) ->
  limit_ordinal (\osup (range X)).
```

```
Lemma sup_range_aleph X:
  fgraph X -> ordinal_fam X -> nonempty (domain X) ->
  \osup (range (Lg (domain X) (fun z => \aleph (Vg X z)))) =
    \aleph  (\osup (range X)).
```

A variant of (11.75) shows that

$$\sum \aleph_{\sigma_\xi} = \aleph_\alpha \qquad (\alpha = \sup \uparrow \sigma_\xi;\ \mathrm{dom}(\sigma_\xi)\ \text{limit}). \tag{11.77}$$

```
Lemma increasing_sup_limit4 X (Y:= Lg (domain X) (fun z => \aleph (Vg X z)))
  (a := \osup (range X)):
  ofg_Mlt_lto X -> limit_ordinal (domain X) ->
  [/\ limit_ordinal a, sub (range X) a & \csup (range Y) = \aleph a].
Lemma aleph_sum_pr6 X  (Y:= Lg (domain X) (fun z => \aleph (Vg X z)))
  (a := \osup (range X)):
  ofg_Mlt_lto X -> limit_ordinal (domain X) ->
   \aleph a = csum Y.
```

11.19 Ordinal cofinality

Ler E be an ordered set, and consider all cofinal subsets F of E. This means: whenever $x \in E$, there exists $y \in F$ such that $x \leq y$. If E has a greatest element z, then F is cofinal if and only if $z \in F$. If E is finite, then F is cofinal if and only if all maximal elements of E are in F. A subset of **N** is cofinal, if and only it is infinite. The cofinality of E defines how small a cofinal subset can be. A possible definition could be $\min \mathrm{card}(F)$.

The Bourbaki definition is $\min \mathrm{ord}(F)$. Here ord(F) could mean the order type of $\leq_F$ (the order induced by $\leq$ on F). It is not clear whether there is a least order type, so Bourbaki considers only well-ordered subsets F (so that ord(F) is an ordinal). Exercise 6.16 says that every totally ordered set has a cofinality. we simplify the situation by assuming E well-ordered. We show here that the cofinality exists.

```
Definition cofinality_aux r :=
   (Zo (powerset (substrate r))
      (fun z => cofinal r z /\ worder (induced_order r z))).
Definition cofinality' r := (fun_image (cofinality_aux r)
     (fun z => ordinal (induced_order r z))).

Lemma cofinality'_pr0 r : worder r ->
   (nonempty (cofinality' r) /\ ordinal_set (cofinality' r)).
Lemma cofinality'_pr2 r (y:= (intersection (cofinality' r))) :
  worder r ->
  (exists z, [/\ sub z (substrate r),cofinal r z, worder (induced_order r z) &
      y = ordinal (induced_order r z)]).
Lemma cofinality'_pr3 r z (y:= (intersection (cofinality' r))):
  worder r -> sub z (substrate r) -> cofinal r z ->
  y <=o ordinal (induced_order r z).
```

We define the cofinality (according to Bourbaki) of an ordinal x as the cofinality of its well-ordering (x). The properties stated here are trivial.

```
Definition cofinality_alt x :=
  intersection (cofinality' (ordinal_o x)).
```

```
Lemma cofinality_pr1 x (r:= ordinal_o x):
  ordinalp x ->
 (exists z, [/\ sub z x, cofinal r z, worder (induced_order r z) &
   (cofinality_alt x) = ordinal (induced_order r z)]).
Lemma cofinality_pr2 x  (r:= ordinal_o x) z:
  ordinalp x -> sub z x -> cofinal r z ->
  (cofinality_alt x) <=o ordinal (induced_order r z).
```

We define the *cofinality* of x, denoted $\mathrm{cf}(x)$, as the least ordinal y such that there exists a *cofinal* function $f : y \to x$; this means a function f such if $t \in x$ there exists $z \in y$ such that $t \le f(z)$.

```
Definition cofinal_function f x y :=
  function_prop f y x /\
  (forall t, inc t x -> exists2 z, inc z y & t <=o Vf f z).

Definition cofinal_function_ex x y:= exists f, cofinal_function f x y.
Definition cofinality x := least_ordinal (cofinal_function_ex x) x.
```

Since the identity function is cofinal, the cofinality of x is well-defined and is an ordinal $\le x$. If x is zero, its cofinality is zero, and conversely. The cofinality of x is one if and only if x is a successor. In all other cases, the cofinality is a limit ordinal.

```
Lemma cofinal_function_pr2 a: ordinalp a -> cofinal_function_ex a a.
Lemma cofinality_rw a (b:= \cf a) :
  ordinalp a -> [/\ ordinalp b, cofinal_function_ex a b &
    forall z, ordinalp z -> cofinal_function_ex a z ->  b <=o z].
Lemma OS_cofinality a: ordinalp a -> ordinalp (\cf a).
Lemma cofinality_pr3 a: ordinalp a -> \cf a <=o a.
Lemma cofinality0: \cf \0o = \0o.
Lemma cofinality_n0 x: ordinalp x -> x <> \0o -> \cf x <> \0o.
Lemma cofinality1: \cf \1o = \1o.
Lemma cofinality_succ x: ordinalp x -> \cf (osucc x) = \1o.
Lemma cofinality_limit1 n: ordinalp n ->
  (\cf n = \1o <-> exists2 m, ordinalp m & n = osucc m).
Lemma cofinality_limit2 x (y:= \cf x): ordinalp x ->
  y = \0o \/ y = \1o \/ limit_ordinal y.
Lemma cofinality_limit3 x: limit_ordinal x -> limit_ordinal (\cf x).
Lemma cofinality_limit4 x: limit_ordinal x -> omega0 <=o \cf x.
```

If x is any ordinal, there exists a strictly increasing cofinal function $f : \mathrm{cf}(x) \to x$. We may even assume f normal and $f(0) = 0$.

Let $y = \mathrm{cf}(x)$. If $x = 0$, then $y = 0$ and the only function $y \to x$ is the empty function. If x is a successor, say $x = z^+$, then $y = 1$ and the only cofinal function $y \to x$ satisfies $f(0) = z$. In all other cases, y is a limit ordinal. In this case, we can assume $f(0) = 0$. Consider a cofinal function $g : y \to x$; its range is a cofinal subset X of x. There is an order-isomorphism $\mathrm{ord}(\mathrm{X}) \to \mathrm{X}$, this is a strictly increasing cofinal function, and $\mathrm{ord}(\mathrm{X}) = y$. To say that this function is normal is the same as to say that X is closed. Since this property may be false, the existence of a normal function f is non-trivial.

We construct the function f by transfinite induction as follows: for $a \in y$, if h is the restriction of f to a, then either the source of h is b^+ and $f(a) = \sup(g(b), h(b)+1)$ or $f(a)$ is the supremum of the image of h (the definition is not completely trivial; we define $f(a)$ to be zero if either b is not in the source of h, or the supremum is not in x; what we get is a function with values in x). A careful analysis shows that the source of h is a, and $f(b+1) = \sup(g(b), f(b)+1)$, and that the image of h is a subset of x. Note that h is not cofinal (by definition of y). This shows $f(a) = \sup_{b<a} f(b)$ for any non-successor a. Thus f is a normal function.

We deduce that the cofinality X of x (according to Bourbaki) is $\mathrm{cf}(x)$. In fact, if f is strictly increasing, and cofinal with range z, then f induces an order isomorphism between $\mathrm{cf}(x)$ and z. Conversely, X is the ordinal of some cofinal z, so that there exists an isomorphism $\mathrm{X} \to z$, that induces a function $\mathrm{X} \to x$ with range z and this function is obviously cofinal.

```
Lemma cofinality_pr4 x (y:= \cf x): ordinalp x ->
  exists2 f, (cofinal_function f x y /\ normal_function f y x) &
    (inc \1o y -> Vf \f 0o = \0o).  (* 95 *)
Lemma cofinality_sd a: ordinalp a -> (* 86 *)
  (cofinality a) = (cofinality_alt a).
```

Let f be a normal OFS, x an ordinal, which is not a fix-point of f. Let $y = \mathrm{N}_f(x)$ be the next fix-point of f after x. This is the supremum of a sequence x_i indexed by $\mathbf{N}$; thus, its cofinality is ω. Assume now that $f : \mathrm{X} \to \mathrm{X}$ is a function, $x \in \mathrm{X}$. We have shown that $y = \mathrm{X}$ or $y \in \mathrm{X}$; if the cofinality of X is not ω, then the first case is excluded, and y is a fix-point of f. Thus f has arbitrarily large fix-points.

```
Lemma cofinality_least_fp_normal x y f:
  normal_ofs f -> f x <> x ->  least_fixedpoint_ge f x y ->
  \cf y = omega0.
Lemma normal_function_fixpoints x f:
  ordinalp x -> normal_function f x x ->
  (\cf x <> omega0) ->
  (forall a, inc a x -> exists b, [/\ inc b x, a <=o b & Vf f b = b]).
```

We say that an ordinal x is *regular* if $\mathrm{cf}(x) = x$, and *singular* otherwise.

We have: whenever $f : x \to y$ is cofinal and strictly increasing, then $\mathrm{cf}(x) = \mathrm{cf}(y)$. Proof. Let f_x and f_y be cofinal mappings $\mathrm{cf}(x) \to x$ and $\mathrm{cf}(y) \to y$ respectively. Composing f and f_x gives a cofinal mapping $\mathrm{cf}(x) \to y$, so that $\mathrm{cf}(y) \le \mathrm{cf}(x)$. On the other hand, if $a \in \mathrm{cf}(y)$, there is $t \in x$ such $f(t) \ge f_y(a)$. Call it $g(x)$. If $a \in x$, there is $b \in \mathrm{cf}(y)$ such that $f(x) \le f_y(b)$, thus $x \le g(b)$. This shows that g is a cofinal function.

If we take $x = \mathrm{cf}(y)$ it follows $\mathrm{cf}(\mathrm{cf}(y)) = \mathrm{cf}(y)$. A cofinality is regular. Zero and one are regular: they are the only regular finite ordinals. The next regular ordinal is ω.

```
Definition regular_ordinal x := ordinalp x /\ \cf x = x.
Definition singular_ordinal x :=  ordinalp x /\ not (regular_ordinal x).

Lemma regular_0: regular_ordinal \0o.
Lemma regular_1: regular_ordinal \1o.
Lemma regular_finite x:
   regular_ordinal x -> [\/ x = \0o, x = \1o | omega0 <=o x].
Lemma cofinality_pr5 a b:
  ordinalp a -> ordinalp b ->
  (exists2 f, cofinal_function f b a & sincr_ofn f a) ->
  \cf a = \cf b.
Lemma cofinality_proj x: ordinalp x -> \cf (\cf x) = \cf x.
Lemma cofinality_reg x: ordinalp x ->
   regular_ordinal (\cf x).
Lemma regular_omega: regular_ordinal omega0.
```

We show here that a and $b+a$ have same cofinality (if a is non-zero), that a and $b\cdot a$ have same cofinality (if b is non-zero and a is limit), that a regular ordinal is indecomposable, and moreover, is 0, 1, ω or ω^y, where y is a limit ordinal.

```
Lemma cofinality_sum a b: ordinalp a -> \0o <o b ->
  \cf (a +o b) = \cf b.
Lemma cofinality_prod a b: \0o <o a -> limit_ordinal b ->
 \cf (a *o b) = \cf b.
Lemma cofinality_prod_omega a: \0o <o a -> \cf (a *o omega0) = omega0.
Lemma regular_indecomposable x:
   regular_ordinal x -> (x = \0o \/ indecomposable x).
Lemma regular_indecomposable1 x:
   regular_ordinal x -> [\/ x = \0o, x= \1o,  x =omega0  |
   exists2 y, limit_ordinal y & x = omega0 ^o y].
```

The cofinality y of an ordinal x is a cardinal (since the cardinal of y is an ordinal equipotent to it and $\le y$). It follows that the cofinality of a countable limit ordinal is ω_0. It follows that a regular ordinal is a cardinal. Let z be a cofinal subset of x, and z' the ordinal of z, well-ordered by $\le_{\mathrm{ord}}$. We have shown $\mathrm{cf}(x) \le z'$. Since $\mathrm{cf}(x)$ is a cardinal, we deduce $\mathrm{cf}(x) \le \mathrm{card}(z)$.

Consider an infinite cardinal x. We pretend that its cardinal cofinality y is $\mathrm{cf}(x)$. First, if $f : \mathrm{cf}(x) \to x$ is a cofinal function, then x is the union of all the $f(i)$, so that $\mathrm{card}(x) \le \sum \mathrm{card}(f(i))$. As $\mathrm{cf}(x) \le x$ and x is infinite, we get $x = \sum \mathrm{card}(f(i))$ so that y is at most $\mathrm{cf}(x)$. Conversely, consider al family X_i, (X) $< x$ and $s(X) = x$. If $\sup X_i = x$, this gives a cofinal function. Otherwise $\sum_{i\in y} X_i = x$, says $y \ge x$, this implies $\mathrm{cf}(x) \le y$.

It follows that, for any cardinal x, $\mathrm{cf}(x)$ is zero, one, or a regular cardinal.

```
Lemma CS_cofinality x: ordinalp x -> cardinalp (\cf x).
Lemma cofinality_limit_countable x: limit_ordinal x -> countable_ordinal x ->
   \cf x = omega0.
Lemma cofinality_pr8 x z: ordinalp x -> sub z x -> \osup z = x ->
   \cf x <=c cardinal z.
Lemma regular_is_cardinal x: regular_ordinal x -> cardinalp x.
Lemma cofinality_infinite_limit x:
   limit_ordinal x -> infinite_c (\cf x).
Lemma cofinality_infinite_cardinal x:
   infinite_c x -> infinite_c (\cf x).
Lemma cofinality_card x: infinite_c x ->
  cofinality_c x = cofinality x.  (* 65 *)
Lemma cofinality_small x: infinite_c x -> \cf x <=c x.
Lemma cofinality_regular x (y:= cofinality x): ordinalp x ->
  [\/ y = \0c, y = \1c | regular_cardinal y].
```

If α is an ordinal, then the cofinality of ω_α is an initial ordinal ω_β. Exercise 16c concludes with: $\beta \le \alpha$ and ω_α is regular if and only if $\alpha = \beta$.

```
Lemma cofinality_index a: ordinalp a ->
   ord_index (\cf (\omega a)) <=o a.
Lemma cofinality_index_regular a (x :=\omega a) : ordinalp a ->
   (regular_ordinal x <-> ord_index (\cf x) = a).
```

If α is a limit ordinal then $\mathrm{cf}(\aleph_\alpha) = \mathrm{cf}(\alpha)$. Proof: let c be the cardinal cofinality of $\aleph_\alpha$; consider a cofinal function $f : \mathrm{cf}(\alpha) \to \alpha$. Since α is limit, it is the supremum of the $f(t)$. Consider now $g(t) = \aleph_{f(t)}$. By (11.75), $\sum g(t) = \aleph_\alpha$. This implies $c \le \mathrm{cf}(\alpha)$. Conversely, assume

$\aleph_\alpha = \sum_{i\in I} x_i$, where card(I) = c. Write $x_i = \aleph_{y_i}$ (if x_i is finite, we let $y_i = 0$). Let $z = \sup y_i$. If $z = \alpha$, we have a cofinal function, and cf(α) ≤ c. Otherwise $x_i \le \aleph_\alpha$ (even when x_i is finite), so that card(I) = $\aleph_\alpha$ and cf(α) ≤ c again.

Thus, if $\alpha < \aleph_\alpha$, then $\aleph_\alpha$ is singular. In particular, $\aleph_\omega$ is the least singular cardinal.

```
Definition singular_cardinal x :=
  infinite_c x /\ \cf x <> x.

Lemma regular_cardinalP x:
  regular_cardinal x <-> infinite_c x /\ \cf x = x.
Lemma regular_initial_limit0 x: limit_ordinal x ->
  \cf (\aleph x) <=o \cf x.
Lemma regular_initial_limit1 x: limit_ordinal x -> (* 74 *)
  \cf (\aleph x) = \cf x.
Lemma regular_initial_limit2 x: limit_ordinal x ->
  \cf (\aleph x) <=o x.
Lemma singular_limit n: ordinalp n ->
    singular_cardinal (\aleph n) -> limit_ordinal n.
Lemma regular_initial_limit3 x: limit_ordinal x ->
    x <o \aleph x -> singular_cardinal (\aleph x).
Lemma regular_initial_limit4: singular_cardinal (\aleph omega0).
Lemma regular_initial_limit5 x:
  singular_cardinal x -> (\aleph omega0) <=c x.
```

If α is a limit ordinal and $\aleph_\alpha$ is regular, it is called *weakly inaccessible*. A necessary condition is that $\alpha = \omega_\alpha$. Note that, if f is normal, x is not a fix-point of f, the next fix-point of f after x has cofinality ω, thus is ω or singular. In particular, for any x, the least $y > x$ such that $y = \aleph_y$ is singular.

```
Definition inaccessible_w x :=
  regular_cardinal x /\ (exists2 n, limit_ordinal n & x = \aleph n).

Lemma inaccessible_pr1 x:
    inaccessible_w x -> x = \aleph x.
Lemma inaccessible_uncountable x:
  inaccessible_w x -> aleph0 <c x.
Lemma cofinality_least_fp_normal2 x y f:
  normal_ofs f -> f x <> x ->  least_fixedpoint_ge f x y ->
  y = omega0  \/ singular_ordinal y.
Lemma cofinality_least_fp_normal3 x y:
  ordinalp x ->
  least_fixedpoint_ge \omega (osucc x) y -> singular_cardinal y.
```

We show here: assume that x and y are two regular ordinals, and f, g two strictly increasing cofinal functions with the same target z. Then $x = y$. This is the uniqueness part of Exercise 16c (this property holds for any ordered set z; for simplicity we consider here an ordinal). There exists $h : x \to y$ such that $g(h(t)) \ge f(t)$ for all t. Let $a \in y$; there is $b \in x$ with $f(b) \ge g(a)$. Thus $g(h(b)) \ge f(b) \ge g(a)$. Since g is strictly increasing we deduce $g(b) \ge a$. This shows that h is cofinal; thus $y \le x$. Exchanging x and y gives $x = y$.

```
Lemma regular_cofinal_si_unique z:
  uniqueness (fun x => regular_ordinal x
            /\ (exists2 f, cofinal_function fz x & sincr_ofn f z)).
```

11.20 Infinite products

We have, for any cardinals a and b,

$$2 \le a \le 2^b \text{ and } \omega \le b \implies a^b = 2^b.$$

In particular, $a^b = 2^b$ holds if $a \le b$ (by Cantor), if $a = b$ or if a is the cardinal successor of b.

```
Lemma infinite_power1 a b:  \2c <=c a -> a <=c (\2c ^c b) -> infinite_c b ->
   a ^c b = \2c ^c b.
Lemma infinite_power1_a a b:  \2c <=c a -> a <=c b -> infinite_c b ->
   a ^c b = \2c ^c b.
Lemma infinite_power1_b x: infinite_c x ->  x ^c x = \2c ^c x.
Lemma infinite_power1_c x: infinite_c x ->
  (cnext x) ^c x = \2c ^c x.
Lemma infinite_power1_d m: infinite_c m ->  omega0 ^c m = \2c ^c m.
```

The result of Exercise 3.3 (see explanations page 593) is known as König's Theorem: if $(x_i)_{i\in I}$ and $(y_i)_{i\in I}$ are two families of cardinals such that $x_i < y_i$, then $\sum x_i < \prod y_i$. We first state a similar result where $<$ is replaced by $\le$; in this case, we need $y_i \ge 2$.

```
Section Exercise3_3.
Variables f g :Set.
Hypothesis  (fgf: fgraph f)(fgg: fgraph g).
Hypothesis sd: domain f = domain g.
Hypothesis hg:  (allf g (fun x => \2c <=c x)).

Lemma compare_sum_prod0 a b : \2c <=c a -> \2c <=c b ->
   a +c b <=c a *c b.
Lemma compare_sum_prod1:  csum g <=c cprod g.   (* 72 *)

Lemma compare_sum_prod2 :
  (forall i, inc i (domain f) -> Vg f i <=c Vg g i) ->
  csum f <=c cprod g.
Lemma compare_sum_prod3 :
  (forall i, inc i (domain f) -> Vg f i <c Vg g i) ->
   csum f <c cprod g.  (* 64 *)
End Exercise3_3.

Lemma compare_sum_prod  f g:
  fgraph f ->  fgraph g -> domain f = domain g ->
  (forall i, inc i (domain f) -> Vg f i <c Vg g i) ->
  csum f <c cprod g.
```

One non-trivial question is: do we have an equivalent of (11.75), (11.76), or (11.77) for a product? We shall first show that we may assume our sequence in increasing order. Let's state:

Let E be a well-ordered set, $(X_i)_{i\in I}$ a sequence of elements of E. There exists an ordinal α and a bijection $f : \alpha \to I$ such that $X_{f(j)}$ is an increasing sequence.

We prove this result in the case where X_i is a family of cardinals, but the proof is the same in the general case. Take some w that is not in I and a cardinal β greater than the cardinal of I. Note that β is an ordinal, thus a well-ordered set. For every non-empty subset J of I, the set of all $\{f(j), j \in J\}$ is non-empty, thus has a least element y. By the axiom of choice, there is k_J

such that $y = f(k_J)$. We define by transfinite induction on β a function g as follows. For each x, let $R_x = \{g(i), i < x\}$ and $J = I - R_x$. If $J = \emptyset$, then $g(x) = w$, otherwise $g(x) = k_J$. Note that $g(x)$ is either w or in I, and that $g(i) = g(j)$ says that either $i = j$ or $g(i) = w$. Assume $g(i)$ is never w. Then g is injective, contradicting the assumption on the cardinal of β. Let α be the least ordinal such that $g(\alpha) = w$, and f the restriction of g to α. Then f is injective and $i \to X_{f(i)}$ increasing by definition of k_J. Let F be the image of f. This is a subset of $I \cup \{w\}$. By definition of α, $w \not\in F$ so that $F \subset I$. The complement in I of F is empty, so that $F = I$. This means that f is bijective.

```
Lemma exists_ordering X: cardinal_fam X ->  (* 150 *)
   exists f,
  [/\ bijection f, ordinalp (source f), target f = domain X &
     fg_Mle_lec (X \cf (graph f))].
```

If $x_i \ge 2$, we have $\sup x_i \le \sum x_i \le \prod x_i$. A slight modification of the argument shows $\sup x_i \le \prod x_i$ provided that no factor is zero . The relation $\prod x_i \le (\sup x_i)^{\mathrm{card(I)}}$ is obvious. We prove here a variant: we assume x_i infinite, so that $x_i = \aleph_{\sigma_i}$; we set $\alpha = \sup \sigma_i$, so that $\sup x_i = \aleph_\alpha$. We have then $\prod x_i \le \aleph_\alpha^{\mathrm{card(I)}}$.

```
Definition cardinal_pos_fam g := (allf g (fun z => \0c <c z)).

Lemma compare_sum_prod5  x  :
  fgraph x -> cardinal_pos_fam x ->
  \csup (range x) <=c cprod x.

Lemma infinite_increasing_power_bound0 X:
  fgraph X -> cardinal_fam X ->
  cprod X <=c (\csup (range X)) ^c (cardinal (domain X)).
Lemma infinite_increasing_power_bound1 X:
  fgraph X -> ordinal_fam X ->
  cprod (Lg (domain X) (fun z => \aleph (Vg X z))) <=c
  \aleph (\osup (range X)) ^c (cardinal (domain X)).
```

We have shown $p \le S^I$, where p is the product of the x_i and S the supremum. A non-trivial question is: can we have equality? It is quite possible that $x_i = a$ on J, $x_i = b$ on K, where $I = J \cup K$, K is rather small (card(K) < card(J) = card(I)), and $a < b$, $a^J < b^I$ and $b^K < b^I$, and the product is less then b^I.

We consider here the following situation. Every x_i is infinite so that $x_i = \aleph_{\sigma_i}$ for some functional term σ. The sequence x_i is strictly increasing (so σ_i is strictly increasing) and the domain is a limit ordinal β (this means that the sequence has no greatest element). We have then

$$\aleph_\alpha < \prod \aleph_{\sigma_\xi} \le \aleph_\alpha^{\mathrm{Card}(\beta)} \le 2^{\aleph_\alpha}.$$

Proof: the nontrivial point is the first inequality. We have $\sum x_i < \prod x_{i+1}$, by König, and $\prod x_{i+1} \le \prod x_i$ (since β is a limit ordinal, every factor x_{i+1} of the first product is a factor of the second product). We shall prove in section 11.23 that if GCH holds, then a more precise result hols.

$$\prod_{\xi<\beta} \aleph_{\sigma_\xi} = \aleph_\alpha^{\mathrm{Card}(\beta)} = \aleph_{\alpha+1} \qquad (\alpha = \sup \sigma_\xi, \mathrm{GCH}). \tag{11.78}$$

```
Lemma infinite_increasing_power4 X
  (Y:= Lg (domain X) (fun z => \aleph (Vg X z)))
```

```
  (a := \osup (range X)):
  ofg_Mlt_lto X -> limit_ordinal (domain X) ->
  (cardinal (domain X) <=c \aleph a /\  \aleph a <c (cprod Y)).
Lemma infinite_increasing_power5 X
  (Y:= Lg (domain X) (fun z => \aleph (Vg X z)))
  (a := \osup (range X)):
  ofg_Mlt_lto X -> limit_ordinal (domain X) ->
  [/\ \aleph a <c cprod Y,
     cprod Y <=c \aleph a ^c cardinal (domain X) &
     \aleph a ^c cardinal (domain X) <=c \2c ^c \aleph a].
```

Assume x_i weakly increasing, the index set being an infinite cardinal; Then

(11.79) $$\prod_{i\in\mathrm{I}} x_i = (\sup x_i)^{\mathrm{I}} \qquad (\mathrm{I}\text{ infinite cardinal}).$$

Let $f : \mathrm{I}\times\mathrm{I} \to \mathrm{I}$ be a bijection. Set $y_{jk} = x_{f(j,k)}$ and $s = \sup x_i$. By associativity $\prod x_i = \prod_j(\prod_k y_{jk}) = \prod_j p_j$. It suffices to show, for every j, that $s \le \sup_k y_{jk}$ since $\sup_k y_{jk} \le p_j$. So, it suffices to show that, for every i, there is k such that $x_i \le y_{jk}$. This is false in general, but the function associated to the canonical ordering of pairs of ordinals (see page 112). satisfies this property.

```
Lemma infinite_increasing_power x (y := domain x):
   fgraph x -> infinite_c y -> cardinal_pos_fam x ->
   fg_Mle_lec x ->
   cprod x = (\csup (range x)) ^c y.   (* §9 *)
```

Let κ be an infinite cardinal. A direct application of König's theorem shows $\kappa < \kappa^{\mathrm{cf}(\kappa)}$. (if κ is finite, $\mathrm{cf}(\kappa)$ has to be interpreted as the cardinal cofinality, hence is equal to 2; the formula holds in this case). We deduce $\kappa < \mathrm{cf}(2^\kappa)$. In particular, the cofinality of $2^{\aleph_0}$ is $> \aleph_0$. Note that $\kappa < \mathrm{cf}(\lambda^\kappa)$ when $\lambda \ge 2$.

```
Lemma power_cofinality x: \2c <=c x -> x <c  x ^c (cofinality_c x).
Lemma power_cofinality1 x: infinite_c x -> x <c  x ^c (\cf x).
Lemma power_cofinality2 x: infinite_c x  -> x <c \cf (\2c ^c x).
Lemma power_cofinality3: aleph0 <c \cf (\2c ^c aleph0).
Lemma power_cofinality5 x y: \2c <=c x -> infinite_c y ->
  y <c \cf (x ^c y).
```

We state [Hausdorff, 1904]

(11.80) $$\aleph_{\alpha+1}^{\mathfrak{m}} = \aleph_\alpha^{\mathfrak{m}}\cdot\aleph_{\alpha+1} \qquad (\mathfrak{m}\ne 0).$$

We deduce (by induction when $\mathfrak{m}$ is infinite) [Bernstein]

(11.81) $$\aleph_n^{\mathfrak{m}} = 2^{\mathfrak{m}}\cdot\aleph_n \qquad (n\in\mathbf{N}, \mathfrak{m}\ne 0).$$

Assume first that x is an infinite cardinal, and $y < \mathrm{cf}(x)$. Let f be a function $y \mapsto x$. This function cannot be cofinal, so that the supremum of f is $< x$. Since x is a limit ordinal, the successor $s(f)$ of the supremum is also in x. By definition, if $t \in y$ we have $f(t) \in s(f)$. Let T_f be the function $y \to s(f)$ that takes the same values as f. Let U be the union of all z^y for $z < x$. We have shown $\mathrm{T}_f \in \mathrm{U}$. Since $f \mapsto \mathrm{T}_f$ is obviously injective, we deduce $x^y \le \mathrm{card}(\mathrm{U})$.

Let $z = \aleph_\alpha$, $x = z^+ = \aleph_{\alpha+1}$ and $y = \mathfrak{m}$. If $x \le y$, then $z^y = x^y = 2^y$ and (11.80) holds trivially. Otherwise, assume $y < x$. Since x is a cardinal successor, it is regular, and we may apply the previous result. It says $x^y \le \sum t^y \le z^y x$, as x is the number of terms in the sum and z an upper bound of these terms.

```
Lemma infinite_power7b x y: (* 58 *)
  infinite_c x -> y <c \cf x ->
  x ^c y <=c cardinal (unionb (Lg x (functions y))).
Lemma infinite_power2 n m (x:=\aleph n) (y:= \aleph (osucc n)):
   ordinalp n -> m <> \0c ->
   y ^c m = (x ^c m) *c y.
Lemma infinite_power2_bis x (y:= cnext x) m:
  infinite_c x ->  m <> \0c ->
   y ^c m = (x ^c m) *c y.
Lemma infinite_power3 n m (x:=\aleph n):
   natp n ->  m <> \0c ->
   x ^c m = (\2c ^c m) *c x.
```

We deduce: if $\aleph_1^{\aleph_0} = \aleph_1$, then $\aleph_n^{\aleph_0} = \aleph_n$, for every non-zero integer n.

We also deduce: if a is an infinite cardinal, then 2^a is the least infinite cardinal b such that $b^a = b$, and the least such that $b^a < (b^+)^a$. Note that $b^a = b$ holds if $b = 0$, $b = 1$, but for no other integer. On the other hand $b^a < (b+1)^a$ is true if and only if $b = 0$ or $b = 1$ (if $b \ge 2$, either b and $b+1$ are finite, so that $b^a < (b+1)^a = 2^a$, or b is infinite, and $b+1 = b$).

```
Lemma aleph_pow_prop1: aleph1 ^c aleph0 = aleph1 ->
  forall n, natp n -> n <> \0c -> (\aleph n) ^c aleph0 = \aleph n.

Lemma infinite_power4 a (b:= \2c ^c a): infinite_c a ->
 [/\ b ^c a = b, b ^c a <c (cnext b) ^c a,
    (forall c, \2c <=c c -> c ^c a = c -> b <=c c) &
     (forall c, infinite_c c -> c ^c a <c (cnext c) ^c a -> b <=c c)].
```

For each ordinal γ we have (Tarski, 1925, [20])

(11.82) $$\aleph_{\alpha+\gamma}^{\mathfrak{m}} = \aleph_\alpha^{\mathfrak{m}} \cdot \aleph_{\alpha+\gamma}^{\operatorname{card}(\gamma)} \qquad (\operatorname{card}(\gamma) \le \mathfrak{m}).$$

If we replace α by zero and rename γ into α, then (11.82) reads:

(11.83) $$\aleph_\alpha^{\mathfrak{m}} = 2^{\mathfrak{m}} \cdot \aleph_\alpha^{\operatorname{card}(\alpha)} \qquad (\operatorname{card}(\alpha) \le \mathfrak{m})$$

The proof is by transfinite induction on γ. The non-trivial case is when γ is limit. In this case we have $\aleph_{\alpha+\gamma}^{\mathfrak{m}} \le \prod_{t<\gamma} \aleph_{\alpha+t}^{\mathfrak{m}}$. We apply the induction property, and write the product as the product of two other factors; there is a trivial bound for the second factor. The LHS is then at most $\aleph_\alpha^{\mathfrak{m}\operatorname{card}(\gamma)} \cdot \aleph_{\alpha+\gamma}^{\operatorname{card}(\gamma)\operatorname{card}(\gamma)}$. Note that card(γ) is infinite, and the exponents simplify. We use $\aleph_{\alpha+n}^{\mathfrak{m}} = \aleph_\alpha^{\mathfrak{m}} \cdot \aleph_{\alpha+n}$, which holds for any non-zero integer n, when $\mathfrak{m}$ is infinite.

```
Lemma infinite_power5 n p m (x:=\aleph n) (y:= \aleph (n +o p)):
   ordinalp n -> ordinalp p -> m <> \0o ->
   cardinal p <=c m ->
   y ^c m = (x ^c m) *c (y ^c (cardinal p)). (* 1112 *)

Lemma infinite_power6  p m (y:= \aleph p):
  ordinalp p ->  m <> \0o ->  cardinal p <=c m ->
```

```
   (infinite_c m \/ p <> \0o) ->
   y ^c m = (\2c ^c m) *c (y ^c (cardinal p)).
Lemma infinite_power6_0  p m (y:= \aleph p):
  ordinalp p -> infinite_c m -> cardinal p <=c m ->
   y ^c m = (\2c ^c m) *c (y ^c (cardinal p)).
```

In (11.82), if γ is finite, we can omit the exponent $\mathrm{card}(\gamma)$. In the special case $\gamma = 1$, we get the Hausdorff formula. In (11.83), if α is countable, we can replace the exponent by $\aleph_0$ (this is clear when α is infinite, but also holds when α is finite, via Bernstein):

$$\aleph_\alpha^{\mathfrak{m}} = 2^{\mathfrak{m}} \cdot \aleph_\alpha^{\aleph_0} \qquad (\mathrm{card}(\alpha) \le \omega_0 \le \mathfrak{m}).$$

```
Lemma infinite_power5' n p m (x:=\aleph n) (y:= \aleph (n +o p)):
   ordinalp n -> natp p -> m <> \0o -> p <=c m ->
   y ^c m = (x ^c m) *c y.
Lemma infinite_power5'' n p m (x:=\aleph n) (y:= \aleph (n +o p)):
   ordinalp n -> natp p -> infinite_c m ->
   y ^c m = (x ^c m) *c y.
Lemma infinite_power6_ct  p m (y:= \aleph p):
  ordinalp p -> infinite_c m -> cardinal p <=c omega0  ->
   y ^c m = (\2c ^c m) *c (y ^c aleph0).
```

Applications

(11.84) $$\aleph_\omega^{\aleph_1} = \aleph_\omega^{\aleph_0} \cdot 2^{\aleph_1}, \qquad \aleph_\alpha^{\aleph_1} = \aleph_\alpha^{\aleph_0} \cdot 2^{\aleph_1}, \qquad \aleph_\beta^{\aleph_2} = \aleph_\beta^{\aleph_1} \cdot 2^{\aleph_2} \qquad (\alpha < \omega_1, \beta < \omega_2).$$

$$\aleph_\alpha^{\aleph_\beta} = \aleph_\alpha^{\aleph_0} \cdot 2^{\aleph_\beta} \qquad (\omega \le \alpha < \omega_1)$$

(note that $\mathrm{card}(\alpha) \le \aleph_0$ and $\mathrm{card}(\beta) \le \aleph_1$ so that (11.82) applies. The result is clear when we have equality (in particular in the last case). Otherwise, we want to prove $a = bc$, and the formula gives $a = b'c$, with $b' \le b$. But this implies $a \le bc$, while $bc \le a$ is trivial.

```
Lemma infinite_power6_1 a: a <o omega1 ->
  (\aleph a) ^c aleph1 =
  (\aleph a) ^c aleph0 *c \2c ^c aleph1.
Lemma infinite_power6_2 b: b <o omega2 ->
  (\aleph b) ^c aleph2 =
  (\aleph b) ^c aleph1 *c \2c ^c aleph2.
Lemma infinite_power6_3:
  (\aleph omega0) ^c aleph1 =
  (\aleph omega0) ^c aleph0 *c \2c ^c aleph1.
Lemma infinite_power6_4 a b:
  infinite_o a -> a <o omega1 -> ordinalp b ->
  (\aleph a) ^c (\aleph b) =
  (\aleph a) ^c aleph0 *c \2c ^c (\aleph b).
```

If $\Omega = \omega_1$, a consequence of (11.82) and (11.83) is

$$\aleph_{\Omega+\omega}^{\aleph_2} = 2^{\aleph_2} \cdot \aleph_\Omega^{\aleph_1} \cdot \aleph_{\Omega+\omega}^{\aleph_0}.$$

```
Lemma infinite_power5_ex:
  \aleph (omega1 +o omega0) ^c (\aleph \2o) =
  (\2c ^c  (\aleph \2o)) *c ((\aleph omega1) ^c aleph1)
   *c (\aleph (omega1 +o omega0) ^c aleph0).
```

We have

$$\aleph_{\omega_\alpha} = c^{\aleph_\beta} \implies \beta < \alpha; \qquad 2^{\aleph_\omega} = \aleph_{\omega_1 alpha} \implies \omega < \alpha. \tag{11.85}$$

The second relation is a special case of the first. Here c is any cardinal (and $c \ge 2$). Let's compute cofinalities; on one hand this is $\mathrm{cf}(\omega_\alpha)$ thus $\le \aleph_\alpha$; on the other hand it is $> \aleph_\beta$, thus $\beta < \alpha$. We deduce that $2^{\aleph_\omega} \neq \aleph_{\omega_n}$, for any integer n. The case $n = 4$ is interesting since one has: $2^{\aleph_\omega} < \aleph_{\omega_4}$, provided that $\aleph_\omega$ is a strong limit cardinal.[4] We also deduce: if $\aleph_{\omega_1}$ has the form x^y, where y is infinite, then y has to be $\aleph_0$.

```
Lemma infinite_power6_5 a b c :
  ordinalp a -> ordinalp b ->
  \aleph (\omega a) = c ^c (\aleph b)  ->
  b <o a.
Lemma infinite_power6_5a a b : infinite_c b ->
  \aleph omega1 = a ^c b  -> b  = aleph0.
Lemma infinite_power6_6 n : ordinalp n ->
 \2c ^c (\aleph omega0) = \aleph (\omega n) -> omega0 <o n.
```

Define $\gimel x = x^{\mathrm{cf}(x)}$. If x is regular, then $\mathrm{cf}(x) = x$, thus $\gimel x = x^x = 2^x$. We have also $\gimel x \le 2^x$.

```
Definition gimel_fct x := x ^c (\cf x).

Lemma gimel_prop1 x: regular_cardinal x ->
  gimel_fct x = \2c ^c x.
Lemma gimel_prop2 x: infinite_c x ->
  gimel_fct x <=c \2c ^c x.
```

Assume $2^{\aleph_1} = \aleph_2$. Then $\aleph_\omega^{\aleph_1} = \aleph_\omega^{\aleph_0}$. We deduce $\aleph_\omega^{\aleph_0} \neq \aleph_{\omega_1}$, for otherwise we would have $\aleph_\omega^{\aleph_1} = \aleph_{\omega_1}$, contradicting the previous result. Assume $\aleph_\omega^{\aleph_0} > \aleph_{\omega_1}$. Then $\aleph_{\omega_1}^{\aleph_1} = \aleph_\omega^{\aleph_0}$ holds.

Assume $2^{\aleph_0} > \aleph_\omega$ (note that these two quantities cannot be equal, since their cofinalities are respectively $> \omega$ and $= \omega$). Then $\aleph_\omega^{\aleph_0} = 2^{\aleph_0}$. This is obvious.

Assume $2^{\aleph_0} \ge \aleph_{\omega_1}$. Then $\gimel(\aleph_\omega) = 2^{\aleph_0}$ and $\gimel(\aleph_{\omega_1}) = 2^{\aleph_1}$. The formulas hold because the cofinalities are $\aleph_0$ and $\aleph_1$.

Note that $\aleph_\omega < \aleph_\omega^{\aleph_0}$ and $\aleph_{\omega_1} < \aleph_{\omega_1}^{\aleph_1}$, as these relations are of the form $x < x^{\mathrm{cf}(x)}$.

```
Lemma infinite_power6_7a: \2c ^c aleph1 = aleph2 ->
 \aleph omega0 ^c aleph1 = \aleph omega0 ^c aleph0.
Lemma infinite_power6_7b:
  \2c ^c aleph1 = aleph2 ->
  (\aleph omega0) ^c aleph0 <> \aleph omega1.
Lemma infinite_power6_7c:
  \2c ^c aleph1 = aleph2 ->
   \aleph omega1 <c (\aleph omega0) ^c aleph0 ->
  \aleph omega1 ^c aleph1   = (\aleph omega0) ^c aleph0.
Lemma infinite_power6_7d:
  \aleph omega0 <c \2c ^c omega0 ->
  (\aleph omega0) ^c omega0 = \2c ^c omega0.
Lemma infinite_power6_7e:
   \aleph omega1  <=c \2c ^c omega0 ->
```

[4] by a famous result of Shelah; we shall not prove this.

```
   ( gimel_fct (\aleph omega0) = \2c ^c aleph0
   /\ gimel_fct (\aleph omega1) = \2c ^c aleph1).
Lemma infinite_power6_7f:
  \aleph omega0 <c  (\aleph omega0) ^c aleph0 /\
  \aleph omega1 <c  (\aleph omega1) ^c aleph1.
```

We consider here a variant of (11.78). It says $\prod x_i = (\sup x_i)^{\mathrm{I}}$. We assume $\mathrm{I} = \omega_\beta$ (recall that I must be an infinite cardinal). We asume x_i infinite hence of the form $\aleph_{\sigma_i}$, where σ is increasing.

(11.86)
$$\prod_{\xi<\omega_\beta} \aleph_{\sigma_\xi} = \aleph_\alpha^{\aleph_\beta} \qquad (\alpha = \sup \sigma_\xi)$$

```
Lemma infinite_increasing_power1 X b:
  ofg_Mle_leo X -> domain X = \omega b -> ordinalp b ->
  cprod (Lg (domain X) (fun z => \aleph (Vg X z))) =
  \aleph (\osup (range X)) ^c \aleph b.
Lemma infinite_increasing_power2 X b:
  ofg_Mlt_lto X -> domain X = \omega b -> ordinalp b ->
  cprod (Lg (domain X) (fun z => \aleph (Vg X z))) =
  \aleph (\osup (range X)) ^c \aleph b.
```

The Tarski conjecture is that

(C)
$$\prod_{\xi<\beta} \aleph_{\sigma_\xi} = \aleph_\alpha^{\mathrm{Card}\,\beta}$$

holds whenever σ_ξ is an increasing family of ordinals, indexed by a limit ordinal β such that $\sigma_\xi < \alpha$ and $\sup_{\xi<\beta} \sigma_\xi = \alpha$. The formula holds when β is countable, and when β has card(β) disjoint cofinal subsets. It holds if $\beta < \omega_1 + \omega$. It holds if SCH holds (this is an assumption weaker than GCH, it will be studied later on). One can prove that if it fails for some β, then it fails for $\omega_1 + \omega$.

Example (Josh and Shelah). Consider

$$p = \prod_{\xi<\omega_1} \aleph_\xi \prod_{n<\omega} \aleph_{\gamma+n}.$$

We have

(C')
$$p = \aleph_{\omega_1}^{\aleph_1} \cdot \aleph_{\gamma+\omega}^{\aleph_0} \le \aleph_{\gamma+\omega}^{\mathrm{card}(\omega_1+\omega)} = \aleph_{\gamma+\omega}^{\aleph_1}.$$

The conjecture says that we have equality. We have a counter-example if

$$\aleph_{\omega_1}^{\aleph_1} < \aleph_{\gamma+\omega}^{\aleph_1}, \qquad \aleph_{\gamma+\omega}^{\aleph_0} < \aleph_{\gamma+\omega}^{\aleph_1}.$$

Let $\mathrm{Y} = \aleph_{\gamma+\omega}$, $\mathrm{Z} = \aleph_\gamma$, $a = \aleph_0$ and $b = \aleph_1$. The Tarski formula says $\mathrm{Y}^b = \mathrm{Z}^b\mathrm{Y}^a$. The second condition can be restated as $\mathrm{Y}^a < \mathrm{Z}^b$. Let $\mathrm{X} = \aleph_{\omega_1}$. The first condition is equivalent to $\mathrm{X}^b < \mathrm{Z}^b$. It implies $\mathrm{X}^b < \mathrm{Z}$. A non-trivial result is that, if (C) fails, one can chose γ so that Z is singular with cofinality $\aleph_1$.

If $\gamma = \omega_1$ then p is the product of all infinite cardinals $\aleph_\alpha$ with $\alpha < \omega_1 + \omega$, and there is equality in (C'). In order to prove this, we start with some helper lemmas. The first one says that the supremum of all $\aleph_{\gamma+n}$ is $\aleph_{\gamma+\omega}$. (the composition of two normal function is a normal function). We also state that an infinite product remains unchanged if we replace one factor x_i by x'_i provided there is another infinite factor x_j such that x_i and x'_i are both $\le x_j$. This lemma will be used as follows: when we compute the product of all $\aleph^m_{a+n}$, (for finite n) we can ignore the case $n = 0$, and apply (11.82). Then $\aleph^n_{a+n} = \aleph_{a+n}$.

```
Lemma ord_sup_aleph_sum x: ordinalp x ->
  \csup (range (Lg omega0 (fun z => \aleph (x +o z)))) =
    \aleph (x +o omega0).
Lemma ord_sup_aleph x: limit_ordinal x ->
  \csup (range (Lg x \aleph)) = \aleph x.
Lemma cprod_inf_eq x y i:
   fgraph x -> fgraph y -> (domain x = domain y) ->
   inc i (domain x) ->  (Vg x i) <> \0c -> Vg y i <> \0c ->
   (exists j, [/\ inc j (domain x), j <> i, infinite_c (Vg x j),
     Vg x i <=c Vg x j & Vg y i <=c Vg x j]) ->
   (forall j, inc j (domain x) -> j = i \/ Vg x j = Vg y j) ->
  cprod x = cprod y.
```

We have

$$\prod_{0<n<\omega} n = 2^{\aleph_0}; \quad \prod_{n<\omega} \aleph_n = \aleph_\omega^{\aleph_0}; \quad \prod_{\alpha<\omega+\omega} \aleph_\alpha = \aleph_{\omega+\omega}^{\aleph_0}; \quad \prod_{\alpha<\omega_1+\omega} \aleph_\alpha = \aleph_{\omega_1+\omega}^{\aleph_1}. \tag{11.87}$$

A possible proof for the first formula is: we first notice that $0 < n < \omega$ can be replaced by $2 \le n < \omega$ so that each factor x satisfies $2 \le x \le \omega$ so that $2^\omega \le p \le \omega^\omega = 2^\omega$.

In the first two cases, the product is s^{I}, where s is the supremum and I the cardinal of the index set. For the first formula, we have $s^{\mathrm{I}} = 2^{\mathrm{I}}$. In the two other cases, the product is over all $\alpha < a + b$; we split the product as $q_1 q_2$, and get $q = \aleph^a_a \aleph^b_{a+b}$. The result is trivial if $a = b$; but non-trivial if $a = \omega_1$ and $b = \omega$.

We have $\aleph^a_{a+b} = q_2 = q_2^a = \prod \aleph^a_{a+n} = \prod \aleph^a_a \aleph_{a+n} = (\aleph^a_a)^b \prod \aleph_{a+n}$ (we use (11.82) for non-zero n, this trick works only for $b = \omega$). This is $q_1^b q_2 = q_1 q_2 = q$.

```
Lemma infinite_prod_pA:
  cprodb Nat csucc = cprodb  Nat (fun z => (csucc (csucc z))).

Lemma infinite_prod_pB1: union (range (L Nat csucc)) = omega0.
Lemma cprod_An1 a b f: ordinalp a -> ordinalp b ->
  cprodb (a +o b) f =
  cprodb a f *c cprodb b (fun z => f (a +o z)).

Lemma infinite_prod_pB2: cprodb Nat csucc = \2c ^c omega0.
Lemma infinite_prod_pB:  cprodb Nat csucc = \2c ^c omega0.
Lemma infinite_prod_pC:
   cprodb Nat \aleph = (\aleph omega0) ^c omega0.
Lemma infinite_prod_pD (o2 := omega0 +o omega0):
   cprodb o2 \aleph = \aleph2 ^c aleph0.
Lemma infinite_prod_pE (o2 := omega01 +o omega0): (* 98 *)
   cprodb o2 \aleph = (\omega o2) ^c aleph1.
```

Assume now that the index set is a power of ω, say $\beta = \omega^\gamma$, with $\gamma > 0$, the sequence $x_i = \aleph_{\sigma_i}$ is strictly increasing. We have [20, Lemma 6]

$$\prod_{i<\omega^\gamma} \aleph_{\sigma_i} = \aleph_\alpha^{\operatorname{card}(\omega^\gamma)} \qquad (\alpha = \sup(\sigma_i)) \tag{11.88}$$

Let A be the product and B the power. Relation $A \le B$ is trivial. Let $C = \prod x_i^{\mathrm{card}(\beta)}$. By application of (11.75) we have $B \le C$. Let $f : (i, j) \to x_i$ be defined on $\beta \times \beta$. By associativity, $\prod f = C$. To each $t \in \beta$ we associate the set $G_t \subset \beta \times \beta$ of all (u, v) such that $u \sharp v = t$. We use the following three properties of the natural sum $u \sharp v$. First, if u and v are in β, so is $u \sharp \beta$, so that the set of these G_t is a partition of $\beta \times \beta$; secondly G_t is a finite nonempty set (these properties hold because β is a power of ω). We use again associativity and get $C = \prod_t p_t$. p_t is the product of all x_u such that $(u, v) \in G_t$. The last property of the natural sum is $u \le t$, so that $x_u \le x_t$ and $p_t \le x_t^{\mathrm{card}(G_x)}$. Since G_t is non-empty finite, one deduces $p_t \le x_t$, thus $C \le A$.

Application: $\prod_{\xi<\mathfrak{m}} \aleph_{\sigma_\xi} = \aleph_\alpha^{\mathfrak{m}}$ if $\mathfrak{m}$ is an infinite cardinal, since an infinite cardinal is indecomposable (hence a power of ω).

```
Lemma infinite_prod3 X c  (Y:= Lg (domain X) (fun z => \aleph (Vg X z)))
   (b:= omega0 ^o c):
   \0o <o c -> ofg_Mlt_lto X -> domain X = b ->
  (cprod Y) = (\csup (range Y)) ^c (cardinal b). (* 84 *)
Lemma infinite_prod3_bis X m (Y:= Lg (domain X) (fun z => \aleph (Vg X z))):
  infinite_c m ->  ofg_Mlt_lto X -> domain X = m ->
  (cprod Y) = (\csup (range Y)) ^c m.
```

In [20, th. 7], Tarski explains how to compute τ^λ. If $\lambda < \mathrm{cf}(\tau)$, it is a sum (11.93), otherwise a product. Consider a strictly increasing function σ_ξ whose supremum is a limit ordinal α; then

$$\prod_{\xi<\omega_{\mathrm{cf}(\alpha)}} \aleph_{\sigma_\xi}^{\aleph_\beta} = \aleph_\alpha^{\aleph_\beta} \qquad (\alpha = \sup \sigma_\xi, \quad \beta \ge \mathrm{cf}(\alpha)).$$

or

$$\prod_{\xi<\mathrm{cf}(\alpha)} \aleph_{\sigma_\xi}^{\mathfrak{m}} = \aleph_\alpha^{\mathfrak{m}} \qquad (\alpha = \sup \sigma_\xi, \quad \mathfrak{m} \ge \mathrm{cf}(\alpha)). \tag{11.89}$$

Assume that the domain I of the sequence σ_ξ is an infinite cardinal. Then $\prod \aleph_{\sigma_\xi} = \aleph_\alpha^{I}$, hence $\prod \aleph_{\sigma_\xi}^{\mathfrak{m}} = \aleph_\alpha^{I\cdot\mathfrak{m}}$, and the exponent simplifies to $\mathfrak{m}$ when $I \le \mathfrak{m}$.

Now comes the trick. The relation $\alpha = \sup_I \sigma_\xi$ says that I must be at least the cofinality of α. We get the second formula when I takes this value (note: since α is a limit ordinal, its cofinality is an infinite cardinal, and $\mathfrak{m}$ is an infinite cardinal). In the first formula, we write $\mathfrak{m} = \aleph_\beta$ and take for I the value $\omega_{\mathrm{cf}(\alpha)}$; this is an initial ordinal, hence an infinite cardinal. The condition $I \le \mathfrak{m}$ becomes $\mathrm{cf}(\alpha) \le \beta$. This weaker statement is proved by Tarski.

```
Lemma infinite_prod44 X a b
  (Y:= Lg (domain X) (fun z => \aleph (Vg X z) ^c \aleph b)):
  limit_ordinal a -> \cf a <=o b -> domain X = \omega (\cf a) ->
  ofg_Mlt_lto X -> a = \osup (range X) ->
  cprod Y = \aleph a ^c (\aleph b).
Lemma infinite_prod4 X a m
  (Y:= Lg (domain X) (fun z => \aleph (Vg X z) ^c m)):
  limit_ordinal a ->  \cf a <=c m -> domain X = \cf a ->
  ofg_Mlt_lto X -> a = \osup (range X) ->
  cprod Y = \aleph a ^c m.
```

Examples

$$\aleph_\omega^{\aleph_0} = \prod_{n<\omega} \aleph_n, \quad \aleph_\omega^{\aleph_\beta} = \prod_{\xi<\omega} \aleph_\xi^{\aleph_\beta}; \quad \aleph_{\omega_1}^{\aleph_1} = \prod_{\xi<\omega_1} \aleph_\xi, \quad \aleph_{\omega_1}^{\aleph_\beta} = \prod_{\xi<\omega_1} \aleph_\xi^{\aleph_\beta} \quad (\beta \ge 1). \tag{11.90}$$

The first relation has already been proved.

```
Lemma infinite_prod_pF:
   cprodb omega1 \aleph = (\aleph omega1) ^c aleph1.
Lemma infinite_prod_pG b: \1o <=o b ->
   cprod omega1 (fun z => (\aleph z) ^c (\aleph b))
   = (\aleph omega1) ^c (\aleph b).
Lemma infinite_prod_pH b: ordinalp b ->
   cprodb omega0 (fun z => (\aleph z) ^c (\aleph b))
   = (\aleph omega0) ^c (\aleph b).
```

We have

$$\prod_{\xi\le\beta}\aleph_\xi=\aleph_\beta^{\operatorname{card}\beta},\qquad \prod_{\xi<\alpha}\aleph_\xi=\aleph_\alpha^{\operatorname{card}\alpha},\qquad (\alpha\text{ limit};\beta>0). \tag{11.91}$$

Denote by p_β the first product, and by q_α the second. Let's show that $p_\alpha=\aleph_\alpha^{\operatorname{card}\alpha}$ by induction on α. We can write $\alpha=c+\omega^n$, and split the product via `cprod_An1`. The first factor is p_c; it is 1 if $c=0$; otherwise c is limit, and the factor is evaluated by induction (note that $\operatorname{card}(c)=\operatorname{card}(\alpha)$). The second factor can be evaluated by (11.88), so that the product becomes $\aleph_c^{\operatorname{card}(\alpha)}\aleph_\alpha^{\operatorname{card}(\omega^n)}$. The conclusion follows by using (11.82).

We prove the first formula by transfinite induction. We have $p_\beta=q_\beta\aleph_\beta$. First $p_1=\aleph_0\aleph_1=\aleph_1$. If β is a successor, we apply (11.82). The non-trivial point is $\aleph_\beta^{\operatorname{card}(\beta)}=\aleph_\beta^{\operatorname{card}(\beta+1)}$. If β is a limit ordinal, we have $q_\beta=\aleph_\beta^{\operatorname{card}(\beta)}$, this is at least $\aleph_\beta$, so that $p_\beta=\aleph_\beta^{\operatorname{card}(\beta)}$.

```
Lemma infinite_prod5 a:
  (limit_ordinal a) ->
  cprodb a \aleph = (\aleph a) ^c (cardinal a).
Lemma infinite_prod6 a:
  ordinalp a -> a <> \0o ->
  cprodb (osucc a) \aleph = (\aleph a) ^c (cardinal a).
```

11.21 Sums of powers

We may consider the supremum of x^y when one of x, y is fixed, and the other one varies. We start with the case where the exponent is fixed. The result depends on how the exponent compares to the cofinality of z.

Assume that x is an infinite cardinal $y<\operatorname{cf}(x)$. We have seen that the cardinal of x^y is $\le\sum_t\operatorname{card}(t^y)$. Let $z=\operatorname{card}(t)$ so that $\operatorname{card}(t^y)=z^y$. For each cardinal z there are at most x ordinals t satisfying $z=\operatorname{card}(t)$. It follows

$$\kappa^\lambda=\kappa\sum_{\tau<\kappa}\tau^\lambda\qquad(\omega\le\kappa,0<\lambda<\operatorname{cf}(\kappa)). \tag{11.92}$$

```
Lemma infinite_power7c x y:
  cardinalp x -> cardinalp y ->
  cardinal (unionb (Lg x (functions y)))
  <=c (csum (Lg (cardinals_lt x) (fun z => z ^c y))) *c x.
Lemma infinite_power7d x (y := cardinal (cardinals_lt x)):
   cardinalp x -> x <> \0c ->
   (y <=c x /\ y <> \0c).
```

```
Lemma infinite_power7 x y:
  infinite_c x -> y <c \cf x -> y <> \0c ->
  x ^c y =  (csumb (cardinals_lt x) (fun z => z ^c y)) *c x.
```

Assume $\kappa = \aleph_\alpha$ where α is a limit ordinal. In this case, the supremum of the τ^λ is at least the supremum of the τ, thus κ. This means that the sum is equal to the supremum, and we do not need the factor κ. Since moreover $\mathrm{cf}(\kappa) = \mathrm{cf}(\alpha)$, we have

$$\aleph_\alpha^\lambda = \sup_{\xi<\alpha} \aleph_\xi^\lambda = \sum_{\xi<\alpha} \aleph_\xi^\lambda \qquad (\alpha \text{ limit}, \lambda < \mathrm{cf}(\alpha)). \tag{11.93}$$

```
Lemma infinite_power7e a y:
  limit_ordinal a -> y <> \0c ->
  \aleph a <=c \osup (fun_image a (fun z => (\aleph z) ^c y)).
Lemma infinite_power7f n y:
  limit_ordinal a -> y <c \cf a ->
  (\aleph a) ^c y = \osup (fun_image a (fun z => (\aleph z) ^c y)). (* 61 *)
Lemma infinite_power7f1 a y:
  limit_ordinal a -> y <c \cf a -> y <> \0c ->
  (\aleph a) ^c y = csumb a (fun z => (\aleph z) ^c y).
```

Application:

$$\aleph_{\omega_1}^{\aleph_0} = \sum_{\xi<\omega_1} \aleph_\xi^{\aleph_0}$$

```
Lemma infinite_prod_pI:
  csumb omega1 (fun z => \aleph z ^c aleph0) = \aleph omega1 ^c aleph0.
```

One has

$$\kappa^\lambda = \Big[\sup_{\alpha<\kappa} \alpha^\lambda\Big]^{\mathrm{cf}(\kappa)} \qquad (\lambda \geq \mathrm{cf}(\kappa)). \tag{11.94}$$

Assume $\kappa = \sum_{i\in I} x_i$, with $x_i < \kappa$. One may assume $x_i \neq 0$ so that $\kappa \leq \prod x_i$ and $\kappa^\lambda \leq \prod x_i^\lambda$. Let $S = \sup \alpha^\lambda$ so that $x_i^\lambda \leq S$ and $\kappa^\lambda \leq S^I$. Note that, if $\lambda \geq \kappa$, then $\alpha^\lambda = 2^\lambda = S$, and the result is trivial. Thus, the formula is interesting when $\mathrm{cf}(\kappa) \leq \lambda < \kappa$ (case where κ is singular).

```
Lemma infinite_power8 n (x:= \aleph n) (z:= \cf x) y:
  ordinalp n -> z <=c y ->
  x ^c y = (\osup (fun_image (cardinals_lt x) (fun t => t ^c y))) ^c z.
```

Given two infinite cardinals, one has

$$x^y = \begin{cases} 2^y & \text{if } x \leq y \\ z^y & \text{if } z < x \text{ and } x \leq z^y \\ x & \text{if } (\forall z, z < x \implies z^y < x) \text{ and } y < \mathrm{cf}(x) \\ x^{\mathrm{cf}(x)} & \text{if } (\forall z, z < x \implies z^y < x) \text{ and } y \geq \mathrm{cf}(x) \end{cases} \tag{11.95}$$

Proof. The first two cases are obvious. Taking $z = 2$ shows $y < x$. This excludes the case $x = \aleph_0$. Assume that $x = \aleph_{n+1}$ is a cardinal successor, thus is regular. Write $x = t^+$; so that (11.80) reads $x^y = t^y x$. By assumption $t^y < x$ so that $x^y = x$. Consider finally the case where $x = \aleph_n$ where n is a limit ordinal. If $y < \mathrm{cf}(x)$, the result follows from (11.93); otherwise from (11.92).

```
Lemma infinite_power9 x y: infinite_c x -> infinite_c y ->
  [/\ (x <=c y ->  x ^c y = \2c ^c y),
      (forall  z, z <c x -> x <=c z ^c y -> x ^c y = z ^c y) &
     ((forall z,  z <c x -> z ^c  y <c x) ->
     ( ( y <c \cf x  -> x ^c y = x)
       /\ (\cf x <=c y -> x ^c y = x ^c (\cf x))))].
```

Relation (11.92) holds for any $\lambda \geq \kappa$, since all powers are 2^λ. We deduce: if κ is a regular cardinal, then (11.92) holds for any non-zero λ.

```
Lemma infinite_power7g x y:
  infinite_c x -> x <=c y ->
  x ^c y =  (csumb (cardinals_lt x) (fun z => z ^c y)) *c x.
Lemma infinite_power7h x y:
  regular_cardinal x -> \0c <c y ->
  x ^c y =  (csumb (cardinals_lt x) (fun z => z ^c y)) *c x.
```

Assume that x is infinite. There is y such that $x < y$ and $y^x = y$. It suffices to take $y = 2^x$. There is y such that $x < y$ and $y^x > y$. We may proceed as follows. Let z be the cardinal successor of x, say $x = \aleph_n$ and $z = \aleph_{n+1}$. Note that z is not a fixed point of ω. Let y be the least fixed point of ω that is $\geq z$. Its cofinality is $\omega \leq x$. We have $y < y^{\mathrm{cf}(y)} \leq y^x$.

```
Lemma infinite_power6w y: infinite_c y ->
  ( (exists2 x, y <c x & x ^c y = x) /\
    (exists2 x, y <c x & x <c x ^c y)).
```

Consider

$$x^{<y} = \sup_{z <_{\mathrm{card}} y} x^z = \sum_{z <_{\mathrm{card}} y} x^z.$$

The first equality here is the definition. Note that $x^{<0} = 0$, $x^{<1} = 1$, $1^{<y} = 1$, $0^{<y} = 1$ (if $y > 0$). We shall ignore these trivial cases. If x is infinite and y finite non-zero, then $x^{<y} = x$. In the case $x \geq 2$ and y infinite the second equality holds (Let c be the cardinal of all z such that $z < y$, we have $c \leq y$; since $x^s > s$ by Cantor, we deduce $c \leq s$, where s is the supremum, so that the sum is the supremum).

In the trivial case where y is the successor of z we have $x^{<y} = x^z$.

```
Definition cpow_less x y :=
  \csup (fun_image (cardinals_lt y) (fun t => x ^c t)).

Notation "x ^<c y" := (cpow_less x y) (at level 30).

Lemma cpow_less_alt x y :
  infinite_c y -> \2c <=c x ->
  x ^<c y = csumb (cardinals_lt y) (fun t => x ^c t).
Lemma cpow_less_pr0 x y:
  cardinal_set (fun_image (cardinals_lt y) (fun t => x ^c t)).
Lemma CS_cpow_less x y: cardinalp (x ^<c y).
Lemma cpow_less_pr1 x y:  \0c <c x -> cardinalp y -> x ^<c y <=c x ^c y.
Lemma cpow_less_pr2 x y z: z <c y -> x ^c z <=c (x ^<c y).
Lemma cpow_less_pr3 x y: \0c <c x -> natp y ->
  x ^<c (csucc y) = x ^c y.
Lemma cpow_less_pr4 x y: \0c <c x -> infinite_c y ->
  x ^<c (cnext y) = x ^c y.
```

We have $\kappa \le \kappa^{<\lambda}$ (provided that $\lambda \ge 2$). In particular $\kappa^{<\lambda}$ is infinite if κ is infinite. If $2 \le \kappa$ and κ is finite then $\kappa^{<\omega} = \omega$. In particular $2^{<\omega} = \omega$. If κ is infinite, then $\kappa^{<\omega} = \kappa$. In particular $\omega^{<\omega} = \omega$. This means that the non-trivial case of $x^{<y}$ is when y is a limit cardinal.

```
Lemma cpow_less_pr5a x y: cardinalp x -> \2c <=c y -> x <=c x ^<c y.
Lemma cpow_less_pr5b x y: infinite_c x -> \2c <=c y -> infinite_c (x ^<c y).
Lemma cpow_less_pr5c x: \2c <=c x -> finite_c x ->  x ^<c  omega0 = omega0.
Lemma cpow_less_pr5d : \2c ^<c  omega0 = omega0.
Lemma cpow_less_pr5e x:  infinite_c x -> x ^<c omega0 = x.
Lemma cpow_less_pr5f (x := omega0):  x ^<c x = x.
```

We have

$$\kappa \le 2^{<\kappa} \le \kappa^{<\kappa} \le \kappa^{\kappa} \qquad (\omega \le \kappa). \tag{11.96}$$

$$\lambda^{\kappa} = (\lambda^{<\kappa})^{\mathrm{cf}(\kappa)} \qquad (\lambda \ge 2, \omega \le \kappa). \tag{11.97}$$

In fact, if $\kappa = \sum x_i$, then $\lambda^{\kappa} = \prod \lambda^{x_i} \le \prod \lambda^{<\kappa} = (\lambda^{<\kappa})^{\mathrm{cf}(\kappa)} \le (\lambda^{\kappa})^{\mathrm{cf}(\kappa)} = \lambda^{\kappa}$.

```
Lemma cpow_less_compare x: infinite_c x ->
  (x <=c \2c ^<c x /\ \2c ^<c x <=c x ^<c x /\ x ^<c x <=c x ^c x).
Lemma cpow_less_pr6 x z: infinite_c x -> \2c <=c z ->
  z ^c x = (z ^<c x ) ^c (\cf x).
Lemma cpow_less_pr6a x: infinite_c x ->
  \2c ^c x = (\2c ^<c x ) ^c (\cf x).
```

Let $s(x)$ be the sum of all x^z for $z \le x$ and $s'(x)$ the sum for $z < x$. In the case where x is a successor, say $x = y^+$, we have $s'(x) = 2^y$. This is because $s'(x) = x^{<x} = x^y$; the result follows from $x \le 2^y$. In any case $s(x) = 2^x$, since $s(x) = x^{<x} + x^x$.

```
Lemma cpow_less_pr7a x: infinite_c x ->
  csumb (cardinals_le x) (fun t => x ^c t) = \2c ^c x.
Lemma cpow_less_pr7b x (y := cnext x) : infinite_c x ->
  csumb (cardinals_lt y) (fun t => y ^c t) = \2c ^c x.
```

Let κ be an infinite ordinal, such that for all $\lambda < \kappa$, we have $2^{\lambda} < \kappa$. Then $2^{\kappa} = \gimel\kappa$ (this follows directly from (11.97)). It follows $\kappa < \mathrm{cf}(\gimel\kappa)$. If $\mathrm{cf}(\kappa) = \omega$, then $2^{\kappa} = \kappa^{\omega}$.

We have $y^{<y} = 2^{<y} = 2^x$ whenever x is an infinite cardinal and y its successor.

```
Lemma cpow_less_pr8 x:
   infinite_c x -> (forall y, y<c x -> \2c ^c y <c x) ->
  [/\ \2c ^c x = gimel_fct x,
     x <c (\cf (gimel_fct x)) &
     (\cf x = omega0 ->  \2c ^c x = x ^c omega0)].
Lemma cpow_less_pr9 x (y := cnext x):
  infinite_c x -> ( y ^<c y = \2c ^c x /\ \2c ^<c y = \2c ^c x).
```

We say that the power function x^y is is "eventually constant below z" if there exists t such that $t \le y < z$ implies $x^y = x^t$. We say that the "continuum function is eventually constant below z" if there exists t such that $t \le y < z$ implies $2^y = 2^t$. This implies $2^{<z} = 2^t$. If x is singular, and the continuum function is eventually constant below x, then $2^x = 2^{<x}$. We may assume $\mathrm{cf}(x) \le t$, so that $2^{<x} = 2^t$; we have $2^x = (2^{<x})^{\mathrm{cf}(x)} = 2^{t.\mathrm{cf}(x)}$. Since x is singular, we may assume $\mathrm{cf}(x) \le t$, so that $t.\mathrm{cf}(x) = t$.

```
Definition cpow_less_ecb x :=
  (exists2 a, a <c x & forall b, a <=c b -> b <c x -> \2c ^c a = \2c ^c b).
Lemma cpow_less_pr10 x:  singular_cardinal x -> cpow_less_ecb x ->
  -> \2c ^c x = \2c ^<c x.
```

Assume $2^{\aleph_\alpha} = \aleph_{\alpha+\beta}$ for any ordinal α. Note that this is GCH if $\beta = 1$. Then β is finite. In effect, consider the least α such that $\alpha+\beta < \beta$. Then $0 < \alpha \le \beta$ and α is a limit ordinal. Let $x = \aleph_{\alpha+\beta}$. If $\xi \le \alpha$, then $2^{\aleph_\xi} \le \aleph_{\xi+\beta} \le \aleph_{\alpha+\beta} = x$. Assume $\alpha \le \xi \le \alpha+\alpha$ so that $\xi = \alpha+\lambda$ with $0 \le \lambda < \alpha$. By minimality of α, $\lambda+\beta = \beta$. By assumption $2^\xi = \aleph_{\alpha+\lambda+\beta} = \aleph_{\alpha+\beta} = x$. Thus $2^{<\alpha+\alpha} = x$, and the supremum is strict. Let $\kappa = \aleph_{\alpha+\alpha}$. Note that $\mathrm{cf}(\alpha+\alpha) = \mathrm{cf}(\alpha)$, so that $\mathrm{cf}(\kappa) < \kappa$ and κ is a singular cardinal. We may apply lemma `cpow_less_pr6`. It says $2^\kappa = 2^{<\kappa}$; and this quantity is x. But the assumption says that is is $\aleph_{\alpha+\alpha+\beta}$, which is greater than x, absurd.

```
Lemma genconthypothesis_alt b: ordinalp b -> (* 80 *)
  (forall a, ordinalp a -> \2c ^c (\omega a) = \omega (a +o b)) ->
  b <o omega0.
```

The behavior of 2^κ is uniquely determined by $\gimel\kappa = \kappa^{\mathrm{cf}(\kappa)}$. Let $p(\kappa)$ be the property that $2^{<\kappa}$ is some 2^μ with $\mu < \kappa$. We have:

$$2^\kappa = \begin{cases} \gimel\kappa & \text{if } \kappa \text{ is a successor} \\ 2^{<\kappa}\cdot\gimel\kappa & \text{if } \kappa \text{ is limit and } p \text{ holds} \\ \gimel 2^{<\kappa} & \text{otherwise} \end{cases} \tag{11.98}$$

A variant is

$$2^\kappa = \begin{cases} \gimel\kappa & \text{if } \kappa \text{ regular} \\ 2^{<\kappa} & \text{if } \kappa \text{ is singular and } p \text{ holds} \\ \gimel 2^{<\kappa} & \text{otherwise} \end{cases}$$

Note that, if κ is regular (for instance if it is a successor) then $\gimel\kappa = \kappa^\kappa = 2^\kappa$. If κ is limit, regular and p holds, then $2^{<\kappa} \le 2^\kappa = \gimel\kappa$; but if κ is singular and p holds, we have $2^{<\kappa} = 2^\kappa$; we conclude by $\gimel\kappa \le 2^\kappa$. In the last case, $2^\kappa = (2^{<\kappa})^{\mathrm{cf}(\kappa)}$ and all we need to show is that κ and $2^{<\kappa}$ have the same cofinality. Let $f : \mathrm{cf}(x) \to x$ be a cofinal function, define $g(t) = 2^{f(t)}$. By assumption $2^y < 2^{<x}$ whenever $y < x$, so that g is a function $\mathrm{cf}(x) \to 2^{<x}$. It is cofinal: let t be any ordinal such that $t <_{\mathrm{ord}} 2^{<x}$. We pretend that there are cardinals $\bar t$ and u such that $t \le_{\mathrm{ord}} \bar t$, $u < x$ and $\bar t \le 2^u$, from which $t \le g(u)$ follows. If t is finite, we choose $\bar t = \omega$ and the result is clear. We pretend that $2^{<x}$ is not the successor of u: since 2^w is never $2^{<x}$, we would have $2^w \le u$, thus $2^{<x} \le u$, absurd. If t is a cardinal, we choose $\bar t = t$; so that $\bar t < 2^{<x}$. In the case where t is not a cardinal, we choose for $\bar t$ the cardinal successor of the cardinal of t. The same relation holds. Obviously $t \le_{\mathrm{ord}} \bar t$.

Conversely, let $g : \mathrm{cf}(\lambda) \to \lambda$, where $\lambda = 2^{<x}$. Let $f(t)$ be such that $g(t) \le 2^{f(t)}$ and $f(t) < x$. This exists, according to the previous discussion. Let u be the supremum of f. Then 2^u is a supremum of g.

```
Lemma gimel_prop n (x:= \aleph n): ordinalp n ->
 [/\ (n = \0c  -> \2c ^c x = gimel_fct x),
   ( (osuccp n) -> \2c ^c x = gimel_fct x,
   (limit_ordinal n -> cpow_less_ecb x ->
       \2c ^c x = \2c ^<c x *c  gimel_fct x) &
```

```
   (limit_ordinal n -> not (cpow_less_ecb x) ->
       \2c ^c x = gimel_fct( \2c ^<c x))).  (* 128 *)

Lemma gimel_prop3 x: infinite_c x ->
  [/\ (regular_cardinal x -> \2c ^c x = gimel_fct x),
      (singular_cardinal x -> cpow_less_ecb x ->  \2c ^c x = \2c ^<c x) &
     (singular_cardinal x -> not (cpow_less_ecb x) ->
     \2c ^c x = gimel_fct (\2c ^<c x))].
```

We consider now a variant. Let $p(\kappa,\lambda)$ be the property that the power function is eventually constant below λ. We shall assume $\kappa \geq 2$ and that λ infinite. Thus, there is $\mu < \lambda$ such that for any μ_0 with $\mu \leq \mu_0 < \lambda$ we have $\kappa^{\mu} = \kappa^{\mu_0}$ (in the previous theorem, we considered the case $\kappa = 2$).

This condition is true for any successor λ. It implies $\kappa^{<\lambda} = \kappa^{\mu_0}$. If λ is singular, then $\kappa^{<\lambda} = \kappa^{\lambda}$ (same argument as in the case $\kappa = 2$) (Bukovsky & Hechler).

If the condition holds we have $\mathrm{cf}(\kappa^{<\lambda}) \geq \lambda$. Proof. Assume first $\lambda = \omega$. Then μ is finite. Taking $\mu_0 = \mu + 1$ shows that κ has to be infinite; so that $\kappa^{\mu} = \kappa^{<\lambda}$ is infinite, as well as its cofinality. Assume λ is a successor, say $\lambda = z^+$. Then $\kappa^{<\lambda} = \kappa^z > z \geq \lambda$. Assume finally that λ is a limit cardinal. We have $\mu_0 \leq \mathrm{cf}(\kappa^{<\lambda})$ for every μ_0 that is $< \lambda$ and big enough. Taking the supremum finishes the proof.

```
Definition cpow_less_ec_prop x y a:=
  a <c y /\ forall b,  a<=c b -> b <c y ->  x ^c a = x ^c b.

Lemma cpow_less_ec_pr0 x y:
  infinite_c y -> \2c <=c x -> infinite_c  (x ^<c y).
Lemma cpow_less_ec_pr1 x y:
  cardinalp y ->  exists a, cpow_less_ec_prop x (cnext y) a.
Lemma cpow_less_ec_pr2 x y a:
  cpow_less_ec_prop x y a -> \2c <=c x ->
  forall b,  a<=c b -> b <c y  ->  x ^<c y = x ^c b.
Lemma cpow_less_ec_pr3 x y a:
  cpow_less_ec_prop x y a -> \2c <=c x -> singular_cardinal y ->
   (x ^<c y = x ^c a /\ x ^<c y = x ^c y).
Lemma cpow_less_ec_pr4 x y:
  infinite_c y -> \2c <=c x ->
  (exists a, cpow_less_ec_prop x y a) ->
  y <=c \cf (x ^<c y).
```

Assume now $p(\kappa,\lambda)$ false. We pretend that there is a sequence of cardinals X_i, indexed by $\mathrm{cf}(\lambda)$, such that, if $Y_i = \kappa^{X_i}$, then $Y_i < \kappa^{<\lambda}$, and the supremum is $\kappa^{<\lambda}$. Our assumption is: If $\mu < \lambda$, there is ν such that $\mu < \nu < \lambda$ and $\kappa^{\mu} < \kappa^{\nu}$. Since λ is an infinite cardinal, there is μ such that $\lambda = \aleph_{\mu}$. If $\mu = 0$, our assumption says κ finite, and we can take $X_i = i$. Otherwise, our assumption says that μ is a limit ordinal, so that $\mathrm{cf}(\mu) = \mathrm{cf}(\lambda)$. There is a strictly increasing cofinal function $f : \mathrm{cf}(\lambda) \to \mu$. Take $X_i = \aleph_{(fi)}$. Let S be the supremum of the X_i; this is an infinite cardinal, $\leq \lambda$, thus is of the form $\aleph_{\alpha}$, and it is clear that $\alpha < \mu$ is impossible. Thus $\sup X_i = \lambda$. It follows easily that $\sup Y_i = \kappa^{<\lambda}$.

It follows $\mathrm{cf}(\lambda) = \mathrm{cf}(\kappa^{<\lambda})$. Note that the sequence Y_i is cofinal in $\kappa^{<\lambda}$. On the other hand, consider a cofinal function $f : \mathrm{cf}(\kappa^{<\lambda}) \to \kappa^{<\lambda}$. For every i, $\mathrm{Card}(f(i))$ is a cardinal less than $\kappa^{<\lambda}$; so that there is $\alpha < \lambda$ such that $\mathrm{Card}(f(i) \leq \kappa^{\alpha}$. The mapping $i \mapsto \alpha$ is cofinal in λ.

```
Lemma cpow_less_ec_pr5 x y:  (* 105 *)
```

```
 infinite_c y -> \2c <=c x ->
  ~ (exists a, cpow_less_ec_prop x y a) ->
  exists X, let Y := Lg (domain X) (fun z => x ^c (Vg X z)) in
  [/\ domain X = \cf y,
     (forall i, inc i (domain X) -> Vg X i <c y),
     (forall i, inc i (domain X) -> Vg Y i <c  x ^<c y),
     (y <> omega0 -> card_nz_fam X) &
     \csup (range Y) = x ^<c y ].
Lemma cpow_less_ec_pr6 x y:  (* 58 *)
  infinite_c y -> \2c <=c x ->
  ~ (exists a, cpow_less_ec_prop x y a) ->
  \cf (x ^<c y) = \cf y.
```

We have

$$(\kappa^{<\lambda})^\nu = \begin{cases} \kappa^{<\lambda} & \text{if } 0 < \nu < \mathrm{cf}(\lambda) \\ \kappa^{\lambda} & \text{if } \mathrm{cf}(\lambda) \le \nu < \lambda \\ \kappa^{\nu} & \text{if } \lambda \le \nu \end{cases} \tag{11.99}$$

Proof. Assume first the power function eventually constant, and consider each of the three cases. The result is clear if ν is finite. Assume first $\lambda \le \nu$. Then $(\kappa^{<\lambda})^\nu = (\kappa^\mu)^\nu = \kappa^{\mu\nu} = \kappa^\nu$ (note that we may assume $\mu \neq 0$). Assume now $\mathrm{cf}(\lambda) \le \nu < \lambda$. In this case λ is singular, and we conclude via $\kappa^\lambda = \kappa^{<\lambda}$.

Assume now the power function not eventually constant. There is a sequence λ_ξ such that $\kappa^{<\lambda} = \sup(\kappa^{\lambda_\xi})$. We have $\mathrm{cf}(\lambda) = \mathrm{cf}(\kappa^{<\lambda})$. Assume $\nu < \mathrm{cf}(\lambda) = \mathrm{cf}(\kappa^{<\lambda})$. By (11.92) $(\kappa^{<\lambda})^\nu = \kappa^\nu \sum_{\mu<\kappa^{<\lambda}} \mu^\nu$. Each term in the sum is bounded above by $(\kappa^\tau)^\mu$, for some $\tau < \lambda$. But $\tau\mu < \lambda$, so that each term is $\le \kappa^{<\lambda}$. It follows $(\kappa^{<\lambda})^\nu = \kappa^{<\lambda}$.

We assume now $\mathrm{cf}(\lambda) \le \nu$. The case $\lambda = \omega$ is trivial (since $\kappa^{<\lambda}$ is ω or κ, depending on whether κ is finite or not; moreover λ is regular).

We have $(\kappa^{<\lambda})^\nu = (\sum \kappa^{\lambda_\xi})^\nu \le (\prod \kappa^{\lambda_\xi})^\nu = \kappa^{\sum \lambda_\xi \nu}$. The number of terms in the sum is $\mathrm{cf}(\lambda)$. If $\lambda \le \nu$, all terms in the sum are equal to ν, and the sum is $\mathrm{cf}(\lambda)\nu = \nu$. Assume $\mathrm{cf}(\lambda) \le \nu \le \lambda$. Then all terms in the sum are $< \lambda$, and the sum is $\le \lambda$. We deduce that $(\kappa^{<\lambda})^\nu$ is $\le \kappa^\nu$ in the first case, $\le \kappa^\lambda$ is the second case. The converse inequality holds, in one case by (11.97).

```
Lemma cpow_less_ec_pr7 x y z (p := (x ^<c y) ^c z):
  infinite_c y -> \2c <=c x ->
  [/\ ( \1c <=c z -> z <c \cf y -> p =  x ^<c y),
    (\cf y <=c z -> z <c y ->  p =  x ^c y)&
    (y <=c z  ->  p =  x ^c z)].  (* 158 *)
```

11.22 Inaccessible cardinals

If x is a weakly inaccessible cardinal, then the number of regular cardinals $< x$ is x (note that $x = \aleph_x$, and all $\aleph_{i+1}$ are regular).

```
Lemma inaccessible_pr2 x:  inaccessible_w x ->
  cardinal (Zo x regular_cardinal) = x.
```

We say that x is *dominant* if x is an infinite cardinal such that $a < x$ and $b < x$ implies $a^b < x$. Note that $x \neq 0$ and $2^b < x$ for all b implies that x is dominant (note that x cannot be finite). Let $N_f(x)$ be the next fix-point of f after x, where $f(t) = 2^t$. Note that f is not normal (when t is a cardinal). However, if $x_0 = x$, $x_{n+1} = 2^{x_n}$, then $N = \sup x_i$ is the least dominant cardinal $> x$. Note that $N = \sum x_i$ so that $2^N = \prod x_i \le N^\omega$. The reverse equality is trivial so that $2^N = N^\omega = \omega^N$. In Exercise 6.21, Bourbaki deduces that if $\mathfrak{b}$ is the least dominant cardinal greater than ω, then $\mathfrak{b}^{\aleph_0} = (2^{\mathfrak{b}})^{\mathfrak{b}}$. This is an example of $a < b$ and $c < d$ but $a^c = b^d$.

```
Definition card_dominant x:=
   infinite_c x /\ forall a b, a <c x -> b <c x -> a ^c b <c x.
Definition next_dominant x :=
  the_least_fixedpoint_ge (cpow \2c) x.
Definition least_non_trivial_dominant := next_dominant omega0.

Lemma card_dominant_pr1 x: cardinalp x ->
   x <> \0c -> (forall m, m <c x -> \2c ^c m <c x) -> card_dominant x.
Lemma card_dominant_pr2: card_dominant omega0.
Lemma next_dominant_pr x (y:= next_dominant x): cardinalp x ->
   [/\ card_dominant y,  x <c y &
   (forall z,  card_dominant z -> x <c z -> y <=c z)].
Lemma card_dominant_pr3 x (y := next_dominant x) :
  cardinalp x -> \2c ^c y  =  y ^c omega0.
Lemma card_dominant_pr4 (b:=  least_non_trivial_dominant):
 [/\ card_dominant b,
    omega0 <c b,
    (forall z, card_dominant z -> omega0 <c z -> b <=c z),
    (b ^c omega0 = omega0 ^c b)
    &   (b ^c omega0 = \2c ^c b) /\
    (b ^c omega0 = (\2c ^c b) ^c b) ].
```

Let x be a dominant cardinal. Then x is not a successor (if $x = y^+$, then $y < x$ and $y^y = 2^y > y$, so that $y^y \ge x$). For this reason, such a cardinal is sometimes called a strong limit cardinal.

Then $2^{<x} = x$, and $2^x = \gimel x$ by (11.97).

```
Lemma dominant_limit a: ordinalp a ->
  ~ (card_dominant (\aleph (osucc a))).
Lemma card_dominant_pr5 x: card_dominant x ->
  (\2c ^<c x = x /\ \2c ^c x = gimel_fct x).
```

We consider here a variant of (11.95). Let α and β be two ordinals, $A = \aleph_\alpha$ and $B = \aleph_\beta$. If γ is a third ordinal, we set $C = \aleph_\gamma$. We want to compute $X = A^B$. If $\alpha \le \beta$, we have $A \le B$ and $X = 2^B$. Assume $\alpha < \beta$, or $A < B$.

We say that A is B-strong if $z^B < A$ whenever $z < A$. Let $s = \sup_{C<A} C^B$, so that $s \le A$. Assume that A is a successor, say $A = C^+$. By (11.79) we get $A^B = C^B A = A$. We have $s = C^B < A$. Assume α limit. Then $A = \sup C$, so that $s = A$.

The last two clauses of relation (11.95) says that, if A is B-strong then $X = A$ if $B < \mathrm{cf}(A)$ and $X = A^{\mathrm{cf}(A)}$ if $\mathrm{cf}(A) \le B$.

Assume A not B-strong. There exists C such that $C < A$ and $C^B \ge A$, and there is a least such one. Assume A and B are infinite. For any C satisfying the property, we have $A^B = C^B$. If 2 satisfies the property, we have $A \le 2^B$. Otherwise C is infinite, thus of the form $C = \aleph_\gamma$.

Let's show: if $A^B > 2^B$, A not B-strong, C the least such that $C < A$ and $C^B \ge A$, then C is B-strong, is singular, $\mathrm{cf}(C) \le B < C$ and $A^B = C^{\mathrm{cf}(C)}$. Proof. Since $A^B = C^B$, the relation $C \le B$

would imply $A^B = 2^B$, which is excluded. Thus, $B < C$ and C is infinite. By minimality, C is B-strong. We can apply (11.95) with B and C. If $\mathrm{cf}(C) \le B$, then $A^B = C^B = C^{\mathrm{cf}(C)}$. Moreover, $\mathrm{cf}(C) < C$ so that C is singular. In the other case, $C^B = C$, but $C < A \le C^B$, absurd.

```
Definition rel_strong_card x y:=
   forall t, t <c x -> t ^c y <c x.

Lemma card_dominant_pr7 a (A := \aleph a) B
  (s := \osup (fun_image a (fun z=> \aleph z ^c B))):
  rel_strong_card A B -> B <> \0c -> ordinalp a ->
  (s <=c A /\ (limit_ordinal a -> s = A)).
Lemma card_dominant_pr8 a b  (A := \aleph a) (B:= \aleph b) (X := A ^c B):
  ordinalp a -> ordinalp b -> rel_strong_card A B ->
    ( (B <c \cf A -> X = A) /\ (\cf A <=c B -> X = gimel_fct A)).
Definition the_nondominant_least A B :=
  select (fun z => [/\ z <c A, A <=c z ^c B  &
    (forall t, t <c A -> A <=c t ^c B -> z <=c t)]) A.

Lemma the_nondominant_least_pr1 A B (C:= the_nondominant_least A B):
  ~ (rel_strong_card A B) -> cardinalp A ->
  [/\ C <c A, A <=c C ^c B  &
    (forall t, t <c A -> A <=c t ^c B -> C <=c t)].
Lemma the_nondominant_least_pr2 a b  (A := \aleph a) (B:= \aleph b)
   (c := (ord_index (the_nondominant_least A B))):
  ~ (rel_strong_card A B) -> ordinalp a -> ordinalp b ->
   (A <=c \2c ^c B) \/
  [/\ c <o a, A <=c \aleph c ^c B  &
    (forall t, t <o a -> A <=c \aleph t ^c B -> c <=o t)].
Lemma card_dominant_pr9 A B C:
  infinite_c B ->  C <c A ->  A <=c C ^c B -> A ^c B = C ^c B.
Lemma card_dominant_pr10 a b  (A := \aleph a) (B:= \aleph b) (X := A ^c B)
  (C := (the_nondominant_least A B)):
  ordinalp a -> ordinalp b ->
  \2c ^c B <c X -> ~ (rel_strong_card A B) ->
  [/\ rel_strong_card C B, singular_cardinal C,
    \cf C <=c B, B <c C & X = gimel_fct C)].
```

Consider now two infinite cardinals A and B. We have then either $A^B = 2^B$ or $A^B = A$ or $A^B = \gimel C$ for some singular B-strong cardinal C (since $2 \le A$ we have $2^B \le A^B$). So, we have either $A^B = 2^B$, and the previous lemma applies. If A is B-strong, we apply (11.95). The second clause does not apply; if the last applies we have $\mathrm{cf}(A) \le B < A$ and A is singular. Otherwise, the previous lemma asserts existence of C with the desired properties. (Josh)

```
Lemma card_dominant_pr11 A B (X := A ^c B):
  infinite_c A -> infinite_c B ->
 [\/ X = A, X =  \2c ^c B |
  exists C, [/\ infinite_c C, rel_strong_card C B,
    \cf C <=c B , B <c C & X = gimel_fct C]].
```

We say that a cardinal x is inaccessible if it is weakly inaccessible and dominant. In particular, x is regular and not a successor. Let $P(x)$ be the property that $\prod x_i < x$, whenever $x_i < x$ and the index set if of cardinal $< x$. This is like regularity, with a product instead of a sum.

If x is singular, then $\mathrm{P}(x)$ holds. Consider a family x_i, with $x_i < x$, indexed by a set whose cardinal is $< x$. Since x is regular, we have $\sum x_i < x$. Let s be the supremum of the x_i. We have $s < x$, $\prod x_i \leq s^{\mathrm{I}}$, and $s^{\mathrm{I}} < x$ since x is dominant.

Assume now $\mathrm{P}(x)$ holds. We exclude the trivial cases $x = 0$ or $x = 2$, thus assume $x > 2$. Consider $x_i = 2$. This show that x is dominant, thus infinite. Assume that x is the cardinal successor of y. Take $x_i = y$, and $\mathrm{I} = y$. Then the product is $y^y \geq x$, absurd. We pretend that x is regular. In fact, we can write $x = \sum x_i$, where the index set is the cofinality of x, and $2 \leq x_i < x$, so that $\sum x_i \leq \prod x_i$.

```
Definition inaccessible x :=
   inaccessible_w x /\ (forall t, t <c x -> \2c ^c t <c x).

Definition cprod_of_small f x:=
  [/\ cardinal_fam f,
  (forall i, inc i (domain f) -> Vg f i <c x) &
  domain f <c x].
Lemma inaccessible_dominant x: inaccessible x -> card_dominant x.
Lemma inaccessible_pr3 x: \2c <c x ->
   (forall f, cprod_of_small f x -> cprod f <c x) ->
   card_dominant x.
Lemma inaccessible_pr4 x: \2c <c x ->
   (forall f, cprod_of_small f x -> cprod f <c x) ->
   (x = omega0 \/ (exists2 n, limit_ordinal n & x = \aleph n)).
Lemma inaccessible_pr5 x: \2c <c x ->
   (forall f, cprod_of_small f x -> cprod f <c x) ->
   (x = omega0 \/ inaccessible x).
Lemma inaccessible_pr6 x:  inaccessible x ->
   (forall f, cprod_of_small f x -> cprod f <c x).
```

Assume κ inaccessible. Then $\kappa^\lambda = \kappa$ whenever $0 < \lambda < \kappa$. This holds, because $\lambda < \mathrm{cf}(\kappa)$, so that $\kappa^\lambda = \kappa \sum \tau^\lambda$. Since κ is dominant, we have $\tau^\lambda < \kappa$, so that the sum is $\leq \kappa$. It follows $k^{<k} = k$.

```
Lemma inaccessible_pr7 x y:  inaccessible x ->
  y <> \0c -> y <c x ->  x ^c y = x.
Lemma inaccessible_pr8 x:  inaccessible x -> x = x ^<c x.
```

If a is a dominant cardinal, then $a = \aleph_n$, where n is zero or a limit ordinal (for otherwise, we have $a = \aleph_{n+1} \geq 2^{\aleph_n}$, and $\aleph_n < a$).

Consider now the two properties of a: (1) for all b, $0 < b < a$ implies $a^b = a$, and (2) for all b, $0 < b$ implies $a^b = a.2^b$. If a is ω or inaccessible, then (1) holds. If a is infinite, then (1) implies (2). Proof: Assume $a \leq 2^b$. In this case, b is infinite, $a^b = 2^b$, and the product is the second factor. Otherwise, the product is the first factor, and by Cantor, $b < a$. If a is dominant, then (1) and (2) are equivalent, since $b < a$ says $2^b < a$. Moreover, if (1) holds, then a is inaccessible or ω. In fact, since $a < a^{\mathrm{cf}(a)}$, relation (1) says that $a \leq \mathrm{cf}(()a)$, so that a is regular, thus inaccessible.

In short, if $\omega < a$, then a is inaccessible if and only if (1) and (3): for all x, y, $x < a$ and $y < a$ imply $x^y < a$.

```
Lemma inaccessible_dominant1 x:
   card_dominant x ->
   (x = omega0 \/ (exists2 n, limit_ordinal n & x = \aleph n)).
```

```
Lemma inaccessible_dominant2 x
  (p1 := forall z, \0c <c z -> z <c x -> x ^c z = x)
  (p2:= forall z, \0c <c z ->  x ^c z = x *c \2c ^c z):
  [/\ (infinite_c x -> p1 -> p2),
    (card_dominant x -> (p1 <-> p2)),
    (card_dominant x -> p1 -> (x = omega0 \/ inaccessible x)) &
    (x = omega0 \/ inaccessible x -> (card_dominant x /\ p1))].
Lemma inaccessible_dominant3 x: omega0 <c x ->
  (inaccessible x <->
  ( (forall a b, a <c x -> b <c x -> a ^c b <c x)
    /\ (forall z, \0c <c z -> z <c x -> x ^c z = x))).
```

We have shown that the cofinality of the ordinal sum $a+b$ is that of b. We deduce $\mathrm{cf}(a+\omega)=\omega$. Thus $\mathrm{cf}(\aleph_{\alpha+\omega})=\aleph_0$, whenever α is an ordinal, and $\aleph_{\alpha+\omega}$ is singular. We deduce that there is no set containing all singular cardinals: if E is such a set, x its supremum, $\aleph_\alpha \ge x$, then $\aleph_{\alpha+\omega}$ is singular and not in the set.

```
Lemma cofinality_sum1 a: ordinalp a ->
   \cf (\aleph(a +o omega0)) = omega0.
Lemma cofinality_sum2 a: ordinalp a ->
   singular_cardinal (\aleph(a +o omega0)).
Lemma singular_non_collectivizing:
  not (exists E, forall x, singular_cardinal x -> inc x E).
```

11.23 Consequences of GCH

We assume here GCH, that says that 2^x is the cardinal successor of x, whenever x is infinite.

```
Section GenContHypothesis_Props.
Hypothesis gch: GenContHypothesis.
```

The main result is

$$x^y = \begin{cases} 1 & \text{in case } y=0 \\ x & \text{in case } 0<y<\mathrm{cf}(x) \\ x^+ & \text{in case } \mathrm{cf}(x)\le y\le x \\ y^+ & \text{otherwise} \end{cases} \tag{11.100}$$

The nontrivial point in the proof is the third case. We use (11.84) and pretend $\tau^\lambda \le x$. Let $w=\sup(\tau,\lambda)$. Then $\tau^\lambda \le w^w = 2^w = w^+$ (assuming w infinite, otherwise the result is trivial). Now $w<x$ says $w^+\le x$.

```
Lemma infinite_power10 x y (z := x ^c y): infinite_c x ->  (* 56 *)
 [/\  (y = \0c -> z = \1c),
   (y <> \0c -> y <c \cf x -> z = x),
   (\cf x <=c  y -> y <=c x -> z = cnext x) &
   (x <c y -> z = cnext y)].
```

We have

1. $\kappa^{\mathrm{cf}(\kappa)}=\kappa^+$ whenever κ is infinite.

2. If $0 < \lambda < \mathrm{cf}(\kappa)$ then $\kappa^\lambda = \kappa$.

3. $\kappa^{<\lambda} = \lambda$ for any infinite λ, whenever $2 \le \kappa < \omega$

4. If κ is regular, then $\kappa^{<\kappa} = \kappa$.

5. If x is weakly inaccessible, it is strongly inaccessible.

6. If x is inaccessible or ω, then $x^{<x} = x$.

7. Relation (11.84) $\kappa^\lambda = \kappa \sum_{\tau<\kappa} \tau^\lambda$ holds for every λ if and only if κ is regular.

8. $\prod \aleph_{\sigma_\xi} = \aleph_{\alpha+1}$ whenever σ_ξ is defined for all $\xi < \beta$, which is a limit ordinal, $\alpha = \sup \sigma_\xi$. (cf relation (11.78)).

9. Assume $\kappa \ge 2$, λ infinite

$$\kappa^{<\lambda} = \begin{cases} \kappa & \text{if } \lambda \le \mathrm{cf}(\kappa) \\ \kappa^+ & \text{if } \mathrm{cf}(\kappa) < \lambda \le \kappa^+ \\ \kappa^\lambda & \text{if } \kappa^+ < \lambda \end{cases} \tag{11.101}$$

In the case κ finite, $\kappa^\lambda = k^{<\lambda}$ as shown earlier. Proof of 3: Assume $\kappa \ge 2$ and finite. Assume $y < \lambda$; then $\kappa^y \le \lambda$, as this is obvious if y is finite, otherwise we have $\kappa^y = 2^y = y^+ \le \lambda$. Then $\kappa^{<\lambda} \le \lambda$ holds. Let $y = \kappa^{<\lambda}$. If $y < \lambda$ then $2^y \le \kappa^y \le \kappa^{<\lambda} = y$, contradicting Cantor.

Proof of 5. All that needs to be done is to show that $t < x$ implies $2^t < x$. This is obvious if t is finite; otherwise $x = \aleph_m$ and $2^t = \aleph_{n+1}$ where $n < m$. The relation relation $n+1 < m$ holds since m is a limit ordinal.

Proof of 6. Assume $x = y^+$, so that $x^{<y} = x^y$. Now GCH says $x = 2^y$ so that $x^y = 2^{yy} = 2^y = x$. Conversely, assume $x^{<x} = x$. This relation holds if $x = 0$, $x = 1$ and $x = 2$; otherwise x is infinite; thus is a successor, $\aleph_0$ or limit. In the case it is regular, thus inaccessible if GCH holds.

Proof of 7. We have already seen that (11.84) holds for $\lambda < \mathrm{cf}(\kappa)$ and $\lambda \ge \kappa$. Thus, κ is regular, it holds for all λ. Assume that it holds for all λ, as well as GCH. An easy consideration shows that κ has to be zero or infinite. Let's exclude the first case, take $\lambda = \mathrm{cf}(\kappa)$. This is an infinite cardinal and the LHS is $> \kappa$; there there is some y, with $y < \kappa$ such that $y^\lambda > \kappa$. If $\lambda \le y$ then y is infinite and $y^\lambda \le y^y = y^+ \le \kappa$, absurd. Thus $y \le \lambda$ and $y^\lambda = 2^\lambda$ (note that $y \ge 2$).

Thus, it holds for all λ if κ is regular. Conversely, assume that it holds for any non-zero λ. Let's exclude the case $\kappa = 0$. If $\kappa = 1$, then the RHS is zero, absurd. If $\kappa = 2$, then the RHS is 2, absurd. Taking $\lambda = 1$, $\tau = 2$ shows that κ is infinite. Each term of the sum is at most 2^x, where x is the supremum of τ and λ. Assume $\lambda < \kappa$. If GCH holds, all terms are $\le \kappa$, and so is the RHS. Take $\lambda = \mathrm{cf}(\kappa)$, then the LHS is $> \kappa$. We deduce $\mathrm{cf}(\kappa) \ge \kappa$ hence κ is regular.

Proof of 9. The case κ finite is point 3, so that we may assume κ infinite. The first two cases are trivial from (11.100). If $\lambda > \kappa^+$, then $\kappa^{<\lambda} \le \lambda$ holds. The reverse inequality holds: assume $\lambda = \aleph_\gamma$. The case $\gamma = 0$ is excluded. If γ is a successor, say $\gamma = \delta^+$, then $\kappa^{<\lambda} = \kappa^{\aleph_\delta}$ and this is $\aleph_{\delta^+} = \lambda$. If γ is a limit ordinal, the result is immediate.

```
Lemma infinite_power10_a x: infinite_c x ->
   x ^c (\cf_c x) = cnext x.
Lemma infinite_power10_b x y:
  infinite_c x -> y <c  (\cf x) -> y <> \0c ->
  x ^c y = x.
```

```
Lemma cpow_less_pr11a x y:
  infinite_c y -> \2c <=c x -> finite_c x ->
  x ^<c y = y.
Lemma cpow_less_pr11b x: infinite_c x ->
  \2c ^<c  x = x.
Lemma cpow_less_pr12 x: regular_cardinal x ->
  x ^<c  x = x.
Lemma inaccessible_weak_strong x:
   inaccessible_w x -> inaccessible x.
Lemma inaccessible_pr8_gch x: \2c <c x ->
   ([\/ (x = omega0),
     (exists2 n, ordinalp n & x = \aleph (osucc n)) | inaccessible x]
   <-> x = x ^<c x).
Lemma infinite_power7h_rev x: \0c <c x -> (* 80 *)
  (forall y, \0c <c y ->
  x ^c y =  (csumb  (cardinals_lt x) (fun z => z ^c y)) *c x)
  -> regular_cardinal x.
Lemma infinite_increasing_power5gch X
  (Y:= Lg (domain X) (fun z => \aleph (Vg X z)))
  (a := \osup (range X)):
  ofg_Mlt_lto X -> limit_ordinal (domain X) ->
  ((cprod Y) = \aleph a ^c (cardinal (domain X)) /\
   (cprod Y) = \aleph (osucc a)).
Lemma cpow_less_ec_pr8 x y ( z := x ^<c y):  (* 67 *)
  infinite_c y -> infinite_c x ->
  [/\ (y <=c \cf x ->  z = x),
   (\cf x <c y -> y <=c cnext x ->  z = cnext x) &
  (cnext x <c y -> z = y)].

End GenContHypothesis_Props.
```

11.24 The beth function

There exists a unique normal OFS $\alpha \mapsto \beth_\alpha$ such that

$$\beth_0 = \aleph_0, \qquad \beth_{\alpha+1} = 2^{\beth_\alpha}.$$

```
Definition beth x :=
  let p := fun z => \2c ^c z in
  let osup := fun y f => \osup (fun_image y (fun z => (p (f z)))) in
  let osupp:= fun f =>  Yo (source f = \0o) omega0 (osup (source f) (Vf f)) in
  let f:= fun x => Vf (transfinite_defined (ordinal_o (osucc x)) osupp) x in
  f x.

Lemma beth_prop:
  [/\ normal_ofs beth, beth \0o = omega0 &
   (forall x,  ordinalp x -> beth (osucc x) = \2c ^c (beth x)) ].
```

Note that $\beth_0$ and $\beth_{\alpha+1}$ are cardinals. By transfinite induction $\beth_\alpha$ is a cardinal. It follows that $\alpha <_{\text{ord}} \beta$ implies $\beth_\alpha <_{\text{card}} \beth_\beta$ In particular, $\beth_\alpha$ is an infinite cardinal. This means that there $\pi(\alpha)$ such that $\beth_\alpha = \aleph_{\pi(x)}$ [in his paper Tarski considers π instead of beth].

```
Lemma beth0: beth \0o = omega0.
Lemma beth_succ x: ordinalp x ->
```

```
    beth (osucc x) = \2c ^c (beth x).
Lemma beth_normal: normal_ofs beth.
Lemma CS_beth x: ordinalp x -> cardinalp (beth x).
Lemma beth_M x y: x <o y -> beth x <c beth y.
Lemma beth_pr1 x: ordinalp x -> infinite_c (beth x).
```

One can rewrite GCH as; for every ordinal α we have $\beth_\alpha = \aleph_\alpha$.

```
Lemma beth_gch: GenContHypothesis <->  beth =1o \aleph.
```

As the beth OFS is normal, it has many fixed points. They are strong limit cardinals (let $y < x$ be any cardinal, where $x = \beth_\alpha$; there is β such that $\beta < \alpha$ and $y \le \beth_\beta$, since α is a limit ordinal, and $\beth$ is normal; now $2^y \le 2^{\beth_\beta} = \beth_{\beta+1} < \beth_\alpha = x$). Note that the next fix-point after a fix-point has cofinality ω.

```
Lemma beth_fixed_point x:
   ordinalp x -> beth x = x -> card_dominant x.
Lemma cofinality_least_fp_beth x y:
  beth x <> x ->  least_fixedpoint_ge beth x y ->
  cofinality y = omega0.
```

As the beth function is normal, for every α there is a β such that $\beth_\beta \le \aleph_\alpha < \beth_{\alpha+1}$. One deduces: if α is a limit ordinal, there is a limit ordinal β such that $\beth_\alpha = \aleph_\beta$.

If κ is an inaccessible cardinal and $a < \kappa$ then $\beth_a < \kappa$. The proof is by transfinite induction. Since κ is dominant, $\beth_a < \kappa$ implies $\beth_{a+1} < \kappa$. If a is a limit ordinal, $\beth_b < \kappa$ for $b < a$, then $\sup_b(\beth_b) < \kappa$ since κ is regular and the supremum is taken over a set of cardinal $< \kappa$.

One deduces: if κ is inaccessible, then $\beth_\kappa = \kappa$.

```
Lemma aleph_and_beth a: ordinalp a ->
  exists b, [/\ ordinalp b, beth b <=c \aleph a & \aleph a <c beth (osucc b)].
Lemma beth_limit a: limit_ordinal a ->
   exists2 b, limit_ordinal b & beth a = \aleph b.
Lemma beth_inaccessible a k : inaccessible k ->
  a <o k -> beth a <c k.
Lemma beth_inaccessible1 k : inaccessible k -> beth k = k.
```

An infinite cardinal is dominant when its is $\aleph_0$ or $\beth_\alpha$ where α is a limit ordinal.

```
Lemma card_dominant_pr12 x: infinite_c x ->
  (forall m, infinite_c m -> m <c x -> \2c ^c m <c x) -> card_dominant x.
Lemma beth_dominant a: limit_ordinal a -> card_dominant (beth a).
Lemma beth_dominantP x: infinite_c x ->
  (card_dominant x <-> exists2 a, (a = \0c \/ limit_ordinal a) & x = beth a).
```

Let's compute $\beth_\alpha^\mathfrak{m}$, where α is an ordinal, and $\mathfrak{m}$ a cardinal. The case $\mathfrak{m} = 0$ is trivial, so that we shall assume $\mathfrak{m}$ non-zero; the case $\mathfrak{m}$ finite is trivial as well, since $\beth_\alpha$ is an infinite cardinal. If α is zero or a successor, the result is obvious. For instance $\beth_{\alpha+1}^\mathfrak{m} = 2^{\beth_\alpha \cdot \mathfrak{m}}$, and the exponent simplifies as the first factor is infinite.

```
Lemma beth_pow0 m:  m <c beth \0c -> m <> \0c ->
  (beth \0c) ^c m = beth \0c.
```

```
Lemma beth_pow1 m: beth \0c <=c m ->
  (beth \0c) ^c m = \2c ^c m.
Lemma beth_pow2 m a:  ordinalp a ->  m <=c beth a -> m <> \0c ->
  (beth (osucc a)) ^c m = beth (osucc a).
Lemma beth_pow3 m a: ordinalp a -> beth a <=c m ->
  (beth (osucc a)) ^c m = \2c ^c m.

Conjectures:
Lemma beth_pow4c m a: limit_ordinal a -> (beth a) <=c m  ->
  (beth a) ^c m = \2c ^c m.
Lemma beth_pow4b m a: limit_ordinal a ->
   \cf a <=c m -> m <=c (beth a)  ->
  (beth a) ^c m = \2c ^c (beth a) /\  (beth a) ^c m = m ^c (beth a).
Lemma beth_pow4 m a: limit_ordinal a -> m <c \cf (beth a) -> m <> \0c ->
  (beth a) ^c m = beth a.
```

11.25 Consequences of SCH

We consider here the Singular Cardinal Hypothesis. It says that, if $2^{\mathrm{cf}(x)} < x$ then $x^{\mathrm{cf}(x)} = x^+$. This is an obvious consequence of GCH, thus is weaker than GCH.

Note that $2^{\mathrm{cf}(x)} < x$ implies that x is singular, for x regular says $\mathrm{cf}(x) = x$, and then $2^{\mathrm{cf}(x)} > x$. Note that $2^{\mathrm{cf}(x)} \neq x$ is trivial.

```
Definition SingCardHypothesis:=
  forall x, infinite_c x ->  \2c ^c (\cf x) <c x ->
      x ^c (\cf x) = cnext x.

Lemma SCH_case1 x: infinite_c x -> \2c ^c (\cf x) <> x.
Lemma sch_gch: GenContHypothesis -> SingCardHypothesis.
```

Let κ be a singular cardinal. Let p be the property that $t \mapsto 2^t$ is eventually constant below κ. Let $A = 2^\kappa$ and $B = 2^{<\kappa}$. We know that $A = B$ if p holds. Otherwise SCH says $A = B^+$. Proof. We apply (11.97) and SCH with $t = 2^{<\kappa}$. The assumption says that $\mathrm{cf}(t) = \mathrm{cf}(\kappa)$ so that $2^{\mathrm{cf}(t)} = 2^{\mathrm{cf}(\kappa)}$. Since κ is singular, $t \mapsto 2^t$ cannot be constant between $\mathrm{cf}(\kappa)$ and κ and $2^{\mathrm{cf}(\kappa)} < 2^{<\kappa}$.

We have (for infinite κ, and $\lambda \geq 1$):

$$\kappa^\lambda = \begin{cases} \kappa & \text{if } 2^\lambda < \kappa, \lambda < \mathrm{cf}(\kappa) \\ \kappa^+ & \text{if } 2^\lambda < \kappa, \lambda \geq \mathrm{cf}(\kappa) \\ 2^\lambda & \text{if } 2^\lambda \geq \kappa. \end{cases}$$

If both κ and λ are finite, so is κ^λ. If κ is finite (and ≥ 2) and λ is infinite, the $\kappa^\lambda = 2^\lambda$. This is why we assume κ infinite. The case λ finite is easy, as $\lambda < \mathrm{cf}(\kappa)$ and $2^\lambda < \kappa$; moreover $\kappa^\lambda = \kappa$. Thus assume λ infinite. If $\kappa \leq 2^\lambda$ we know $\kappa^\lambda = 2^\lambda$.

Finally, consider the case $2^\lambda < \kappa$ and λ infinite. We write $\kappa = \aleph_\alpha$ and proceed by transfinite induction on α. The assumption $2^\lambda < \kappa$ implies $\alpha \neq 0$. Assume $\alpha = \beta + 1$. We have $2^\lambda \leq \aleph_\beta$. We deduce $\aleph_\beta^\lambda \leq \kappa$ (if $2^\lambda = \aleph_\beta$, then $2^\lambda = (\aleph_\beta)^\lambda$; otherwise, by induction $\aleph_\beta^\lambda$ is $\aleph_\beta$ or its successor). We deduce, with the help of (11.79) that $\kappa^\lambda = \kappa$. Note that $\aleph_\beta$ is regular, so that the first clause applies. Assume finally that α is a limit ordinal. By induction, if $\nu < \aleph_\alpha$, then $\nu^\lambda < \aleph_\alpha$. We can apply (11.95). The case $\lambda < \mathrm{cf}(\kappa)$ is trivial. Otherwise $\kappa^\lambda = \kappa^{\mathrm{cf}(\kappa)}$, which is κ^+ by SCH.

```
Lemma SCH_prop2 x (p:= cpow_less_ecb x):
   singular_cardinal x -> SingCardHypothesis ->
   ( (p -> \2c ^c x = \2c ^<c x)
    /\ (~ p -> \2c ^c x = cnext (\2c ^<c x))).
Lemma SCH_prop3 x y (z := x ^c y): infinite_c x -> \1c <=c y ->
  SingCardHypothesis ->
  ( (x <=c \2c ^c y  -> z =  \2c ^c y)
    /\ ((\2c ^c y <c x) ->
      (   (y <c \cf x -> z = x)
       /\ ((\cf x <=c y) -> z = cnext x)))).  (* 81 *)
```

11.26 Von Neumann universe

The Von Neumann universe We define here a functional term V_α, by transfinite induction, so that V_α is the union of the powersets of its elements.

```
Definition universe:=
    transdef_ord (fun f => unionf (range f) powerset).

Lemma universe_rec z: ordinalp z ->
   universe z = unionf z (fun t => powerset (universe t)).
```

We show here some trivial properties. If $\alpha < \beta$, then by construction, $\mathfrak{P}(V_\alpha) \subset V_\beta$. Since, by induction, V_α is a transitive set, we deduce $V_\alpha \subset V_\beta$. We deduce the following: $V_0 = \emptyset$, $V_{\alpha+1} = \mathfrak{P}(V_\alpha)$ and if α is a limit ordinal, then $V_\alpha = \bigcup_{\beta<\alpha} V_\beta$. This could be restated: V is a normal OFS (of course, V_α is not an ordinal).

```
Lemma universe_P a: ordinalp a ->
  forall x, inc x (universe a) <-> exists2 b, b<o a & sub x (universe b).

Lemma universe_trans a: ordinalp a -> transitive_set (universe a).
Lemma universe_inc1 a b: a <o b -> sub (powerset (universe a)) (universe b).
Lemma universe_inc1' a b: a <o b -> inc (universe a) (universe b).
Lemma universe_inc2 a b: a <=o b -> sub (universe a) (universe b).

Lemma universe_0: universe \0o = emptyset.
Lemma universe_succ a: ordinalp a ->
  universe (osucc a) = powerset (universe a).
Lemma universe_limit a: limit_ordinal a ->
  universe a = unionf a universe.
```

By induction, if α is an ordinal, then $\alpha \subset V_\alpha$. Moreover, α is the set of all ordinals contained in V_α.

If n is finite, so is V_n (by induction); this shows that V_ω is countable and has cardinal $\aleph_0$. By trandfinite induction, one gets $\mathrm{card}(V_{\omega+\alpha}) = \beth_\alpha$. It follows that, when α is inaccessible, every element of V_α has cardinal $< \alpha$, and V_α has cardinal α.

```
Lemma ordinal_in_universe a: ordinalp a -> sub a (universe a).
Lemma ordinals_of_universe a: ordinalp a ->
  Zo (universe a) ordinalp = a.
Lemma card_universe_fin n: natp n -> finite_set (universe n).
Lemma card_universe_omega: cardinal (universe omega0) = aleph0.
```

```
Lemma card_universe a: ordinalp a ->
   cardinal (universe (omega0 +o a)) = beth a.
Lemma universe_inaccessible x a :
  inaccessible a -> inc x (universe a) -> cardinal x <c a.
Lemma universe_inaccessible_bis a :
  inaccessible a -> cardinal (universe a) = a.
```

We say that x is of *rank* α if α is the least ordinal such that $x \subset V_\alpha$, and we write it $R(x,\alpha)$. We denote by $r_\alpha(x)$ the least ordinal β (less than α, if it exists), such that $x \subset V_\beta$. Alternatively, we could consider the condition $x \in V_\alpha$, with the notations $R'(x,\alpha)$ and $r'_\alpha(x)$.

We define the *von Neumann universe* V as the collection of all x that belong to at least one V_α (since every ordinal is in V, this is not a set).

```
Definition universe_i x := exists2 a, ordinalp a & inc x (universe a).
Definition urank_prop x a :=
  [/\ ordinalp a, sub x (universe a) &
    forall c, c <o a -> ~(sub x (universe c)) ].
Definition urankA_prop x a :=
  [/\ ordinalp a, inc x (universe a) &
    forall c, c <o a -> ~(inc x (universe c)) ].
Definition urank a x:= least_ordinal (fun b => sub x (universe b)) a.
Definition urankA a x:= least_ordinal (fun b => inc x (universe b)) a.
```

For any x, there is at most one α such that $R(x,\alpha)$ or $R'(x,\alpha)$. If $x \subset V_\alpha$, then $r_\alpha(x)$ satisfies $R(x)$. If $x \in V_\alpha$, then $r'_\alpha(x)$ satisfies $R'(x)$. Moreover $r'_\alpha(x) = r_\alpha(x)+1$, thus $r_\alpha(x) < \alpha$.

```
Lemma urank_uniq x: uniqueness (urank_prop x).
Lemma urankA_uniq x: uniqueness (urankA_prop x).
Lemma OS_urank x a: ordinalp a -> ordinalp (urank a x).
Lemma urank_pr x a (b:= urank a x): ordinalp a -> sub x (universe a) ->
  b <=o a /\  urank_prop x b.
Lemma urankA_pr x a (b:= urankA a x): ordinalp a -> inc x (universe a) ->
  b <=o a /\  urankA_prop x b.
Lemma urankA_succ x a: urankA_prop x a -> osuccp a.
Lemma urankA_ex x: universe_i x ->
   exists2 b, ordinalp b & urankA_prop x (osucc b).
Lemma urank_alt x b: urankA_prop x (osucc b) <-> urank_prop x b.
Lemma urank_alt1 a x: ordinalp a -> inc x (universe a) ->
   urankA a x = osucc (urank a x).
Lemma urank_alt2 a x: ordinalp a -> inc x (universe a) ->
  inc x (universe (osucc (urank a x))).
Lemma urank_pr1 x a (b:= urank a x): ordinalp a -> inc x (universe a) ->
  b <o a /\  urank_prop x b.
Lemma urank_uniq2 x a b:
   ordinalp a -> ordinalp b ->
   inc x (universe a) -> inc x (universe b) ->
   urank a x = urank b x.
```

The rank of the ordinal α is α (by induction), so that the universe contains all ordinals. The rank of V_α is α (obvious). If x has rank α, and $y \in x$, the alternate rank of y is at most α, thus the rank of y is α. It follows that if $y \subset x$, its rank is at most α; so that $\mathfrak{P}(x)$ has rank $\alpha+1$. The union of x has rank one less.

If X is in V, all its elements are in V; the converse is equally true. In effect, consider the ranks of the elements of x. This is some set of ordinals, thus has a supremum α. This means that $y \in x$ implies $y \in V_\alpha$; it follows $X \in V_{\alpha+1}$.

```
Lemma urank_universe a: ordinalp a -> urank_prop (universe a) a.
Lemma urank_ordinal x: ordinalp x -> urank_prop x x.

Lemma universe_ordinal x: ordinalp x -> universe_i x.
Lemma urank_inc a x y : ordinalp a -> inc x (universe a) -> inc y x ->
   inc y (universe a) /\ (urank a y) <o (urank a x).
Lemma urank_sub a x y : ordinalp a -> inc x (universe a) -> sub y x ->
   inc y (universe a) /\ (urank a y) <=o (urank a x).
Lemma urank_powerset a x : ordinalp a -> inc x (universe a) ->
   inc (powerset x) (universe (osucc a)) /\
   (urank a (powerset x)) = osucc (urank a x).
Lemma urank_union a x : ordinalp a -> inc x (universe a) ->
   inc (union x) (universe a) /\ (urank a (union x)) =  (opred (urank a x)).
Lemma universe_stable_inc x:
  universe_i x <-> (forall y, inc y x -> universe_i y).
```

We may also consider $r(x)$, the rank of x, which exists if x is in the universe. If $x \in y$ then $r(x) < r(y)$.

```
Definition urank0 x :=
   choose (fun z => exists a, [/\ ordinalp a, inc x (universe a) &
    z = urank a x]).
Lemma urank0_pr a x (r:=  urank0 x): ordinalp a -> inc x (universe a) ->
   r = urank a x /\ urank_prop x r.
Lemma urank0_pr1 x: universe_i x -> urank_prop x (urank0 x).
Lemma urank0_ordinal a: ordinalp a -> urank0 a = a.
Lemma urank0_pr3 a b:
   universe_i b -> inc a b -> urank0 a <o urank0 b.
```

The axiom of foundation

The *transitive closure* of a set X, denoted Cl(X) is the least transitive set containing X; in other words, Cl(X) is transitive, and if Z is any transitive set such that $X \subset Z$, then $\mathrm{Cl}(X) \subset Z$. Uniqueness is obvious. If $X_0 = X$ and $X_{n+1} = \bigcup X_n$, then the transitive closure is the union of the X_i. One has

$$\mathrm{Cl}(X) = X \cup \bigcup_{y \in X} \mathrm{Cl}(y).$$

```
Definition transitive_closure X:=
  union (target (induction_defined union X)).

Definition transitive_closure_pr X Y:=
   [/\ transitive_set Y, sub X Y &
     forall Z, transitive_set Z -> sub X Z  -> sub Y Z].

Lemma tc_unique X: uniqueness (transitive_closure_pr X).
Lemma tc_exists X:
  transitive_closure_pr X (transitive_closure X).
Lemma tc_pr1 X:
  transitive_closure X = X \cup (unionf X transitive_closure).
```

Let's say that a set X has the *foundation property* if X is empty or has an element y that does not meet X. The axiom of foundation (AF) says that all sets have the foundation property. Let's say that a set x is *well-founded* if there is no infinite sequence a_i such that $a_{i+1} \in a_i$ and $a_0 = x$. Every element in the universe is well-founded (since a_i would be in the universe, and the sequence of ranks would be strictly decreasing). So: $x \in V_\alpha$ if and only if x is well-founded and its rank is $< \alpha$.

```
Definition foundation_prop x :=
  x = emptyset \/ exists2 y, inc y x & disjoint y x.
Definition foundation_axiom:= forall x, foundation_prop x.
Definition well_founded_set x :=
  forall f, function f  -> source f = Nat ->
    (forall n, (natp n -> inc (Vf f (csucc n)) (Vf f n))) ->
  x <> Vf f \0c.
Definition ordinal_altp x:=
   transitive_set x /\
   (forall a b, inc a x -> inc b x -> [\/ inc a b, inc b a | a = b]).

Lemma ordinal_with_AF x:
 (forall y, sub y x -> foundation_prop y) ->
 (ordinalp x <-> ordinal_altp x).
Lemma well_founded_in_universe x: universe_i x ->
  well_founded_set x.
Lemma well_founded_universe a: ordinalp a ->
  forall x, inc x (universe a) <->
     (well_founded_set x /\ exists2 b, b<o a & urank_prop x b).
```

Some consequence of AF: every set is well-founded (the set of these a_i has not the foundation property), in particular is irreflexive and asymmetric. Every set X is in V: in effect, the subset b of Cl(X) formed of elements not in V has the foundation property. If it is empty, then X is in V (as all elements of X are in Cl(X), thus in V). Otherwise, there is $y \in b$, disjoint from b, not in V. This last condition says that there is u in y not in V. By transitivity of Cl(X), we have $u \in b$, absurd.

Assume that every subset of X has the foundation property. Then X is an ordinal if and only if X is transitive, and whenever u and v belong to X, then $u \in v$, $v \in u$ or $u = v$ (note that, for any subset y of X, the element of y that does not meet y is the least element of y for $\in$).

```
Section Foundation.
Hypothesis AF: foundation_axiom.

Lemma AF_infinite_seq x: well_founded_set x.
Lemma AF_irreflexive  x: ~(inc x x).
Lemma AF_asymmetric  x: asymmetric_set x.
Lemma AF_universe x: universe_i x.
Lemma AF_ordinal x: (ordinalp x <-> ordinal_altp x).
Lemma urank0_pr2 x: urank_prop x (urank0 x).

End Foundation.
```

Every element of V satisfies the foundation property. Thus AF is equivalent to: every set is in V.

```
Lemma universe_AF x: universe_i x -> foundation_prop x.
Lemma AF_universe':  foundation_axiom <-> forall x, universe_i x.
```

Hereditarily finite sets

Consider the following question. Let α be an ordinal, $x \subset V_\alpha$, so that $x \in V_{\alpha+1}$. Do we have $x \in V_\alpha$? A necessary condition is that all elements of x have rank $< \alpha$. Assume $\alpha = \beta + 1$. The condition becomes $x \subset V_\beta$ and is sufficient. However, if α is a limit ordinal, this condition is always satisfied and is no more sufficient.

In the case V_ω, the answer is easy: every finite subset of V_ω is an element of V_ω (note that if $x \in V_\omega$, it is a finite subset of V_ω, and has finite rank; conversely, if x is a finite subset of V_ω, there is an element of maximal rank). Note: every ordinal in V_ω is finite.

```
Lemma universe_omega_props x: inc x (universe omega0) ->
   [/\ finite_set x, sub x (universe omega0) & inc (urank omega0 x) Nat].
Lemma universe_omega_hi x:
   finite_set x -> sub x (universe omega0) -> inc x (universe omega0).
Lemma integer_in_Vomega x: inc x (universe omega0) ->
  (inc x Nat <->  ordinal_altp x).
Lemma doubleton_in_Vomega x y (V := universe omega0):
   inc x V -> inc y V -> inc (doubleton x y) V.
Lemma pair_in_Vomega x y (V := universe omega0):
   inc x V -> inc y V -> inc (J x y) V.
```

We say that a set x is *hereditarily finite* (in short HF) if x is finite, and its elements are HF. We must refine this notion so as to exclude pathological sets such that $x = \{x\}$. Let's define a sequence x_i by $x_0 = x$ and $x_{i+1} = \bigcup x_i$. We say that x has finite depth if there is n such that x_n is empty (so that x_i is empty for $i \geq n$).

Thus, we say that x is *hereditarily finite* if x has finite depth, and all x_i are finite.

```
Definition rept_union x :=  Vf (induction_defined union x).
Definition finite_depth x :=
   exists2 n, n & rept_union x n = emptyset.
Definition rec_finite x:= forall n, natp n -> finite_set (rept_union x n).
Definition hereditarily_finite x := rec_finite x /\ finite_depth x.
```

Note that ω has infinite depth (since $\omega_i = \omega$). Any set x with infinite depth, for which all x_i are finite, is not well-founded (assume that all elements of x have finite depth; since x is finite, there is n, such that, for all $y \in x$, y_n is empty; it follows that x_{n+1} is empty, absurd. We deduce that there is $y \in x$, of infinite depth, such that all y_i are finite. By the axiom of choice, there is a function $x \mapsto y$ that we can iterate).

We show here that if x is HF, then x is finite, and all its elements are HF (if $y_{i+1} = \bigcup y_i$, and $y \in x$, then $y_i \subset x_{i+1}$). Moreover $x \in V_\omega$. The converse holds (note that if $x \in V_\omega$ has rank $k+1$, then $\bigcup x$ has rank k, so that x is of finite depth).

```
Lemma rept_union_pr x (f := rept_union x):
   f \0c = x /\ (forall n, natp n -> f (csucc n) = union (f n)).
Lemma rept_union_inc x y: inc y x ->
  forall n, inc n Nat -> sub (rept_union y n) (rept_union x (csucc n)).
Lemma rept_union_inc2 x y n: natp n -> inc y (rept_union x (csucc n)) ->
  exists2 z, inc z x & inc y (rept_union z n).
Lemma infinite_depth_prop x (f := rept_union x):
  (forall n, natp n -> (finite_set (f n) /\ nonempty (f n))) ->
  ~ (well_founded_set x).
Lemma hereditarily_finite_pr x:
```

```
  hereditarily_finite x ->
  finite_set x /\ (forall y, inc y x -> hereditarily_finite y).
Lemma universe_omega_HF x:
   inc x (universe omega0) <-> hereditarily_finite x.
```

Extensional Sets. We say that X is *extensional* whenever, for any a, b in X, $a \cap X = b \cap X \implies a = b$ (in other words, two elements a and b of X are equal when $x \in a \iff x \in b$, for any element x of X). Every transitive set is extensional. If the axiom of foundation holds, there is, for any extensional set X, a unique bijection $f : X \to T$, where T is transitive and $a \in b \iff f(a) \in f(b)$.

Proof. The function f satisfies

$$(*) \qquad \forall x, x \in X \implies f(x) = f\langle x \cap X\rangle.$$

Assume that we have two solutions, f and f'. Let A be the set of elements a such that $f(a) \neq f'(a)$. If this set is empty, then $f = f'$. Otherwise, apply (*) to an element a of A which is disjoint from A and use $f\langle a \cap X\rangle = f'\langle a \cap X\rangle$ to obtain a contradiction. Conversely, we define by stratified induction on the rank of x a function f satisfying (*). (here $H(x, f)$ is the set of all t in the range of f such that $t = f(u)$ for some u in $x \cap X$). This function is injective (consider the least α so that f is injective for rank less than α; then $f\langle x \cap X\rangle = f\langle y \cap X\rangle$ says $x \cap X = y \cap X$, thus $x = y$.

```
Definition extensional_set x :=
   (forall a b, inc a x -> inc b x -> a \cap x = b \cap x -> a = b).

Lemma transitive_extensional x:
    transitive_set x -> extensional_set x.
Lemma extensional_pr X:   (* 110 *)
  foundation_axiom -> extensional_set X ->
  exists! f, [/\ bijection f, source f = X, transitive_set(target f) &
      forall a b, inc a X -> inc b X -> (inc a b <-> inc (Vf f a) (Vf f b))].
```

Consistency properties

We shall now show that some axioms (like AF) are relatively consistent with ZF. This means that there exists a property U, and a relation R, such that the axioms of Zermelo Fraenkel (together with some additional properties) hold, when we replace "x is a set" by $U(x)$, and $x \in y$ by $R(x, y)$. By abuse of language, we say "x is in $\mathscr{U}$" instead of "$U(x)$ holds".

Let's recall the axioms of ZFC:

1. The axiom of extent says that if x and y are in $\mathscr{U}$, then $x = y$, provided that, for all z in $\mathscr{U}$, the propositions $R(z, x)$ and $R(z, y)$ are equivalent.

2. The axiom of the union says for any x in $\mathscr{U}$, there is y in $\mathscr{U}$ such that, for any z in $\mathscr{U}$, $R(z, y)$ is equivalent to $R(z, t)$ and $R(t, x)$ for some t in $\mathscr{U}$.

3. The axiom of the powerset says that for any x in $\mathscr{U}$, there is y in $\mathscr{U}$, such that, for any z in $\mathscr{U}$, $R(z, y)$ is equivalent to: for all t in $\mathscr{U}$, $R(t, z)$ implies $R(t, x)$.

4. The axiom of replacement says that, for any x in $\mathscr{U}$, any property p, any function f such that, if t in $\mathscr{U}$ and $R(t, x)$, then $f(t)$ is in $\mathscr{U}$, there is y in $\mathscr{U}$ such that for all t in $\mathscr{U}$, $R(t, y)$ is equivalent to: there is z in $\mathscr{U}$, such that $R(z, x)$ and $t = f(x)$ and $p(x)$.

5. The axiom of choice says that, if x is a set whose elements are non-empty and mutually disjoint, there is a set y that meets each element of x exactly once.

6. The axiom of infinity says there is a set, containing the empty set and stable by ordinal successor.

Comments. By the axiom of extent, the sets defined by items 2, 3 and 4, are unique. The sets defined by the other axioms are not unique. If we take the identity function in the axiom of replacement, we get: for any x, there is y, such that $\mathrm{R}(z, y)$ is equivalent to $\mathrm{R}(z, x)$ and $p(z)$. This is called the axiom of comprehension. If we take $p(z)$ to be false, we get: there is a unique y such that $\mathrm{R}(z, y)$ is false (for simplicity, we omit the statements, that x, y, z, etc., should be in $\mathscr{U}$). We denote this set by 0; we denote by $\mathrm{P}(x)$ the set obtained by applying the axiom of the powerset, we denote by 1 the quantity $\mathrm{P}(0)$ and by 2 the quantity $\mathrm{P}(1)$. This set has exactly two elements, 0 and 1; by the axiom of replacement, for any a and b, there exists a set c such that $\mathrm{R}(z, c)$ is equivalent to $z = a$ or $z = b$ (this is the axiom of the pair). We denote it by $d(a, b)$. If we apply the axiom of the union to this set, we get a set z such that $\mathrm{R}(t, z)$ is equivalent to $\mathrm{R}(z, a)$ or $\mathrm{R}(z, b)$. We denote it by $u(a, b)$. The ordinal successor of x is $u(x, d(x, x))$. We denote it by $s(x)$. The axiom of infinity is now: there exists x in $\mathscr{U}$ such that $\mathrm{R}(0, x)$ and whenever y in $\mathscr{U}$ satisfies $\mathrm{R}(y, z)$ we have also $\mathrm{R}(s(y), x)$.

Consider now the axiom of choice. For simplicity, we write here "is a set" and $x \in y$ instead of "x in $\mathscr{U}$" and $\mathrm{R}(x, y)$. We use in GAIA a strong version of AC, of the form: there exists a choice function C, such that, for any property p, if for some set x, $p(x)$ holds, then $p(\mathrm{C}(p))$ holds; moreover, for any p, $\mathrm{C}(p)$ is a set. A weak version would be: for any set y (or any set y is some big set Y), there is a choice function if we take $x \in y$ for $p(x)$. If we take for Y the powerset of X, the weak axiom is equivalent to the existence of a well-ordering on X. We show here two equivalent forms: given a partition P of a set X, there is a set Y (that can be chosen as subset of X) that meets each element of P once, and: a product of non-empty sets is non-empty. In fact, consider a family $(\mathrm{E}_i)_{i\in\mathrm{I}}$ of non-empty sets, and take for P the set of all $\{i\} \times \mathrm{E}_i$; this is obviously a partition of its union X (the elements are non-empty are pairwise disjoint). The set Y provided by AC is a functional graph, an element of the product of the E_i.

The precise formalization of AC is the following. Denote by $i(a, b)$, for a and b in $\mathscr{U}$, the c in $\mathscr{U}$ such that that $\mathrm{R}(t, c)$ is equivalent to "$\mathrm{R}(t, a)$ and $\mathrm{R}(t, b)$" (replacement for a, with f identity and $p(t) = \mathrm{R}(t, b)$). This is the intersection of the two sets. Assume that x in $\mathscr{U}$ satisfies: no a in $\mathscr{U}$ such that $\mathrm{R}(a, x)$ is zero, and if a, b in $\mathscr{U}$ are such that $\mathrm{R}(a, x)$, $\mathrm{R}(b, x)$ and $a \neq b$, then $i(a, b)$ is zero; in this case there is y in $\mathscr{U}$ such that for any $a \in \mathscr{U}$ with $\mathrm{R}(a, x)$ there is c in $\mathscr{U}$ such that $i(y, a) = d(c, c)$.

```
Lemma AC_variants:
   let AC:= forall x, (forall z, inc z x -> nonempty z) ->
      (forall z z', inc z x -> inc z' x -> z = z' \/ disjoint z z') ->
      (exists y, forall z, inc z x -> singletonp (y \cap z)) in
   (AC -> (forall f, nonempty_fam f -> nonempty (productb f)))
   /\ (AC -> forall x, exists y,
      [/\ fgraph y, domain y = powerset x -s1 emptyset &
        forall z, sub z x -> nonempty z -> inc (Vg y z) z])
   /\ ((forall x, exists r, forall z, inc z x -> nonempty z -> inc (r z) z)
        -> AC).
```

We give here a formal definition of the first four axioms. We give also a definition for the empty set and the axiom of the pair.

```
Section ModelTheory.
Variables (U: property)(R: relation).


Definition unionA_pr x u :=
  U u /\ (forall y, U y -> (R y u <->
          (exists z, [/\ U z, R z x & R y z]))).
Definition powersetA_pr x p:=
  U p /\  (forall y, U y -> (R y p <->
           (forall t, U t -> R t y -> R t x))).
Definition comprehensionA_pr x (p:property) c :=
  U c /\  (forall z, U z -> (R z c <-> (R z x /\ p z))).
Definition replacementA_pr x (p:property) (f: fterm) r :=
  U r /\ (forall z, U z ->
        (R z r <-> (exists2 t, U t & [/\ R t x, p t & z = f t]))).
Definition emptysetA_pr e:= U e /\ forall t, U t -> ~ (R t e).
Definition pairA_pr a b p :=
  U p /\  (forall t, U t -> (R t p <-> t = a \/ t = b)).

Definition extensionalityA :=
  (forall x y, U x -> U y -> (forall z, U z -> (R z x <-> R z y)) ->
       x = y).

Definition unionA :=  forall x, U x -> exists u, unionA_pr x u.
Definition powersetA :=  forall x, U x -> exists p, powersetA_pr x p.
Definition comprehensionA :=
   forall x (p:property), U x -> exists c, comprehensionA_pr x p c.
Definition replacementA :=
  forall x (p: property) f, U x ->
   (forall t, U t ->  R t x -> p t -> U (f t)) ->
     exists r, replacementA_pr x p f r.
Definition pairA :=  forall a b, U a -> U b -> exists c, pairA_pr a b c.
```

We show here that, if the axiom of extent holds, then quantities considered by the other axioms are unique, and that replacement implies comprehension. Using our axiom of choice, we define some quantities, such as union, powerset, etc, and show that they satisfy the desired property. We assume that our universe contains at least one set, so that it contains the empty set. It contains $\mathfrak{P}(\mathfrak{P}(\emptyset))$, which is a set of two elements. The axiom of the pair follows by replacement.

```
Definition ZF_axioms1 :=
  [/\  (exists x, U x), extensionalityA, unionA, powersetA & replacementA].

Definition emptysetU := choose (fun z => emptysetA_pr z).
Definition unionU x:= choose (fun z => unionA_pr x z).
Definition powersetU x:= choose (fun z => powersetA_pr x z).
Definition setofU x p := choose (fun z => comprehensionA_pr x p z).
Definition funimageU x p f := choose (fun z => replacementA_pr x p f z).
Definition doubletonU a b := choose (fun z => pairA_pr a b z).

Lemma model_uniqueness: extensionalityA ->
  [/\ forall x, uniqueness (unionA_pr x),
      forall x, uniqueness (powersetA_pr x),
      forall x p, (uniqueness (comprehensionA_pr x p)),
      forall x p f, (uniqueness (replacementA_pr x p f)) &
```

```
      uniqueness emptysetA_pr /\
      forall a b, uniqueness (pairA_pr a b)].
Lemma model_replacement_comprehension:  replacementA -> comprehensionA.

Lemma model_existence: ZF_axioms1 ->
 [/\  emptysetA_pr (emptysetU),
       forall x, U x -> unionA_pr x (unionU x),
       forall x, U x -> powersetA_pr x (powersetU x),
       forall x r, U x -> comprehensionA_pr x r (setofU x r) &
       forall x (p: property) f,
           U x -> (forall t, U t -> R t x -> p t -> U (f t)) ->
       replacementA_pr x p f (funimageU x p f)].
Lemma model_existence2 : ZF_axioms1 ->
  forall a b, U a -> U b -> pairA_pr a b (doubletonU a b).
```

We define here $a\cup b$ and $a\cap b$, and show the basic properties. We also state the axiom of choice, of infinity and of foundation.

```
Definition unionU2 a b :=  unionU (doubletonU a b).
Definition intersectionU2 a b := setofU a (fun z => R z b).

Lemma model_union: ZF_axioms1 ->
  forall a b, U a -> U b ->
  ( (forall z, U z -> (R z (unionU2 a b) <-> R z a \/ R z b)))
   /\ (forall z, U z -> (R z (intersectionU2 a b) <-> R z a /\ R z b)).

Definition choiceA:= forall x,
    U x -> (forall z, U z -> R z x -> z <> emptysetU) ->
    (forall z1 z2, U z1 -> U z2 -> R z1 x -> R z2 x ->
        (z1 = z2 \/ intersectionU2 z1 z2 = emptysetU)) ->
   exists2 y, U y &
     forall z, U z -> R z x -> exists2 s, U s &
       intersectionU2 y z = doubletonU s s.
Definition infiniteA:=
   exists2 x, U x &
    (R emptysetU x /\
     forall t, U t -> R t x -> R (unionU2 t (doubletonU t t)) x).
Definition foundationA :=
  forall x, U x ->
  x = emptysetU \/ exists2 y, U y &
     R y x /\ intersectionU2 y x = emptysetU.
End ModelTheory.
```

We consider now a model, for which $\mathrm{R}(x,y)$ is $x\in y$. We shall assume that if y is in $\mathscr{U}$, this relation says that x is in $\mathscr{U}$. The relation: "for every t in $\mathscr{U}$, if $\mathrm{R}(t,y)$ then $\mathrm{R}(t,x)$" can be simplified (when y is in $\mathscr{U}$) to $y\subset x$. Thus, we can restate the axiom of the powerset set: if x is in $\mathscr{U}$, there is p in $\mathscr{U}$ such that for all y in $\mathscr{U}$, $y\in p$ is equivalent to $y\subset x$, thus to $y\in\mathfrak{P}(x)$. We shall assume that if $y\subset x$ and x is in $\mathscr{U}$ then y is in $\mathscr{U}$; this is the axiom of comprehension. With it, we can further simplify the axiom of the powerset: removing the condition y in $\mathscr{U}$ gives: for any x in $\mathscr{U}$, $\mathfrak{P}(x)$ is in $\mathscr{U}$. The axiom of infinity is equivalent to ω is in $\mathscr{U}$ (modulo some technical conditions).

```
Lemma universe_mo1 (U: property):
   (forall x y, U x ->  inc y x -> U y) ->
```

```
   (forall x y, U x ->  sub y x -> U y) ->
  (extensionalityA U inc)
  /\ (forall x r, U x -> (setofU U inc x r) = (Zo x r))
  /\ (comprehensionA U inc)
  /\ (forall x, U x -> U (union x) ->
      (unionA_pr U inc x (union x)) /\ (unionU U inc x = union x))
  /\ (forall x, U x -> U (powerset x) ->
      (powersetA_pr U inc x (powerset x)) /\ (powersetU U inc x = powerset x))
  /\ ( forall a b, U a -> U b -> U (doubleton a b) ->
       (pairA_pr U inc a b  (doubleton a b))
       /\ (doubletonU U inc a b = doubleton a b))
  /\ (forall a b, U a -> intersectionU2 U inc a b = a \cap b)
  /\ (forall a b, U a -> U b -> U (doubleton a b) -> U (a\cup b) ->
     unionU2 U inc a b = a \cup b)
  /\ ((exists x, U x) -> emptysetU U inc = emptyset)
  /\ ((forall a, inc a omega0 -> U (singleton a))
   -> (forall a, inc a omega0 -> U (doubleton a (singleton a)))
   -> U omega0
   -> infiniteA U inc)
   /\ ( (forall a b, U a -> U b -> U (doubleton a b)) ->
       (forall a b, U a -> U b -> U (a \cup b)) ->
        infiniteA U inc -> U omega0)
  /\ ( (forall x, U x -> x = emptyset \/ exists2 y, U y &
     inc y x /\ disjoint y x) -> foundationA U inc).
```

If we take for $\mathscr{U}$ the whole von Neumann universe, then all axioms are satisfied. If we take $\mathscr{U} = V_\alpha$, the assumptions of the previous lemma are satisfied. In particular, the axiom of comprehension is satisfied, but in general, not the axiom of replacement. If α is a limit ordinal, all axioms are satisfied, with the possible exception of the axiom of infinity. In fact V_ω satisfies ZFC, the axiom of foundation, but not the axiom of infinity.

```
Lemma universe_mo2 :
   let U := universe_i in
     [/\ ZF_axioms1 U inc, comprehensionA U inc,
        infiniteA U inc, choiceA U inc & foundationA U inc].
Lemma universe_mo' a (U :=fun z => inc z (universe a)) :
  ordinalp a ->
  ( (forall x y, U x ->  inc y x -> U y) /\
    (forall x y, U x ->  sub y x -> U y)).

Lemma universe_mo3 a (U :=fun z => inc z (universe a)) :
  limit_ordinal a ->
  [/\ [/\ (exists x, U x), (extensionalityA U inc), (unionA U inc),
      (powersetA U inc) & comprehensionA U inc],
       ((forall x : Set, U x -> U (powerset x)) /\
       (forall x y, U x -> U y -> U (doubleton x y))),
       (omega0 <o a <-> infiniteA U inc), choiceA U inc & foundationA U inc].
Lemma universe_mo4 (U :=fun z => inc z (universe omega0)) :
  [/\ ZF_axioms1 U inc, ~(infiniteA U inc), choiceA U inc & foundationA U inc].
```

The set $V' = V_{\omega+\omega}$ does not satisfy the axiom of Replacement, since the collection of all $\omega + n$ for $n \in \omega$, is of rank $\omega + \omega + 1$, thus is a subset of V' that does not belong to V'. As noted above, it does satisfy Comprehension and Infinity, thus the set of axioms of Zermelo. One can show: there is a well-ordering on a subset of V' not isomorphic to any ordinal of V'. Proof:

the following relation between a and b on ω: "a is odd and b is even, or both a and b have the same parity and $a \le b$" is isomorphic to the ordinal $\omega + \omega$, which is not in V'.

We consider now the case of $\mathscr{U} = V_\alpha$, where α is an inaccessible cardinal. We shall show below that this set satisfies ZFC. We shall also show that the ordinals, cardinals and inaccessible cardinals of $\mathscr{U}$ of those of the current universe that are in $\mathscr{U}$. Recall that an inaccessible cardinal is a dominant regular cardinal, of the form $\aleph_n$, where n is a limit ordinal. In particular, $\omega < \alpha$. Dominant means that if $x < \alpha$ and $y < \alpha$, then $x^y < \alpha$. An alternative definition is: α is dominant, $\omega < \alpha$ and for all x, $0 < x < \alpha$ implies $\alpha^x = \alpha$.

We introduce here a lot of definitions. We say that X is a $\mathscr{U}$-subset of Y if for all t in $\mathscr{U}$, $t \in X$ implies $t \in Y$. The main assumption will be that, if X is in $\mathscr{U}$, and $t \in X$, then t is in $\mathscr{U}$; so that if X and Y are in $\mathscr{U}$, X is a $\mathscr{U}$-subset of Y if and only if $X \subset Y$. We say that X is $\mathscr{U}$-transitive if any element x of X in $\mathscr{U}$ is a $\mathscr{U}$-subset of X. We say that X is a $\mathscr{U}$-ordinal if every $\mathscr{U}$-transitive $\mathscr{U}$-subset of X is either X or an element of X. It is easy to show that X is a $\mathscr{U}$-ordinal if and only if X is an ordinal and in $\mathscr{U}$. Definitions of $\mathscr{U}$-cardinal and $\mathscr{U}$-inaccessible are similar. If a and b are $\mathscr{U}$-ordinals (resp. $\mathscr{U}$-cardinals) we have $a \le b$ if and only if $a \subset b$, i.e., a is a $\mathscr{U}$-subset of b. We have $a < b$ if a is a strict subset of b.

Given x and y in $\mathscr{U}$, we can consider the set $d(x, y)$ whose only elements are x and y, and the set $p(x, y) = d(d(x, x), d(x, y))$. Given X and Y in $\mathscr{U}$ we can consider the set $P(X, Y)$ of all $p(x, y)$ for $x \in X$ and $y \in Y$. It happens that $d(x, y) = \{x, y\}$, $p(x, y) = (x, y)$ and $P(X, Y) = X \times Y$. For simplicity, we shall use (x, y) and $X \times Y$ instead of $p(x, y)$ and $P(X, Y)$ in the definition of $\mathscr{U}$-equipotent: we say that X and Y are $\mathscr{U}$-equipotent if there is a subset Z of $X \times Y$, such that the relation $(x, y) \in G$ induces a 1-1 relation between X and Y. Note that, if X and Y are in $\mathscr{U}$, any subset of $X \times Z$ is in $\mathscr{U}$, so that $\mathscr{U}$-equipotent is the same as equipotent. We can thus say that X is a $\mathscr{U}$-cardinal if it is a $\mathscr{U}$-ordinal, such that if Z is a $\mathscr{U}$-ordinal $\mathscr{U}$-equipotent to X, then X is a $\mathscr{U}$-subset of Z; a $\mathscr{U}$-cardinal is nothing else than a cardinal that belongs to $\mathscr{U}$. We say z is the $\mathscr{U}$-power of x and y if z is a cardinal $\mathscr{U}$-equipotent to the set of $\mathscr{U}$-functions $y \to x$. If x and y are in $\mathscr{U}$, there is at most one such z, and it is x^y (the set of $\mathscr{U}$-functions exists in $\mathscr{U}$, as being a subset of the powerset of the cartesian product of y and x; it is the set of functional graphs $y \to x$). We define $\mathscr{U}$-inaccessible in terms of $\mathscr{U}$-power, and pretend that if x is in $\mathscr{U}$, inaccessible is the same as $\mathscr{U}$-inaccessible. Note: if x is a cardinal, to say $x < \alpha$ is the same as to say $x < \alpha$ (considered as a relation between ordinals). Since α is a limit ordinal, for any ordinal x, $x < \alpha$ is equivalent to $x \in V_\alpha$.

```
Definition subU U x y:= forall t, U t -> inc t x -> inc t y.
Definition transitive_setU U X := forall x, U x -> inc x X -> subU U x X.
Definition ordinalU U X:= forall Y, U Y -> subU U Y X -> transitive_setU U Y ->
   Y <> X -> inc Y X.
Definition equipotentU U X Y := exists2 Z, U Z & [/\ subU U Z (X \times Y),
   (forall x, inc x X -> exists2 y, U y & inc (J x y) Z),
   (forall y, inc y Y -> exists2 x, U x & inc (J x y) Z),
   (forall x x' y, U x -> U x' -> U y -> inc (J x y) Z -> inc (J x' y) Z ->
       x = x') &
   (forall x y y', U x -> U y -> U y' -> inc (J x y) Z -> inc (J x y') Z ->
       y = y')].
Definition cardinalU U x :=
    [/\ U x, ordinalU U x & forall z, U z -> ordinalU U z ->
      equipotentU U x z -> subU U x z].
Definition funsetU U X Y Z :=
  U Z /\  forall f, inc f Z <->
      [/\ U f,
          (forall p, inc p f -> inc p (Y \times X)),
```

```
         (forall x, inc x Y -> exists2 y, U y & inc (J x y) f) &
         (forall x y y', inc (J x y) f -> inc (J x y') f -> y = y') ].
Definition powerU_pr U x y z :=
    cardinalU U z /\ (exists2 Z, funsetU U x y Z & equipotentU U z Z).
Definition powerU U x y := choose (powerU_pr U x y).
Definition inaccessibleU U x :=
  [/\ cardinalU U x, ssub omega0 x,
    (forall a b, cardinalU U a ->  cardinalU U b -> ssub a x -> ssub b x ->
       ssub (powerU U a b) x) &
    (forall a, cardinalU U a ->  a <> emptyset -> ssub a x ->
       (powerU U x a) = x) ].

Definition Ufacts U a:=
 [/\
   (forall x, (U x /\ ordinalU U x) <-> (ordinalp x /\ x <o a)),
   (forall x y, U x -> U y -> (x \Eq y <-> equipotentU U x y)),
   (forall x, cardinalU U x <-> (cardinalp x /\ x <c a)),
   (forall x y, cardinalU U x -> cardinalU U y ->
     powerU U x y = (x ^c y)) &
   (forall x, inaccessibleU U x <-> (inaccessible x /\ x <c a)) ].
```

The only non-trivial point is to show that if α is an inaccessible cardinal, then V_α satisfies the axiom of Replacement. Recall that if $a < b$ is ordinal comparison, then $a <_{\text{card}} b$ implies $a < b$, and $a < b$ is equivalent $\text{Card}(a) <_{\text{card}} b$ if b is a cardinal

First, $\beta < \alpha$ implies $\text{Card}(V_\beta) <_{\text{card}} \alpha$. The proof is by transfinite induction. We have $V_\beta = \bigcup_{\gamma\in\beta} \mathfrak{P}(V_\gamma)$, so that $\text{Card}(V_\beta) \le_{\text{card}} \sum_\beta 2^{\text{Card}(V_\gamma)}$. By induction, each exponent is $<_{\text{card}} \alpha$; since α is inaccessible, each power is $<_{\text{card}} \alpha$. Since $\text{Card}(\beta) <_{\text{card}} \alpha$, regularity of α gives the result. It follows that $\text{Card}(V_\alpha) \le_{\text{card}} \sum x_i$, where the index set is α and each $x_i <_{\text{card}} \alpha$. We deduce $\text{Card}(V_\alpha) = \alpha$.

Now, V_α is the set of subsets of V_α formed of elements of cardinal $<_{\text{card}} \alpha$. First, if $x \in V_\alpha$, then x is a subset of V_α (by transitivity of V_α) and $x \in V_\beta$ for some $\beta < \alpha$, so that $x \subset V_\beta$ and $\text{Card}(x) \le_{\text{card}} \text{Card}(V_\beta) <_{\text{card}} \alpha$. Conversely, assume that x is a subset of V_α of cardinal $<_{\text{card}} \alpha$. If $y \in x$, its rank $r(y)$ is $< \alpha$, so that $\text{Card}(r(y)) <_{\text{card}} \alpha$. Since α is inaccessible we get $s = \sum_{y\in x} \text{Card}(r(y)) <_{\text{card}} \alpha$. Let $s' = 2^s$. We have $\text{Card}(r(y)) \le_{\text{card}} s <_{\text{card}} s'$; so that $r(y) < s'$. By definition of the rank, we deduce that x is a subset of $V_{s'}$. Since α is inaccessible, we get $s' <_{\text{card}} \alpha$, thus $\mathfrak{P}(V_{s'}) \subset V_\alpha$.

Consider now $x \in V_\alpha$, assume $f(t) \in V_\alpha$ whenever $t \in x$. The previous criterion says $f\langle x\rangle \in V_\alpha$, so that V_α satisfies the axiom of replacement.

We show here informally: the axiom "there is no inaccessible cardinal" is compatible with ZF. If our universe has no inaccessible cardinal, we can add the axiom "there is no inaccessible cardinal". Otherwise, there is a least inaccessible cardinal, say π. Thus V_π satisfies the axioms of ZF, and has no inaccessible cardinal.

```
Section InaccessibleUniverse.
Variable a: Set.
Hypothesis ia: inaccessible a.


Lemma universe_inaccessible_card1 b:
   b <o a -> cardinal(universe b) <c a.
Lemma universe_inaccessible_card2:  cardinal(universe a) = a.
Lemma universe_inaccessible_inc x:
```

```
   inc x (universe a) <-> (sub x (universe a) /\ cardinal x <c a).
Lemma universe_inaccessible_mo1:
   replacementA (fun z => inc z (universe a)) inc.
Lemma universe_inaccessible_mo2 :  (* 206 *)
   let U := (fun z => inc z (universe a)) in
   [/\ ZF_axioms1 U inc, comprehensionA U inc,
        infiniteA U inc, choiceA U inc & foundationA U inc]
   /\ Ufacts U a.
End  InaccessibleUniverse.
```

We present here another way of constructing a universe $\mathscr{U}'$ satisfying ZFC from a universe $\mathscr{U}$ satisfying ZFC. It has the same sets, but membership is defined by $x \in f(y)$, for some f. Let's use primes in $\mathscr{U}'$. The relation $z \in' x \iff z \in' y$ is equivalent to $f(x) = f(y)$. Thus, the axiom of extent in $\mathscr{U}'$ holds if f is injective. We also assume f surjective, (for instance, if $\emptyset$ is not in the range of f, then there is no empty set in $\mathscr{U}'$).

```
Section UniversePermutation.
Variables (f g: fterm).
Hypotheses (fg: forall x, f (g x) = x) (gf: forall x, g (f x) = x).

Let U:= fun x: Set => True.
Let inc':= fun x y => inc x (f y).

Lemma up_fi x y: f x = f y -> x = y.
Lemma up_Ut t: U t.
Lemma up_exten: extensionalityA U inc'.
```

The model obviously satisfies the axioms of ZF, and we can explicit some values. In particular, the ordinal successor of x is $g(f(x) \cup \{x\})$; we deduce the axiom of infinity. Proving the axiom of choice is easy.

```
Lemma up_ZF1: ZF_axioms1 U inc'.
Lemma up_values:
 [/\ emptysetU U inc' = g emptyset,
     forall x, unionU U inc' x = (g (union (fun_image (f x) f))),
     forall x, powersetU U inc' x = (g (fun_image (powerset (f x)) g)),
     forall x p, setofU U inc' x p = (g (Zo (f x) p)) &
    (forall x p f0, funimageU U inc' x p f0 = (g (fun_image (Zo (f x) p) f0)))
    /\ (forall x y, doubletonU U inc' x y = g (doubleton x y))].
Lemma up_union2 x y: unionU2 U inc' x y = g (f x \cup f y).
Lemma up_succ x: (unionU2 U inc' x (doubletonU U inc' x x)) = g (f x +s1 x)
Lemma up_infinite: infiniteA U inc'.
Lemma up_inter2 x y: intersectionU2 U inc' x y = g (f x \cap f y).
Lemma up_disjoint x y:
  (intersectionU2 U inc' x y = emptysetU U inc') <-> disjoint (f x) (f y).
Lemma up_choice: choiceA U inc'.
```

Take for instance the function that permutes 0 and 1, leaves other sets unchanged. All axioms are satisfied. Moreover 1 becomes empty and 0 becomes a singleton. More precisely $0 = \{0\}$ and $x = \{x\}$ holds only for zero, or if it holds in the initial theory. By induction, one can then add a finite number of sets of the form $x = \{x\}$ to the theory.

```
Lemma Universe_permutation1
  (f:= fun x => Yo (x = \0c) \1c (Yo (x = \1c) \0c x))
```

```
  (U := fun z: Set => True)
  (R := fun x y => inc x (f y))
  (x0 := \1c) (x1:= \0c):
  [/\ ZF_axioms1 U R, infiniteA U R, choiceA U R, emptysetU U R = x0 &
   ( doubletonU U R x1 x1 = x1 /\
    forall t, doubletonU U R t t = t -> t = x1 \/ (t = singleton t))].
```

Let's permute x and $\{x\}$, for every non-zero integer x. All axioms of ZF are satisfied. Moreover $x = \{x\}$ if either this holds in the initial theory, or if x is a non-zero integer. This means that we have added a countable number of sets of the form $x = \{x\}$.

```
Section UniversePermutation2.

Definition universe_permutation_fun x :=
  Yo (inc x Nat /\ x <> \0c) (singleton x)
  (Yo (exists y, [/\ inc y Nat, y <> \0c & x = singleton y]) (union x) x).

Let f :=  universe_permutation_fun.

Lemma up2_f0: f \0c = \0c.
Lemma up2_fnat x : natp x -> x <> \0c -> f x = singleton x.
Lemma up2_fnats x : natp x -> x <> \0c -> f (singleton x) = x.
Lemma up2_fperm x :  f (f x) = x.


Lemma Universe_permutation2
  (U := fun z: Set => True)
  (R := fun x y => inc x (f y))
  (x1 := \1c) (x0:= \0c):
  [/\ ZF_axioms1 U R, infiniteA U R, choiceA U R, emptysetU U R = x0 &
   ((forall x, inc x Nat -> x <> \0c -> doubletonU U R x x = x) /\
    forall t, doubletonU U R t t = t ->
        (inc t Nat /\ t <> \0c) \/  singleton t = t)].
End UniversePermutation2.
```

11.27 The set of formulas

This section corresponds to Chapter 5 of [15] or [14]. We consider a set $\mathcal{V}$ whose elements are called *variables*, and a set of operators; in [15], it is assumed that these sets are disjoint, but this is not really necessary; thus we take for $\mathcal{V}$ the set of natural integers. The *operators* are $\vee$, $\neg$, $\bigvee$, ε and $\approx$. If x is a variable, we shall write the ordered pair $(\bigvee, x)$ as $\bigvee x$. This expression is not an operator (recall that an ordered pair is a doubleton whose elements are non-empty; now, the only integers that are doubletons are 1 and 2).

```
Section Formulae.

Let val_eq := \0c.
Let val_in := \1c.
Let val_or := \2c.
Let val_not := \3c.
Let val_ex := \4c.

Definition variables := Nat.
```

```
Definition variablep x := inc x variables.

Lemma int_not_pair x: inc x Nat -> pairp x -> False.
Lemma val_compare:
  [/\ val_eq <> val_in, val_eq <> val_or, val_eq <> val_not, val_eq <> val_ex
  & [/\ val_in <> val_or, val_in <>  val_not, val_in <> val_ex &
    [/\ val_or <> val_not, val_or <> val_ex & val_not <> val_ex]]].
```

We define the *set of formulas* $\mathscr{F}$ as follows

$$\mathscr{F}_0 = \{\varepsilon, \approx\} \times (\mathscr{V} \times \mathscr{V}),$$
$$\mathscr{F}_{n+1} = \mathscr{F}_n \cup (\{\neg\} \times \mathscr{F}_n) \cup (\{\vee\} \times \mathscr{F}_n \times \mathscr{F}_n) \cup (\{\bigvee\} \times \mathscr{V} \times \mathscr{F}_n),$$
$$\mathscr{F} = \bigcup_n \mathscr{F}_n.$$

```
Definition atomic_formulas :=
   ((doubleton val_in val_eq) \times (variables \times variables)).

Definition next_formulas F :=
   F \cup (singleton val_not \times F) \cup
   ((singleton val_or) \times (F \times F)) \cup
   ((singleton val_ex) \times (variables \times F)).

Definition formulas_rec :=
   induction_defined next_formulas atomic_formulas.
Definition all_formulas := union (target formulas_rec).

Definition formulap x := inc x all_formulas.
Definition atomic_formulap x := inc x atomic_formulas.
```

The set of formulas. A formula x is a set such that there exists an integer n such that $x \in \mathscr{F}_n$. Note that $\mathscr{F}_n \subset \mathscr{F}_{n+1}$, so that this integer is not unique. If ϕ and ψ are formulas, there are two integers n and m such that $\phi \in \mathfrak{F}_n$ and $\psi \in \mathfrak{F}_m$; there is an integer p such that $n \le p$ and $m \le p$, so that $\phi \in \mathfrak{F}_p$ and $\psi \in \mathfrak{F}_p$. By definition the pair $(\vee, (\phi, \psi))$ belongs to F_{p+1}, thus is a formula (the proof will be given below). For simplicity, this quantity will be denoted $\phi \vee \psi$. Similarly, $(\neg, \phi)$ is a formula, it will be denoted by $\neg\phi$. If x is a variable, then $(\bigvee, (x, \phi))$ is a formula; it will be denoted $\bigvee_x \phi$.

Note: in [15], the expression $(\bigvee, (x, \phi))$ is replaced by $((\bigvee, x), \phi)$. Our approach has the advantage that every formula is an ordered pair, whose first projection is an operator. Thus, the theorem that asserts that $(\bigvee, x)$ is never equal to $\neg$ becomes unnecessary. With our definition, as well with the definition of [15], the set of formulas is hereditarily finite. This will not be used here. However, we shall use the fact that the set of formulas is countable.

If a and b are variables, then $(\varepsilon, (a, b))$ and $(\approx, (a, b))$ belong to $\mathscr{F}_0$, thus are formulas; they are called *atomic formulas*, and denoted by $a\varepsilon b$ or $a \approx b$.

```
Lemma atomic_formulaP x:
   atomic_formulap x <->
   [/\ pairp x, pairp (Q x), variablep (P (Q x)), variablep (Q (Q x)) &
         (P x = val_in \/ P x = val_eq)].
Lemma next_formulasP F x :
  inc x (next_formulas F) <->
  [\/ (inc x F),
```

```
      [/\ pairp x, P x = val_not & inc (Q x) F ],
      [/\ pairp x, P x = val_or, pairp (Q x), inc (P (Q x)) F &
       inc  (Q (Q x))  F] |
      [/\ pairp x, P x = val_ex, pairp (Q x), variablep (P (Q x)) &
       inc  (Q (Q x))  F] ].

Lemma formulas_rec0: Vf formulas_rec \0c = atomic_formulas.
Lemma formulas_rec_succ n: natp ->
   Vf formulas_rec (csucc n) = next_formulas (Vf formulas_rec n).
Lemma formula_hi x n: natp n -> inc x (Vf formulas_rec n) -> formulap x.
Lemma formula_i1 x : formulap x ->
  exists2 n, inc n Nat & inc x (Vf formulas_rec n).
Lemma formula_sub_rec n m: n <=c m -> inc m Nat ->
  sub (Vf formulas_rec n) (Vf formulas_rec m).
Lemma formula_Vomega n: natp n ->
  sub (Vf formulas_rec n) (universe omega0).
Lemma formulas_Vomega: sub all_formulas (universe omega0).
Lemma formula_HF x: formulap x -> hereditarily_finite x.
Lemma countable_formulas: countable_set all_formulas.
```

The length of a formula. If x is a formula, the least integer n such that $x \in \mathscr{F}_n$ is called the *length* of the formula. The length of a formula is zero if and only if the formula is atomic. If x has length n and $n \le m$ then $x \in \mathscr{F}_m$. If n is non-zero, and $m = n+1$, then $x \in \mathscr{F}_{m+1} - \mathscr{F}_m$. This means that the formula is one of $\neg\phi$, $\phi \vee \psi$ and $\bigvee_v \phi$ where ϕ, ψ are in $\mathscr{F}_m$, v is a variable.

```
Definition formula_len x :=
  intersection (Zo Nat (fun n => inc x (Vf formulas_rec n))).

Lemma formula_len_pr x (n := formula_len x): formulap x ->
  [/\ natp n, inc x (Vf formulas_rec n) &
   forall m, natp m -> inc x (Vf formulas_rec m) -> n <=c m].
Lemma BS_formula_len x: formulap x -> natp (formula_len x).
Lemma flength_rec x m (n := formula_len x):
   formulap x -> n <=c m -> natp m -> inc x (Vf formulas_rec m).
Lemma flength0P x: formulap x ->
   (formula_len x = \0c <-> atomic_formulap x).
Lemma formula_i x: formulap x ->  inc x (Vf formulas_rec (formula_len x)).
Lemma flength_nz x (n := formula_len x) (F := (Vf formulas_rec (cpred n))):
  formulap x -> n <> \0c ->
  [/\ natp (cpred n), inc x (next_formulas F) & ~ inc x F].
```

It is obvious by induction that a formula is an ordered pair, whose first component is an operator. If for instance this operator is $\vee$, then the formula x is $\phi \vee \psi$, where ϕ and ψ are formulas of length smaller then the length of x. Conversely, if ϕ and ψ are formulas, so is $\phi \vee \psi$ and the length is one more than the maximum of the lengths of ϕ and ψ. The length of $\neg\phi$ or $\bigvee_x \phi$ is one more than the length of ϕ.

```
Lemma fkind_pa x: formulap x ->
  ((P x = val_in \/ P x = val_eq )  <-> atomic_formulap x).

Lemma fkind_eq_p1 x (b:= (P (Q x))) (c := Q (Q x)):
  formulap x -> P x = val_eq ->
  [/\  x = J val_eq (J b c), variablep b & variablep c].
```

```
Lemma fkind_eq_p2 a b (x := J val_eq (J a b)):
  variablep a -> variablep b -> (formulap x /\ formula_len x = \0c).

Lemma fkind_in_p1 x (b:= (P (Q x))) (c := Q (Q x)):
  formulap x -> P x = val_in ->
  [/\  x = J val_in (J b c), variablep b & variablep c].
Lemma fkind_in_p2 a b (x := J val_in (J a b)):
  variablep a -> variablep b -> (formulap x /\ formula_len x = \0c).

Lemma fkind_or_p1 x (a := P (Q x)) (b := Q (Q x)):
  formulap x ->  P x = val_or ->
  [/\ x = J val_or (J a b), formulap a, formulap b,
    formula_len a <c formula_len x & formula_len b <c formula_len x].
Lemma fkind_or_p2 a b (x := J val_or (J a b)):
  formulap a -> formulap b ->
 (formulap x /\ formula_len x = csucc (cmax (formula_len a) (formula_len b))).

Lemma fkind_not_p1 x (a := Q x):
  formulap x -> P x = val_not ->
  [/\ x = J val_not a, formulap a & formula_len a <c formula_len x].
Lemma fkind_not_p2 a (x := J val_not a):
  formulap a ->
  (formulap x /\ formula_len x = csucc (formula_len a)).

Lemma fkind_ex_p1 x (a := (P (Q x))) (b := Q (Q x)):
  formulap x -> P x = val_ex ->
  [/\ x = J val_ex (J a b), variablep a, formulap b
       & formula_len b <c formula_len x].
Lemma fkind_ex_p2 a b (x := J val_ex (J a b)):
  variablep a -> formulap b ->
  (formulap x /\ formula_len x = csucc (formula_len b)).
```

Definition by induction. Consider the following: let's say that ϕ is equalitarian if ε does not appear in it. This is defined by a boolean function f as: if ϕ is $a \approx b$, then f is true, if ϕ is $a\varepsilon b$, then f is false, if ϕ is $\psi\vee\psi'$ then f is the conjonction of $f(\psi)$ and $f(\psi')$, etc. One could define f by induction on n on each $\mathscr{F}_n$, the trouble being that the union of the $\mathscr{F}_n$ is not disjoint. Making it disjoint is easy: $\mathscr{F}$ is the disjoint union of the $\mathscr{F}'_n$, the set of formulas of length n; however the recursion properties of $\mathscr{F}'_n$ are not obvious.

For these reasons, we define f via stratified induction, using the length for ρ. Here W_α is: the empty set when $\alpha = 0$, $\mathscr{F}'_n$ if $\alpha = n+1$ for some integer n, and $\mathscr{F}$ for all other ordinals.

We deduce: for every $H(x, g)$ there exists f such that $f(x) = H(x, f_x)$ where f_x is the functional graph that coincides with f on $W_{\text{length}(x)}$. Assume for instance that x is $\neg\phi$, where ϕ is of length k. Then $k < n$, $\phi \in W_n$, and $f_x(\phi) = f(\phi)$ (see examples below).

```
Definition length_smaller_formulas i :=
   Yo (i = \0c) emptyset
   (Yo (natp i) (Vf formulas_rec (cpred i)) all_formulas).
Definition small_formulas n := Zo all_formulas (fun z => formula_len z <c n).

Lemma stratified_formula_ax1:
  (forall x, formulap x -> ordinalp (formula_len x)).
Lemma stratified_formula_ax2' i: ordinalp i ->
  (forall x, inc x (length_smaller_formulas i)  <->
```

```
    formulap x /\ formula_len x <o i).
Lemma stratified_formula_ax2:
 (forall i,ordinalp i ->
   exists E, forall x, inc x E <-> formulap x /\ formula_len x <o i).
Lemma stratified_formula_rec  (H : fterm2)
  (f := stratified_fct formulap H formula_len):
  (forall x, formulap x ->
     f x = H x (Lg (small_formulas (formula_len x)) f)).
```

We define now $\mathrm{fv}(\phi)$, by induction on the length of the formula ϕ. This is a finite subset of $\mathcal{V}$; it will be called the *set of free variables* of ϕ and an element of this set will be called a *free variable*. The different cases are

- If ϕ is $a \approx b$ or $a \varepsilon b$ then $\mathrm{fv}(\phi)$ is $\{a, b\}$.
- If ϕ is $\neg\psi$ then $\mathrm{fv}(\phi) = \mathrm{fv}(\psi)$,
- If ϕ is $\psi \vee \psi'$ then $\mathrm{fv}(\phi) = \mathrm{fv}(\psi) \cup \mathrm{fv}(\psi')$,
- If ϕ is $\bigvee_x \psi$ then $\mathrm{fv}(\phi) = \mathrm{fv}(\psi) - \{x\}$.

```
Definition free_vars_aux x f :=
  Yo (P x = val_in \/ P x = val_eq)  (doubleton (P (Q x)) (Q (Q x)))
  (Yo (P x = val_not ) (Vg f (Q x))
   (Yo (P x = val_or) (Vg f (P (Q x)) \cup  (Vg f (Q (Q x))))
     ((Vg f (Q (Q x)) -s1 (P (Q x)))))).


Definition free_vars := stratified_fct formulap free_vars_aux formula_len.
Definition closed_formula x := free_vars x = emptyset.

Lemma free_vars_rec x: formulap x ->
  free_vars x =
  free_vars_aux x (Lg (small_formulas (formula_len x)) free_vars).

Lemma free_vars_eq b c (x := J val_eq (J b c)):
  variablep b -> variablep c -> free_vars x = doubleton b c.
Lemma free_vars_in b c (x := J val_in (J b c)):
  variablep b -> variablep c -> free_vars x = doubleton b c.
Lemma free_vars_not a (x := J val_not a): formulap a ->
  free_vars x = free_vars a.
Lemma free_vars_or a b (x := J val_or (J a b)):
  formulap a -> formulap b ->
  free_vars x = free_vars a \cup free_vars b.
Lemma free_vars_ex  a b (x := J val_ex (J a b)):
  variablep a -> formulap b ->
  free_vars x = (free_vars b) -s1 a.

Lemma free_vars_variables x: formulap x ->
  sub (free_vars x) variables /\ finite_set (free_vars x).
```

Let X be a set, ϕ a formula. We define, by induction on the length of ϕ, the value $\mathrm{Val}(\phi, \mathrm{X})$ of ϕ with respect to X; this is a subset of $\mathrm{X}^{\mathrm{fv}(\phi)}$, i.e., if $f \in \mathrm{Val}(\phi, \mathrm{X})$ then f is a functional graph, its domain is the set of free variables of ϕ, and its range is a subset of X; moreover, it is assumed:

- if ϕ is $a \approx b$, that $f(a) = f(b)$;
- if ϕ is $a \varepsilon b$, that $f(a) \in f(b)$;
- if ϕ is $\neg\psi$, that f is not in Val(ψ,X);
- if ϕ is $\psi \vee \psi'$, that f, restricted to the free variables of ψ, is in Val(ϕ,X), or that f, restricted to the free variables of ψ', is in Val(ϕ',X).
- and if ϕ is $\bigvee_x \psi$, that f can be extended to an element of Val(ψ,X).

We start with the definitions.

```
Definition formula_values X x := gfunctions (free_vars x) X.

Definition Val_of_eq X a b :=
  Zo (gfunctions (doubleton a b) X) (fun z => Vg z a = Vg z b).
Definition Val_of_inc X a b :=
  Zo (gfunctions (doubleton a b) X) (fun z => inc (Vg z a) (Vg z b)).
Definition Val_of_not X v V := (gfunctions v X) -s V.
Definition Val_of_or X v1 v2 V1 V2 :=
   Zo (gfunctions (v1 \cup v2) X)
   (fun z => inc (restr z v1) V1 \/inc (restr z v2) V2).
Definition Val_of_ex X v V :=
  Zo (gfunctions v X) (fun z => exists2 f, inc f V & z = restr f v).

Definition formula_val_aux X x f (t:= P x) :=
  Yo (P x = val_in) (Val_of_inc X (P (Q x)) (Q (Q x)))
     (Yo (P x = val_eq) (Val_of_eq X  (P (Q x)) (Q (Q x)))
         (Yo (P x = val_not) (Val_of_not X (free_vars x) (Vg f (Q x)))
             (Yo (P x = val_or) (Val_of_or X (free_vars (P (Q x)))
                    (free_vars (Q (Q x))) (Vg f (P (Q x))) (Vg f (Q (Q x)))))
Definition formula_val X:= stratified_fct formulap
    (formula_val_aux X) formula_len.
```

In each case we give a necessary and sufficient condition for $f \in$ Val(ϕ,X).

```
Lemma formula_val_rec X x: formulap x ->
  formula_val X x = formula_val_aux X x
      (Lg (small_formulas (formula_len x)) (formula_val X)).

Lemma formula_val_eq X a b (x := J val_eq (J a b)) (V := formula_val X x):
  variablep a -> variablep b ->
  [/\ V  =  Val_of_eq X a b, sub V (formula_values X x) &
    forall f,  inc f V <->
    [/\ fgraph f, domain f = doubleton a b, Vg f a = Vg f b & inc (Vg f a) X]].

Lemma formula_val_inc X a b (x := J val_in (J a b)) (V := formula_val X x):
  variablep a -> variablep b ->
  [/\ V  =  Val_of_inc X a b, sub V (formula_values X x) &
   forall f, inc f V <->
   [/\ fgraph f, domain f = doubleton a b, inc (Vg f a) X, inc (Vg f b) X &
         inc (Vg f a) (Vg f b)]].
Lemma formula_val_not X a  (x := J val_not a)
    (V := formula_val X x) (Va := formula_val X a):
```

```
  formulap a ->
  [/\ V  = Val_of_not X (free_vars a) Va, sub V (formula_values X x) &
    forall f,  inc f V <->
      inc f  (formula_values X x) /\ ~ inc f Va].

Lemma formula_val_or X  a b (x := J val_or (J a b))
    (V := formula_val X x) (Va := formula_val X a)(Vb := formula_val X b):
  formulap a -> formulap b ->
  [/\ V = Val_of_or X (free_vars a) (free_vars b) Va Vb,
     sub V (formula_values X x) &
   forall f, inc f V <->
     (inc f  (formula_values X x) /\
         (inc (restr f (free_vars a)) Va \/ inc (restr f (free_vars b)) Vb))].

Lemma formula_val_ex X  a b (x := J val_ex (J a b))
  (V := formula_val X x) (Vb := formula_val X b):
  variablep a -> formulap b ->
  [/\ V = Val_of_ex X (free_vars x) Vb,
     sub V (formula_values X x) &
   forall f, inc f V <->
     (inc f  (formula_values X x) /\
         exists2 g, inc g Vb & f = restr g (free_vars x))].
```

By case analysis, we have $\mathrm{Val}(\phi, X) \subset X^{\mathrm{fv}(\phi)}$. If $\mathrm{fv}(\phi)$ is empty, then ϕ is said *closed.* In this case, $f \in \mathrm{Val}(\phi, X)$ says that f is the empty function. Thus $\mathrm{Val}(\phi, X)$ is either empty or $\{\emptyset\}$ (this is the integer one). We say that ϕ is false or true in X in these cases.

We can simplify some formulas: We have $\mathrm{Val}(\phi \vee \phi, X) = \mathrm{Val}(\phi, X)$, $\mathrm{Val}(\neg\neg\phi, X) = \mathrm{Val}(\phi, X)$, and, whenever v is not a free variable of ϕ, $\mathrm{Val}(\bigvee_v \phi, X) = \mathrm{Val}(\phi, X)$,

```
Lemma formula_val_i X x:
    formulap x -> sub (formula_val X x) (formula_values X x).
Lemma formula_val_ic X x f:
  formulap x -> closed_formula x -> inc f (formula_val X x) ->
  f = emptyset.
Lemma gfunctions_empty X: (gfunctions emptyset X) = \1c.

Lemma formula_or_simp X a: formulap a ->
   formula_val X (J val_or (J a a)) =  formula_val X a.
Lemma formula_not_simp X a: formulap a ->
   formula_val X (J val_not (J val_not a)) =  formula_val X a.
Lemma formula_ex_simp X a b:
  variablep a -> formulap b -> ~(inc a (free_vars b)) ->
  formula_val X (J val_ex (J a b)) =  formula_val X b.
```

Example. Consider $\phi_0 = \neg\bigvee_0(0 \approx 0)$. This is a free formula of length two. If X is non-empty, then ϕ_0 is false in X. The formula $\phi_1 = \neg\bigvee_0\neg(0 \approx 0)$ has length three and is true in any set. The expression can be interpreted as $\forall x \in X, x = x$.

```
Definition formula_ex0 :=
  J val_not (J val_ex (J \0c (J val_eq (J \0c \0c)))).
Definition formula_ex1 :=
 J val_not (J  val_ex (J \0c (J val_not (J val_eq (J \0c \0c))))).

Lemma formula_ex0_pr (x := formula_ex0) X:
```

```
   nonempty X ->
  [/\ formulap x, formula_len x = \2c, free_vars x = emptyset &
     (formula_val X x) = \0c].

Lemma formula_ex1_pr (x := formula_ex1) X:
  [/\ formulap x, formula_len x = \3c, free_vars x = emptyset &
     (formula_val X x) = \1c].
```

Example 2. Write $a \to b$ instead of $\neg a \vee b$. Consider $\neg\bigvee_0\neg(0\varepsilon 1 \to 0\varepsilon 2)$. This is a formula of length five, with two free variables, 1 and 2. If f is a functional graph $\{1,2\} \to \mathrm{X}$, then $f \in \mathrm{Val}(\phi, \mathrm{X})$ if and only if $f(1) \subset f(2)$.

Note. We really have: if $f \in \mathrm{Val}(\phi, \mathrm{X})$ then $\forall t \in \mathrm{X}, t \in f(1) \implies t \in f(2)$. In order to conclude $f(1) \subset f(2)$ one needs $f(1) \in \mathrm{X}$; since $f(1)$ is arbitrary, we require X to be transitive.

```
Definition formula_ex2 :=
  J val_not (J val_ex (J \0c (J val_not (J val_or
     (J (J val_not (J val_in (J \0c \1c)))(J val_in (J \0c \2c))))))).


Lemma formula_ex2_pr (x := formula_ex2) X: (* 125 *)
  transitive_set X ->
  [/\ formulap x, formula_len x = card_five, free_vars x = doubleton \1c \2c &
   forall f, inc f (formula_val X x) <->
     (inc f (formula_values X x) /\ sub (Vg f \1c) (Vg f \2c))].
```

We say that $\Phi = (\phi, f)$ is a *formula with parameters* when ϕ is a formula, f is a functional graph whose domain is a subset of fv(ϕ). We say that is it *with values in* X when the range of f is a subset of X. The set of free variables of Φ, denoted by fv(Φ), is the complement of the domain of f in fv(ϕ). If Φ has no free variable, it is said to be *closed*. This means that the domain of f is the whole of fv(ϕ).

We show here: the cardinal of the set F of formulas with values in X is $\mathrm{card(X)} + \aleph_0$. Let c be this quantity. Since $\aleph_0$ is infinite, this is the maximum of $\aleph_0$ and card(X). Remember that $\aleph_0$ is the number of variables, as well as the number of formulas. We have $\mathrm{card(X)} \le \mathrm{card(F)}$: take for ϕ the formula $0 \approx 0$ and for f the function such that $f(0) = x$, for any $x \in \mathrm{X}$. We have $\aleph_0 \le \mathrm{card(F)}$: take for ϕ the formula $i \approx i$, where i is a variable and for f the empty function.

Let F_ϕ be the subset of F where the first component is ϕ, and $\mathrm{F}_{\phi,\mathrm{T}}$ the set of all pairs (ϕ, f), where f is a functional graph with domain T and range in X. If T is finite, the cardinal of $\mathrm{F}_{\phi,\mathrm{T}}$ is finite when X is finite or T empty, is card(X) otherwise. In any case, it is $\le c$. Now F_ϕ is the union of the $\mathrm{F}_{\phi,\mathrm{T}}$, where T is a subset of fv(ϕ). Since fv(ϕ) is finite, each T is finite and there is a finite number of T. Thus $\mathrm{card}(\mathrm{F}_\phi) \le c$. Since F is the union of the F_ϕ and there is a countable number of formulas, we deduce $\mathrm{card(F)} \le c$.

```
Definition pformula x :=
  [/\ pairp x, formulap (P x), fgraph (Q x)
     & sub (domain (Q x)) (free_vars (P x))].
Definition pformula_in X x := pformula x /\ sub (range (Q x)) X.
Definition pfree_vars x := (free_vars (P x)) -s (domain (Q x)).
Definition pclosed x := pfree_vars x = emptyset.

Definition pformulas_in X :=
   Zo (all_formulas \times (sub_fgraphs variables X)) (pformula_in X).
```

```
Definition pcformulas_in X :=
   Zo (all_formulas \times (sub_fgraphs variables X))
      (fun f => pformula_in X f /\ free_vars (P f) = domain (Q f)).

Lemma pformulas_inP X f: inc f (pformulas_in X) <->  pformula_in X f.
Lemma pcformulas_inP X f:
   inc f (pcformulas_in X) <->  (pformula_in X f /\ pclosed f).
Lemma aleph0_pr X (x := cardinal X) (c := x +c aleph0):
  [/\  aleph0 = cardinal Nat, aleph0 = cardinal variables,
       (x <=c aleph0 -> c = aleph0),   aleph0 <=c x -> c = x
      & [/\ infinite_c c,  aleph0 <=c c,& cardinal X <=c c &
        (forall z, aleph0 <=c z -> cardinal X <=c z -> c <=c z  ]].
Lemma cardinal_pformulas_in X :
  cardinal (pformulas_in X) = cardinal X +c aleph0.   (* 79 *)
```

Assume that $\Phi = (\phi, f)$ and that f takes its values in X. We define Val(Φ,X) to be the set of all functional graphs g with domain fv(Φ) such that $f \cup g$ is in Val(ϕ,X) (note that fv(ϕ) is the disjoint union of the domains of f and g so that $f \cup g$ is the functional graph that associates $f(x)$ or $g(x)$ to every free variable x of ϕ).

If Φ is closed then fv(Φ) is empty, and g is the empty function; so that $f \cup g = f$. Thus Val(Φ,X) is either 0 (when $f \notin$ Val(ϕ,X)) or 1 (when $f \in$ Val(ϕ,X)). If the value is one we write $X \Vdash \Phi$.

Example. Take for ϕ the formula ϕ_2 studied above and for f the function $(1 \mapsto a; 2 \mapsto b)$. Then we get a closed formula with parameters. Assume a and b are in X and X is transitive. Then $X \Vdash \Phi$ is equivalent to $a \subset b$.

```
Definition pformula_vals X x :=
  Zo (gfunctions (pfree_vars x) X)
      (fun g => inc (Q x \cup g) (formula_val X (P x))).
Definition pformula_valid_on X f := pformula_vals X f = \1c.

Lemma pformula_vals_aux X x g:
  pformula_in X x -> inc g (gfunctions (pfree_vars x) X) ->
   inc (Q x \cup g) (formula_values X (P x)).
Lemma pfree_vars_closed_pr1 X x (V:= formula_param_vals X x):
  pclosed x ->
   (V = \0c \/ V = \1c) /\ (V = \1c <-> inc (Q x) (formula_val X (P x))).
Lemma pfree_vars_closed_pr X x:
  pclosed x ->
  (pformula_valid_on X x <-> inc (Q x) (formula_val X (P x))).

Lemma pformula_ex a b X (x := formula_ex2)
   (g := variantL \1c \2c a b) (y := J x g):
  transitive_set X -> inc a X -> inc b X ->
  [/\  pformula y, pclosed y,  pformula_in X y &
  (pformula_vals X y) = \1c <-> (sub a b)].
```

The theorem of Löwenheim-Skolem. Let X a set, $P \subset X$. Let $\mathscr{A}$ be the set of closed formulas with parameters $\Phi = (\phi, f)$ where the range of f is a subset of P and $X \Vdash \Phi$. There is a subset Y of X that contains P, such that $Y \Vdash \Phi$ whenever $\Phi \in \mathscr{A}$; moreover the cardinal of Y is at most the cardinal of P plus $\aleph_0$ (This can be restated as: if P is finite, then Y is countable, otherwise Y has the same cardinal as P).

We start with some definitions (the last one is $\mathscr{A}$) and trivial lemmas.

```
Definition sub_fgraphs A B :=  unionf (powerset A) (gfunctions ^~ B).

Definition pformulas_in X :=
   Zo (all_formulas \times (sub_fgraphs variables X))
      (pformula_in X).
Definition pformulas_in_val X P :=
   Zo (pcformulas_in P) (pformula_valid_on X).


Lemma sub_fgraphsP A B f:
   inc f (sub_fgraphs A B) <-> exists2 C, sub C A & inc f (gfunctions C B).
Lemma pformulas_inP X f: inc f (pformulas_in X) <->  pformula_in X f.
Lemma pcformulas_inP X f:
   inc f (pcformulas_in X) <->  (pformula_in X f /\ pclosed f).
Lemma pcformulas_in_valP X P f:
 inc f (pformulas_in_val X P) <->
  [/\ pformula_in P f, pclosed f& pformula_valid_on X f].
```

From now on, the set X will be fixed. We consider a formula $\Phi = (\phi, f)$ with one free variable x and parameters in X. For any $a \in \mathrm{X}$, we can extend f to f' by $f'(x) = a$, so that (ϕ, f') becomes closed. We denote by U_ϕ the set of all a such that $\mathrm{X} \Vdash (\phi, f')$.

We may also consider $\phi' = \bigvee_x \phi$ and the closed formula $\Phi' = (\phi', f)$ with parameters in X. Assume $\mathrm{X} \Vdash \Phi'$. Then $f \in \mathrm{Val}(\bigvee_x \phi, \mathrm{X})$. By definition f can be extended to an element of $\mathrm{Val}(\phi, \mathrm{X})$. Let g be the extension, and $a = g(x)$. Then $a \in \mathrm{U}_\Phi$, so that U_Φ is non-empty.

```
Definition pformula_inst f :=
  J (J val_ex (J (union (pfree_vars f)) (P f))) (Q f).
Definition LS_setU X f (x:= (union (pfree_vars f))) :=
   Zo X (fun a => pformula_valid_on X (J (P f) ((Q f) +s1 (J x a)))).

Lemma LS_setU_p1 X f a (x:= (union (pfree_vars f)))
   (h:= (J (P f) ((Q f) +s1 (J x a)))):
  pformula_in X f -> (singletonp (pfree_vars f)) -> inc a X ->
  pformula_in X h /\ pclosed h.
Lemma LS_setU_P X f a (x:= (union (pfree_vars f)))
   (h:= (J (P f) ((Q f) +s1 (J x a)))):
  pformula_in X f -> (singletonp (pfree_vars f)) ->
  ( inc a (LS_setU X f) <->
    [/\ inc a X, pformula_in X h,  pclosed h & pformula_valid_on X h] ).

Lemma pformula_inst_p1 f (g := pformula_inst f):
  pformula f -> (singletonp (pfree_vars f)) ->
  [/\ pformula g, pclosed g ,
    (forall X, pformula_in X f -> pformula_in X g) &
     forall X,  pformula_valid_on X g -> nonempty (LS_setU X f)].
```

Let $\mathscr{G}_\mathrm{P}$ be the set of all formulas Φ such that, with the same notations as above, f takes its values in P and Φ' is satisfied in X. Let $r(\Phi)$ be the representative of this U_Φ. If $\Phi \in \mathscr{G}_\mathrm{P}$, then U_Φ in then nonempty so that[5] $r(\Phi) \in \mathrm{U}_\Phi$.

The set of all $r(\Phi)$ is denoted N(P). We define P_n by induction as $\mathrm{P}_0 = \mathrm{P}$ and $\mathrm{P}_{n+1} = \mathrm{N}(\mathrm{P}_n)$. We define Y to be the union of the P_i. It is easy to show that $\mathrm{card}(\mathrm{Y}) \le \mathrm{card}(\mathrm{P}) + \aleph_0$.

[5]We use here the axiom of choice

```
Definition LS_nextP_aux X P :=
 (Zo  (pformulas_in P)
     (fun f => [/\ pformula f, (singletonp (pfree_vars f))  &
       pformula_valid_on X (pformula_inst f)])).

Definition LS_nextP X P :=
  fun_image (LS_nextP_aux X P) (fun z => rep (LS_setU X z)).
Definition LS_rec X P :=  induction_defined  (LS_nextP X) P.
Definition LS_res X P := union (target (LS_rec X P)).

Lemma LS_res_p0 X P x:
   inc x (LS_res X P) <-> (exists2 n, inc n Nat & inc x (Vf (LS_rec X P) n)).
Lemma card_LS_nextP X P :
   cardinal (LS_nextP X P) <=c cardinal P +c aleph0.
Lemma card_LS_res X P :
   cardinal (LS_res X P) <=c cardinal P +c aleph0.
```

Consider the formula $\Phi = (\phi, f)$ where ϕ is $0 \approx 1$, and $f(1) = a$. There is one free variable 0; extend f to f' by $f'(0) = b$. Here U_Φ is the set of all $b \in X$ such that $X \Vdash (\phi, f')$. This simplifies to $a = b$, so that $U_\Phi = \{a\}$ (provided $a \in X$). We deduce $r(\Phi) = a$. Assume $a \in Q$; then $a \in N(Q)$. Thus $Q \subset X$ says $Q \subset N(Q)$. By induction, $P_i \subset P_{i+k}$; so $P \subset X$ implies $P \subset Y \subset X$. Moreover, given a finite subset E of Y, there is (by induction on E) an integer n such that every element of E is in P_n.

```
Lemma formula_ext3 a X (z := J (J val_eq (J \0c \1c))(singleton (J \1c a))):
  inc a X ->
  (LS_setU X z = singleton a /\
     forall Y, inc a Y -> (inc z (LS_nextP_aux Y X))).  (* 82 *)

Lemma rep_set1 a: rep (singleton a) = a.
Lemma LS_nextP_pr1 X P: sub P X -> sub P (LS_nextP X P).

Lemma LS_nextP_pr2 X P n (Y:= (Vf (LS_rec X P) n)):
   sub P X -> natp n ->  sub P Y /\ sub Y X.
Lemma LS_nextP_pr3 X P (Y:= LS_res X P):  sub P X ->  sub P Y /\ sub Y X.
Lemma LS_nextP_pr4 X P n m: sub P X -> n <=c m -> inc m Nat ->
  sub  (Vf (LS_rec X P) n) (Vf (LS_rec X P) m).
Lemma LS_nextP_pr5 X P E:
  sub P X -> finite_set E -> sub E (LS_res X P)  ->
  exists2 m, natp m & sub E (Vf (LS_rec X P) m).
```

The theorem of Löwenheim-Skolem follows trivially from: if $\Phi = (\phi, f)$ is a closed formula with parameters in Y, then $X \Vdash \Phi$ is equivalent to $Y \Vdash \Phi$. The proof in by induction on the length of ϕ. The non-trivial point is to show that, if ϕ is $\bigvee_x \psi$ then $X \Vdash \Phi$ is equivalent to $Y \Vdash \Phi$. Recall that $X \Vdash \Phi$ is equivalent to $f \in \mathrm{Val}(\phi, X)$, in this case it is: f can be extended into f', such that $f' \in \mathrm{Val}(\psi, X)$, and $f'(x) \in X$. One implication follows from $Y \subset X$.

Since all parameters are in Y, there is an integer m such that all parameters are in P_m. We may assume that x is a free variable of ψ (for otherwise the formula simplifies). This says that $\Psi = (\psi, f)$ is formula with one free parameter x. Recall the previous discussion about formulas with one parameter (and replace ϕ or Φ by ψ or Ψ). Note that ψ' is ϕ and Ψ' is Φ. As $X \Vdash \Phi$, we get that U_Ψ is non-empty. Let $a = r(\Psi)$. This is in $\mathscr{G}_{P_m}$, thus in P_{m+1}, thus in Y. It is also in U_Ψ. This means that, if we extend f to f' by imposing $f'(x) = a$ we get $X \Vdash (\psi, f')$. Note that the range of f' is a subset of Y (since f takes its values in Y and $a \in Y$). By induction, $Y \Vdash (\psi, f')$. This says $f' \in \mathrm{Val}(\psi, Y)$, thus $f \in \mathrm{Val}(\phi, Y)$ and $Y \Vdash (\phi, f)$.

```
Lemma LS_main X P f (Y:=LS_res X P) :  (* 131 *)
    sub P X -> pformula_in Y f -> pclosed f ->
   (pformula_valid_on X f <-> pformula_valid_on Y f).

Theorem Lowenheim_Skolem X P (Y:=LS_res X P):
   sub P X ->
   [/\ sub P Y, sub Y X, cardinal Y <=c cardinal P +c aleph0 &
    forall F, inc F (pformulas_in_val X P) ->  pformula_valid_on Y F].
```

11.28 Order types

Let a_i and b_i be two families of orders, index by an ordered set I. We may compare orders, thus assume $a_i \le b_i$ for every i. We may consider the ordinals sums $\sum a_i$ and $\sum b_i$. We have then $\sum a_i \le \sum b_i$. This holds if r is well-ordered, and the orders are replaced by ordinals.

```
Lemma osum_increasing1 r f g:
  orsum_ax r f -> orsum_ax r g ->
  (forall x, inc x (domain f) -> order_le (Vg f x) (Vg g x)) ->
  order_le (order_sum r f) (order_sum r g). (* 61 *)
Lemma osum_increasing2 r f g:
  worder r ->
  substrate r = domain f -> substrate r = domain g ->
  (forall x, inc x (domain f) -> (Vg f x) <=o (Vg g x)) ->
  (osum r f) <=o (osum r g).
```

We deduce $\sum_J a_i \le \sum_I a_i$ when $J \subset I$.

```
Lemma osum_increasing3 r f j:
  orsum_ax r f ->
  sub j (domain f) ->
  order_le (order_sum (induced_order r j) (restr f j)) (order_sum r f).
Lemma osum_increasing4 r f j:
  worder_on r (domain f) -> sub j (domain f) -> ordinal_fam f ->
  (osum (induced_order r j) (restr f j)) <=o (osum r f).
```

The same holds for the lexicographic product (in the case of ordinals the set I has to be finite).

```
Lemma oprod_increasing1 r f g:
  orprod_ax r f -> oprod_ax r g ->
  (forall x, inc x (domain f) -> order_le (Vg f x) (Vg g x)) ->
  order_le (order_prod r f) (order_prod r g). (* 66 *)
Lemma oprod_increasing2 r f g: worder r ->
  substrate r = domain f -> substrate r = domain g ->
  (forall x, inc x (domain f) -> (Vg f x) <=o (Vg g x)) ->
  finite_set (substrate r) ->
  (oprod r f) <=o (oprod r g).
```

We deduce $\prod_J a_i \le \prod_I a_i$ when $J \subset I$ and factors in $I - J$ are non-zero.

```
Lemma oprod_increasing3 r f j:
  orprod_ax r f ->
```

```
  sub j (domain f) ->
  (forall x, inc x  ((domain f) -s j) ->
     substrate (Vg f x) <> emptyset) ->
  order_le (order_prod (induced_order r j) (restr f j)) (order_prod r f).
Lemma oprod_increasing4 r f j:
  worder_on r (domain f) -> sub j (domain f) -> ordinal_fam f ->
  finite_set (substrate r) ->
  (forall x, inc x ((domain f) -s j)  -> Vg f x <> \0o) ->
  (ord_prod (induced_order r j) (restr f j)) <=o (ord_prod r f).

Lemma cardinal_du2 A B :
   cardinal (canonical_du2  A B) = (cardinal A) +c (cardinal B).
```

We assume the existence of a function Ord(x) that satisfies two criteria: First, if x is an ordering, then x is order-isomorphic to Ord(x). Second, if x and y are order-isomorphic, then Ord(x) = Ord(y).

Objects of the form Ord(x) are called order-type; they can be compared by $x \prec_{\mathrm{Ord}} y$, meaning that there is an order-isomorphism of x onto some subset z of y.

```
Parameter order_type_of: Set -> Set.
Axiom order_type_exists:
  forall x, order x -> x \Is (order_type_of x).
Axiom order_type_unique:
  forall x y, x \Is y -> (order_type_of x = order_type_of y).
Definition order_type x := exists2 y, order y & x = order_type_of y.
Definition order_type_le x y:=
   [/\ order_type x, order_type y &
   exists f z,
    sub z (substrate y) /\ order_isomorphism f x (induced_order y z)].
Notation "x <=t y" := (order_type_le x y) (at level 60).
Notation "x <=O y" := (order_le x y) (at level 60).
```

We have Ord(r) = Ord(o(ord(r))) whenever r is well-ordered. This relation $x \prec_{\mathrm{Ord}} y$ is reflexive and transitive (but not antisymmetric).

```
Lemma OT_ordinal_compat x: worder x ->
  order_type_of x = order_type_of (ordinal_o (ordinal x)).
Lemma OT_prop0 x: order x -> (order_type_of x) \Is x.
Lemma OT_prop1 x: order x -> order_type (order_type_of x).
Lemma OT_prop2 x: order_type x -> order x.
Lemma OT_prop3 x: order x -> order (order_type_of x).
Lemma OT_prop4 x: order x ->
  order_type_of (order_type_of x) = order_type_of x.

Lemma order_le_alt4 r r': r <=O r' ->
   (exists f, order_morphism f r r').
Lemma order_le_alt3P r r':
  r <=O r' <-> (exists f, order_morphism f r r').
Lemma order_le_transitive: forall x y z,
  x <=O y -> y <=O z -> x <=O z. (* 28 *)
Lemma OTorder_le_altP r r':
  r <=t r' <-> [/\ order_type r, order_type r' &
    exists f, order_morphism f r r'].
Lemma OTorder_le_alt2P r r':
```

```
  r <=t r' <-> [/\ order_type r, order_type r' & r <=O  r'].
Lemma OTorder_le_compat1 r: order r ->
  r <=O (order_type_of r).
Lemma OTorder_le_compat2 r: order r ->
  (order_type_of r) <=O r.
Lemma OTorder_le_compatP r r': order r -> order r' ->
  (r <=O r' <-> (order_type_of r) <=O  (order_type_of r')).

Lemma OT_order_le_reflexive x: order_type x -> x <=t x.
Lemma OT_order_le_transitive x y z:
  x <=t y -> y <=t z -> x <=t z.
```

We define here the ordinal sum and ordinal product, according to Bourbaki.

```
Definition OT_sum r g :=
   order_type_of (order_sum r (Lg (domain g)
     (fun z => (order_type_of (Vg g z))))).
Definition OT_prod r g :=
   order_type_of (order_prod r (Lg (domain g)
     (fun z => (order_type_of (Vg g z))))).
Definition OT_sum2 a b :=
  order_type_of (order_sum2 (order_type_of a) (order_type_of b)).
Definition OT_prod2 a b :=
  order_type_of (order_prod2 (order_type_of a) (order_type_of b)).
Notation "x +t y" := (OT_sum2 x y) (at level 50).
Notation "x *t y" := (OT_prod2 x y) (at level 40).
```

We show some basic properties of the ordinal sum and product.

```
Lemma OT_sum2_pr a b:
  a +t b = OT_sum canonical_doubleton_order (variantLc a b).
Lemma OT_prod2_pr a b:
  a *t b = OT_prod canonical_doubleton_order (variantLc b a).

Lemma OT_sum_ordertype r g:
  order r -> substrate r = domain g -> order_fam g ->
  order_type (OT_sum r g).
Lemma OT_prod_ordertype r g:
  worder r -> substrate r = domain g -> order_fam g ->
  order_type (OT_prod r g).
Lemma OT_sum2_ordertype a b: order_type a -> order_type b ->
  order_type (a +t b).
Lemma OT_prod2_ordertype a b: order_type a -> order_type b ->
  order_type (a *t b).
```

We show what happens when an ordering is replaced by an isomorphic one.

```
Lemma OT_sum_invariant3 r g:
  order r -> substrate r = domain g -> order_fam g ->
  order_type_of (order_sum r g) =
    OT_sum r (Lg (substrate r) (fun i => order_type_of (Vg g i))).
Lemma OT_prod_invariant3 r g:
  worder r -> substrate r = domain g -> order_fam g ->
  order_type_of (order_prod r g) =
    OT_prod r (Lg (substrate r) (fun i => order_type_of (Vg g i))).
```

```
Lemma OT_sum_invariant5 a b c: order a -> order b -> order c ->
  (order_sum2 a b) \Is c ->
  (order_type_of a) +t (order_type_of b) = order_type_of c.
Lemma OT_prod_invariant5 a b c: order a -> order b -> order c ->
  (order_prod2 a b) \Is c ->
  (order_type_of a) *t (order_type_of b) = order_type_of c.
```

We may consider a set with three elements, well-order it, and use this to define $a+b+c$ or $a\cdot b\cdot c$, the sum or product of three terms. We could then partition our set, taking apart the least or greatest element, and apply twice the associativity theorem, then get

$$a+(b+c)=(a+b)+c \text{ and } a\cdot(b\cdot c)=(a\cdot b)\cdot c.$$

This is rather difficult; we prefer a direct proof; there is no difficulty here, except that the proofs are rather long, especially for the sum. We have similarly

$$a\cdot(b+c)=a\cdot b+a\cdot c.$$

Direct proof is easy. Note that there is a natural bijection between $(b+c)\cdot a$ and $b\cdot a+c\cdot a$ but it is not always order preserving.

```
Lemma osum_assoc2 a b c:
  order a -> order b -> order c ->
  (order_sum2 a (order_sum2 b c))
   \Is (order_sum2 (order_sum2 a b) c).   (* 141 *)
Lemma oprod_assoc2 a b c:
  order a -> order b -> order c ->
  (order_prod2 a (order_prod2 b c))
  \Is (order_prod2 (order_prod2 a b) c).  (* 82 *)

Lemma osum_distributive x y z:
  order x -> order y -> order z ->
  (order_prod2 z (order_sum2 x y))
   \Is (order_sum2 (order_prod2 z x) (order_prod2 z y)).   (* 125 *)
```

We show here associativity of the sum and product, and show that a product of two terms is a sum.

```
Definition order_type_fam g := (allf g order_type).

Lemma OT_sum_assoc1 r g r' g':
  order r -> substrate r = domain g  -> order_type_fam g ->
  order r' -> substrate r' = domain g' -> order_fam g' ->
  r = order_sum r' g' ->
  let order_sum_assoc_aux :=
    fun l =>
    OT_sum (Vg g' l) (Lg (substrate (Vg g' l)) (fun i => Vg g (J i l))) in
  OT_sum r g = OT_sum r' (Lg (domain g') (order_sum_assoc_aux)).  (* 30 *)

Lemma OT_prod_assoc1 r g r' g':
  worder r -> substrate r = domain g  -> order_type_fam g ->
  worder r' -> substrate r' = domain g' -> worder_fam g' ->
  r = order_sum r' g' ->
  (forall i, inc i (domain g') ->  finite_set (substrate (Vg g' i))) ->
  let order_prod_assoc_aux :=
```

```
    fun l =>
    OT_prod (Vg g' l) (Lg (substrate (Vg g' l)) (fun i => Vg g (J i l))) in
  OT_prod r g = OT_prod  r' (Lg (domain g') order_prod_assoc_aux). (* 31 *)
Lemma OT_sum_assoc3 a b c:
  order_type a -> order_type b -> order_type c ->
  a +t (b +t c) = (a +t b) +t c. (* 17 *)
Lemma OT_prod_assoc3 a b c:
  order_type a -> order_type b -> order_type c ->
  a *t (b *t c) = (a *t b) *t c. (* 17 *)
Lemma OT_sum_distributive3 a b c:
  order_type a -> order_type b -> order_type c ->
  c *t (a +t b) = (c *t a) +t (c *t b). (* 21 *)

Lemma OT_prod_pr1 a b c:
  order_type a -> order_type b -> worder c -> b \Is c ->
  a *t b = OT_sum c (cst_graph (substrate c) a).
```

We show compatibility of sum and product with the order.

```
Lemma OT_sum_increasing2 r f g: order r ->
  substrate r = domain f -> substrate r = domain g ->
  (forall x, inc x (domain f) -> (Vg f x) <=t (Vg g x)) ->
  (OT_sum r f) <=t (OT_sum r g). (* 20 *)
Lemma OT_prod_increasing2 r f g: worder r ->
  substrate r = domain f -> substrate r = domain g ->
  (forall x, inc x (domain f) -> (Vg f x) <=t (Vg g x)) ->
  (OT_prod r f) <=t (OT_prod r g).  (* 20 *)
Lemma OT_sum_increasing4 r f j:  order r ->  (* 33 *)
  substrate r = domain f ->
  sub j (domain f) -> order_type_fam f ->
  (OT_sum (induced_order r j) (restr f j)) <=t (OT_sum r f).
Lemma OT_prod_increasing4 r f j:  worder r ->  (* 41 *)
  substrate r = domain f ->
  sub j (domain f) -> order_type_fam f ->
  (forall x, inc x  ((domain f) -s j) ->
     substrate (Vg f x) <> emptyset) ->
  (OT_prod (induced_order r j) (restr f j)) <=t (OT_prod r f).
```

We study now the properties of opposite order-types.

```
Definition OT_opposite x := order_type_of (oppo_order x).

Lemma OT_opposite1 a b: a \Is b ->
  (opp_order a) \Is (opp_order b).
Lemma OT_opposite2 x: order x ->
   oppo_order (opp_order x) = x.
Lemma OT_opposite3 a: order a -> OT_opposite (order_type_of a) = OT_opposite a.

Lemma OT_double_opposite x: order_type x ->
   OT_opposite (OT_opposite x) = x.
Lemma OT_opposite_sum r f: order r ->
  substrate r = domain f -> order_type_fam f ->
  OT_opposite (OT_sum r f) =
  OT_sum (opp_order r) (Lg (substrate r) (fun z => OT_opposite (Vg f z))).
```

11.28.1 The eta ordering of Cantor

In section 9.3, we introduced the notion of "being like eta", and studied some properties. In particular the interval $]0,1[$ of **Q**, the sets $\mathbf{Q}^+$ and $\mathbf{Q}^+$ are order isomorphic. Cantor defines η as the order type of $]0,1[$. We deduce: r is like eta if and only if its order-type is η.

```
Definition Cantor_eta := order_type_of BQ_int01_ordering.

Lemma Cantor_eta_prop2 r: eta_like r -> order_type_of r = Cantor_eta.
Lemma Cantor_etaP r: order r ->
   (eta_like r <-> order_type_of r = Cantor_eta).
```

Let's show some properties of η. It is not ω, neither $\eta+1$ neither $1+\eta$ (Here ω is the order type of **N**, it has a least element, + is the sum of order types, and 1 is the order type of a singleton).

```
Definition singleton_order := singleton (J emptyset emptyset).
Definition singleton_OT := order_type_of singleton_order.
Notation "\1t" := singleton_OT.

Lemma Cantor_eta_pr1: order_type_of Nat_order <> Cantor_eta.
Lemma singleton_order_or: order singleton_order.
Lemma OT_1_or : order \1t.
Lemma OT_eta_1 : order Cantor_eta.
Lemma singleton_OT_pr: exists x, \1t = singleton (J x x).
Lemma OT_cardinal_1: cardinal (substrate singleton_order) = \1c.

Lemma Cantor_eta_pr2: Cantor_eta +t \1t <> Cantor_eta.
Lemma Cantor_eta_pr3: \1t +t Cantor_eta <> Cantor_eta.
```

The quantities $\eta+\eta$, $\eta+1+\eta$, $\eta\cdot\eta$, η^* are all equal to η.

```
Lemma Cantor_eta_pr4: Cantor_eta  +t Cantor_eta = Cantor_eta.  (* 67 *)
Lemma Cantor_eta_pr5: Cantor_eta  +t \1t +t Cantor_eta = Cantor_eta. (* 122 *)
Lemma Cantor_eta_pr6: Cantor_eta  *t Cantor_eta = Cantor_eta.   (* 50 *)
Lemma Cantor_eta_pr7:  OT_opposite Cantor_eta = Cantor_eta.
```

11.29 The eta ordering of Cantor

Note: this section describes the initial implementation of the theory properties, and is now useless.

11.29.1 A partial implementation of Q

The set of rational integers **Q** is the set of all a/b, where a is a rational integer (an element of **Z**) and b a positive integer. We identify a/b with c/d when $ad=bc$. The sum of the two numbers is $(ad+bc)/(bd)$. One generally assumes $b\in\mathbf{Z}$, but it is sometimes easier to assume $b\in\mathbf{N}^*$. In this case bc is the product of an element of **Z** and an element of **N** considered as an element of **Z**. There is another approach: recall that **Z** is the disjoint union of $\mathbf{Z}^-$, $\{0\}$, and $\mathbf{Z}^+$, where $\mathbf{Z}^+$ is $\mathbf{N}^*$, and there is a bijection $f:\mathbf{Z}^-\to\mathbf{Z}^+$, that associates to each number its opposite. Thus **Q** is the disjoint union of $\mathbf{Q}^-$, {0}, and $\mathbf{Q}^+$, where $\mathbf{Q}^+$ is the quotient of $\mathbf{N}^*$ by $\mathbf{N}^*$, and there is a bijection $f : \mathbf{Q}^- \to \mathbf{Q}^+$, that associates to each number its opposite.

Let A and B be the order types of $\mathbf{Z}^+$ and $\mathbf{Q}^+$. Let's denote by a star the order type of the opposite. Since $x < y$ is equivalent to $-y < -x$ (where $-z$ is the opposite of z in **Z** or **Q**) the order types of $\mathbf{Z}^-$ and $\mathbf{Q}^-$ are A^* and B^*. Moreover, if $x \in \mathbf{Z}^-$ and $y \in \mathbf{Z}^+$, we have $x < 0 < y$ (the same holds for **Q**). We deduce that the order type of **Z** and **Q** are $\mathrm{A}^* + 1 + \mathrm{A}$ and $\mathrm{B}^* + 1 + \mathrm{B}$ respectively (here 1 is the order type of a singleton).

We know that $\mathrm{A} = \omega$ and $1 + \omega = \omega$, so that the order-type of **Z** is $\omega^* + \omega$. Note: there is a unique order-isomorphism $\omega \to \omega$, and the order isomorphisms of **Z** are the functions $x \mapsto x + a$, where a is a constant.

We shall denote by η the order type of the set of elements x in **Q** such that $0 < x < 1$, and show that it is the order type B of $\mathbf{Q}^+$. Thus the order type of **Q** is $\eta^* + 1 + \eta$. We shall show below that: $\eta^* = \eta$, and $\eta + 1 + \eta = \eta$. This shows that the order type of **Q** is η.

Note that the cardinal of the number of order isomorphisms $\mathbf{Q} \to \mathbf{Q}$ is $c = 2^{\aleph_0}$. Proof. Since **Q** is countable, the number of isomorphisms is at most c. Let $f : \mathbf{N} \to \mathbf{N}$ be any function and $g(n)$ the sum of the n first terms, plus n. Then g is strictly increasing, $g(0) = 0$, and $f(n) = g(n+1) - g(n) - 1$. Let h be defined as follows: if $x \le 0$ then $h(x) = x$. If n is an integer, then $h(n) = g(n)$. If $n \le x \le n+1$, then $h(x) = a_n x + b_n$ where $a_n = g(n+1) - g(n)$ and $b_n = g(n) - n a_n$. Note that $h(x) \ge x$, so that h is surjective. Since h is strictly increasing, it is an order isomorphism. Since $f \mapsto h$ is injective, the number of such h is at least c.

In what follows, we shall not introduce **Q**, but only Q, the union of {0} and $\mathbf{Q}^+$, as a quotient **N** by $\mathbf{N}^*$. In one of the exercises we have defined the following objects:

```
Definition Nstar := Nat -s1 \0c.
Definition Qplus1 := Nat \times Nstar.
Definition Qplus_eq_r x y := (P x) *c (Q y) = (P y) *c (Q x).
Definition Qplus1_le_r x y := (P x)*c (Q y) <=c (P y) *c (Q x).
Definition Qplus_eq := graph_on Qplus_eq_r Qplus1.
Definition Qplus := quotient Qplus_eq.
Definition Qplus_or:= graph_on (fun x y => Qplus1_le_r (rep x) (rep y)) Qplus.
```

We have defined the quotient and shown ≤ is a total order on this quotient:

```
Lemma Qplus_equiv: equivalence Qplus_eq.
Lemma Qplus_eq_sr : substrate Qplus_eq = Qplus1.
Lemma Qplus_or_osr: order_on  Qplus_or Qplus.
Lemma Qplus_or_tor: total_order  Qplus_or.
```

Some additional notations and lemmas.

```
Notation "x <=q y" := (gle Qplus_or x y) (at level 60).
Notation "x <q y" := (glt Qplus_or x y) (at level 60).


Lemma Qplus_or_sr: substrate Qplus_or = Qplus.
Lemma Qplus_or_sr_bis x y: x <=q y -> inc x Qplus /\ inc y Qplus.
Lemma Qplus_cc x y: inc x Qplus1 -> inc y Qplus1 ->
  (class Qplus_eq x = class Qplus_eq y  <->
   Qplus_eq_r x y).
Lemma Qplus_cc1 a b c d:
   inc a Nat -> inc b Nstar -> inc c Nat -> inc d Nstar ->
```

```
  (class Qplus_eq (J a b) = class Qplus_eq  (J c d) <->  a *c d = c *c b).
Lemma Qplus_inc3 x: inc x Qplus -> (inc (rep x) x /\ inc (rep x) Qplus1).
Lemma NsS1 : inc \1c Nstar.
```

We show here: if (a,b) an element of $z \in \mathrm{Q}$, then $(ax, bx) \in z$, provided that x is non-zero. This can be rephrased as (a,b) and (ax, bx) are identical in the quotient.

```
Lemma Qplus_stable1 a b x: inc a Nat -> inc b Nats -> inc x Nats ->
   related Qplus_eq (J a b) (J (a *c x) (b *c x)).
Lemma Qplus_stable2 a b x: inc a Nat -> inc b Nats -> inc x Nats ->
  inc (J (a *c x) (b *c x)) (class Qplus_eq (J a b)).
Lemma Qplus_class_stable3 z y x: inc z Qplus -> inc y z -> inc x Nats ->
  inc  (J ((P y) *c x) ((Q y) *c x)) z.
```

Let's show that there is an order-preserving injection $\mathbf{N} \to \mathrm{Q}$. This allow us to consider 0_Q and 1_Q, thus the interval $]0_\mathrm{Q}, 1_\mathrm{Q}[$. Note that (a,b) is in 0_Q if $a = 0$ and is in 1_Q if $a = b$.

```
Definition Nat_to_Qplus n := class Qplus_eq (J n \1c).
Definition Qplus_0 := Nat_to_Qplus \0c.
Definition Qplus_1 := Nat_to_Qplus \1c.
Notation "\0q" :=  Qplus_0.
Notation "\1q" :=  Qplus_1.

Lemma Nat_to_Qplus_aux n: natp n -> inc (J n \1c) Qplus1.
Lemma Nat_to_Qplus_Q n: natp n -> inc (Nat_to_Qplus n) Qplus.
Lemma Nat_to_Qplus_inj n m: natp n -> natp m ->
   (Nat_to_Qplus n) = (Nat_to_Qplus m) -> n = m.
Lemma Nat_to_Qplus_le n m: natp n -> natp m -> n <=c m ->
   (Nat_to_Qplus n) <=q (Nat_to_Qplus m).
Lemma Nat_to_Qplus_lt n m: natp n -> natp m -> n <c m ->
   (Nat_to_Qplus n) <q (Nat_to_Qplus m).

Lemma Qplus_0_pr: \0q =  singleton \0c \times Nstar.
Lemma Qplus_1_pr: \1q = diagonal Nstar.
```

The least element of Q is 0_Q; we have $x < 1_\mathrm{Q}$ when x is the class of (a,b) with $a < b$.

```
Definition Qplus_star :=  Qplus -s1 \0q.

Lemma Qplus_0_least x: inc x Qplus -> \0q <=q x.
Lemma Qplus_0_least1 x:  (\0q <q x) <-> inc x Qplus_star.
Lemma Qplus_le1_aux a b:
  inc a Nat -> inc b Nstar -> a <=c b ->
  (class Qplus_eq (J a b)) <=q \1q.
Lemma Qplus_le1 x:
   (x <=q \1q) <-> (inc x Qplus /\ (P (rep x) <=c (Q (rep x)))).
Lemma Qplus_lt1 x:
    (x <q \1q) <->
   (inc x Qplus /\ (P (rep x) <c (Q (rep x)))).
```

We define here half, double, middle and show $h(x) < x < d(x)$ and $x < m(x,y) < y$.

```
Definition Qplus_half x :=
    class Qplus_eq (J (P (rep x)) ((Q (rep x)) *c \2c)).
```

```
Definition Qplus_double x :=
    class Qplus_eq (J (P (rep x)  *c \2c) (Q (rep x))).
Definition Qplus_middle x y (a:= (P (rep x))) (b:= (Q (rep x)))
   (c:= (P (rep y))) (d:= (Q (rep y))) :=
  class Qplus_eq (J (a *c d +c b *c c) (b *c d *c \2c)).

Lemma Qplus_halfp x (y := Qplus_half x) : inc x Qplus_star ->
  inc y Qplus_star  /\  y <q x.
Lemma Qplus_doublep x (y := (Qplus_double x)): inc x Qplus_star ->
  inc y Qplus_star /\ x <q y.
Lemma Qplus_middlep x y (z := Qplus_middle x y):  x <q y ->
   [/\ inc z Qplus_star, x <q z & z <q y].
```

11.29.2 Definition of eta

Let's say that an ordered set set is like η if it is totally ordered, there is no greatest element, no least element, and no interval $]a,b[$ is empty. Obviously, the set is empty or infinite. We shall assume that it is non-empty and countable, thus has cardinal $\aleph_0$.

```
Definition eta_like0 r  (E:=substrate r) :=
  [/\ total_order r, countable_set E,
     (forall x, inc x E ->  exists y, glt r y x),
     (forall x, inc x E ->  exists y, glt r x y) &
     (forall x y, glt r x y -> exists z, glt r x z /\ glt r z y)].
Definition eta_like r := eta_like0 r  /\ cardinal (substrate r) = aleph0.

Lemma eta_like_pr1 r: eta_like0 r ->
    substrate r = emptyset \/ cardinal (substrate r) = aleph0.
```

We consider now two ordered sets: A is the set of all non-zero elements of Q, and B is the interval $]0_Q, 1_Q[$. These sets are countable and non-empty (note that $1_Q \in A$ and the middle of 0_Q and 1_Q is in B).

```
Definition Qplus01 := interval_oo  Qplus_or Qplus_0 Qplus_1.
Definition Qplus_star_o := induced_order Qplus_or Qplus_star.
Definition Qplus_01_o := induced_order Qplus_or Qplus01.

Lemma Qplus_countable: countable_set Qplus.
Lemma Qplus_star_countable: countable_set Qplus_star.
Lemma Qplus01_countable: countable_set Qplus01.
```

We show here that the two sets A and B are like η. The objective is to show that the two sets are order isomorphic (note: a trivial solution is $x \mapsto 1/(1-x)$, more precisely, if $x \in A$, and $(a,b) \in x$ we consider the class of $(a, b-a)$).

```
Lemma Qplus_star1: inc Qplus_1 Qplus_star.
Lemma Qplus_star_o_sr: substrate  Qplus_star_o =  Qplus_star.
Lemma Qplus_star_eta_like0:  eta_like0 (Qplus_star_o).
Lemma Qplus_star_eta_like: eta_like (Qplus_star_o).

Lemma Qplus_01_sr: substrate  Qplus_01_o =  Qplus01.
Lemma Qplus_01_eta_like0:  eta_like0 (Qplus_01_o).
Lemma Qplus_01_eta_like: eta_like (Qplus_01_o).
```

Let h be a function $\mathrm{I}_n \to \mathbf{N}$, such that $f(i) < f(j)$ if and only if $g(h(i)) < g(h(j))$. We call this $\mathrm{H}(h)$. Let $\psi(i) = g(h(s(i)))$ where $s = s_n$ is the permutation of I_n studied above. Note that H says that ψ is strictly increasing. If h' is the extension of h to I_{n+1} such that $h'(n) = \mathrm{N}(\psi, k)$, then $\mathrm{H}(h')$ holds. Here k is $\mathrm{C}_k(s,n)$, it says how x_n compares to the other x_i.

```
Definition EPpermi_next ph n (s:= EPperm_rec n)
   (h := fun i => Vf g (Vf ph (Vf s i)))
   (k := (EPperm_next_index n s)) :=
  Yo (n = \0c) \0c (EPpermi_fct h n k).
Definition EPpermi_prop ph n :=
  [/\ function ph, source ph = Nint n, sub (target ph) Nat &
     (forall i j, i <c n -> j <c n ->
       (glt r1 (Vf f i) (Vf f j) <->
       glt r2 (Vf g (Vf ph i)) (Vf g (Vf ph j))))].
Lemma EPpermi_next_pr1 ph n
  (k1 := EPpermi_next ph n)
  (ph1 := extension ph n k1):
  natp n -> (EPpermi_prop ph n) ->
  [/\ (Vf ph1 n) = k1, forall i, i <c n -> Vf ph1 i = Vf ph i &
    (EPpermi_prop ph1 (succ n))].
```

Let now η be the order type of Q. If r is η like, its order type is η; the converse holds if f is an ordering, since being η-like is clearly invariant by order isomorphism.

```
Definition Cantor_eta := order_type_of Qplus_star_o.

Lemma Cantor_eta_pr r1 r2:
  eta_like r1 -> eta_like r2 -> r1 \Is r2.
Lemma Cantor_eta_prop r1 r2:
  eta_like  r1 -> eta_like r2 ->
  exists f, order_isomorphism f r1 r2.
Lemma Cantor_eta_prop2 r: eta_like r -> order_type_of r = Cantor_eta.
Lemma Cantor_etaP r: order r ->
  (eta_like r <-> order_type_of r = Cantor_eta).
```

11.30 The cardinals, according to Zermelo, 1908

We implement here a part of the theory of Zermelo as described in his paper "Investigations in the foundations of set theory I" (see [23]). It has seven axioms: Axiom I is the axiom of extent, axiom II corresponds to the axiom of the pair, Axiom III is the axiom of separation, axiom IV is the axiom of the powerset, Axiom V is the axiom of the union, axiom VI is the axiom of choice and axiom VII the axiom of infinity. The bold face numbers are the paragraph numbers of the Zermelo paper.

Note that Zermelo uses $\mathfrak{S}$A for the union of a family of sets, $\mathfrak{D}$A for its intersection and $\mathfrak{U}$T for the powerset of T. He uses A + B and [A, B], for the union or intersection of two sets. There is no mention of ordered pairs, thus no graphs. The product of two or more sets is given by the following definition.

13. "Let T be a set whose elements M, N, R, ... are various (mutually disjoint) sets, and let S_1 be any subset of its union $\mathfrak{S}$T. Then it is definite for every element M of T whether the intersection $[\mathrm{M}, \mathrm{S}_1]$ consists of a single element or not. Thus all those elements of T that have exactly one element in common with S_1 are the elements of a certain subset T_1 of T, and it is again definite whether $T_1 = T$ or not. All subsets S_1 of $\mathfrak{S}T$ that have exactly one element in common with each element of T then are, according to Axiom III, the elements of a set $P = \mathfrak{P}T$, which according to axioms III and IV is a subset of $\mathfrak{U}\mathfrak{S}T$ and will be called the connection set associated with T, or the product of the sets $M, N, R, \ldots$. If $T = \{M, N\}$ we write $\mathfrak{P}T = MN$."

Note the product is independent of the ordering of the factors. An element of a product of n sets has exactly n elements, but if we drop the condition that the sets are disjoint, there may be less than n elements.

```
Definition zprod a := Zo (powerset (union a))
  (fun y => forall x, inc x a -> singletonp (y \cap x)).
Definition zprod2 a b:= zprod (doubleton a b).
```

We have $X \in AB$ if and only if $X \subset A \cup B$ and the two sets $X \cap A$ and $X \cap B$ are singletons.

```
Lemma zprod2_P a b y:
  inc y (zprod2 a b) <->
  [/\ sub y (a \cup b), singletonp (y \cap a) & singletonp (y \cap b)].
```

We denote by $s_X(A)$ the union of $X \cap A$. If $X \cap A = \{t\}$, then $s_X(A) = t$. If $X \in AB$, then $X \cap A = \{s_X(A)\}$ and $X \cap B = \{s_X(B)\}$. From this we deduce that $s_X(A)$ is in A and X, and also that $s_X(B)$ is in B and X.

```
Definition zpr x a := union (x \cap a).
Lemma zprod2_pr1 a b x:
  inc x (zprod2 a b) ->
  ((x \cap a = singleton (zpr x a)) /\
   (x \cap b = singleton (zpr x b))).
Lemma zprod2_pr0 a b x:
  inc x (zprod2 a b) ->
  [/\ inc  (zpr x a) a, inc (zpr x b) b, inc (zpr x a) x & inc (zpr x b) x].

Lemma zprod2_pr0aa a b x:
  inc x (zprod2 a b) -> inc (zpr x a) a.
Lemma zprod2_pr0ax a b x:
  inc x (zprod2 a b) -> inc (zpr x a) x.
Lemma zprod2_pr0bb a b x:
  inc x (zprod2 a b) -> inc (zpr x b) b.
Lemma zprod2_pr0bx a b x:
  inc x (zprod2 a b) -> inc (zpr x b) x.
```

Conversely, an element in A and X must be $s_X(A)$. From this we deduce $X = \{s_X(A), s_X(B)\}$. If $x \in A$ and $x \in X$, then $s_X(A) = x$.

```
Lemma zprod2_pr1a a b x z:
  inc x (zprod2 a b) ->
  inc z a -> inc z x -> z = zpr x a.
Lemma zprod2_pr1b a b x z:
  inc x (zprod2 a b) ->
  inc z b -> inc z x -> z = zpr x b.
Lemma zprod2_pr2 a b x:
  inc x (zprod2 a b) ->  x = doubleton (zpr x a) (zpr x b).
```

We characterize here the product AB when B is a singleton $\{b\}$, as the set of all $\{a, b\}$ for $a \in A$.

```
Lemma intersection_singletonP a b c:
  (a \cap (singleton b) = singleton c) <->
  (inc c a /\ c = b).
Lemma is_singleton_int a b c:
  inc c a -> inc c b -> (forall u, inc u a -> inc u b -> u = c) ->
  singletonp (a \cap b).
Lemma zprod_singleton M r: ~ inc r M ->
  let N := zprod2 M (singleton r) in
    ( (forall u, inc u M -> inc (doubleton u r) N) /\
      (forall x, inc x N -> exists2 u, inc u M & x = doubleton u r)).
```

15. “A *mapping of* M *onto* N is a subset ϕ of the product MN such that each element of $M + N$ occurs as an element in one and only one element $\{m, n\}$ of ϕ. Two elements m and n that occur together in one element of ϕ are said to be ‘mapped onto each other’. Two sets are said to be *immediately equivalent* if there exists at least one such ϕ.”

The definition is symmetric with respect to M and N. The union of these two sets is uniquely determined by ϕ as the union of ϕ. Zermelo assumes the two sets disjoint. In fact, if $M = N$, there is only one mapping, the set of all singletons. We add the disjointness condition to the definition of “equivalent” (but this really changes nothing).

```
Definition zmap f a b := sub f (zprod2 a b) /\
  (forall x, inc x (a \cup b)-> exists !z, inc z f /\ inc x z).
Definition ziequivalent a b := disjoint a b /\ exists f, zmap f a b.

Lemma zmap_symm f a b:
  zmap f a b -> zmap f b a.
Lemma zequiv_symm a b: ziequivalent a b -> ziequivalent b a.
Lemma zmap_pr1 f a b: zmap f a b ->
  union f = union2 a b.
```

Examples: disjoint singletons are equivalent as well as disjoint doubletons.

```
Lemma zmap_example1 a b: a <> b ->
  let A := singleton a in
    let B := singleton b in
    zmap (singleton (doubleton a b)) A B.
Lemma zmap_example2 a b c d: let A := doubleton a b in
  let B := doubleton c d in
    disjoint A B -> a <> b -> c <> d ->
    zmap (doubleton (doubleton a c) (doubleton b d)) A B.
Lemma zequiv_example1 a b: a <> b ->
  ziequivalent (singleton a) (singleton b).
Lemma zequiv_example2  a b c d: let A := doubleton a b in
  let B := doubleton c d in
    disjoint A B -> a <> b -> c <> d ->
    ziequivalent A B.
```

Assume $f(a) \in B$ whenever $a \in A$, where A and B are two disjoint sets. The set of all $z \in A \cup B$ of the form $\{a, f(a)\}$ with $a \in A$ is an element of AB. Assume f bijective; this set is then a mapping.

```
Definition zbijective F a b :=
  [/\ (forall x, inc x a -> inc (F x) b) ,
      (forall x x', inc x a -> inc x' a -> F x = F x' -> x = x') &
      (forall y, inc y b -> exists2 x, inc x a & y = F x)].

Lemma is_singleton_int a b c:
  inc c a -> inc c b -> (forall u, inc u a-> inc u b -> u = c) ->
  is_singleton (a \cap b).
Lemma zmap_example3 F a b: disjoint a b -> zbijective F a b ->
  zequivalent a b.
```

Let's denote by $w_f(x)$ the element such that $w_f(x) \in f$ and $x \in w_f(x)$. If f is a mapping, $x \in \mathrm{A} \cup \mathrm{B}$, such an object exists and is unique. We can restate this as: if x and y are in f, if $s_x(\mathrm{A}) = s_y(\mathrm{A})$ then $x = y$; the same holds for B.

```
Definition zmap_aux f x := select (fun z => inc x z) f.

Lemma zmap_aux_pr1 f a b x:
  zmap f a b -> inc x (a \cup b) ->
  (inc (zmap_aux f x) f /\ inc x (zmap_aux f x)).
Lemma zmap_aux_pr2 f a b x y:
  zmap f a b -> inc x (a \cup b) -> inc y f -> inc x y ->
  y = (zmap_aux f x).
Lemma zmap_aux_pr3a f a b x y:
  zmap f a b -> inc x f -> inc y f ->  zpr x a = zpr y a ->
  x = y.
Lemma zmap_aux_pr3b f a b x y:
  zmap f a b -> inc x f -> inc y f ->  zpr x b = zpr y b ->
  x = y.
```

Consider now $s_\mathrm{A}(w_f(x))$. This is the unique element of A in $w_f(x)$. This is obviously x if $x \in \mathrm{A}$. It is also defined for $x \in \mathrm{B}$. We shall sometimes denote this by $f_\mathrm{A}(x)$. Similarly for $f_\mathrm{B}(x)$. We have $f_\mathrm{A}(f_\mathrm{B}(x)) = x$ if $x \in \mathrm{A}$ and $f_\mathrm{B}(f_\mathrm{A}(x)) = x$ if $x \in \mathrm{B}$. This implies that f_B is bijective (in the sense given above).

```
Definition zmap_val f a x:= zpr (zmap_aux f x) a.

Lemma zmap_val_pr1a  f a b x: zmap f a b ->  inc x (a \cup b) ->
  inc (zmap_val f a x) a.
Lemma zmap_val_pr1b  f a b x: zmap f a b ->  inc x (a \cup b) ->
  inc (zmap_val f b x) b.

Lemma zmap_val_pr2a f a b x: zmap f a b ->  inc x a ->
  (zmap_val f a x) = x.
Lemma zmap_val_pr2b  f a b x: zmap f a b ->  inc x b ->
  (zmap_val f b x) = x.

Lemma zmap_val_pr3a  f a b x: zmap f a b ->  inc x a ->
  (zmap_val f a (zmap_val f b x)) = x.
Lemma zmap_val_pr3b f a b x: zmap f a b ->  inc x b ->
  (zmap_val f b (zmap_val f a x)) = x.

Lemma zmap_bijective  f a b: zmap f a b ->
  zbijective (zmap_val f b) a b.
```

16. Zermelo says: "It is definite for two disjoint sets whether they are equivalent or not", since this is the same checking whether a set Ω is empty or not. We just show here that the set of mappings $A \to B$ exists.

```
Lemma zmap_setP a b: exists s,
   forall f, zmap f a b <-> inc f s.
```

17. If ϕ is a mapping $M \to N$, and M_1 is a subset of M, there exists a subset N_1 of N which is equivalent to M_1 via a subset of ϕ. We first note that if M and N are disjoint, if M_1 is any subset of M and N_1 any subset of N, then M_1 and N_1 are disjoint. Let ϕ_1 be the set of all $X \in \phi$ such that $s_X(M) \in M_1$, then N_1 is the set of all $z \in N$ of the form $z = s_X(N)$ for some $X \in \phi_1$.

```
Lemma sub_disjoint a b a' b':
  disjoint a b -> sub a' a -> sub b' b -> disjoint a' b'.
Lemma zmap_sub f a b a': zmap f a b -> sub a' a ->
  exists f' b', [/\ sub f' f, sub b' b & zmap f' a' b'].
Lemma zequiv_sub a b a': ziequivalent a b -> sub a' a ->
  exists b',  (sub b' b /\ ziequivalent a' b').
```

18. Assume that f maps A to B, and g maps B to C. For instance, we map $\{a, b\}$ to $\{c, d\}$ and then to $\{b, a\}$, where all four elements are distinct. Composition of these mappings yield the permutation on $\{a, b\}$. But this is not a mapping according to Zermelo. Thus, in the following lemma, we assume A and C disjoint. The set of all $z = \{s_x(A), s_y(C)\}$ in $\mathfrak{P}(A \cup C)$ such that there exist $x \in f$ and $y \in g$ such that $s_x(B) = s_y(B)$ is a mapping $A \to C$. We give here a simpler proof: both f_A and g_B are bijective, hence the composition is also bijective. Thus we have a bijection $A \to C$, which gives a mapping, since A and C are disjoint.

```
Lemma zmap_transitive a b c: disjoint a c ->
  zequivalent a b -> zequivalent b c -> zequivalent a c.
```

19. In §10, Zermelo says that for every set M there is a set N such that $N \subset M$ and $N \not\in M$ (consider the set of elements x of M such that $x \not\in x$). This theorem can be used as: for every set M there is a set N such that $N \not\in M$.

Thus, given M and N, there exists r not in M such that, if $R = \{r\}$, the set MR is disjoint from M and N (an element x of MR has the form $\{y, r\}$, if it is in M or N, it is in $M \cup N$ and r is in the union of this set). The function $x \mapsto \{x, r\}$ is bijective, thus we get a mapping $M \to MR$.

It follows that, for any M and N, there exists M' disjoint from M and N, equivalent to M.

```
Lemma disjointness M: exists N,
  sub N M /\ ~ inc N M.
Lemma disjointness1  M N: exists r,
  let M1 := zprod2 M (singleton r) in
   [/\ ~ (inc r M) , disjoint M M1 & disjoint N M1].
Lemma zmap_example4 M r:
  let N := zprod2 M (singleton r) in
    ~ (inc r M) -> disjoint M N ->
    ziequivalent M N.
Lemma zequiv_example4 M N: exists M',
  disjoint N M' /\ ziequivalent M M'.
```

19. Zermelo deduces that there is no set containing all sets equivalent to M, since if T is such a set, we have a set x equivalent to M disjoint from $\mathfrak{S}$T, thus cannot be in T (note that this argument does not hold if x is empty, so that we start with a lemma: if M is empty, so is x).

```
Lemma zequiv_empty M: ziequivalent M emptyset -> M = emptyset.

Lemma zequiv_no_graph M: nonempty M ->
  ~ (exists S, forall M',  ziequivalent M M' -> inc M' S).
```

21. Zermelo says that is "definite" the existence of a set R disjoint form both sets M and N and equivalent to them. This justifies the following definition of "mediately equivalent". This is an equivalence relation.

```
Definition zequiv  M N := exists2 R, ziequivalent M R & ziequivalent N R.

Lemma zequiv_reflexive M: zequiv M M.
Lemma zequiv_symmetric M N:
   zequiv  M N -> zequiv  N M.
Lemma zequiv_transitive M N P:
   zequiv  M N -> zequiv N  P -> zequiv M P.
```

One could now define a cardinal of a set, then the sum and product of two cardinals, and prove the usual properties. This is a bit tedious.

We finish by showing that equivalent is the same as equipotent. In fact `zmap_example3` says that two disjoint equipotent sets are immediately equivalent. Conversely, if f is a mapping from A onto B, then f_B induces a bijection $A \to B$.

```
Lemma zequiv_equipotent M N: zequiv M N <->  M \Eq N.
```

Chapter 12

The size of one

When I was young, I was intrigued by the following quote of Bourbaki:

Bien entendu il ne faut pas confondre le *terme* mathématique *désigné* (chap. I, § 1, n° 1) par le symbole « 1 » et le mot « *un* » du langage ordinaire. Le terme désigné par « 1 » est égal, en vertu de la définition donnée ci-dessus, au terme désigné par le symbole

$$(*)\qquad \begin{aligned} &\tau_Z((\exists u)(\exists U)(u = (U,\{\emptyset\},Z) \text{ and } U \subset \{\emptyset\} \times Z \\ &\text{and } (\forall x)((x \in \{\emptyset\}) \implies (\exists y)((x, y) \in U)) \\ &\text{and } (\forall x)(\forall y)(\forall y')(((x, y) \in U \text{ and } (x, y') \in U) \implies (y = y')) \\ &\text{and } (\forall y)((y \in Z) \implies (\exists x)((x, y) \in U)))). \end{aligned}$$

Une estimation grossière montre que le terme ainsi *désigné* est un assemblage de plusieurs dizaines de milliers de signes (chacun de ces signes étant l'un des signes τ, $\square$, $\vee$, $\neg$, $=$, $\in$, $\supset$).

English translation is ([4, p. 158]): The mathematical *term denoted* (Chapter 1, § 1, no. 1) by the symbol "1" is of course not to be confused with the *word* "one" in ordinary language. The term denoted by "1" is equal, by virtue of the definition above, to the term denoted by the symbol (*). As a rough estimate, the term so *denoted* is an assembly of several tens of thousands of signs (each of which is one of τ, $\square$, $\vee$, $\neg$, $=$, $\in$, $\supset$).

The same expression appears in the English version and in the French one (except that "and" is replaced by "et" in French). By definition, 1 is $\mathrm{Card}(\{\emptyset\}) = \tau_Z(\mathrm{Eq}(\{\emptyset\},Z))$.

If 1 is a big object, then how big is 2? or the set N of integers, or the set R of real numbers? If $P(X, n)$ is: $X = (x, y, z)$ and $x^n + y^n = z^n$ and $x \neq 0$ and $y \neq 0$ and $z \neq 0$, what is the size of the formula $(\forall n)(n \in N \implies (\exists X)(X \in R^3 \text{ and } P(X, n)))$? If this is an assembly with millions of signs, how big will be a proof of it? If billions of signs are needed, how can one check the proof? I am convinced that it is a bad idea to try to reduce the object under consideration to its basic components (a kind of normal form) and apply low-level theorems to it; Fermat's theorem cannot be proved by exhibiting an assembly that is a proof in the sense of Bourbaki. On the other hand, the estimation of Bourbaki shows that it should be possible to print the whole assembly denoting 1 on an A0-size poster. Alas, the estimation is wrong.

12.1 Comments

The assembly (*) is obviously not 1 but is *equal* to 1. For simplicity, we shall write Y instead of $\{\emptyset\}$. Introduce the variant

$$(**)\qquad \begin{aligned}\tau_Z((\exists u)(\exists U)(\quad & u=(U,Y,Z)\\ & \text{and } U\subset Y\times Z\\ & \text{and } (\forall x)((x\in Y)\implies(\exists y)((x,y)\in U))\\ & \text{and } (\forall x)(\forall y)(\forall y')(((x,y)\in U \text{ and } (x,y')\in U)\implies(y=y'))\\ & \text{and } (\forall y)((y\in Z)\implies(\exists x)((x,y)\in U))\\ & \text{and } (\forall x)(\forall x')(\forall y)(((x,y)\in U \text{ and } (x',y)\in U)\implies(x=x')))).\end{aligned}$$

Define a triple t to be an object of the form (x,y,z). Denote by $\mathrm{pr}'_1 t$, $\mathrm{pr}'_2 t$ and $\mathrm{pr}'_3 t$ the three components. Let $C(G,A,B)$ be a statement specified below. There is an ambiguity in the definition of a bijection as explained below. First interpretation: f is a bijection of X onto Y if there exists G such that $f=(G,X,Y)$ and $C(G,X,Y)$. In this interpretation 1 is $\tau_Z(W^3_Z)$ where W^3_Z is $(\exists u)(\exists U)(W^3)$ and W^3 is $u=(U,Y,Z)$ and $C(U,Y,Z)$. Second interpretation: f is a triple such that $C(\mathrm{pr}'_1 f,\mathrm{pr}'_2 f,\mathrm{pr}'_3 f)$, $\mathrm{pr}'_2 f=X$, and $\mathrm{pr}'_3 f=Y$ hold. In this interpretation 1 is $\tau_Z(W^4_Z)$ where W^4_Z is $(\exists f)(W^4)$, for some W^4. Note that (**) has the form $\tau_Z(W^2_Z)$ where W^2_Z is $(\exists u)(\exists U)(W^2)$ and where W^2 is of the form "P1 and P2 and P3 and P4 and P5 and P6". Finally note that (*) has the form $\tau_Z(W^1_Z)$ where W^1_Z is $(\exists u)(\exists U)(W^1)$ and where W^1 is like W^2 without the P6.

We pretend that for every Z, the relations W^1_Z, W^2_Z, W^3_Z, and W^4_Z are equivalent. First, it is obvious that W^3 and W^4 are equivalent. Second, and this will be explained below, W^2 is equivalent to W^3. It says that U is the graph of a bijection of Y onto Z; hence W^1 says that U is the graph of a surjection of Y onto Z. Since Y is a set with one element, every function with source Y is injective, hence W^1_Z and W^2_Z are equivalent. Now Axiom Scheme S7 says that the three sets $\tau_Z(W^1_Z)$, $\tau_Z(W^2_Z)$, $\tau_Z(W^3_Z)$ and $\tau_Z(W^4_Z)$ are equal.

We believe that Bourbaki made a mistake and meant (**) instead of (*). We also believe that Bourbaki simplified a bit the definition W^3 of 1 as W^2, assuming that the simplified definition would have approximatively the same size, see below.

Definition 2 of Chapter II §3 no. 1 says: *A correspondence between a set* A *and a set* B *is a triple* $\Gamma=(G,A,B)$ *where* G *is a graph such that* $\mathrm{pr}_1 G\subset A$ *and* $\mathrm{pr}_2 G\subset B$. G *is said to be the graph of* Γ, A *is the source and* B *is the target of* Γ.

Later on, Bourbaki says: *Let* A *and* B *be two sets; a mapping of* A *into* B *is a function* f *whose source is equal to* A *and whose target is equal to* B. Instead of "let f be a mapping of A into B" one can say: "let $f: A\to B$ be a mapping". We shall consider two interpretations. In the first interpretation, correspondence, mapping, bijection, etc., are objects that depend on two parameters A and B. In the case of a Coq function of type $A\to B$, the parameters can be deduced from the type; in the case of Bourbaki $A=\mathrm{pr}'_2 f$ and $B=\mathrm{pr}'_3 f$. There is however an important difference: in order to define in Coq the composition $g\circ f$ of a function $f: A\to B$ and a function $g: B'\to C$ it is necessary that B and B′ be *convertible*. The Bourbaki definition is independent of B and B′; in the case where $B=B'$ we obtain a mapping $A\to C$ and such that $(g\circ f)(x)=g(f(x))$ whenever $x\in A$. There is a second interpretation, that is used for the implementation of Bourbaki in Coq: one says that f is a correspondence, function, bijection, etc; independently of the context. The composition $g\circ f$ is defined whatever g and f, for instance $\emptyset\circ\emptyset$ is defined. This is coherent with the fact that the theory of sets is an *untyped* theory.

Let's try to formalize definition 2. Denote by $c(G,A,B)$ the statement "G is a graph such that $\mathrm{pr}_1 \mathrm{G} \subset \mathrm{A}$ and $\mathrm{pr}_2 \mathrm{G} \subset \mathrm{B}$", so that a correspondence is a triple that satisfies c. Consider a context where A, B and G are sets; it makes sense to define a "correspondence between A and B with graph G" as a triple $\Gamma = (\mathrm{G}, \mathrm{A}, \mathrm{B})$ satisfying c, then define the graph, source and target of the correspondence to be G, A and B. This not a good to interpretation of Definition 2 (it would make composition of functions, and statements about it, a nightmare). The only reasonable interpretation is: Γ is a correspondence if there exists G such that $\Gamma = (\mathrm{G}, \mathrm{A}, \mathrm{B})$ satisfying $c(\mathrm{G}, \mathrm{A}, \mathrm{B})$. Write this as there exists G such that $c'(\Gamma, \mathrm{G}, \mathrm{A}, \mathrm{B})$. Now "the graph" of Γ becomes undefined. However, if Γ is a correspondence, then $c'(\Gamma, \mathrm{pr}'_1\Gamma, \mathrm{A}, \mathrm{B})$ holds; and from $c'(\Gamma, \mathrm{G}, \mathrm{A}, \mathrm{B})$ one deduces $\mathrm{G} = \mathrm{pr}'_1\Gamma$. For this reason, we define the graph as $\mathrm{pr}'_1\Gamma$. We also define the source and target as $\mathrm{pr}'_2\Gamma$ and $\mathrm{pr}'_3\Gamma$. This is not exactly the definition of Bourbaki, but $\mathrm{A} = \mathrm{pr}'_2\Gamma$ and $\mathrm{B} = \mathrm{pr}'_3\Gamma$ hold. Write $c''(\Gamma)$ for $c(\mathrm{pr}'_1\Gamma, \mathrm{pr}'_2\Gamma, \mathrm{pr}'_2\Gamma)$. Now Γ is a correspondence if there exists G such that $\Gamma = (\mathrm{G}, \mathrm{A}, \mathrm{B})$ and $c''(\Gamma)$. Here c'' is in the scope of the exists G but we can move it out of the scope. Now "there exists G such that $\Gamma = (\mathrm{G}, \mathrm{A}, \mathrm{B})$" is equivalent to: Γ is a triple and $\mathrm{A} = \mathrm{pr}'_2\Gamma$ and $\mathrm{B} = \mathrm{pr}'_3\Gamma$ (see below how we interpret "is a triple"). Let $c'''(\Gamma)$ be: Γ is a triple and $c''(\Gamma)$. So, an equivalent definition of a correspondence between A and B is: $c'''(\Gamma)$ and $\mathrm{A} = \mathrm{pr}'_2\Gamma$ and $\mathrm{B} = \mathrm{pr}'_3\Gamma$. This is our second interpretation; moreover $c'''(\Gamma)$ means Γ is a correspondence

Let's go down to the details. Given two sets x and y one can define an ordered pair (x, y), see comments below. If z is pair, there are two sets x and y such that $z = (x, y)$; these are unique, and denoted by $\mathrm{pr}_1 z$ or $\mathrm{pr}_2 z$. The notations stand for $\tau_x((\exists y)(z = (x, y)))$ and $\tau_y((\exists x)(z = (x, y)))$. A triple $t = (x, y, z)$ is a pair of pairs $((x, y), z)$ so that $\mathrm{pr}'_1 t = \mathrm{pr}_1\mathrm{pr}_1 t$, $\mathrm{pr}'_2 t = \mathrm{pr}_2\mathrm{pr}_1 t$ and $\mathrm{pr}'_3 t = \mathrm{pr}_2 t$. By abuse of notations, $\mathrm{pr}_1 \mathrm{G}$ and $\mathrm{pr}_2 \mathrm{G}$ are the sets of all $\mathrm{pr}_1 z$ or $\mathrm{pr}_2 z$ for z in G, hence

$$\tau_z((\forall x)((x \in z) \iff ((\exists y)((x, y) \in \mathrm{G})))) \text{ and } \tau_z((\forall y)((y \in z) \iff ((\exists x)((x, y) \in \mathrm{G}))).$$

With the same notations as above, let Γ be a correspondence and $z \in \mathrm{G}$. Since G is a graph, z is a pair, say $z = (x, y)$. Obviously $x \in \mathrm{pr}_1 \mathrm{G}$ hence $x \in \mathrm{A}$. Similarly $y \in \mathrm{B}$ so that $z \in \mathrm{A} \times \mathrm{B}$, and $\mathrm{G} \subset \mathrm{A} \times \mathrm{B}$. Conversely, if $\mathrm{G} \subset \mathrm{A} \times \mathrm{B}$ then G is the graph of a correspondence.

So we claim: the relations "P1 and P2" are equivalent to "u is a correspondence between Y and Z whose graph is U". This is the first simplification made by Bourbaki.

Definition 9 of §3 no. 4 says: *A graph* F *is said to be a functional graph if for each x there is at most one object which corresponds to x under* F *(Chapter I, §5 no. 3)*. In other terms, for every x, there is at most one y such that $(x, y) \in \mathrm{F}$. *A correspondence $f = (\mathrm{F}, \mathrm{A}, \mathrm{B})$ is said to be a function if its graph* F *is a functional graph and if its source* A *is equal to its domain* $\mathrm{pr}_1 \mathrm{F}$. Relation P4 says that U is functional, and relation P3 says $\mathrm{Y} \subset \mathrm{pr}_1 \mathrm{U}$. By P2 we have $\mathrm{pr}_1 \mathrm{U} \subset \mathrm{Y}$, so that P3 becomes equivalent to $\mathrm{Y} = \mathrm{pr}_1 \mathrm{U}$. Hence "P1 and P2 and P3 and P4" are equivalent to "u is a correspondence between Y and Z; it is a function, and its graph is U". This is the second simplification made by Bourbaki.

There are two interpretations of Definition 9. Interpretation 1: in the context where A and B are sets, f is a function if there exists a set (called G in Definition 2 and F in Definition 9), satisfying the properties indicated in these two definitions. Second interpretation: f is a function if it is a correspondence and an additional property holds. So f is a function if it is a triple and $(\forall z)((z \in \mathrm{pr}'_1 f) \implies ((\exists x)(\exists y)(z = (x, y))))$ and $(\forall x)(\forall y)(\forall y')(((x, y) \in \mathrm{pr}'_1 f \text{ and } (x, y') \in \mathrm{pr}'_1 f) \implies (y = y'))$ and $\mathrm{pr}'_2 f = \mathrm{pr}_1\mathrm{pr}'_1 f$ and $\mathrm{pr}_1\mathrm{pr}'_1 f \subset \mathrm{pr}'_2 f$ and $\mathrm{pr}_2\mathrm{pr}'_1 f \subset \mathrm{pr}'_3 f$.

Definition 9 continues as follows *In other terms, a correspondence $f = (\mathrm{F}, \mathrm{A}, \mathrm{B})$ is a function if for every x belonging to the source* A *of f, the relation $(x, y) \in \mathrm{F}$ is functional in y (Chapter I, §5 no. 3)*. Assume that f is a function and $x \in \mathrm{A}$. Since $\mathrm{A} = \mathrm{pr}_1 \mathrm{F}$, we get $x \in \mathrm{pr}_1 \mathrm{F}$, so there exists y such that $(x, y) \in \mathrm{F}$. Since F is functional, this y is unique, so that the relation $(x, y) \in \mathrm{F}$ is functional in y. The converse holds. First $\mathrm{A} \subset \mathrm{pr}_1\mathrm{F}$ since whenever $x \in \mathrm{A}$, there exists y such that $(x, y) \in \mathrm{F}$. As mentioned above this implies $\mathrm{A} = \mathrm{pr}_1\mathrm{F}$. Assume now $(x, y) \in \mathrm{F}$ and $(x, y') \in \mathrm{F}$. This implies $x \in \mathrm{pr}_1\mathrm{F}$, hence $x \in \mathrm{A}$. Since the relation is functional, we get $y = y'$. This says that F is a functional graph. So the claim of Bourbaki is correct but non-trivial. *the unique object which corresponds under f to x is called the value of f at the element x of* A*, and is denoted by $f(x)$ (or f_x or* $\mathrm{F}(x)$ *or* F_x*).* Obviously the first f has to be replaced by F, so that $f(x)$ becomes a notation for $\tau_y((x, y) \in \mathrm{F})$. According to Chapter I, §5 no. 3, since the relation is functional we have: whenever $x \in \mathrm{A}$, $(x, y) \in \mathrm{F} \iff y = f(x)$.

This last statement is expressed by Bourbaki as: *if f is a function,* F *its graph, and x an element of the domain of f, the relation $y = f(x)$ is then equivalent to* $(x, y) \in \mathrm{F}$. Note that the domain of f is $\mathrm{pr}_1\mathrm{F}$.

Definition 10 of §3 no. 6 says: *Let f be a mapping of* A *into* B. So f is a function whose source is equal to A and whose target is equal to B. Let F be its graph. *The mapping f is said to be injective or an injection, if any two distinct elements of* A *have distinct images under f.* If $p_i(\mathrm{F}, \mathrm{A})$ is short for $(\forall x)(\forall x')((x \in \mathrm{A} \text{ and } x' \in \mathrm{A} \text{ and } x \neq x') \implies f(x) \neq f(x'))$, then f is injective when $p_i(\mathrm{F}, \mathrm{A})$ holds (as mentioned above $f(x)$ depends only on x and the graph F). This is obviously equivalent to: if x and x' are in A then $f(x) = f(x')$ implies $x = x'$. Assume the function injective, $(x, y) \in \mathrm{F}$, $(x', y) \in \mathrm{F}$. As noticed above, this implies $x \in \mathrm{A}$ and $x' \in \mathrm{A}$; moreover $y = f(x)$ and $y = f(x')$. So $f(x) = f(x')$ hence $x = x'$. So we get P6. Conversely assume P6, $x \in \mathrm{A}$, $x' \in \mathrm{A}$, and $f(x) = f(x')$. The first two relations yield $(x, f(x)) \in \mathrm{F}$, $(x', f(x')) \in \mathrm{F}$, and the third gives $(x', f(x)) \in \mathrm{F}$, so that P6 says $x = x'$. Hence P6 says f is injective. *The mapping f is said to be surjective, or a surjection, if f*(A) = B. Here $f(\mathrm{A})$ is an abuse of notations for the set of all $f(x)$ for $x \in \mathrm{A}$. Let $p_s(\mathrm{F}, \mathrm{A}, \mathrm{B})$ stand for $\tau_z((\forall y)((y \in z) \iff (\exists x)(x \in \mathrm{A} \text{ and } y = f(x))))) = \mathrm{B}$. This is $f(\mathrm{A}) = \mathrm{B}$. Note that $f(\mathrm{A}) \subset \mathrm{B}$ because $f(x) \in \mathrm{pr}_2\mathrm{F} \subset \mathrm{B}$. So surjectivity becomes equivalent to: whenever $y \in \mathrm{B}$, there is $x \in \mathrm{A}$ such that $y = f(x)$; this can be restated as $(x, y) \in \mathrm{F}$. We get P5. Conversely, $(x, y) \in \mathrm{F}$ says $x \in \mathrm{A}$ and $y = f(x)$. Hence P5 is equivalent to surjectivity of f. *If f is both injective and surjective, it is said to be bijective, or a bijection.*

Definition 1 of Chapter III §3 no. 1 says: *A set* X *is said to be equipotent to a set* Y*, if there exists a bijection of* X *onto* Y. This means: there is a mapping f of X into Y such that f is injective and surjective. This can be interpreted in two ways.

Interpretation one. We are in a context where the source is Y (in reality $\{\emptyset\}$) and the target is Z. So W^3_Z is $(\exists u)$ such that u is a mapping (i.e., there is G such that $u = (\mathrm{G}, \mathrm{Y}, \mathrm{Z})$ and Pc and Pf) satisfying Pi and Ps. These two conditions are $p_i(\mathrm{G}, \mathrm{Y})$ and $p_s(\mathrm{G}, \mathrm{Y}, \mathrm{Z})$, they depend on G so the interpretation is: there is G such that $u = (\mathrm{G}, \mathrm{Y}, \mathrm{Z})$ and Pc and Pf and Pi and Ps. Interpretation two (context independent): there exists f, f is an injection, f is a surjection; the source is Y, the target is Z. Hence W^4_Z is: $(\exists f)$ such that f is a function and Pi and f is a function and Ps and $\mathrm{pr}'_2 f = \mathrm{Y}$ and $\mathrm{pr}'_3 f = \mathrm{Z}$. Here Pi and Ps are p_i and p_s evaluated at $\mathrm{pr}'_1 f$, $\mathrm{pr}'_2 f$ and $\mathrm{pr}'_3 f$.

12.2 Computations

In 2002, Mathias published a paper, [16], where he explains that 1 is huge (it has $4 \cdot 10^{12}$ symbols; not counting the number of links, which amounts to 10^{12}). These numbers disagree with our computations; in 2017, we have redone our computations, and corrected some mistakes. Our computations give the number of τ, $\square$, $\vee$, $\neg$, $=$, $\in$ and $\supset$. Since a link connects a τ and a $\square$ and since each $\square$ has exactly one link, we deduce that the number of links is the number of $\square$.

In the 1970 French edition [5], the symbol $\supset$ has been withdrawn, and a pair is no more a primitive. We shall denote by "original" the term that contains the funny symbol, and "modified" the term that does not contain it. We denote by $\mathrm{L}'[x]$ the size of the original x and by $\mathrm{L}[x]$ the size of the modified x. If these quantities coincide, we give only the values of L; moreover, in the case of the "real one", we give only L.

The empty set. The empty set $\emptyset$ is defined as $\tau_z((\forall x)(x \not\in z))$, this corresponds to the following assembly

$$\tau\neg\neg\neg\in\tau\neg\neg\in\square\square\square$$

where each $\square$ is linked to a τ; without the links the formula has four different interpretations (the final $\square$ is not in the scope of the second τ, hence is linked to the first τ). Let x_1, x_2, x_3 and x_4 be the four interpretations. It is very difficult to understand the meaning of these objects. First, let's write them as $\tau_z(\neg\neg\neg y \in z)$, where y stands for $\tau\neg\neg\square\in\square$, it has four meanings y_1 which is $\tau_a(\neg\neg a \in a)$, y_2 which is $\tau_a(\neg\neg a \in z)$, y_3 which is $\tau_a(\neg\neg z \in a)$, and y_4 which is $\tau_a(\neg\neg z \in z)$. Note that y_1 does not depend on z, other terms depend on it, and $\emptyset$ is x_2. Axiom scheme S7 says that if for every z, R is equivalent to S then $\tau_z\mathrm{R} = \tau_z\mathrm{S}$. For instance $x_1 = \tau_z(\neg y_1 \in z)$. Denote this quantity by x'_1. Axiom scheme S6 says that, if $u = v$ for every relation S, $\mathrm{S}(u)$ is equivalent to $\mathrm{S}(v)$. Let $\mathrm{E}(z)$ be the relation $(\forall x)(x \not\in z)$; this is interpreted as z is empty. So by S6 x_1 is empty whenever x'_1 is empty. Let now y'_1 be $\tau_a(a \in a)$. By S7 we have $y_1 = y'_1$, and by S6 and S7 $x_1 = x''_1$, where x''_1 is like x_1, with y_1 replaced by y'_1. So x_1 is empty whenever x''_1 is empty. (Here "is empty" can be replaced by any predicate and 1 by any other index).

If we want to go further, we must use Axiom Scheme S5 (the only scheme that uses τ). It says: if x is a letter, T a term, R a relation, if R holds when x is replaced by T, then $(\exists x)\mathrm{R}$ is true. The only case where the inner τ corresponds to a $\exists$ is x_2, and the axiom scheme does not apply. Take for R the relation $y \not\in z$. By S7, $x = \tau_z\mathrm{R}$. Let T be an empty set. Now R(T) holds since, whatever the value of y, y is not in T. It follows that $(\exists z)\mathrm{R}$ is true. By definition of $\exists$, this is $\mathrm{R}(x)$, i.e., $y \not\in x$. The problem is that y is some set that depends on x (except for y_1). In the special case of x_2, the relation R is equivalent to $\mathrm{E}(z)$. It follows $\mathrm{E}(\emptyset)$. So $\emptyset$ is empty.

Note. Bourbaki introduces $\emptyset$ in Chapter II, §1 no. 7. The axiom of extent says that if T is any empty set, then $\mathrm{T} = \emptyset$. Axiom A2 asserts the existence of the doubleton $\{x, y\}$. Axiom Scheme S8 (scheme of selection and union) together with A2 says that for any set E, any property P, there is a set T containing all x of E that satisfy P. Take $\mathrm{E} = \emptyset$, P the property $x \not\in \emptyset$. Now $x \in \mathrm{T}$ leads to a contradiction, so that T is empty. It follows that there exista an empty set. This concludes the proof that $\emptyset$ is empty.

Preliminary computations. Let P be some formula that contains α occurrences of z and β other signs. Remember that $(\exists z)\mathrm{P}$ is $(\tau_z\mathrm{P}|z)\mathrm{P}$, this is the formula obtained by replacing every z by $\tau_z\mathrm{P}$. Thus we have

$$\mathrm{L}[(\exists z)\mathrm{P})] = (\alpha+1)(\alpha+\beta).$$

Since $(\forall z)\mathrm{P}$ is $\neg(\exists z)(\neg\mathrm{P})$, we get

$$\mathrm{L}[(\forall z)\mathrm{P}] = 1+(\alpha+1)(\alpha+\beta+1).$$

The set of all x such that P is $\tau_y((\forall z)(z \in y) \iff \mathrm{P})$ hence

$$\mathrm{L}[\{z, \mathrm{P}\}] = 4\alpha^2+36\alpha+47+(4\alpha+6)\beta.$$

Since $\{x, y\}$ is the set of all z such that $z = x$ or $z = y$ we get

$$L[\{x, y\}] = 177 + 14x + 14y.$$

Example: the empty set $\emptyset$ is defined by $\tau_x(\forall z)P$ where P is $z \not\in x$. In this case $\alpha = 1$ and $\beta = 3$ so that $(\forall z)P$ has length 11, and $\emptyset$ has length 12. We also deduce

$$L[\{\emptyset\}] = 513.$$

The algorithm. The previous formula is different from that of [16], because Mathias considers that $\{x\}$ is the term $\tau_y \forall z(z \in y \iff z = x)$ *slightly simplified from the actual definition as* $\{x, x\}$. He obtains an assembly of size 217 with 56 links. Since there are 25 τ, this means that, on average each τ has slightly more than two links. In fact, the formula has ten τ_1, ten τ_2, four τ_4 and one τ_{10}, where the index is the number of links. In the case of the unsimplified $\{\emptyset\}$, we have 28 τ_1, 28 τ_2, six τ_6 and a τ_{14}.

Assume that 1 has the form $\tau_Z((\exists f)\,F(f,Z))$, where F is the formula that says that f is a bijection of $\{\emptyset\}$ onto Z. Assume that this formula has size $\alpha f + \beta + Z$. By the previous remarks, 1 has size $1 + (\alpha+1)(\alpha+\beta+1)$. We can be more precise. By definition of $\exists$, each f has to be replaced by an assembly, whose size is $1 + \alpha + \beta + F$. This assembly starts with a τ that has α links. After substitution, we obtain α occurrence of τ_α and the number of Z is increased by one. This means that the external τ has $\alpha + 1$ links. In the case of (*), (**) and W^3 the situation is a bit different: 1 has the form $\tau_Z((\exists u)(\exists U)\,W)$; assume that W has size $\alpha + \beta U + \gamma Z + u$. In this case 1 has size $1 + (\beta+1)(\beta+2)(\alpha+\beta+\gamma+1)$; the outer τ has $\gamma(2+3\beta+\beta^2)$ links, there are $\beta + 1$ occurrence of $\tau_{\beta+1}$ for the $\exists u$, and $(\beta+2)\beta$ occurrences of τ_β for the $\exists U$.

We represent the size of an object x as a sum $c_1A + c_2B + c_3C + c_4D + c_5E + c_6F + c_7G + y$ where the capital letters are symbols that stand for τ, $\square$, etc, the c_i are integers, and y counts for the variables in x. For instance the size of $x \iff x'$ is $2x + 2x' + 3C + 5D$, and the size of $\emptyset$ is $2A + 3B + 5D + 2F$. The size of $\{x, y\}$ is $7A + 50B + 35C + 43D + 28E + 14F + 14x + 14y$. We get the formula $L[\{x, y\}] = 177 + 14x + 14y$ shown above by substituting A, B, C, etc, by 1.

Let e be a formula of size s, and x a variable. The size s' of $\tau_x e$ is $A + S(B)$ where $S(t)$ is the quantity obtained by replacing x by t in S. The size of $(\exists x)e$ is $S(s')$. We mentioned above that the τ in $\{\emptyset\}$ have 1, 2, 6 or 14 links; this can be checked by hand, but is obtained by computation. Instead of c_1A we have a sum $\sum a_i A^{b_i}$ where a_i is the number of τ with b_i links. The number of τ is $\sum a_i$, this replaces c_1. To say that the number of links is the number of $\square$ is the following equality $c_2 = \sum a_i b_i$, this is an invariant of our algorithm. The actual formula for the size of $\tau_x e$ is $A^n + S(B)$, where $n = S(1) - S(0)$. In fact, assume that e is a formula that contains α occurrences of x, and β other terms. Then $s = \alpha x + \beta$ and $S(t) = \alpha t + \beta$; so $S(1) = \alpha + \beta$, $S(0) = \beta$ and $n = \alpha$. Replacing A by 1 replaces A^n by 1, whatever n. So the actual formula for $\{x, y\}$ has $6A^6 + A^{14}$ rather than 7A. There is a further refinement: the actual size is $A^{n+l} + S(B)$ where l is a label. Choosing a different label allows us to say: the τ associated to the $\forall y$ of P4 has 6 links and occurs 3406199076 times in (**).

Size of a triple. By definition a triple (a, b, c) is a couple $((a, b), c)$. Hence

$$L'[u = (U, Y, Z)] = u + U + Z + 516.$$

In the modified version, a pair (x, y) is defined by $\{\{x\}, \{x, y\}\}$. We have

$$L[(x, y)] = 5133 + 588x + 196y.$$

$$\mathrm{L}[((x,y),z)] = 3023337 + 345744x + 115248y + 196z$$

$$\mathrm{L}[u = (\mathrm{U},\mathrm{Y},\mathrm{Z})] = 62\,145\,562 + 345\,744\mathrm{U} + 196\mathrm{Z} + u$$

We deduce from this that the modified size of one is at least 10^{18}, much larger than the "several tens of thousands" of Bourbaki. In what follows, we compute the two sizes in parallel. Since Mathias uses a simplified version of a singleton, he also uses a simplified version of a pair.

Size of a graph. We try to find the size of the expression P2, namely $\mathrm{U} \subset \mathrm{Y} \times \mathrm{Z}$. If B is $\mathrm{Y} \times \mathrm{Z}$, this expression is $(\forall z)(z \in \mathrm{A} \implies z \in \mathrm{B})$, its size is $22 + 3\mathrm{A} + 3\mathrm{B}$. The size of "$z = (x,y)$ and $x \in \mathrm{Y}$ and $y \in \mathrm{Z}$" is $z+2x+2y+\mathrm{Y}+\mathrm{Z}+12$. If this is P7, then $\mathrm{Y}\times\mathrm{Z}$ is the set of all z so that there exists x such that there exists y such that P7. Quantifying $\exists y$ gives $42+6x+3\mathrm{Y}+3\mathrm{Z}+3z$, quantifying $\exists x$ gives $336 + 21(\mathrm{Y} + \mathrm{Z} + z)$. Taking the set of all z gives $32807 + 1890(\mathrm{Y} + \mathrm{Z})$.

$$\mathrm{L}'[\mathrm{U} \subset \mathrm{Y} \times \mathrm{Z}] = 98443 + 3\mathrm{U} + 5670(\mathrm{Y} + \mathrm{Z});$$

$$\mathrm{L}'[\mathrm{P2}] = 3\,007\,153 + 3\mathrm{U} + 5670\mathrm{Z}.$$

For the modified version, the length of P7 is

$$589x + 5144 + 197y + z + \mathrm{Y} + \mathrm{Z}.$$

Quantifying over y gives

$$1057518 + 198\mathrm{Y} + 198\mathrm{Z} + 116622x + 198z.$$

Quantifying over x gives

$$136931729220 + 23091354(\mathrm{Y} + \mathrm{Z} + z).$$

Taking the set of all z gives

$$\mathrm{L}[\mathrm{Y} \times \mathrm{Z}] = 12649889797944532895 + 2132842656761388(\mathrm{Y} + \mathrm{Z});$$

$$\mathrm{L}[\mathrm{U} \subset \mathrm{Y} \times \mathrm{Z}] = 37949669393833598707 + 3\mathrm{U} + 6398527970284164(\mathrm{Y} + \mathrm{Z});$$

$$\mathrm{L}[\mathrm{P2}] = 41232114242589374839 + 3\mathrm{U} + 6398527970284164\mathrm{Z}.$$

This number is so big that it is impossible to put in a computer the full expansion of P2.

Size of a bijection. Let's try to find the size of the expressions P3, P4, P5 and P6. They are similar. We start with $(x,y) \in \mathrm{U}$, the size is

$$5134 + 588x + 196y + \mathrm{U} \text{ or } 2 + x + y + \mathrm{U}.$$

The size is $(\exists y)((x,y) \in \mathrm{U})$ is

$$1050010 + 197\mathrm{U} + 115836x \text{ or } 6 + 2\mathrm{U} + 2x.$$

The size of $x \in \mathrm{Y} \implies (\exists y)((x,y) \in \mathrm{U})$ is

$$1050013 + 197\mathrm{U} + 115837x + \mathrm{Y} \text{ or } 9 + 2\mathrm{U} + 3x + \mathrm{Y}.$$

The size of $(\forall x)(x \in \mathrm{Y} \implies (\exists y)((x,y) \in \mathrm{U}))$ is

$$135049848139 + 22820086\mathrm{U} + 115838\mathrm{Y} \text{ or } 53 + 8\mathrm{U} + 4\mathrm{Y}.$$

The size of P3 is

$$135109273033 + 22820086\mathrm{U} \text{ or } 2105 + 8\mathrm{U}.$$

The size of $(x, y) \in \mathrm{U}$ and $(x, y') \in \mathrm{U}$ is

$$10272 + 1176x + 196y + 196y' + 2\mathrm{U} \text{ or } 8 + 2x + y + y' + 2\mathrm{U}.$$

The size of $(x, y) \in \mathrm{U}$ and $(x, y') \in \mathrm{U} \implies y = y'$ is

$$10275 + 1176x + 197y + 197y' + 2\mathrm{U} \text{ or } 11 + 2x + 2y + 2y' + 2\mathrm{U}.$$

The size of $(\forall y')((x, y) \in \mathrm{U}$ and $(x, y') \in \mathrm{U} \implies y = y')$ is

$$2073655 + 396\mathrm{U} + 232848x + 39006y \text{ or } 43 + 6\mathrm{U} + 6x + 6y.$$

The size of $(\forall y)(\forall y')((x, y) \in \mathrm{U}$ and $(x, y') \in \mathrm{U} \implies y = y')$ is

$$82408606635 + 15446772\mathrm{U} + 9082701936x \text{ or } 351 + 42\mathrm{U} + 42x.$$

Finally the size of P4 is

$$830988285585569103965 + 140298425964797364\mathrm{U} \text{ or } 16943 + 1806\mathrm{U}.$$

The size of $(\exists x)((x, y) \in \mathrm{U})$ is

$$3370258 + 589\mathrm{U} + 115444y \text{ or } 6 + 2\mathrm{U} + 2y.$$

The size of $y \in \mathrm{Z} \implies (\exists x)((x, y) \in \mathrm{U})$ is

$$3370261 + 589\mathrm{U} + 115445y + \mathrm{Z} \text{ or } 9 + 2\mathrm{U} + 3y + \mathrm{Z}.$$

Hence the size of P5 is

$$402410930323 + 67997694\mathrm{U} + 115446\mathrm{Z} \text{ or } 53 + 8\mathrm{U} + 4\mathrm{Z}.$$

The size of $(x, y) \in \mathrm{U}$ and $(x', y) \in \mathrm{U} \implies x = x'$ is

$$10275 + 589x + 589x' + 392y + 2\mathrm{U} \text{ or } 11 + 2x + 2x' + 2y + 2\mathrm{U}.$$

The size of $(\forall y)((x, y) \in \mathrm{U}$ and $(x', y) \in \mathrm{U} \implies x = x')$ is

$$4192525 + 786\mathrm{U} + 231477(x + x') \text{ or } 43 + 6\mathrm{U} + 6x + 6x'.$$

The size of $(\forall x')(\forall y)((x, y) \in \mathrm{U}$ and $(x', y) \in \mathrm{U} \implies x = x')$ is

$$1024059366435 + 181941708\mathrm{U} + 53581833006x \text{ or } 351 + 42\mathrm{U} + 42x.$$

Hence the size of P6 is

$$577419907899644255820 95 + 9748770215064355956\mathrm{U} \text{ or } 16943 + 1806\mathrm{U}.$$

The sum of the sizes of the expressions P3 to P6 is hence

$$585729790760875148894 16 + 9889068641119971100\mathrm{U} + 115446\mathrm{Z} \text{ or } 36044 + 3628\mathrm{U} + 4\mathrm{Z}.$$

Final computations. Assume W has length $\alpha+\beta U+\gamma Z+u$. In the case of W^2 we have

$$\alpha = 3043733, \beta = 3632, \gamma = 5675.$$

We deduce, that the size of 1 is

$$L'[1] = 40\,307\,230\,361\,203.$$

The details are: 4817276453682 τ, 10590599054995 $\square$, 2825270503356 $\vee$, 14308755797532 $\neg$, 2196562727394 $=$, 5445984229644 $\in$, 122781594600 $\supset$. In billions (10^{12}), we get a total of 40 = 4.8+10.6+2.8+14.3+3.2+5.4+0.1. Hence the symbol that appears most is negation, with a frequency of 35%, followed by a square (or link), with frequency 26%. Each τ has an average of 2.2 links.

The details of links per τ is given in the table below. In the first column have k, the number of links, in the second the number of occurrences c, then the frequency c/t, then the contribution $1000kt/c$. Note that 95% of the links come from Y, either directly from $\emptyset$ or from $\{\emptyset\}$.

1	2097848965800	0.43	435	$\emptyset$
1	52809288	1e-5		P5
1	52809288	1e-5		P3
2	2097848965800	0.43	871	$\emptyset$
2	49904777160	0.001	21	$Y\times Z$
2	7947797844	0.002	3	P4
2	7947797844	0.002	3	P6
2	26404644	5e-6		P2
3	39606966	8e-6		P3
3	39606966	8e-6		P5
6	449539064100	0.093	560	$\{\emptyset\}$
6	21387761640	0.004	26	$Y\times Z$
6	3406199076	7e-4	4	P4
6	3406199076	7e-4	4	P6
14	74923177350	0.015	218	$\{\emptyset\}$
42	554497524	1e-4	5	P4
42	554497524	1e-4	5	P6
44	1742706504	4e-3	16	$Y\times Z$
90	39606966	8e-6		$Y\times Z$
3632	13198688	3e-6		FU
3633	3633	7e-10		Fu
74923177350	1	2e-13	16	Fz

Table 12.1: Number of links par τ for (**)

In the case of (*) we have

$$\alpha = 3026786, \quad \beta = 1826, \quad \gamma = 5675.$$

The size of (*) is $10133781553729 \approx 10.1\cdot 10^{12}$. This means that the size is reduced by a factor 4 if we omit P6 (whose size is exactly the same as P4). The distribution of links is similar to that of (**), except for the three outer τ with 1826, 1827 and 18953115300 links.

Consider now the simplified definition of a singleton. We have

$$\alpha = 1363933, \quad \beta = 3632, \quad \gamma = 5675 \qquad \text{for (**)};$$

$$\alpha = 1346986, \quad \beta = 1826, \quad \gamma = 5675 \qquad \text{for (*)}.$$

This gives a length of $4523659424929 \approx 4.52 \cdot 10^{12}$ and $18129969865603 \approx 18.1 \cdot 10^{12}$ for (*) and (**) respectively. The number of links of (*), in agreement with [16] is 179618517981. In all four cases, the number of links is 26% of the size of the formula, and the average numbers of links for each τ is between 2.2 and 2.4. The distribution of links is similar to the previous case, except that $\{\emptyset\}$ give τ_4 and τ_{10} instead of τ_{14}. In the case of (*) the contributions of the τ_1, τ_2, τ_4 and τ_{10} from Y are now 0.383, 0.765, 0.612 and 0.382. This gives 2.14 out of 2.38.

Modified 1. In this case, the numbers become drastically larger. The size is

573306704401838125661472729409307306784967637530522131918 0433	SS
17171367179050008152549470214937315341641246258381426417	S
3156282763109242044567148677179431686312165927279032398 2733	SM
2409875496393137472149767527877436912979508338752092897	M

The number of links is

216340973779765017662124495881515896212511239392590425057 2945	10.98	SS
64721418601644823282069968924446326070425926610821984 49	10.86	S
114233082399268622399966195815537414473620555964425454 00925	9.38	SM
871880233733949069946182804910912227472430953034182177	9.36	M

In these table S and SS stand for (*) and (**), while M and SM are the versions of (*) and (**) with a simplified singleton. The average number of links pet τ has increased; in fact, as the next table shows, the leading symbol is $\square$ with frequency of 38%, followed by negation 21%.

τ	$\square$	$\vee$	$\neg$	$=$	$\in$	$\supset$	
34	377	171	211	137	69		SS
34	377	171	211	137	69		S
39	362	162	238	123	77		SM
39	362	162	238	123	77		M
119	262	70	355	54	135	3	uSS
109	260	73	369	47	136	6	uM

Table 12.2: Frequency of symbols in per mille; first for the modified version (four cases), then for the original definition that contains a $\supset$ (two cases).

We give in the tables below the number of links per τ is case SS and M. The first table is a bit simplified. It shows that 99% of all τ come from P6, more precisely from the pairs (x, y) and (x, y') that appear in P6. In the second table we consider the simplified (*); here the dominant term is P4 (since there is no P6). Half of the τ have 6 links and come from the pairs in P4. (in the table p3p, p4p and p5p stand for pair in P3, P4 and P5).

1	9e-5		∅
2	9e-5		∅ P2
6	0.854	5.13	mainly P6
14	0.140	1.99	mainly P6
196	1e-1		P3
197	4e-5	0.008	Y × Z P4
392	0.002	0.946	P6
588	3e-14		P5
39006	2e-7	0.007	P4
115837	5e-17		P3
115445	4e-17		P5
116622	2e-8	0.002	Y × Z
231477	5e-6	1.42	P6
46182710	7e-14		Y × Z
92365422	1e-21		Y × Z
9082701936	4e-12	0.04	P4
53581833006	3e-11	1.42	P6
n_1	5e-22	0.005	FU
n_1+1	1e-41		Fu
n_2	5e-60		Fz

n_1= 9889068641120316847
n_2= 625735587778656516230695832329759701991558005824241312

Table 12.3: Links per τ; modified (**)

The size of the real one. In what follows, we consider the definition of 1, without any simplification. This means that we do not follow Mathias and simplify singletons, we do not forget injectivity, etc. There are four results, since we have two interpretations and two versions, with or without ⊃.

Recall the definitions of pr_1 and pr_2 given above. We get

$$\mathrm{L}[\mathrm{pr}_1 z] = 197z + 1165847, \quad \mathrm{L}[\mathrm{pr}_2 z] = 589z + 3485703.$$

The numbers will become rapidly huge, so we mention only the size of the initial version. Here $\mathrm{L}'[\mathrm{pr}_1 z] = \mathrm{L}'[\mathrm{pr}_2 z] = 9+2z$. Let A, B, G be the source, graph, target of f. These quantities are $\mathrm{pr}'_2 f$, $\mathrm{pr}'_3 f$ and $\mathrm{pr}'_1 f$ hence

$$\mathrm{L}'[\mathrm{A}] = 27+4f, \quad \mathrm{L}'[\mathrm{B}] = 9+2f, \quad \mathrm{L}'[\mathrm{G}] = 27+4f.$$

To say that z is a pair means $(\exists x)(\exists y)(z = (x, y))$; the size is $24+6z$. Now G is a graph if every element is a pair; the size is The size is

$$261 + 8\mathrm{U} \text{ or } 497 + 32f$$

Here the first value corresponds to interpretation 1 (replace source target and graph by Y, Z and U everywhere) and the second valur corresponds to interpretation 2 (use A, B, G, these are values that depend on f). Note that the modified version of this relation has size 10^{18}. To say that G is functional is P4; we have computed its size above; it suffices to replace U by G to get:

$$6943 + 1806\mathrm{U} \text{ or } 65705 + 7224f.$$

1	0.001	1	∅
2	0.001	3	∅
2	1e-19		P2
4	1e-15		P1
4	5e-11		p3p
4	8e-11		p5p
4	0.31	1240	p4p
4	8e-3	32	$Y \times Z$
4	5e-4	2	{∅}
6	2e-15		P1
6	9e-11		p3p
6	0.50	2987	p4p
6	1e-10		p5p
6	0.013	78	$Y \times Z$
10	0.077	774	p4p
10	1e-16		P1
10	1e-11		p3p
10	2e-11		p5p
10	1e-4	1	{∅}
10	0.002	20	$Y \times Z$
14	0.08	1161	p4p
14	3e-16		P1
14	3e-11		p3p
14	2e-11	1	p5p
14	0.002	30	$Y \times Z$
196	9e-13		P3
197	1e-4	28	$Y \times Z$
197	0.002	542	P4
336	1e-12		P5
39006	1e-5	544	P4
66053	5e-15		P5
66193	5e-15		P3
66727	7e-7	48	$Y \times Z$
26423894	5e-12		$Y \times Z$
52847790	2e-20		$Y \times Z$
5190115392	3e-10	1858	P4
n_U	7e-20	6	$\exists U$
$n_U + 1$	9e-37		$\exists u$
n_F			outer tau

$n_U = 80170529164774711$.
$n_F = 1346307842460763835203449580562034369020611715112$0

Table 12.4: Number of links per τ, case of simplified (*)

Because pr_1 and pr_2 are overloaded, the sizes of $\mathrm{pr}_1\mathrm{G}$ and $\mathrm{pr}_2\mathrm{G}$ are $219+28\mathrm{G}$ (on the modified version the two quantities are a bit different). We deduce that the size of $\mathrm{A} \subset \mathrm{pr}_1\mathrm{G}$ and $\mathrm{B} \subset \mathrm{pr}_2\mathrm{G}$:

$$2901 + 168\mathrm{U} + 3\mathrm{Z} \text{ or } 6006 + 690f.$$

One can note that this is smaller than the equivalent $\mathrm{G} \subset \mathrm{A} \times \mathrm{B}$.

Let's say that f is a triple; this means "$(\exists a)(\exists b) f = (a, b)$ and a is a pair". This has size $1140+30f$. We prefer the alternative "f is a pair and $\mathrm{pr}_1 f$ is a pair", whose size is $106+18f$. In the modified version of Bourbaki the sizes become $1.196 \cdot 10^{23}$ and $2.687 \cdot 10^{13}$; the difference is huge.

Our first result is now: the size of the assembly that says that f is a function is

$$17680 + 1834\mathrm{U} \text{ or } 73333 + 8080f.$$

The expression $f(x)$ is short for $\tau_y((x, y) \in \mathrm{G})$; its size is hence

$$4 + \mathrm{U} + x \text{ or } 31 + 4f + x.$$

The size of the term that says that a function is injective is

$$54991 + 104\mathrm{U} \text{ or } 7255 + 832f$$

The size of the term that says that a function is surjective is

$$28542 + 54\mathrm{U} + \mathrm{Z} \text{ or } 3765 + 434f.$$

Let's put everything together. Interpretation 1, original Bourbaki. With the same notations as above we have

$$\alpha = 104931, \quad \beta = 2169, \quad \gamma = 5.$$

This gives a size of $504583863421 \approx 0.5 \cdot 10^{12}$. This is a bit smaller than the results shown above. Interpretation 1, modified Bourbaki. We have

$$\alpha = 834070598979660692906, \quad \beta = 140298427582600295, \quad \gamma = 200.$$

This gives a size of

$$1642031431480645956466162930607999962764297936549315662 5 \approx 1.64 \cdot 10^{55}$$

This number is comparable to the size of (**). Interpretation 2. Here 1 has the form $\tau_{\mathrm{Z}}((\exists f)\,\mathrm{W})$. If W has size $\alpha f + \beta + \mathrm{Z}$, then 1 has size $1 + (\alpha+1)(\alpha+\beta+1)$. In this case

$$\beta = 158257, \quad \alpha = 17432$$

and

$$\beta = 6477400268819447467497511 3, \quad \alpha = 1088968338402435642781 9.$$

This gives size of 3 062 803 771 or

$$7054869659945846633088034340300875774509507960 61 \approx 7 \cdot 10^{47}$$

τ	□	∨	¬	=	∈	⊃	□/τ	
108	277	74	337	53	129	21	2.55	I1o
163	384	24	69	161	26	172	2.35	I2o
119	262	70	355	54	135	3	2.2	SSo
34	377	171	211	137	69		10.98	I1m
34	377	171	211	137	69		10.95	I2m
34	377	171	211	137	69		10.98	SSm

Table 12.5: Frequency of symbols in per mille; (o=original, m=modified

τ	54761477680	499420584
□	139758602621	1176483439
∨	37292830120	73619559
¬	170305180500	210416310
=	26740033320	492081291
∈	65314274480	80958852
⊃	10411464700	529823736

Table 12.6: Number of signs, interpretation 1 and 2, original

16420314314806459564661629306079999627642979365493156652	total
564073585029977154146795690970067394284202536239148080	τ
6196422173531692723350614472358911901851695485535741377	□
2816174294250857784601909390694422253783746474648296648	∨
3463396382713096963813604670392108429655385858182525792	¬
2252121210124743929682191676578639510232293382144795888	=
1128126669156091009066513405085850137835655628742648840	∈

Table 12.7: Number of signs, Interpretation 1, modified

705486965994584663308803434030087577450950796061	total
24314804679835023673226414023140543962456989380	τ
266270970602380970656184591066844038965101856741	□
120978082961272973491479088521851747501322433680	∨
148630220109987698323207837873258955558290093200	¬
96901652623232249770153642145778113503186060140	=
48391235017875747394551860399214177960593362920	∈

Table 12.8: Number of signs, Interpretation 2, modified

1	418392	8	triple
1	488124	10	$\emptyset$
1	836784	17	G
1	1464372	29	ls2
1	1952496	39	ls1
1	2754414	55	$f(x)$
1	8733933	174	T
1	218505222	4375	G
2	69732	3	ls1
2	69732	3	ls2
2	139464	6	T
2	348660	14	triple
2	488124	20	$\emptyset$
2	557856	22	G
2	627588	25	surjective
2	8733933	349	S
2	20989332	840	P4
2	218505222	8750	G
3	679887	40	injective
6	104598	13	$\{\emptyset\}$
6	627588	75	ls2
6	836784	100	ls1
6	8995428	1080	P4
7	244062	34	G
8	139464	22	surjective
12	209196	50	injective
14	104598	29	ls2
14	139464	39	ls1
14	17433	5	$\{\emptyset\}$
18	17433	6	surjective
42	1464372	1231	P4
17432	17432	6084	$\exists f$
17433	1	0	outer τ

Table 12.9: Statistics for Interpretation 2, original; column 3 is contribution times 10000

1	113065680	21	G
1	197864940	36	ls2
1	263819920	48	ls1
1	744349060	136	$f(x)$
1	21501323480	3926	$\emptyset$
2	9422140	3	ls1
2	9422140	3	ls2
2	75377120	28	G
2	169598520	62	surjective
2	2836064140	1035	P4
2	21501323480	7853	$\emptyset$
3	183731730	101	injective
6	84799260	93	ls2
6	113065680	124	ls1
6	1215456060	1331	P4
6	4607426460	5048	$\{\emptyset\}$
7	32977490	42	G
8	37688560	55	surjective
12	56532840	123	injective
14	14133210	36	ls2
14	18844280	48	ls1
14	767904410	1963	$\{\emptyset\}$
19	4711070	16	surjective
42	197864940	1517	P4
2169	4708899	1865	fU
2170	2170	1	Fu
23555350	1	4	outer τ

Table 12.10: Statistics for Interpretation 1, original; column 3 is contribution times 10000

Statistics. In the next tables we give the frequency of each symbol and the number of links per τ. Note that in the modified version, the numbers are the same as those of (**).

The next table show detailed statistics for interpretation 2, original. In the table ls1 stands for $A \subset \mathrm{pr}_1 G$ or $A = \mathrm{pr}_1 G$, and ls2 stands for $B \subset \mathrm{pr}_3 G$. The major contributions of links come from the τ_1 and τ_2 of G (which is $\mathrm{pr}'_1 f$). Then comes the links of the τ associated to $\exists f$, then comes P5 (this says that the graph is functional).

The next table shows statistics for interpretation 1, original version. This corresponds to (**) without simplifications. Here 80% of all links come from the $\emptyset$ in Y (which is $\{\emptyset\}$) woth a contribution of 1.88 (out of2.55). The contribution of P4 is 0.39, that of $\exists$U is 0.19.

The table that follow correspond to the modified 1. We give both interpretations in parallel. The dominant term is P4; it says that the graph is functional. This term has five τ with 6, 14, 197, 39006 and n_2 links, where n_2 is given in the table below. This term depends on U; in interpretation 2, U is replaced by $\mathrm{pr}'_1 f$ which is an object with 6, 14, 196 or 115835 links. In both interpretations, 99% of all τ come from the τ_6 or τ_{14} of P5 or of the U in P4. The contribution of these terms is 7.1. The τ with 196, 197 or 39006 links have a contribution of ≈ 1. The other links have a contribution of 2.88 (this seems a strange coincidence).

1	4.041e-10		1.254e-23		∅
2	4.041e-10		1.254e-23		∅
2	6.979e-20		1.791e-24		ls1
2	6.979e-20		1.791e-24		ls2
6	3.576-15				P1
6	2.217e-9		5.691e-14		ls1
6	4.954e-9		1.271e-13		ls2
6	8.661e-11		2.687e-24		{∅}
6	0.00316	189	8.116e-8		simple graph
6	2.511e-9		3.222e-14		$f(x)$
6	0.852	51112	2.187e-5	1	P4
6			0.853	51175	G
6			3.758e-9		S
6			3.672e-19		T
6			7.078e-13		is triple
14	5.960e-16				P1
14	3.695e-10		9.484e-15		ls1
14	8.2573e-10		2.120e-14		ls2
14	1.443e-11		4.479e-20		{∅}
14	4.185e-10		5.372e-15		$f(x)$
14	0.0005	73	1.352e-8		simple graph
14	0.142	19876	3.644e-6	1	P4
14			0.142	19901	G
14			6.264e-10		S
14			6.120e-20		T
14			1.189e-13		is triple
196	1.268e-11		3.253e-16		ls1
196	1.443e-11		1.852e-16		$f(x)$
196	1.808e-5	35	4.641e-10		simple graph
196			0.0049	9559	G
196			2.138e-11		S
196			4.047e-15		is triple
197	0.00244	4798	6.251e-8		P4
588	2.842e-11		7.296e-16		ls2
588			1.089e-13		S
588			2.107e-21		T
589	4.862e-14		6.241e-19		surjective
590	7.179e-12		8.214e-17		injective

Table 12.11: Distribution of links for Interpretations 1 and 2, modified

1182	4.124e-17		5.293e-22		surjective
2366	3.490e-20		4.479e-25		surjective
39006	1.362e-5	4822	3.173e-10		P4
115444			1.842e-16		S
115444			3.582e-24		T
115836	9.224e-8	107	2.368e-12		simple graph
115836			2.488e-5	28822	G
115836			2.064e-1	7	is triple
115836			1.091e-13		S
230890	2.417e-14		6.204e-19		ls2
231674	3.233e-14		8.300e-19		ls1
348690	1.216e-14		1.562e-19		injective
461782	3.5e-20		2.687e-24		ls2
463350	1.396e-19		3.582e-24		ls1
22819890	7.963e-13		2.044e-17		simple graph
n_2	3.169e-10	28787	8.136e-15	1	P4
n_1	3.490e-20	49			FU
n_1+1	2.5e-37				Fu
n_3	1.772e-54				Fz
n_4			4.478e-25	49	exists f
n_4+1			4.113e-47		outer τ

n_1= 140298427582600295
n_2= 9082701936
n_3= 39367297564300279353580632963775 82400
n_4= 108896833840243564 27819

Table 12.12: Distribution of links for Interpretations 1 and 2, modified (continued)

Chapter 13

Exercises

There are 111 exercises for this chapter. We give here the number of lines of the code of all those that are solved. Some Exercise of section 5 are implemented using the *bigop* file of the SSREFLECT library.

Section 1. 1 (12), 2 (580), 3 (994), 4 (373), 5 (137), 6 (975), 7 (137), 8 (37), 9 (183), 10 (185), 11 (482), 12 (14), 13 (107), 14 (79), 15 (615), 16 (307), 17 (326), 18 (626), 19 (191), 20 (497), 21 (627), 22 (688), 23 (274*), 24(532*).

Section 2. 1 (278), 2 (137), 3 (190), 4 (136), 5 (14), 6 (294), 7 (1030) , 9 (38), 10 (39), 11 (193), 12 (420), 13 (3650), 14(400), 15(1200), 16(190), 17 (1460), 18(550), 19(1440), 20(220).

Section 3. 1 (78), 2 (59), 3 (276), 4 (52), 5 (85), 6 (18).

Section 4. 1 (100), 2 (51), 3 (21), 4 (193), 5 (774), 6 (642*), 7 (270), 8 (735), 9 (760), 10 (1000).

Section 5. 1 (290/46), 2 (98/75), 3 (), 4 (360/277), 5 (650), 10 (220).

Section 6. 1 (134), 2 (49), 3 (55), 4 (73), 5 (332), 6 (108), 7 (131), 8 (94), 9 (99), 10 (650), 11(105), 12 (1150*), 13(340), 14(500), 15 (403), 17 (24), 18 (321), 19 (460), 20 (393), 21 (164), 22 (187).

Ex 1.23, 1.24, 2.8, 3.4, are un-complete

13.1 Additional theorems

Zermelo's theorem. We start with an alternate proof of the transfinite principle: assume E well-ordered, and (H) for all $x \in \mathrm{E}$, the relation $\forall y, y < x \implies p(y)$ implies $p(x)$. Then p holds on E. Let A the set of all x that does not satisfy p; if non-empty, this set has a least element x; but (H) says $p(x)$, absurd.

```
Lemma transfinite_principle_bis r (p:property):(* 9 *)
  worder r ->
  (forall x, inc x (substrate r) ->
    (forall y, glt r y x -> p y) -> p x) ->
  forall x, inc x (substrate r) -> p x.
```

We present here a simple (150 lines) proof of Zermelo's theorem. It says that any set E with a choice function can be well-ordered. Moreover, it is the only order such that the choice function, applied to the set of elements $\geq x$, chooses x. A "choice function" is a function r defined on $\mathfrak{P}(\mathrm{E}) - \{\emptyset\}$ such that $r(\mathrm{A}) \in \mathrm{A}$.

The idea is to use remainders instead of segments. We start by studying these objects. We assume that E is ordered by $\leq$. A segment S is subset of E such that $x \in S$ and $y \leq x$ implies $y \in S$; a *remainder* is a set R such that $x \in R$ and $x \leq y$ implies $y \in R$. The complement of a segment is a remainder. The quantity $S_x =]\leftarrow, x[$ is the segment with endpoint x, while $R_x = [x, \rightarrow[$ is the remainder starting with x. In a totally ordered set, R_x is the complement of S_x. The two sets R_x and $R_x - \{x\}$ are remainders. Since the union of segments is a segment, the intersection of remainders is a remainder. Evert remainder R_x is either E, of the form $R_z - \{z\}$, or is the intersection of all R_z for $z < x$ (if S_x is non-empty, it has a least upper bound z; if $z \in S_x$ we are in the second case, otherwise in the third).

```
Definition segment_rp r s:=
  sub s (substrate r) /\ (forall x y, inc x s -> gle r x y -> inc y s).

Lemma segment_rpC r x : order r -> sub x (substrate r) ->
    (segmentp r x <-> segment_rp r (substrate r -s x)). (* 7 *)
Lemma segment_rpC' r x : order r -> sub x (substrate r) ->
    (segment_rp r x <-> segmentp r (substrate r -s x)).  (* 2 *)
Lemma segment_rC r x: total_order r -> inc x (substrate r) ->
  (segment_r r x) = substrate r -s (segment r x).  (* 6 *)
Lemma segment_rC' r x: total_order r -> inc x (substrate r) ->
  (segment r x) = substrate r -s (segment_r r x).  (* 1 *)
Lemma segment_r_p0 r: segment_rp r emptyset. (* 1 *)
Lemma segment_r_p1 r x: order r -> inc x (substrate r) ->
  segment_rp r (segment_r r x). (* 3 *)
Lemma segment_rp_p2 r z:  order r -> inc z (substrate r) ->
   segment_rp r (segment_r r z -s1 z). (* 5 *)
Lemma segment_rp_p3 r s: order r ->
   (alls s (segment_rp r)) -> segment_rp r (intersection s). (* 7 *)
Lemma segment_r_succ_or_limit r x: worder r -> inc x (substrate r) -> (* 35 *)
  [\/ (segment_r r x = substrate r),
  (exists2 z, inc z (segment r x) & segment_r r x = segment_r r z -s1 z) |
  (segment_r r x) = intersection (fun_image (segment r x) (segment_r r)) ].
```

We consider the set Ω of all R_x, together with the empty set. In a well-ordered set, a segment is E or an initial segment. This means that Ω is the set of all remainders.

```
Definition segment_rs r:= (fun_image (substrate r) (segment_r r) +s1 emptyset).
Lemma segment_rs_P r x:   (* 2 *)
  (inc x (segment_rs r) <->
     x = emptyset \/ (exists2 y, inc y (substrate r) & x = segment_r r y)).
Lemma segment_rs_p1 r: worder r -> (* 9 *)
  forall x, inc x (segment_rs r) <-> segment_rp r x.
```

We say that a well-ordered set is *Zermelo-like* if $r(R_x) = x$ and denote this by (Z). If this condition holds, X is a non-empty subset of E, x its least element, $Y = R_x$, then Y is a segment such that $X \subset Y$ and $r(Y) \in X$, and no other segment satisfies this property.

Let's recall the definition of a chain F of E. It (a) is a subset of $\mathfrak{P}(E)$, (b) contains E, (c) is stable by intersection, and (d) is such that $A \in F$ implies $p(A) \in F$. Here $p(A) = A - \{r(A)\}$ where r is the choice function. Note that Ω satisfies (a), (b) and (c). Condition (Z) implies (d) so that Ω is a chain. Note that any chain F contains the empty set (the intersection x of F is in F by (c), so $p(x) \in F$ by (d); since x is the smallest element of F, it has to be empty). It follows that Ω is the least chain (it is contained in all chains, and is the intersection of all chains).

The proof is by transfinite induction: every R_x is in F. It suffices to assume $R_y \in F$ whenever, $y < x$. Now R_x is either E (we conclude by (a)), $R_z - \{z\}$ with $z < x$ (we conclude by (d) and (Z)) or the intersection of all R_z, $z < x$ (we conclude by (c)).

```
Definition Zermelo_chain E F :=
  let p :=  fun a => a -s1 (rep a) in
   [/\ sub F (\Po E), inc E F,
    (forall A, inc A F -> inc (p A) F)
    & (forall A, sub A F -> nonempty A -> inc (intersection A) F)].
Definition Zermelo_like r:= worder r /\
  forall x, inc x (substrate r) -> rep (segment_r r x) = x.

Lemma Zermelo_like_chain_least1 r X:
  Zermelo_like r ->  sub X (substrate r) -> nonempty X ->
  exists!Y, (sub X Y /\ inc Y (segment_rs r) /\ inc (rep Y) X). (* 24 *)
Lemma Zermelo_omega_chain r (E := substrate r): Zermelo_like r ->
  Zermelo_chain E (segment_rs r). (* 12 *)
Lemma Zermelo_chain_least E F: Zermelo_chain E F -> inc emptyset F. (* 6 *)
Lemma Zermelo_chain_minimal1 r (E := substrate r):  (* 14 *)
  Zermelo_like r ->
  forall F, Zermelo_chain E F -> sub (segment_rs r) F.
Lemma Zermelo_chain_minimal r (E := substrate r): (* 9 *)
  Zermelo_like r ->
  (segment_rs r) = intersection (Zo (\Po (\Po E)) (Zermelo_chain E)).
```

Assume that we have two orders, $\leq$ and $\leq'$, which are Zermelo like. By the previous result, they have the same remainders. So every R_x is a R'_y. So $x = r(R_x) = r(R'_y) = y$. The relation $R_x = R'_x$ says that the two orders must be the same.

```
Lemma Zermelo_unique r r': Zermelo_like r -> Zermelo_like r' ->
   substrate r = substrate r' -> r = r'.
```

The construction is now the following. We let Ω be the intersection of all chains. If A is a subset of E, we define $d(A)$ to be the intersection of all elements of Ω that contain A. Assume the problem solved, $A = \{a\}$. Now, $d(A)$ is the intersection of all R_b such that $b \leq a$, thus is R_a. We use this as the definition of R. The order is defined by $x \leq y$ if and only if $R(y) \subset R(x)$.

This gives us the following definition:

```
Definition worder_of E :=
let om :=  intersection (Zo (\Po (\Po E)) (Zermelo_chain E)) in
let d:= fun x => intersection (Zo om (sub x)) in
let R := fun x => d (singleton x) in
graph_on (fun x y => (sub (R y) (R x))) E.
```

Let's show that this is a well-ordering on E such that $R_x = [x, \to[$. We first unfold the definitions and introduce some local variables.

```
Lemma Zermelo_ter E (r := worder_of E):
   worder_on r E /\ Zermelo_like r.
Proof.
rewrite /r /worder_of; clear r.
set chain := (Zermelo_chain E).
set om := intersection (Zo (\Po (\Po E)) chain).
```

```
set d:= fun p => intersection (Zo om (sub p)).
set R := fun x => d (singleton x).
set res:= graph_on (fun x y => (sub (R y) (R x))) E.
```

We have $p(\mathrm{A}) \subset \mathrm{A}$, and if $\mathrm{A} = \emptyset$, then $p(\mathrm{A}) = \mathrm{A}$.

```
set p:=  fun a => a -s1 (rep a).
have pe: p emptyset = emptyset by apply /set0_P => x /setC_P [/in_set0 xe _].
have sp: forall a, sub (p a) a by move=> t; apply:sub_setC.
```

We show here that the power set of E is a chain. This will have as consequence that Ω is well-defined.

```
have cp:chain (\Po E).
  split; fprops; first  by apply /setP_P.
    move=> A /setP_P=>  AE; apply/setP_P;apply: sub_trans AE; apply: sp.
  move=> A AP [x xA]; move: (AP _ xA) => /setP_P xE; apply/setP_P.
  move=> t ti; exact: (xE _ (setI_hi ti xA)).
```

We show here that Ω is the least chain (for set inclusion).

```
have co :chain om.
  have aux: nonempty (Zo (\Po (\Po E)) chain).
    by exists (\Po E); apply: Zo_i; aw; apply: setP_Ti.
  split.
  + by apply:setI_s1; apply: Zo_i=> //; apply: setP_Ti.
  + by apply: (setI_i aux) =>y /Zo_hi [_].
  + move=> A Ai; apply:(setI_i aux) => y yi.
    move/Zo_hi: (yi) => [_ _ q _];apply: q;apply: (setI_hi Ai yi).
  + move=> A sAi neA;  apply: (setI_i aux) => y yi.
    by move/Zo_hi: (yi) =>  [_ _ _]; apply => // t /sAi /setI_hi; apply.
move: (co)=> [sop Eo po io].
have cio: forall x, chain x -> sub om x.
  by move=> x [ha hb hc hd]; apply: setI_s1; apply: Zo_i =>//;apply /setP_P.
```

Now comes a big part of the proof. Let $m(\mathrm{A})$ be the property that for all $\mathrm{X} \in \Omega$, we have either $\mathrm{X} \subset \mathrm{A}$ or $\mathrm{A} \subset \mathrm{X}$. We pretend that $m(\mathrm{A})$ holds for all elements of Ω. In fact, the set of elements that satisfy m is a chain, thus has to be Ω. The non-obvious point is to show that $m(\mathrm{A})$ implies $m(\mathrm{A}')$ where A' is short for $p(\mathrm{A})$. In fact, let T be the set of elements B of Ω such that $\mathrm{B} \subset \mathrm{A}'$ or $\mathrm{A} \subset \mathrm{B}$. It contains E and is stable by intersection (if for all i, $\mathrm{A} \subset \mathrm{T}_i$, then $\mathrm{A} \subset \bigcap \mathrm{T}_i$, otherwise there is i such that $\mathrm{T}_i \subset \mathrm{A}'$ and $\bigcap \mathrm{T}_i \subset \mathrm{A}'$). Assume $\mathrm{B} \in \mathrm{T}$. If $\mathrm{B} \subset \mathrm{A}'$, then $\mathrm{B}' \subset \mathrm{A}'$. Since B' is in Ω, we have either $\mathrm{A} \subset \mathrm{B}'$ or $\mathrm{B}' \subset \mathrm{A}$. In the first case, $\mathrm{A}' \subset \mathrm{B}'$. Now we have $\mathrm{B}' \subset \mathrm{A} \subset \mathrm{B}$. Let b' be $r(\mathrm{B}')$. If $b' \in \mathrm{A}$ then $\mathrm{B} \subset \mathrm{A}$, then $\mathrm{A} = \mathrm{B}$, hence $\mathrm{B}' \subset \mathrm{A}'$. Otherwise $\mathrm{A} \subset \mathrm{B}'$. In summary, T is a chain, thus must be Ω.

```
have am: om = Zo om m.
  apply: extensionality; last by apply: Zo_S.
  apply: (cio); split => //.
  + by apply: sub_trans sop; apply: Zo_S.
  + by apply: Zo_i=>//; move=> x xom; left;move: (sop  _ xom); move/setP_P.
  + move=> A /Zo_P [Aom mA]; apply: Zo_i; first by apply: (po _ Aom).
    suff aux: sub om (Zo om (fun x=> sub x (p A) \/ sub A x)).
      move=> x xom; case: (mA _ xom) => hyp.
        move: (aux _ xom) => /Zo_hi; case =>xpB; first by left.
```

```
        rewrite (extensionality xpB hyp); right; apply: sp.
      right;  apply: (sub_trans (sp A) hyp).
    apply: cio; split.
    - by apply: (@sub_trans om); first by apply: Zo_S.
    - by apply: Zo_i=>//; right; move: (sop _ Aom);move/setP_P.
    - move => B /Zo_P [Bom ors]; apply: Zo_i; first by apply: (po _ Bom).
      case: ors => orsi; first by left; apply: sub_trans orsi; apply: sp.
      case: (mA _ (po _ Bom)) => aux; last by right.
      case: (inc_or_not (rep B) A)=> aux2.
        rewrite (extensionality orsi _); first by left.
        move=> t tB; case: (equal_or_not t (rep B)); first by move=> ->.
        by move=> trB; apply: aux; apply/setC1_P; split.
      right; move=> t tA;apply/setC1_P;split; [ by apply: orsi| dneg trB; ue].
    - move=> B sB neB; apply: Zo_i.
        apply: io =>//;  apply: (sub_trans sB) ; apply: Zo_S.
      case: (p_or_not_p (exists x, inc x B /\ sub x (p A))) => H.
        move: H=> [x [xB xp]]; left; move=> t ti; apply:(xp _ (setI_hi ti xB)).
      right; move=> t tA;  apply: setI_i=>//.
      move=> y yB; move: (sB _ yB) => /Zo_P [yom ]; case; last by apply.
      by move => sy; case: H; exists y.
  + move=> A sAZ neA; apply: Zo_i.
      apply: io =>//; apply: (sub_trans sAZ); apply: Zo_S.
    move=> x xom.
    case: (p_or_not_p (exists2 y, inc y A & sub y x)) => H.
      move: H=> [y yA yx]; right; move => t ti;apply: (yx _ (setI_hi ti yA)).
    left; move=> t tx; apply: setI_i=>//.
    move=> y yA; move: (sAZ _ yA)=> /Zo_P  [yom my].
    case: (my _ xom);[ by apply| move=> yx; case: H;ex_tac; apply: Zo_S].
```

Consequence: if A and B are in Ω, then $A \subset B$ or $B \subset A$.

```
have st: forall a b, inc a om -> inc b om -> sub a b \/ sub b a.
  move=> a b; rewrite {2} am; move => aom  /Zo_P [bom ba]; apply: (ba _ aom).
```

We pretend here that $d(X)$ is the least element of Ω that contains X; this amounts to the three following conditions: if $X \subset E$, then $d(X) \in \Omega$, $X \subset d(X)$, and if $X \subset Y$, where $Y \in \Omega$, then $d(X) \subset Y$.

```
have dpo: forall X, sub X E -> inc (d X) om.
  by move=> X XE; rewrite /d; apply: io;[ apply:Zo_S | exists E;apply: Zo_i].
have pdp: forall X, sub X E -> sub X (d X).
  rewrite /d=> X XE t tX; apply: setI_i; first by exists E; apply: Zo_i.
  by move => y /Zo_hi; apply.
have dpq: forall X Y, inc Y om -> sub X Y -> sub X E -> sub (d X) Y.
  by rewrite /d=> X Y Yom XY XE; apply: setI_s1; apply: Zo_i.
```

Fix a set $X \subset E$ and consider $x = r(d(X))$. We pretend that if $x \in d(X)$, then $x \in X$. Recall that $d(X)' = d(X) - \{x\}$. If x is not in X, we get $X \subset d(X)'$, and by minimality, $d(X) = d(X)'$, absurd. *We assume from now on that r is a choice function*, i.e., $r(X) \in X$, whenever X is a nonempty subset of E. Thus $x \notin X$ implies that $d(X) = \emptyset$, hence that X is empty (since $X \subset d(X)$). In other words: if X is non-empty, then $r(d(X)) \in X$.

```
have rdq: forall X, sub X E -> nonempty X -> inc (rep (d X)) X.
  move=> X XE neX; case: (inc_or_not (rep (d X)) X)=>// ni.
  have aux: (sub X (p (d X))).
```

```
    move=> t tX;apply /setC1_P; split; [ by apply: (pdp _ XE)|].
    move => h;case: ni; ue.
  move: (dpq _ _ (po _ (dpo _ XE)) aux XE).
  rewrite /p; case: (emptyset_dichot (d X)).
    by move => dqe; case /nonemptyP: neX; apply/sub_set0;rewrite - pe - dqe.
  by move=> ned; move: (rep_i ned) => rd dc; move: (dc _ rd) => /setC1_P [_].
```

Conversely: assume $r(Y) \in X$ and $X \subset Y$; then $Y = d(X)$. We know $d(X) \subset Y$. If $d(X) \subset Y'$, one gets $r(Y) \in X \subset d(X) \subset Y'$, absurd; so that $Y' \subset d(X)$. The conclusion follows since $r(Y) \in d(X)$.

```
have qdp: forall X Y, inc Y om -> sub X Y -> inc (rep Y) X -> Y = d X.
  move => X Y Yom sXY YX.
  have sXE: sub X E by apply: (sub_trans sXY);  apply /setP_P; apply: sop.
  apply: extensionality  =>// t tr; last by apply: (dpq _ _ Yom sXY sXE).
  case: (st _ _  (dpo _ sXE) (po _ Yom)) => ch.
    by case /setC1_P: (ch _  ( (pdp _ sXE) _ YX)) => _.
  case: (equal_or_not t (rep Y)); first by move=> ->; apply: (pdp _ sXE).
  by move=> tnr; apply: ch; apply /setC1_P.
```

Denote by $R(a)$ the quantity $d(\{x\})$. For $x \in E$, this is in Ω and contains x. The choice function has no choice here: $r(R(x)) = x$. As a consequence R is injective.

```
have Rp: forall x, inc x E ->
    [/\ inc (R x) om, inc x (R x) & rep (R x) = x].
  move => x xE.
  have p1: sub (singleton x) E by apply: set1_sub.
  split; [ by apply: dpo | exact:(pdp _ p1 _ (set1_1 x)) | ].
  exact:(set1_eq (rdq _ p1 (set1_ne x))).
have Ri:forall x y, inc x E -> inc y E -> R x = R y -> x = y.
  move=> x y xE yE; move: (Rp _ xE)(Rp _ yE).
  by move=> [_ _ p1][_ _ p2] p3; rewrite -p3 in p2; rewrite -p1 -p2.
```

We have $R(r(X)) = X$ for $X \in \Omega$ by uniqueness.

```
have Rrq: forall X, inc X om -> nonempty X -> R (rep X) = X.
  move=> X Xom neX; symmetry; apply: qdp =>//; last by fprops.
  by move=> t /set1_P ->;  apply: rep_i.
```

Fix two elements x and y and define $D = \{x, y\}$. We have $r(R(y)) \in D$ since $r(R(y)) = y$. Assume $D \subset R(y)$. This implies $R(y) = d(D)$. Thus $D \subset R(y)$ and $D \subset R(x)$ implies $R(x) = R(y)$. Claim: if $x \in R(y)$, then $R(x) \subset R(y)$. The assumption says $D \subset R(y)$. Since Ω is totally ordered by inclusion, we have either our conclusion or $R(y) \subset R(x)$. But this implies $D \subset R(x)$ hence $R(x) = R(y)$.

```
have sRR: forall x y, inc x E -> inc y E -> inc x (R y) -> (sub (R x) (R y)).
  move=> x y xE yE xRy.
  move: (Rp _ xE)(Rp _ yE) => [Rom xRx rR] [Rom' yRy rR'].
  case (st _ _ Rom Rom') =>// hyp.
  move: (sub_set2 xRy yRy) => p1; move: (sub_trans p1 hyp) => p2.
  have p3: (inc (rep (R x)) (doubleton x y)) by rewrite rR; fprops.
  have p4: (inc (rep (R y)) (doubleton x y)) by rewrite rR'; fprops.
  by rewrite (qdp _ _ Rom p2 p3) (qdp _ _ Rom' p1 p4).
```

Since R is injective, the relation $x \le y$, short for $R(y) \subset R(x)$, is an ordering on E. The previous remarks says: if $x \in R(y)$ then $y \le x$.

```
have [or sr]:order_on res E.
  split; last by apply: graph_on_sr => a _.
  apply: order_from_rel1.
  + by move=> x y z /= xy yz; apply: sub_trans yz xy.
  + by move=> u v uE vE vu uv; apply: Ri=>//; apply: extensionality.
  + by move => u ue.
```

Given any nonempty subset A of E, the quantity $x = r(d(\mathrm{A}))$ is in A and is its least element, since $\mathrm{A} \subset d(\mathrm{A}) = \mathrm{R}(x)$.

```
have wor:worder res.
  split => // x xsr nex; exists (rep (d x)); hnf;rewrite  iorder_sr //.
  rewrite sr in xsr.
  move: (rdq _ xsr nex) => rdx; split => //.
  move => a ax; apply/iorder_gleP => //;  apply/graph_on_P1.
  split => //;try apply: xsr=>//.
  move: ((pdp _ xsr) _ ax)=> adx;  apply: sRR; fprops.
  have ne: (nonempty (d x)) by exists a.
  by rewrite  (Rrq _ (dpo _ xsr) ne).
```

That $\mathrm{R}(x) = [x, \rightarrow[$ is nearly obvious; this concludes the proof.

```
split=>//;split=>//; rewrite sr -/res => x xE.
suff: (segment_r res x) = R x by move => ->; exact:(proj33 (Rp _ xE)).
set_extens t.
  move /segment_rP /(graph_on_P0 (fun x y=> sub (R y) (R x)) E x t).
  move => [_ tE ]; apply; exact (proj32 (Rp _ tE)).
move => tR; move: ((setP_hi (sop _ (proj31 (Rp _ xE)))) _ tR) => tE.
by apply/segment_rP; apply/graph_on_P1; split => //; apply: sRR.
QED.
```

Monotonicity examples. We show here that $\mathfrak{P}(\mathrm{A} \cap \mathrm{B}) = \mathfrak{P}(\mathrm{A}) \cap \mathfrak{P}(\mathrm{B})$. We first note that any subset of both A and B is a subset of $\mathrm{A} \cap \mathrm{B}$, then that, if $\mathrm{A} \subset \mathrm{B}$ then $\bigcup \mathrm{A} \subset \bigcup \mathrm{B}$ and $\mathfrak{P}(\mathrm{A}) \subset \mathfrak{P}(\mathrm{B})$.

```
Lemma union_monotone3 A B: (* 2 *)
  sub A B -> sub (union A) (union B).
Lemma powerset_mono A B:  (* 2 *)
  sub A B -> sub (\Po A)(\Po B).
Lemma intersection_greatest A B x:  (* 1 *)
  sub x A -> sub x B -> sub x (A \cap B).
Lemma powerset_inter:  (* 7 *)
  {morph \Po : A B / A \cap B}
```

An example of well-ordering. This is example 1 page 148: *Let* $\mathrm{E} = \{\alpha, \beta\}$ *be a set whose elements are distinct. It is easily verified that the subset* $\{(\alpha,\alpha),(\beta,\beta),(\alpha,\beta)\}$ *of* $\mathrm{E} \times \mathrm{E}$ *is the graph of a well-ordering on* E.

Note that, if $\alpha = \beta$, the set E becomes a singleton, but it is still a well-ordering.

```
Definition example_worder a b:= (tripleton (J a a) (J b b)  (J a b))
Lemma example_worder_gleP a b x y: (* 2 *)
```

```
    related  (example_worder a b) x y <->
    [\/ (x = a /\ y = a),  (x = b /\ y = b) |  (x = a /\ y = b)].
Lemma substrate_example_worder a b: (* 10 *)
  substrate (example_worder a b) = doubleton a b.
Lemma example_is_worder a b:  (* 24 *)
  worder_on (example_worder a b) (doubleton a b).
```

Examples of inductive sets. Let $\mathfrak{F}$ be a set of subsets of A, ordered by inclusion, and such that for every totally ordered subset $\mathfrak{G}$ of $\mathfrak{F}$, the union of the sets of $\mathfrak{G}$ belongs to $\mathfrak{F}$. Then $\mathfrak{F}$ is inductive with respect to the relation $\subset$.

```
Lemma inductive_example1 A F: (* 8 *)
  sub A (\Po F) ->
  (forall S, (forall x y, inc x S -> inc y S -> sub x y \/ sub y x) ->
    inc (union S) A) ->
  inductive (sub_order A).
```

The set $\Phi(E,F)$ of mappings of subsets of E into subsets of F is inductive, with respect to the order "v extends u" between u and v (i.e., the opposite of the extension ordering), because there is a common extension on a totally ordered subset.

We give here the initial version of the proof script.

```
Lemma inductive_graphs: forall a b,
  inductive_set (opposite_order (extension_order a b)).
Proof. ir. cp. (extension_is_order a b). red. ir. nin H1. ee. awii H2. awii H0.
  assert (Hd: forall i j, inc i X -> inc j X ->
   agrees_on (intersection2 (source i) (source j)) i j).
  ir. cp (H0 _ H3). cp (H0 _ H4).  bwi H5; bwi H6. ee.
  cp (H2 _ _ H3 H4). ufi gge H11. red. ee.
  app intersection2sub_first. app intersection2sub_second.
  awii H11. nin H11; ee. ir. app W_extends. inter2tac.
  ir. sy. app W_extends. inter2tac.
  assert (He:forall i, inc i X -> function_prop i (source i) b).
  ir. red.  cp (H0 _ H3). bwi H4. eee.
  cp (extension_covering _ _ He Hd). nin H3. clear H4. nin H3. ee.
  red in H3. ee. assert (sub (source x) a). rw H5.
  red. ir. nin (unionf_exists H7). nin H8. cp (H0 _ H8). awi H10. ee.
  bwi H10. ee. app H11.
  assert (inc x (set_of_sub_functions a b)). bw. eee.
  exists x. red. ee. aw. ir.  aw. eee. red. cp (H0 _ H9). bwi H10. ee. am.
  am.  cp (H4 _ H9). red in H13. ee.
  red. ir. cp (in_graph_W H10 H16). rwi H17 H16.
  cp (inc_pr1graph_source H10 H16). rw H17. wr (H15 _ H18).  app W_pr3.
  app H13. rw H6. rw H12. fprops.
  app opposite_is_order.
Qed.
```

Here is the new version (ssreflect style).

```
Lemma inductive_graphs a b:
  inductive (opp_order (extension_order a b)).
```

```
Proof.
have [or ssi] := (extension_osr a b).
have [ooi oos]:= (opp_osr or).
hnf; rewrite oos ssi => X Xs toX.
have sXs :sub X (substrate (extension_order a b)) by rewrite ssi.
have Ha:forall i, inc i X -> function i by move=> i /Xs /sfun_set_P [].
have Hb:forall i, inc i X -> target i = b by move=> i /Xs  /sfun_set_P [_].
move: toX=> [orX]; aw => tor; last by ue.
set si:= Lg X source.
have Hd: forall i j, inc i (domain si) -> inc j (domain si) ->
    agrees_on ((Vg si i) \cap (Vg si j)) i j.
   rewrite /si; bw; move=> i j iX jX; bw.
   split; [by apply: subsetI2l | by apply: subsetI2r | ].
   move=> t /setI2_P [ti tj].
   case: (tor _ _ iX jX)=> h; move: (iorder_gle1 h)=> h'.
     apply: (extension_order_pr h' ti).
  symmetry;  apply: (extension_order_pr h' tj).
have He:forall i, inc i (domain si) -> function_prop i (Vg si i) b.
   rewrite /si; bw; move=> i iX; red; bw;split;fprops.
move: (extension_covering He Hd) => [[fg sg tg] _ _ agg].
set g:= (common_ext si id b).
have gs: (inc g (sub_functions a b)).
  apply /sfun_set_P;split => // t tsg.
  rewrite sg in tsg; move: (setUf_hi tsg)=> [v].
  rewrite {1}/si; bw => vx; rewrite /si; bw => tv.
  by move: (Xs _ vx) =>  /sfun_set_P [_ sv _]; apply: sv.
exists g; red; rewrite oos ssi; split=>//.
move: agg; rewrite /si; bw => agg y yX.
move: (Xs _  yX) (agg _ yX)=> ys ag.
have fy: function y by move: ys; bw; fprops.
apply /igraph_pP; apply/extension_order_P1;split => //.
rewrite (sub_function fy fg).
move: ag; rewrite /agrees_on; bw;move=> [p1 p2 p3]; split => //.
by move=> u; symmetry; apply: p3.
Qed.
```

An exercise of the first part. Exercise 5.3 reads: * Let $(X_i)_{1\le i\le n}$ be a finite family of sets. For each subset H of the index set $[1,n]$ let $P_H=\bigcup_{i\in H}X_i$ and $Q_H=\bigcap_{i\in H}X_i$. Let $\mathfrak{F}_k$ be the set of subsets of $[1,n]$ which have k elements. Show that

$$\bigcup_{H\in\mathfrak{F}_k} Q_H \supset \bigcap_{H\in\mathfrak{F}_k} P_H \text{ if } k\le (n+1)/2$$

and that

$$\bigcup_{H\in\mathfrak{F}_k} Q_H \subset \bigcap_{H\in\mathfrak{F}_k} P_H \text{ if } k\ge (n+1)/2. \quad *$$

Comments. One can generalize this exercise as follows. Let X_i be a family of sets, indexed by a finite set I. Let U and V be two subsets of I. Let P be the intersection of all unions P_H where H is equipotent to U, and let Q be the union of all intersections Q_H where H is equipotent to V. Then P is a subset of Q or Q a subset of P. The second claim is: if U and V are non-disjoint and $U\cup V=I$, then $Q\subset P$. Proof. Assume $x\in Q$, say $x\in Q_H$ for some H equipotent to V. Take H′ equipotent to U. Then H and H′ are non-disjoint so that $x \in P_{H'}$. (if H and H′ were disjoint, the cardinal of the union would be the sum of the cardinals of U and V, thus greater than the cardinal of I, since U and V are non-disjoint). The first claim is similar, using complements.

The exercise deals with the case where U and V have the same cardinal k. In what follows, we shall consider the case where the cardinals may be different. Define

$$P_k = \bigcap_{H \in \mathfrak{F}_k} P_H, \qquad Q_k = \bigcup_{H \in \mathfrak{F}_k} Q_H.$$

We have to show $Q_k \supset P_k$ if $k \le (n+1)/2$ and $Q_k \subset P_k$ if $k \ge (n+1)/2$. Note that $k \le (n+1)/2$ has to be understood as $2k \le n+1$. We show here a more general result:

$$i + j \le n+1 \implies P_i \subset Q_j, \qquad n+1 \le i+j \implies Q_i \subset P_j.$$

There are some technical conditions: if $k = 0$, then H is empty, so that $P_0 = Q_0 = \emptyset$. On the other hand if $k > n$, there is no H, so that $P_k = Q_k = \emptyset$. In both these cases, the result holds if $i = j = k$.

Assume $i+j \le n+1$; fix some x in P_i. Let J be the subset of I formed of all i such that $x \in X_i$, and k the cardinal of J. Assume $k < j$, so that $n+1 \le (n-k)+j$. It follows $i+j \le (n-k)+j$ thus $i \le n-k$. Thus, there is a subset K of I, with i elements in the complement of J. By assumption, $x \in P_K$, so that for some $k \in K$, $x \in X_k$, contradicting the definition of J. Thus $j \le k$, and there is a subset K with j elements in J, thus $x \in Q_K$. The proof of the other claim is similar.

```
Lemma exercise5_3 X i j (I:= domain X) (n := cardinal I) (* 67 *)
   (ssI := fun k => subsets_with_p_elements k I)
   (uH := fun H => unionb (restr X H))
   (iH := fun H => intersectionb (restr X H))
   (iuH := fun H => intersectionf (ssI H) uH)
   (uiH := fun H => unionf (ssI H) iH):
   fgraph X -> natp n -> natp i -> natp j ->
   (((i = j \/ j <> \0c) -> i +c j <=c succ n -> sub (iuH i) (uiH j))
   /\ (csucc n <=c i +c j -> (i = j <\/ j <=c n) -> sub (uiH i) (iuH j))).
```

Order relations. Let $R(x, y)$ be a relation that depends only on x, say it is $p(x)$. Then R is an order relation if and only if it is identically false. Obviously a false relation is an order. Assume R an order and $p(x)$ true. Then $R(x, y)$ holds; by reflexivity $R(y, y)$ holds so $p(y)$ and $R(y, x)$ hold. By antisymmetry, $x = y$, absurd.

Let R be the relation $X \subset Y \subset A$. If this is considered as a relation of X and Y, it is an order whatever A. Considered as a relation of A and X, it is not an order relation, whatever Y. Proof: by reflexivity $X \subset Y \subset A$ implies $X = A = Y$. Take $X = \emptyset$ and $A = Y \cup \{Y\}$. The assumption holds, the conclusion is absurd.

Transitivity of R says: for all x, y and z, if $R(x, y)$ and $R(y, z)$ holds then $R(x, z)$ holds. Assume that R is $x \subset y \subset z$ a variant of the relation studied above. Here $R(y, z)$ is $y \subset z \subset z$, that simplifies to $y \subset z$. This reasoning is false. Proof. Let R be the relation $x \subset y$ and $f(x, y, z)$ for some z. The right way is to write transitivity as: if $x \subset y \subset t$ and $f(x, y, z)$ and $f(y, t, z)$ then $f(x, t, z)$ (whatever x, y, t, this depending on z), the wrong way is to write $x \subset y \subset z$

and $f(x,y,z)$ and $f(y,z,z)$ then $f(x,z,z)$ (whatever x, y, z, no free variables here). The first relation implies the second, the converse is obviously false. The example we provides is a bit weird, since we want also reflexivity and antisymmetry. Reflexivity says that $f(x,x,z)$ should be true. For the bad relation to be true we assume $f(x,z,z)$. For the relation to be non-transitive, we take $z = 3$, assume $f(0,1,z)$ true, $f(1,2,z)$ true and $f(0,2,z)$ false.

```
Lemma order_indep (p: property) (r:= fun x y => p x):
  order_r r <-> forall x, ~(p x).

Lemma not_ord_example (r:= fun A X Y => [/\ sub X Y, sub X A & sub Y A]):
   (forall A, order_r (r A)) /\ (forall Y, ~ order_r(fun A X => r A X Y)).

Definition weird_comp x y z :=
  sub x y /\ [\/ x = C0 /\ y = C1, x = C1 /\ y = C2 , x = y | y = z].

Lemma weid_comp_prop:
  [/\ forall x y z, weird_comp x y z -> weird_comp y x z -> x = y,
     forall x y z, weird_comp x y z -> weird_comp x x z /\ weird_comp y y z,
     forall x y z, weird_comp x y z -> weird_comp y z z -> weird_comp x z z &
      exists z, ~ (order_r (fun x y => weird_comp x y z))].
```

The Cantor Bernstein Theorem. The objective is to give a proof of the Cantor-Bernstein theorem that says that if there is an injection from A into B and an injection from B into A, there is a bijection. (In the main text, we proved this theorem, using an alternate method; it has 55 + 24 lines of proof script).

We start with a lemma. Assume that $g : \mathfrak{P}(\mathrm{E}) \to \mathfrak{P}(\mathrm{E})$ is an increasing function for inclusion. Then g has a fixed-point. Indeed, let A be the set of all x such that $x \subset g(x)$, and U the union of A. If $x \in \mathrm{A}$ then $x \subset \mathrm{U}$, thus $x \subset g(x) \subset g(\mathrm{U})$. This gives $\mathrm{U} \subset g(\mathrm{U})$. Thus $g(\mathrm{U}) \in \mathrm{A}$, hence $g(\mathrm{U}) \subset \mathrm{U}$, so that $\mathrm{U} = g(\mathrm{U})$.

Consider two injections $f : \mathrm{E} \to \mathrm{F}$ and $g : \mathrm{F} \to \mathrm{E}$. For $\mathrm{X} \subset \mathrm{E}$, consider $h(\mathrm{X}) = \mathrm{E} - g\langle \mathrm{F} - f\langle \mathrm{X}\rangle\rangle$. Since taking the complement of a set is a decreasing function and the composition of two decreasing functions is increasing, this function is increasing. It has a fixed point M. We have $\mathrm{E} - \mathrm{M} = g\langle \mathrm{F} - f\langle \mathrm{M}\rangle\rangle$. Let $\mathrm{T} = g\langle \mathrm{F} - f\langle \mathrm{M}\rangle\rangle$. This is the complement of M. Every element in T is uniquely of the form $g(y)$ for some y not of the form $f(x)$ for $x \in \mathrm{T}$. We define $f_2(x)$ to be y (note that this function does not depend on the axiom of choice). We consider the function f_1 whose value is f on M and f_2 on T.

This function is injective. Assume $f_1(x) = f_1(y)$. If one of x, y is in M and the other one is in T, one image is in $f(\mathrm{M})$, and the other is in the complement, absurd. If both elements are in M, we use injectivity of f. Otherwise $f_1(x) = x'$ where $x = g(x')$ and $f_1(y) = y'$ where $y = g(y')$. We use injectivity of g. Since f_1 is clearly surjective, it is a bijection $\mathrm{E} \to \mathrm{F}$.

```
Lemma Cantor_Bernstein_aux E (g: fterm)    (* 11 *)
  (m:= union (Zo (\Po E)(fun x=> sub x (g x)))):
  (forall x, sub x E -> sub (g x) E) ->
  (forall x y, sub x y -> sub y E -> sub (g x) (g y)) ->
  (sub m E /\ g m = m).
Lemma Cantor_Bernstein2_full f g  (* 40 *)
  (E:= source f) (F:= source g)
```

```
  (h:= fun x => E -s (Vfs g (F -s (Vfs f x))))
  (m:= union (Zo (\Po E) (fun x : Set => sub x (h x))))
  (T:= Vfs g (F -s (Vfs f m)))
  (p := fun a y => [/\ inc y F, ~ inc y (Vfs f m) & a = Vf g y])
  (f2:= Lf (fun a =>Yo (inc a T) (select (p a) F) (Vf f a)) E F):
  injection f -> injection g -> source f = target g -> source g = target f ->
  bijection_prop f2 (source f)(source g).

Lemma Cantor_Bernstein2 f g: (* 3 *)
  injection f -> injection g -> source f = target g -> source g = target f ->
  (source f) \Eq (source g).
```

The transfinite principle. In the main text, we proved the principle of transfinite induction by applying `transfinite_principle2`. We give here a direct proof. We assume that E is well-ordered, and whenever $x \in E$, the relation $p(x)$ is a consequence of $H(x)$ that says that all $y < x$ satisfy p. Assume that there is some x not satisfying p. Then there is a least y not satisfying p. This implies $H(y)$, thus $p(y)$, absurd.

```
Theorem transfinite_principle_bis r (p:Set-> Prop):
  worder r ->
  (forall x, inc x (substrate r) ->
    (forall y, inc y (substrate r) -> glt r y x -> p y) -> p x) ->
  forall x, inc x (substrate r) -> p x.
Proof.
move => [or wor] hyp x xsr; ex_middle npx.
set (X:=Zo (substrate r) (fun x => ~ p x)).
have neX: (nonempty X) by exists x; apply: Zo_i.
have Xsr: sub X (substrate r) by apply: Zo_S.
move:(wor _ Xsr neX)=> [y []]; aw => /Zo_P [ysr npy] yle.
case: npy; apply: hyp =>//.
move=> t tsr ty; ex_middle npt.
move: (iorder_gle1 (yle _ (Zo_i tsr npt))) => nty; order_tac.
Qed.
```

Well-ordering according to Cantor. According to Cantor, a totally ordered set F is well-ordered if “(1) there is in F an element f_1 which is lowest in rank and (2) if F' is any part of F and if F has one or many elements of higher rank than all elements of F', then there is an element f' of F which follows immediately after the totality F', so that no element in rank between f' and F' occur in F” (English translation by Jourdain).

We can restate this as: if $U(F')$ denotes the set of all $x \in F$ such that $y < x$ whenever $y \in F'$, then (1) F has a least element, and (2) for any part F' of F, if $U(F')$ is non-empty it has a least element (a part of F is a non-empty subset of F). We can also restate this as: any subset that has a strict upper bound has a least strict upper bound. If F is well-ordered, both conditions hold trivially; conversely, consider a non-empty subset F' of F; if it contains the least element of F, then it has a least element. Otherwise, we may consider the set F_1 of all strict lower bounds of F', and the set F_2 of all strict upper bounds of F_1. This is a superset of X thus is nonempty. By assumption it has a least element a, which is a lower bound of F'.

```
Lemma cantor_worder r (* 30 *)
  (ssub := fun E => Zo(substrate r) (fun z => forall x, inc x E -> glt r x z)):
   order r -> nonempty (substrate r) ->
   ( worder r <->
```

```
    ( (has_least r)
     /\ (forall F, sub F (substrate r) -> nonempty F -> nonempty (ssub F) ->
       (has_least (induced_order r (ssub F)) x))).
```

The Anti-Foundation Axiom. Assume $\Omega = \{\Omega\}$. Our definition of ordinals says that Ω is `infinite_o` (of course, Ω has a single element, thus is a finite set; it is not an ordinal since ordinals are irreflexive). One question is: does Ω exist? and if so, is there another solution to $x = \{x\}$? The Foundation Axiom says that there is no such set. Assume $x = \{x\}$ and $y = \{y\}$. By extensionality $x = y$ if and only if they have the same elements; so, for any z, $z \in x \iff z \in y$. But $z \in x$ is the same as $z = x$ and $z \in y$ is the same as $z = y$; hence we get $z = x \iff z = y$, which reduces to $x = y$. This shows that the axiom of extent is useless in such situations.

The Anti-Foundation Axiom says that the solution of such systems is unique. Obviously $x = \mathfrak{P}(x)$ has no solution. So, AFA says something only of "flat" systems of equations. An example of a flat system is: $x = \{y\}$, $y = \{\emptyset, x\}$. Eliminating y gives $x = \{\mathfrak{P}(x)\}$. Similarly $x = \{\mathfrak{P}(\mathfrak{P}(x))\}$ is not flat, but equivalent to a flat system.

```
Lemma afa_ex1 x: x <> \Po x. (* 4 *)
Lemma afa_ex2 a b:  (* 1 *)
  a = singleton b -> b = doubleton emptyset a ->
  a = singleton (\Po a).
Lemma afa_ex2_inv a: a = singleton (\Po a) ->  (* 2 *)
  exists b,  a = singleton b /\ b = doubleton emptyset a.
Lemma afa_ex3 a b c d:   (* 19 *)
  a = singleton b ->
  b = (doubleton emptyset (singleton emptyset)) +s1 c +s1 d ->
  c = doubleton emptyset a -> d = singleton a ->
  a = singleton (\Po (\Po a)).

Lemma afa_ex3_inv a : a = singleton (\Po (\Po a)) -> (* 18 *)
  exists b c d, [/\
    a = singleton b,
    b = (doubleton emptyset (singleton emptyset)) +s1 c +s1 d,
    c = doubleton emptyset a &
    d = singleton a].
```

Recall that X^Y is the set of functional graphs $Y \to X$. The study of $X = X^\emptyset$ is clear: there is a single solution $X = \{\emptyset\}$. Consider now $X = A^X$. This is not a flat system, because if A has two elements, A^X is isomorphic to the power set of X. If fact, take a and b in A. To any element f of X we associate a if $f(f) = b$ and b otherwise. This function g is in X. If $g(g) = b$ then $g(g) = a$, so that $a = b$. Otherwise, the same conclusion holds. It follows that A has at most one element. It is clear that X cannot be empty, so that there is $f \in X$, and if $a = f(f)$, we have $a \in A$. Since A has at most one element, it is a singleton, so that X contains only the constant function f.

Consider now $X = X^X$. The previous discussion says $X = \{x\}$, and x is a functional graph on X such that $x(x) = x$. This last statement is also $x = \{(x, x)\}$.

```
Lemma afa_ex4 x: (* 1 *)
   x = gfunctions emptyset x <-> x = singleton emptyset.
```

```
Lemma afa_ex5 X A: X = gfunctions X A <->   (* 54 *)
  (exists a f, [/\ A = singleton a, X = singleton f & f = singleton (J f a)]).
Lemma afa_ex6 X: X = gfunctions X X <-> (* 4 *)
  (exists2 x, X = singleton x & x = singleton (J x x)).
```

We assume now that pairs are defined according to Kuratowski. Then $(x, x) = \{\{x\}\}$. Now $X = X^X$ becomes $X = \{\{\{X\}\}\}$. This equation is equivalent to a flat system of three equations. Since it is satisfied by Ω, AFA would say that it is equivalent to $X = \Omega$.

The same argument as above applies to $X = \mathscr{F}(X;A)$ (X is the set of functions $X \to A$). Thus $X = \mathscr{F}(X;X)$ means that $X = \{x\}$, where x is the constant function with value x. We can simplify it as $x = \{\{\{\{\{x\}\}\}\}\}$. If follows $X = \{\{\{\{\{X\}\}\}\}\}$. Since Ω is a solution of this equation, AFA would say that $X = \mathscr{F}(X;X)$ is equivalent to $X = \Omega$.

```
Lemma afa_ex7 X: (* 8 *)
  X = gfunctions X X <-> X = singleton (singleton (singleton X)).
Lemma afa_ex8 X: (* 1 *)
  X = singleton X -> X = gfunctions X X.
Lemma afa_ex9 X A: X = functions X A <->  (* 45 *)
  (exists a f, [/\ A = singleton a, X = singleton f & f = Lf(fun _ => a) X A]).
Lemma afa_ex10 X : X = functions X X <-> (* 15 *)
  (exists2 f, X = singleton f & f = triple (singleton (J f f)) X X).
Lemma afa_ex11 X : X = functions X X <-> (* 6 *)
  X = singleton (singleton (singleton (singleton (singleton X)))).
Lemma afa_ex12 X : (* 1 *)
    X = singleton X ->  X = functions X X.
```

Ordinal subtraction and division. Let F be a totally ordered set, A a segment, B its complement; then F is order isomorphic to the ordinal sum of A and B. Assume that F is well-ordered; then A and B are also well-ordered. Assume now that E is a well-ordered set such that $E \le F$. This means that there is an order morphism $f : E \to F$, whose image A is a segment. We deduce: F is isomorphic to the sum of E and some B. This is how Bourbaki defines the ordinal difference.

Let a and b be two ordinals, with $a \le b$. Let C be the complement of a in b, ordered by $\le_{\text{ord}}$; this is a well-order and we can consider its ordinal c. Then $b = a + c$.

Let c be a third ordinal, such that $a < b \cdot c$. In this case, a is an initial segment S_x of the product of b and c. We can consider x as a pair (β, γ) with $\beta \in b$ and $\gamma \in c$. Let y be the pair $(0, \gamma)$. Now if $t = (t_1, t_2)$, $t < x$ is equivalent to either $t_1 < b$ and $t_2 < \gamma$ or $t_1 < \beta$ and $t_2 = \gamma$. The first condition is $t < b \cdot \gamma$. Thus S_x is the ordinal sum of $b \cdot \gamma$ and β. This shows existence of ordinal division.

If x is an ordinal and y is limit, then $x + y$ is limit. Proof: assume $z < x + y$; we want to show that $z + 1 < x + y$. The result is clear if $z \le x$. Otherwise $z = x + t$, with $t < y$, so that $t + 1 < y$, and $x + t + 1 < x + y$. In the main text, we use a simpler argument: if $f(y)$ is $x + y$ then f is a normal OFS, thus maps limit ordinals to limit ordinals.

```
Lemma order_diff_p1 A r
  (r1 := induced_order r A)
  (r2 := induced_order r ((substrate r) -s A)):
  total_order r -> segmentp r A ->
  r \Is order_sum2 r1 r2.   (* 47 *)
Lemma order_diff_p2 r1 r2 f
```

```
      (r3 := induced_order r2 ((substrate r2) -s (Imf f))):
  worder r1 -> worder r2 ->
  order_morphism f r1 r2 -> segmentp r2 (Imf f) ->
  worder r3 /\  r2 \Is order_sum2 r1 r3. (* 24 *)

Lemma order_diff r1 r2 f (r3 := induced_order r2 ((substrate r2) -s (Imf f))):
  worder r1 -> worder r2 ->
  order_morphism f r1 r2 -> segmentp r2 (Imf f) ->
  worder r3 /\  r2 \Is order_sum2 r1 r3.  (* 63 *)

Lemma odiff_pr_alt a b
  (c := ordinal (induced_order (ordinal_o b) (b -s a))):
  a <=o b -> (ordinalp c /\ b = a +o c).  (* 70 *)
Lemma odiff_wrong_alt a b
  (c := ordinal (induced_order (ordinal_o b) (b -s a))):
  b <=o a -> c = \0c.  (* 5 *)
Lemma ord_div_nonzero_b a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  a <o (b *o c)  -> b <> \0o.  (* 3 *)
Lemma ord_div_nonzero_b_bis  a b:
  ordinalp a -> \0o <o b ->
  exists2 c, ordinalp c & a <o (b *o c). (* 3 *)
Lemma odivision_exists_alt a b c:
  ordinalp a -> ordinalp b -> ordinalp c ->
  a <o (b *o c) ->
  exists q r, odiv_pr1 a b c q r. (* 178 *)
Lemma osum_limit_alt x y: ordinalp x -> limit_ordinal y ->
  limit_ordinal (x +o y).  (* 17 *)
```

13.2 Section 1

1. Let E be an ordered set in which there exists at least one pair of distinct comparable elements. Show that, if $R\{x, y\}$ denotes the relation "$x \in E$ and $y \in E$ and $x < y$", then R satisfies the first two conditions of no. 1 but not the third.

Note. We must show that the relation is antisymmetric, transitive and not reflexive. This is immediate. Here r denotes the ordering on E.

```
Lemma Exercise1_1 r (E:= substrate r) (* 5 *)
     (R := fun x y => [/\ inc x E, inc y E & glt r x y]) :
    order r ->  (exists x y, x <> y /\ related r x y) ->
    [/\ transitive_r R, antisymmetric_r R & ~(reflexive_rr R)].
```

---

2. (a) Let E be a preordered set and let $S\{x, y\}$ be an equivalence relation on E. Let $R\{X, Y\}$ denote the relation "$X \in E/S$ and $Y \in E/S$ and for each each $x \in X$ there exists $y \in Y$ such that $x \le y$". Show that R is a preorder relation on E/S, called the *quotient* by S of the relation $x \le y$. The quotient E/S, endowed with this preorder relation, is called (by abuse of language; cf. Chapter IV, § 2, no. 6) the quotient by S of the preordered set E.

(b) Let ϕ be the canonical mapping of E onto E/S. Show that if g is a mapping of the preordered quotient set E/S into a preordered set F such that $g \circ \phi$ is an increasing mapping, then g is an increasing mapping. The mapping ϕ is increasing if and only if S satisfies the following condition

(C) the relations $x \le y$ and $x \equiv x' \pmod{S}$ in E imply that there exists $y' \in E$ such that $y \equiv y' \pmod{S}$ and $x' \le y'$.

If this condition is satisfied, the equivalence relation S is said to be *weakly compatible* (in x and y) with the preorder relation $x \le y$. Every equivalence relation S which is *compatible* (in x) with the preorder relation $x \le y$ (Chapter II, § 6, no. 3) is *a fortiori weakly* compatible (in x and y) with this relation.

(c) Let E_1 and E_2 be two preordered sets. Show that if S_1 is the equivalence relation $\mathrm{pr}_1 z = \mathrm{pr}_1 z'$ on $E_1 \times E_2$, then S_1 is weakly compatible in z and t with the product preorder relation $z \le t$ on $E_1 \times E_2$ (but is not usually compatible with this relation in z or t separately); moreover if ϕ_1 is the canonical mapping of $E_1 \times E_2$ onto $(E_1 \times E_2)/S_1$, and if $\mathrm{pr}_1 = f_1 \circ \phi_1$ is the canonical decomposition of pr_1 with respect to the equivalence relation S_1, then f_1 is an isomorphism of $(E_1 \times E_2)/S_1$ into E_1.

(d) With the hypothesis of (a), suppose that E is an *ordered* set and that the following condition is satisfied:

(C') The relations $x \le y \le z$ and $x \equiv z \pmod{S}$ in E imply $x \equiv y \pmod{S}$.

Show that $R\{x, y\}$ is then an *order* relation between X and Y on E/S.

(e) Give an example of a totally ordered set E with four elements and an equivalence relation S on E such that neither of the conditions (C) and (C') is satisfied, but such that E/S is an ordered set.

(f) Let E be an ordered set, let f be an increasing mapping of E into an ordered set F, and let $S\{x, y\}$ be the equivalence relation $f(x) = f(y)$ on E. Then the condition (C') is satisfied. Moreover the condition (C) is satisfied if and only if the relations $x \le y$ and $f(x) = f(x')$ imply that there exists $y' \in E$ such that $x' \le y'$ and $f(y) = f(y')$. Let $f = g \circ \phi$ be the canonical decomposition of f. Then g is an isomorphism of E/S onto $f(E)$ if and only if this condition is satisfied and, in addition, the relation $f(x) \le f(y)$ implies that there exists x', y' such that $f(x) = f(x')$, $f(y) = f(y')$, and $x' \le y'$.

Note: Given an equivalence relation and a structure (here, the structure of preordered set), it is sometimes possible to endow the quotient with this structure. This is explained in Chapter IV, § 2, no. 6.

In (c) we must assume $E_2 \neq \emptyset$, since otherwise $(E_1 \times E_2)/S_1$ is empty. Any set with at least three elements satisfies (e).

Solution. We start by introducing the two conditions (C) and (C'), the two conditions of point (f), and the strong compatibility conditions (in x or y). The variable r will denote the preorder (so that E is the substrate of r) and s will denote the equivalence. The implicit assumption is that r and s have the same substrate.

```
Definition ne_substrate r := nonempty (substrate r).

Definition Ex1_2_hC r s:=
  forall x y x', gle r x y -> related s x x' -> exists2 y',
    related s y y' & gle r x' y'.
Definition Ex1_2_hC' r s:=
  forall x y z, gle r x y -> gle r y z -> related s x z -> related s x y.
Definition Ex1_2_hD r f :=
   forall x y x', gle r x y -> inc x' (source f) -> Vf f x = Vf f x' ->
   exists y', [/\ inc y' (source f), gle r x' y' & Vf f y = Vf f y'].
```

```
Definition Ex1_2_hD' r r' f :=
  forall x y, inc x (source f) -> inc y (source f) ->
    gle r' (Vf f x) (Vf f y) -> exists x' y',
     [/\ Vf f x = Vf f x', Vf f y = y' Vf f & gle r x' y'].
Definition Ex1_2_strong_l r s:=
  (forall x x' y,  gle r x y -> related s x x' -> gle r x' y).
Definition Ex1_2_strong_r r s:=
   (forall x y y',  gle r x y -> related s y y' -> gle r x y').
Definition preorder_quo_axioms r s:=
  [/\ preorder r, equivalence s &  substrate s = substrate r].
Definition weak_order_compatibility r s:=
  preorder_quo_axioms r s /\ Ex1_2_hC r s.
```

We shall also define the notion of increasing function and isomorphism for two preorder relations.

```
Definition increasing_pre f r r':=
 [/\ preorder r, preorder r', function_prop f (substrate r) (substrate r')
  & fincr_prop f r r'].
Definition preorder_isomorphism f r r' :=
 [/\ preorder r, preorder r', bijection_prop f (substrate r) (substrate r')
  & fiso_prop f r r'].
```

(a) We define here the quotient preorder relation and the quotient preorder. We show that this is a preorder on the quotient.

```
Definition quotient_order_r r s X Y :=
  [/\ inc X (quotient s), inc Y (quotient s) &
  forall x, inc x X -> exists2 y, inc y Y & gle r x y].
Definition quotient_order r s := graph_on (quotient_order_r r s) (quotient s).

Lemma Exercise1_2a r s: (* 6 *)
  preorder_quo_axioms r s -> preorder_r (quotient_order_r r s).
Lemma quotient_orderP r s x y: (* 2 *)
  related (quotient_order r s) x y <-> quotient_order_r r s x y.
Lemma quotient_is_preorder r s: (* 1 *)
  preorder_quo_axioms r s ->  preorder (quotient_order r s).
Lemma substrate_quotient_order r s: (* 7 *)
  preorder_quo_axioms r s ->  substrate (quotient_order r s) = quotient s.
```

(b) Let ϕ be the canonical projection $\mathrm{E} \to \mathrm{E}/\mathrm{S}$. If $g \circ \phi$ is increasing so is g. Strong compatibility (in x) implies weak compatibility, which is equivalent to (C).

```
Lemma Exercise1_2b1 r s g r': (* 12 *)
  preorder_quo_axioms r s ->
  function g -> quotient s = source g ->
  increasing_pre (g \co (canon_proj s)) r r' ->
  increasing_pre g (quotient_order r s) r'.
Lemma strong_order_compatibility r s: (* 3 *)
  preorder_quo_axioms r s -> Ex1_2_strong_l r s ->
  weak_order_compatibility r s.
Lemma compatibility_proj_increasing r s: (* 15 *)
  preorder_quo_axioms r s ->
  (weak_order_compatibility r s <->
   increasing_pre (canon_proj s) r (quotient_order r s)).
```

(c) We consider here the equivalence associated to pr_1 in a product. The product preorder is weakly compatible with this equivalence.

```
Lemma Exercise1_2c1 r1 r2:  (* 11 *)
  preorder r1 -> preorder r2 ->
  weak_order_compatibility (order_product2 r1 r2)
  (first_proj_eq (substrate r1) (substrate r2)).
```

If S_1 is compatible with $\leq$, then if $x \leq x$ and if x and y are related by S_1, then x and y are comparable (we have $x \leq y$ if S_1 is compatible in x). Consider the case of the product order and the first projection. Then, for all a, b and c, if $x = (a, b)$ and $y = (a, c)$, then x and y are comparable, so that b and c are comparable. This means that all elements of the second factor are related, and this is generally false.

```
Lemma Exercise1_2c2 r1 r2 (* 18 *)
  (p :=first_proj_eq (substrate r1) (substrate r2)) :
  preorder r1 -> preorder r2 -> ne_substrate r1 ->
  (Ex1_2_strong_l (order_product2 r1 r2) p \/
   Ex1_2_strong_r (order_product2 r1 r2) p) ->
  r2 = coarse (substrate r2).
```

We show that $\mathrm{pr}_1 = f_1 \circ \phi_1$ implies that f_1 is an isomorphism. If $f_1(x) = f_1(y)$, where x and y are the classes of x' and y', then $\mathrm{pr}_1 x' = \mathrm{pr}_1 y'$, hence $x' \equiv y'$ and $x = y$, so that f_1 is injective. With the same notations, $f_1(x) \leq f_1(y)$ if and only if $\mathrm{pr}_1 x' \leq \mathrm{pr}_1 y'$.

```
Lemma Exercise1_2c4 r1 r2 f:  (* 52 *)
   (s := first_proj_eq (substrate r1) (substrate r2))
   (r:= order_product2 r1 r2) :
  function_prop f (quotient s) (substrate r1) ->
  preorder r1 -> preorder r2 -> ne_substrate r2 ->
  f \co (canon_proj s)= first_proj (product (substrate r1) (substrate r2)) ->
  preorder_isomorphism f (quotient_order r s) r1.
```

(d) If (C') holds, the quotient of an order is an order.

```
Lemma Exercise1_2d r s: (* 14 *)
  is_equivalence s -> order r -> substrate s = substrate r ->
  Ex1_2_hC' r s ->
  order (quotient_order r s).
```

(e) Assume that E is a totally ordered finite set, and consider two classes X and Y; they have a greatest element x and y. The condition $x \leq y$ is equivalent to $X \leq Y$, so that the quotient is totally ordered. Assume $a < b < c$, S is such that a and c are related by S, but no other pair of distinct elements are related. Then (C$'$) is false. In (C), take a, b and c for x, y and y'. Since $y \equiv y'$ implies $y = y'$, condition (C) is false.

Any totally ordered finite set with at least three elements gives thus a counter-example.

```
Lemma Exercise1_2e1 r s: (* 31 *)
  equivalence s -> total_order r -> substrate s = substrate r ->
  finite_set (substrate r) ->
  total_order (quotient_order r s).
```

```
Lemma Exercise1_2e2 r a b c (E:= substrate r)  (* 33 *)
  (s := (diagonal E) \cup (doubleton (J a c) (J c a))):
  order r -> glt r a b -> glt r b c ->
  [/\ equivalence s, substrate s = substrate r,
  ~ ( weak_order_compatibility r s) &
  ~ ( Ex1_2_hC' r s)].
Lemma Exercise1_2e4 r:  total_order r -> (* 6 *)
  \3c <=c cardinal (substrate r) ->
  exists a b c,  glt r a b /\ glt r b c.

Lemma Exercise1_2e5 r:  (* 4 *)
  total_order r -> finite_set (substrate r) ->
  \3c <=c cardinal (substrate r) ->
  exists s,
  [/\ equivalence s, substrate s = substrate r,
    total_order (quotient_order r s),
  ~ ( weak_order_compatibility r s) &
  ~ ( Ex1_2_hC' r s) ].
```

(f) Let $f : \mathrm{E} \to \mathrm{F}$ be increasing, and S the equivalence associated to f. Then (C') holds and (C) is equivalent to (D).

```
Lemma Exercise1_2f1 r r' f: (* 5 *)
  increasing_fun f r r' ->
  Ex1_2_hC' r (equivalence_associated f).

Lemma Exercise1_2f2 r r' f: (* 15 *)
  increasing_fun f r r' ->
  (weak_order_compatibility r (equivalence_associated f) <->
   (Ex1_2_hD r f)).
```

The next point is straightforward, but a bit longish. Assume $f = g \circ \phi$, and f increasing. We must show that g is an isomorphism if and only if (D) and (D') hold. If $\mathrm{X} \in \mathrm{E}/\mathrm{S}$ we denote by x the representative of X. We have then $g(\mathrm{X}) = f(x)$. The equivalence relation is defined by: $a \in \mathrm{X}$ if and only if $f(x) = f(a)$, $a \in \mathrm{E}$ and $x \in \mathrm{E}$.

If x and y are in E, X and Y are their equivalence classes, then $f(x) \le f(y)$ implies $g(\mathrm{X}) \le g(\mathrm{Y})$. Assume that g is a morphism; we deduce $\mathrm{X} \le \mathrm{Y}$. Expanding this, we get: if x' is in the class of x, there exists y' such that $f(y) = f(y')$ and $x' \le y'$. The conclusion follows easily.

```
Lemma Exercise1_2f3 r r' f g (s := (equivalence_associated f)): (* 62 *)
  increasing_fun f r r' ->
  composable g (canon_proj s) -> f = compose g (canon_proj s) ->
  (order_morphism g (quotient_order r s) r'
    <-> (Ex1_2_hD r f  /\ Ex1_2_hD' r r' f)).
```

---

3. Let I be an ordered set and let $(\mathrm{E}_\iota)_{\iota\in\mathrm{I}}$ be a family of non-empty ordered sets indexed by I.

(a) Let F be the *sum* (Chapter II, § 4, no. 8) of the family $(\mathrm{E}_\iota)_{\iota\in\mathrm{I}}$; for each $x \in \mathrm{F}$, let $\lambda(x)$ be the index ι such that $x \in \mathrm{E}_\iota$; and let G be the graph consisting of all the pairs $(x, y) \in \mathrm{F}\times\mathrm{F}$ such that either $\lambda(x) < \lambda(y)$ or else $\lambda(x) = \lambda(y)$ and $x \le y$ in $\mathrm{E}_{\lambda(x)}$. Show that G is the graph of an ordering on F. The set F endowed with this ordering is called the *ordinal sum* of the family

$(E_\iota)_{\iota\in I}$ (relative to the ordering on I) and is denoted $\sum_{\iota\in I} E_\iota$. Show that the equivalence relation corresponding to the partition $(E_\iota)_{\iota\in I}$ of F satisfies conditions (C) and (C') of Exercise 2, and that the quotient ordered set (Exercise 2) is canonically isomorphic to I.

(b) If the set I is the ordinal sum of a family $(J_\lambda)_{\lambda\in L}$ of ordered sets, where L is an ordered set, show that the ordered set $\sum_{\iota\in I} E_\iota$ is canonically isomorphic to the ordinal sum $\sum_{\lambda\in L} F_\lambda$, where $F_\lambda = \sum_{\iota\in I_\lambda} E_\iota$ ("associativity" of the ordinal sum). If I is the linearly ordered set $\{1,2\}$, we write $E_1 + E_2$ for the ordinal sum of E_1 and E_2. Show that $E_2 + E_1$ and $E_1 + E_2$ are not necessarily isomorphic.

(c) An ordinal sum $\sum_{\iota\in I} E_\iota$ is right directed if and only if I is right directed and E_ω is right directed for each maximal element ω of I.

(d) An ordinal sum $\sum_{\iota\in I} E_\iota$ is totally ordered if and only if I and each E_ι is totally ordered.

(e) An ordinal sum $\sum_{\iota\in I} E_\iota$ is a lattice if and only if the following conditions are satisfied:

(I) The set I is a lattice, and for each pair (λ,μ) of non-comparable indices in I, $E_{\sup(\lambda,\mu)}$ (resp. $E_{\inf(\lambda,\mu)}$) has a least (resp. greatest) element.

(II) For each $\alpha \in I$ and each pair (x, y) of elements of E_α such that the set $\{x, y\}$ is bounded above (resp. bounded below) in E_α, the set $\{x, y\}$ has a least upper bound (resp. greatest lower bound) in E_α.

(III) For each $\alpha \in I$ such that E_α contains a set of two elements which has no upper bound (resp. no lower bound) in E_α, the set of indices $\lambda \in I$ such that $\lambda > \alpha$ (resp. $\lambda < \alpha$) has a least element (resp. a greatest element) β, and E_β has a least element (resp. greatest element).

Note. The ordinal sum has been defined in section 11.1 without the assumption E_ι non-empty. This allows us to compute $0 + x$.

Solution. In what follows, r will denote an ordering (and I its substrate). The quantity g will denote a family of orderings (indexed by I). The substrate of the i-th term will be E_i. The variable f may denote the family $(E_i)_{i\in I}$. The disjoint union F of this family is the set of all (x, i), where $i \in I$ and $x \in E_i$. There is a canonical injection $E_i \to F$, whose image is $\bar{E}_i = E_i \times \{i\}$. By abuse of language, we may identify $\bar{E}_i$ and E_i. Note that, if no E_i is empty, then the family $(\bar{E}_i)_i$ defines a partition of F.

(a) If (H1) holds, lemma `orsum_osr` says that G is an ordering on F. Note that (H2) says that g is a graph (this is a consequence of (H1), that says that g is a functional graph).

```
Section Exercise1_3a.
Variables r g: Set.
Definition E13_F:= order_sum r g.
Definition E13_sF:= sum_of_substrates g.
Definition E13_lam := second_proj E13_sF.
Definition E13_S:= equivalence_associated (second_proj E13_sF).
Definition E13_H1:= orsum_ax r g.
Definition E13_H2:= sgraph g /\ allf g ne_substrate.
```

Note that λ is nothing else than pr_2; its image is the set of all i such that E_i is non-empty. Under assumption (H2), it is I. Obviously pr_2 is increasing.

```
Lemma Exercise1_3a0: function E13_lam. (* 1 *)
Lemma Exercise1_3a1: sgraph E13_sF. (* 2 *)
Lemma Exercise1_3a2: surjection E13_lam. (* 3 *)
Lemma Exercise1_3a3: E13_H2 -> domain g = target E13_lam. (* 5 *)
Lemma Exercise1_3a3': substrate E13_S = E13_sF. (* 1 *)
Lemma Exercise1_3a3'': substrate E13_S = source E13_lam. (* 1 *)
Lemma Exercise1_3a4: (* 9 *)
   E13_H1 -> E13_H2 ->  increasing_fun E13_lam E13_F r.
```

Let f be the family $(\mathrm{E}_i)_{i\in\mathrm{I}}$, and T its range. Then f is a functional graph, that associates E_i to i. We may consider this a function $\tilde{f}:\mathrm{I}\to\mathrm{T}$. We then consider the equivalence relation S_1 on the disjoint union of f such that x and y are related iff there is i such that $x\in\tilde{f}(i)$ and $y\in\tilde{f}(i)$. This simplifies to: x and y are in F and $\mathrm{pr}_2 x = \mathrm{pr}_2 y$. This says that S_1 is S, the equivalence associated to pr_2 on F.

```
Definition disjointU_function f :=
  triple (domain f)(range (disjointU_fam f))(disjointU_fam f).

Lemma disjointU_function_pr f: (* 2 *)
  function (disjointU_function f) /\
  graph (disjointU_function f) =  (disjointU_fam f).
Lemma Exercise1_3a5P x y (f :=  (fam_of_substrates g)):  (* 13 *)
  related (partition_relation (disjointU_function f) (disjointU f)) x y
  <-> [/\ inc x E13_sF, inc y E13_sF & Q x = Q y].
Lemma Exercise1_3a6P x y:  (* 4 *)
  related E13_S x y <-> [/\ inc x E13_sF, inc y E13_sF & Q x = Q y].
```

The classes of S are the sets $\mathrm{E}_i\times\{i\}$. If ϕ is the canonical projection of F on F/S, the function h induced by pr_2 by passing on the quotient is the unique function satisfying $\mathrm{pr}_2 = h\circ\phi$ (canonical decomposition). We can restate it as: an element X of the quotient is of the form $i\mapsto \mathrm{E}_i\times\{i\}$, and $h(\mathrm{X}) = i$. Note that h is bijective

```
Lemma Exercise1_3a7: equivalence E13_S. (* 1 *)
Lemma indexed_p2 a b c: inc a (b *s1 c) -> Q a = c. (* 1 *)
Lemma Exercise1_3a8P a: E13_H2 ->  (* 32 *)
  (inc a (quotient E13_S) <-> exists2 i,
    inc i (domain g) & a = (Vg (fam_of_substrates g) i) *s1 i).
Lemma Exercise1_3a9: function (fun_on_quotient E13_S E13_lam). (* 2 *)
Lemma Exercise1_3a10:  (* 3 *)
  (fun_on_quotient E13_S E13_lam) \coP (canon_proj E13_S).
Lemma Exercise1_3a11: (* 1 *)
  E13_lam = (fun_on_quotient E13_S E13_lam) \co (canon_proj E13_S).
Lemma Exercise1_3a12 x:  E13_H2 -> (* 9 *)
  inc x (quotient E13_S) -> exists i,
    [/\ inc i (domain g), x = (Vg (fam_of_substrates g) i) *s1 i &
    Vf (fun_on_quotient E13_S E13_lam) x = i].
Lemma Exercise1_3a13: E13_H2 -> (* 15 *)
   bijection (fun_on_quotient E13_S E13_lam).
```

All four conditions of Exercise 2 are satisfied. This implies that h is an order morphism; since it is bijective, it is an order isomorphism.

```
Lemma Exercise1_3a14: E13_H1 ->  E13_H2 -> (* 36 *)
  [/\ Ex1_2_hC E13_F E13_S, Ex1_2_hC' E13_F E13_S,
```

```
  Ex1_2_hD E13_F E13_lam & Ex1_2_hD' E13_F r E13_lam].
Lemma Exercise1_3a15: E13_H1 ->  E13_H2 -> (* 10 *)
  order_isomorphism  (fun_on_quotient E13_S E13_lam)
  (quotient_order E13_F E13_S) r.
End Exercise1_3a.
```

(b) The associativity formula has been shown in the main text as `orsum_assoc_iso`. Non-commutativity is a consequence of $\omega+1 \neq 1+\omega$. More generally, if E_1+E_2 has a greatest element, and if E_2 is not empty, this element projects onto a greatest element of E_2. An isomorphism maps a greatest element onto a greatest element. Now 1 has a greatest element and ω does not. Instead of ω we can chose a set with two elements ordered with the trivial ordering (two distinct elements are non-comparable).

```
Lemma orsum2_greatest r r' x: (* 9 *)
  order r -> order r' -> ne_substrate r' ->
  greatest (order_sum2 r r') x -> greatest r' (P x).
Lemma orsum2_greatest' r r' x: order r -> order r' -> (* 7 *)
  greatest r' x -> greatest (order_sum2 r r') (J x C1).
Lemma image_of_greatest r r' f x: (* 5 *)
  order_isomorphism f r r' -> greatest r x ->
  greatest r' (Vf f x).
Lemma orsum2_nc: exists r r', (* 15 *)
  [/\ order r, order r' & ~ ( (order_sum2 r r') \Is (order_sum2 r' r))].
```

(c) We assume here no E_i empty. Assume the ordinal sum right directed. For every pair of indices i and j, there is $x \in E_i$ and $y \in E_j$, identified with $\bar{x}$ and $\bar{y}$ in the ordinal sum. Let $\bar{z}$ be an upper bound, and assume $\bar{z} \in \bar{E}_k$. Then k is an upper bound for i and j. This shows that I is right directed. Assume now i maximal, x and y in E_i. If $\bar{z}$ is as above, we get $k=i$. Thus $\bar{z}$ is the image of some z, which is an upper bound of x and y in E_i.

Conversely, assume I right directed, and E_k right directed whenever k is maximal. Let $\bar{x} \in E_i$ and $\bar{y} \in E_j$ be in the disjoint union, and k an upper bound of i and j. If k is not maximal, there is l with $k<l$, and any element $z \in E_l$ gives $\bar{z}$, a strict upper bound of $\bar{x}$ and $\bar{y}$. If k is one of i and j, but not equal to both, then i and j are comparable and distinct, so that $\bar{x}$ and $\bar{y}$ are comparable and distinct, thus have an upper bound. If $\bar{x}$ and $\bar{y}$ come from x and y in E_k, any upper bound z of x and y yields an upper bound $\bar{z}$ of $\bar{x}$ and $\bar{y}$.

```
Definition orsum_ax2 g:= allf g ne_substrate.

Section Exercise13b.
Variables r g: Set.
Hypothesis oa: orsum_ax r g.
Hypothesis oa2: orsum_ax2 g.

Lemma orsum_pr0: (* 4 *)
  forall i, inc i (substrate r) ->
  exists2 y, inc y (Vg i (fam_of_substrates g) i) &
    inc (J y i) (sum_of_substrates g).
Lemma orsum_pr1: (* 2 *)
  forall i, inc i (domain g) ->
   exists2 y, inc y (Vg (fam_of_substrates g) i) &
    inc (J y i) (substrate (order_sum r g)).
Lemma orsum_directed:  (* 51 *)
  (right_directed (order_sum r g) <-> (right_directed r /\
    forall i, maximal r i -> right_directed (Vg g i))).
```

(d) If I and each E_i are totally ordered, then the ordinal sum is totally ordered; conversely if the sum is totally ordered so is each E_i. If moreover no E_i is empty, then I is totally ordered.

```
Lemma orsum_total1:  (* 12 *)
  total_order (order_sum r g) -> (total_order r /\
    forall i, inc i (domain g) -> total_order (Vg g i)).
Lemma orsum_total2:  (* 12 *)
  total_order r ->
  (forall i, inc i (domain g) -> total_order (Vg g i)) ->
  total_order (order_sum r g).
```

(e) We show here that if the sum is a complete lattice, then (I) holds. In fact, consider two indices i and j, $x \in \mathrm{E}_i$, $y \in \mathrm{E}_j$ and $\bar{z} = \sup(\bar{x}, \bar{y})$. Let k be the index of $\bar{z}$; this is easily seen to be the least upper bound of i and j. This says that I is a lattice. Assume i and j non-comparable. Any t in E_k gives an upper bound $\bar{t}$ of $\bar{x}$ and $\bar{y}$; it follows that E_k has a least element. The same argument says that $\mathrm{E}_{\inf(i,j)}$ has a greatest element.

```
Lemma orsum_g1 i x i' x', (* 2 *)
  inc (J i x) (sum_of_substrates g) -> inc (J i' x') (sum_of_substrates g) ->
  gle r x x' -> x <> x' ->
  gle (order_sum r g) (J i x) (J i' x').
Lemma orsum_lattice1:  (* 37 *)
  lattice (order_sum r g) -> lattice r.
Let orsum_lattice_H1:= forall i j, inc i (domain g) -> inc j (domain g) ->
    [\/ gle r i j, gle r j i | (exists u v,
      least (Vg g (sup r i j)) u /\
      greatest (Vg g (inf r i j)) v)].
Let orsum_lattice_H2 := forall i x y t,
  inc i (domain g) -> gle (Vg g i) x t -> gle (Vg g i) y t ->
    has_supremum (Vg g i) (doubleton x y).
Let orsum_lattice_H3 := forall i x y t,
  inc i (domain g) -> gle (Vg g i) t x -> gle (Vg g i) t y ->
    has_infimum (Vg g i) (doubleton x y).
Let orsum_lattice_H4 := forall i x y,
  inc i (domain g) ->  inc x (Vg (fam_of_substrates g) i) ->
  inc y (Vg (fam_of_substrates g) i) ->
  (forall t, inc t (Vg  (fam_of_substrates g) i) ->
     ~ (gle (Vg g i) x t /\ gle (Vg g i) y t)) ->
  exists j, [/\ inc j (domain g),
    least (induced_order r (Zo (domain g) (fun k=> glt r i k))) j &
    has_least  (Vg g j)].
Let orsum_lattice_H5 := forall i x y,
  inc i (domain g) ->  inc x (Vg (fam_of_substrates g) i) ->
  inc y (Vg (fam_of_substrates g) i) ->
  (forall t, inc t (Vg (fam_of_substrates g) i) ->
     ~ (gle (Vg g i) t x /\ gle (Vg g i) t y)) ->
  exists j, [/\ inc j (domain g),
    greatest (induced_order r (Zo (domain g) (fun k=> glt r k i))) j &
    has_greatest (Vg g j)].
Lemma orsum_lattice2: (* 55 *)
  lattice (order_sum r g) -> orsum_lattice_H1.
```

We show here (II). Assume $x \in \mathrm{E}_i$ and $y \in \mathrm{E}_i$, and t is an upper bound. Then $\bar{t}$ is an upper bound of $\bar{x}$ and $\bar{y}$. It follows that the index of the least upper bound $\bar{z}$ is i, so that z is the upper bound of x and y in E_i.

```
Lemma orsum_lattice3: (* 34 *)
  lattice (order_sum r g) -> orsum_lattice_H2.
Lemma orsum_lattice4: (* 34 *)
  lattice (order_sum r g) -> orsum_lattice_H3.
```

We show here (III). Assume $x \in \mathrm{E}_i$ and $y \in \mathrm{E}_i$, these elements have no upper bound. Then the least upper bound $\bar{z}$ is in some k such that there is no index between i and k, and moreover z is the least element of E_k.

```
Lemma orsum_lattice5: (* 39 *)
  lattice (order_sum r g) -> orsum_lattice_H4.
Lemma orsum_lattice6: (* 39 *)
  lattice (order_sum r g) -> orsum_lattice_H5.
```

The converse is straightforward.

```
Lemma orsum_lattice: (* 163 *)
  (lattice (order_sum r g) <->
  ((lattice r) /\
   [/\ orsum_lattice_H1, orsum_lattice_H2, orsum_lattice_H3, orsum_lattice_H4
    & orsum_lattice_H5])).

End Exercise13b.
```

---

4. *Let E be an ordered set, and let $(\mathrm{E}_\iota)_{\iota\in\mathrm{I}}$ be the partition of E formed by the connected components of E (Chapter II, § 6, Exercise 10) with respect to the reflexive and symmetric relation "either $x = y$ or x and y are not comparable".

(a) Show that if $\iota \neq \kappa$ and if $x \in \mathrm{E}_\iota$ and $y \in \mathrm{E}_\kappa$, then x, y are comparable; and that if, for example, $x \le y$, $y' \in \mathrm{E}_\kappa$, and if $y' \neq y$, then also $x \le y'$ (use the fact that there exists no partition of E_κ into two sets A and B such that every element of A is comparable with every element of B).

(b) Deduce from (a) that the equivalence relation S corresponding to the partition (E_ι) of E is compatible (in x and y) with the order relation $x \le y$ on E, and that the quotient ordered set E/S (Exercise 2) is totally ordered.

(c) What are the connected components of an ordered set $\mathrm{E} = \mathrm{F} \times \mathrm{G}$ which is the product of two totally ordered sets?$_*$

Note. Let R be a reflexive and symmetric relation on a set E. Let S be the relation (between x and y) defined by: there exists a finite sequence x_1, x_2, up to x_n such that $x = x_1$, $y = x_n$ and each x_i is related to x_{i+1}. The stars surrounding the Exercise indicate that it depends on results not yet established (i.e., properties of integers). The relation S is an equivalence on E and its classes are called the connected components. The partition formed by the connected components is the partition formed by the classes; there is no need to introduce the set I and the functional graph $i \mapsto \mathrm{E}_i$.

Solution. Let r be the ordering, E its substrate, R the relation "either $x = y$ or x and y are not comparable" and S the equivalence associated; the substrate of S is E. The connected component of x for R is the class of x. If x and y are related by R they are also related by S.

```
Definition not_comp_rel r := fun x y =>
  [/\ inc x (substrate r), inc y (substrate r) &
  (x = y \/ ~ (ocomparable r x y))].

Definition ncr_equiv r :=
  chain_equivalence (not_comp_rel r) (substrate r).
Definition ncr_component r :=
  connected_comp (not_comp_rel r) (substrate r).

Lemma ncr_properties r: order r ->  (* 12 *)
 [/\ equivalence (ncr_equiv r),
  (substrate  (ncr_equiv r) =  substrate r),
  (forall x, inc x (substrate r) -> class(ncr_equiv r) x = ncr_component r x) &
  (forall x y, not_comp_rel r x y -> related (ncr_equiv r) x y)].
```

(a) We have to prove "if $i \neq k$ and if $x \in \mathrm{E}_i$ and $y \in \mathrm{E}_k$, then x, y are comparable". This can be restated as: if x and y are in E and not in the same class (mod S), then they are comparable. A simpler form is: two elements x and y are in the same class or comparable. This is equivalent to $\mathrm{R} \implies \mathrm{S}$.

```
Lemma Exercise1_4a1 r x y: order r -> (* 5 *)
  inc x (substrate r) -> inc y (substrate r) ->
   ocomparable r x y \/ class (ncr_equiv r) x = class (ncr_equiv r) y.
```

Let C be a class, $\mathrm{C} = \mathrm{A} \cup \mathrm{B}$. Assume every element of A is comparable to every element of B. Let $x \in \mathrm{A}$ and $y \in \mathrm{B}$. Consider a chain $(x_i)_i$ linking x to y, and the least i such that $x_{i+1} \in \mathrm{B}$. Then x_i is in A, thus comparable to x_{i+1}; since it is related to x_{i+1} by R it must be equal to x_{i+1}, so that x_i is in B. Thus A is empty, B is empty, or $\mathrm{A} \cap \mathrm{B}$ is non-empty.

Fix z outside C; let A the set of all $x \in \mathrm{C}$ such that $x \le z$, B the set of $y \in \mathrm{C}$ such that $z \le y$. We have shown $\mathrm{C} = \mathrm{A} \cup \mathrm{B}$. Any element x of A is comparable to any element y of B (in fact $x < z < y$), so that $\mathrm{A} \cap \mathrm{B} = \emptyset$. It follows: one of A and B must be empty.

Assume $y \sim y'$ and $z \le y$. Assume $z \sim y$ is false. If we take for C the class of y, it follows that y is in B, so that A is empty, and $y' \in \mathrm{B}$, thus $z \le y'$.

```
Lemma Exercise1_4a2 r y: (* 25 *)
  order r -> inc y (substrate r) ->
  forall a b, a \cup b = class (ncr_equiv r) y ->
    (forall u v, inc u a -> inc v b -> ocomparable r u v) ->
     [\/ a = emptyset, b = emptyset | nonempty (a \cap b)].
Lemma Exercise1_4a3 r x y y': order r -> (* 29 *)
  inc x (substrate r) -> related (ncr_equiv r) y y' -> gle r x y ->
  related (ncr_equiv r) x y \/ gle r x y'.
```

(b) Let's show the compatibility of the equivalence and the order. We must show that if x and x' are in the same class, if y and y' are in the same class, then $x \le y$ is equivalent to $x' \le y'$. The assumption is that the classes are distinct. We know that $x' \le y'$, $y' \le x'$ or $\mathrm{R}(x', y')$. The first possibility is the desired result, the last one says that the classes are the same. We have shown that $y' \le x'$ implies $y' \le x$ and $x \le y$ implies $x \le y'$. Thus, the second possibility says $x = y'$, absurd.

```
Lemma Exercise1_4b1 r x y x' y': order r ->  (* 21 *)
  related (ncr_equiv r) x x' -> related (ncr_equiv r) y y' ->
```

```
class (ncr_equiv r) x <> class (ncr_equiv r) y ->
gle r x y -> gle r x' y'.
```

The quotient order is an order, according to condition (C'). To show it, assume $x \le y \le z$, x and z in the same class. If x and y are not in the same class, then $x \le y$ implies $z \le y$. This says $y = z$, absurd. The quotient order is total, since, given two distinct classes, the representatives are comparable. Now, if $x \in \mathrm{X}$ and $y \in \mathrm{Y}$, the compatibility condition says $\mathrm{X} \le \mathrm{Y}$ in the quotient if and only if $x \le y$ in the substrate.

```
Lemma Exercise1_4b r: order r -> (* 29 *)
  total_order(quotient_order r (ncr_equiv r)).
```

(c) Consider a product of two totally ordered sets $\mathrm{E} = \mathrm{F} \times \mathrm{G}$. Let's determine the set of components. We have $x \le y$ if and only if $\mathrm{pr}_1 x \le \mathrm{pr}_1 y$ and $\mathrm{pr}_2 x \le \mathrm{pr}_2 y$. We exclude the case where the sets are empty. If G is a singleton, then $\mathrm{pr}_2 x \le \mathrm{pr}_2 y$ is always true, and the product is totally ordered. The set of components is then isomorphic to E. If a and a' are the least elements of F and G, $l = (a, a')$, then l is the least element of E, and $\{l\}$ is a component. In the same fashion, if g is the greatest element of E then $\{g\}$ is a component.

We assume now that F has at least two elements b and c, G has at least two elements b' and c'. We may assume $b < c$ and $b' < c'$. We pretend that the class C of (b, c') contains all elements of E, with the possible exception of l and g. It contains obviously (c, b'). Let $y = (b, b')$, and assume that $y \neq l$. Then $y \in \mathrm{C}$. In fact, if $a < b$, then $\mathrm{R}((a, c'), (b, b'))$ and $\mathrm{R}((a, c'), (c, b'))$, so that (b, b') and (c, b') are in the same class. On the other hand, if $a' < b'$, then $\mathrm{R}((c, a'), (b, b'))$ and $\mathrm{R}((c, a'), (b, c'))$, so that (b, b') and (b, c') are in the same class. In the same way, (c, c') is in the class if it is not the greatest element. Assume $c < d$. Then (c, b') and (d, b') are related to (b, c'). Then (c, b') and (d, b') are related to (b, c'), and (b, c') and (c, c') are related to (d, b'). This shows that the six elements of the product $\{b, c, d\} \times \{b', c'\}$, minus the greatest and least elements are in the same class. By symmetry, if $d' \in \mathrm{F}$, the six elements of the product $\{b, c, d\} \times \{b', c', d'\}$, minus the greatest and least elements are in the same class.

Thus, if E has a least and a greatest element l and g, there are three classes, $\mathrm{E} - \{l, g\}$, $\{l\}$, and $\{g\}$. If E has a least element l and no greatest element, there are two classes, $\mathrm{E} - \{l\}$, and $\{l\}$. If E has a greatest element g and no least element, there are two classes, $\mathrm{E} - \{g\}$, and $\{g\}$. Otherwise there is a single class.

```
Lemma tor_prop r: {inc substrate r &, forall x y , ocomparable r x y} ->
  forall x y, inc x (substrate r) -> inc y (substrate r) ->
  gle r x y \/ glt r y x.  (* 3 *)

Lemma Exercise1_4c1 r x: (* 11 *)
   order r -> greatest r x ->
   ncr_component r x = singleton x.
Lemma Exercise1_4c2 r x:  (* 11 *)
   order r -> least r x ->
   ncr_component r x = singleton x.
Lemma Exercise1_4c3 r r' x y: (* 22 *)
  order r -> total_order r' ->
  substrate r = singleton x -> inc y (substrate r') ->
  ncr_component (order_product2 r r') (J x y) = singleton (J x y).
Lemma Exercise1_4c4 r r' b c b' c' u: (* 104 *)
  total_order r -> total_order r' ->
  glt r b c -> glt r' b' c' ->
  inc u (substrate (order_product2 r r')) ->
```

```
  [\/ least (order_product2 r r') u,
      greatest (order_product2 r r') u |
      inc u (ncr_component (order_product2 r r') (J b c'))].
```

---

5. Let E be an ordered set. A subset X of E is said to be *free* if no two distinct elements of X are comparable. Let $\mathfrak{I}$ be the set of free subsets of E. Show that, on $\mathfrak{I}$, the relation "given any $x \in X$, there exists $y \in Y$ such that $x \le y$" is an order relation between X and Y, written $X \le Y$. The mapping $x \to \{x\}$ is an isomorphism of E onto a subset of the ordered set $\mathfrak{I}$. If $X \subset Y$, where $X \in \mathfrak{I}$ and $Y \in \mathfrak{I}$, show that $X \le Y$. The ordered set $\mathfrak{I}$ is totally ordered if and only if E is totally ordered, and then $\mathfrak{I}$ is canonically isomorphic to E.

Solution. We restate the definition as: if $x \in X$ and $y \in X$ and $x \le y$ then $x = y$.

```
Definition free_subset r X := forall x y, inc x X -> inc y X ->
  gle r x y -> x = y.
Definition free_subsets r:=
  Zo (\Po (substrate r)) (free_subset r).
Definition free_subset_compare r X Y:=
  [/\ inc X (free_subsets r), inc Y (free_subsets r) &
  forall x, inc x X -> exists2 y, inc y Y & gle r x y].
Definition free_subset_order r:=
  graph_on (free_subset_compare r) (free_subsets r).
```

The relation $\forall x \in X, \exists y \in Y, x \le y$ is clearly a preorder relation. Antisymmetry follows from the fact that if $x \in X, y \in Y, x \le y$ and $x' \in X, y \le x'$ we have $x \le x'$ by transitivity, then $x = x'$ when X is free, and $x = y$ by antisymmetry.

```
Lemma Exercise1_5w r x a: order r -> (* 1 *)
 inc x (free_subsets r) -> inc a x ->
 gle r a a.
Lemma Exercise1_5a r: order r -> (* 14 *)
  order_r (free_subset_compare r).
Lemma fs_order_gleP r x y: (* 2 *)
  gle (free_subset_order r) x y <-> free_subset_compare r x y.
Lemma fs_order_osr r: (* 4 *)
  order r -> order_on (free_subset_order r) (free_subsets r).
```

A singleton is free; the mapping $x \mapsto \{x\}$ is an isomorphism onto its range. If E is a totally ordered set, then the only nonempty free subsets are the singletons, so that the range is $\mathfrak{I} - \{\emptyset\}$.

```
Lemma Exercise1_5b r x: order r -> (* 2 *)
  inc x (substrate r) -> inc (singleton x) (free_subsets r).
Lemma Exercise1_5cP r x y: order r -> (* 5 *)
  inc x (substrate r) -> inc y (substrate r) ->
  (gle r x y <-> gle (free_subset_order r) (singleton x) (singleton y)).
Lemma Exercise1_5d r: order r -> (* 6 *)
  order_morphism (Lf singleton (substrate r)  (free_subsets r))
  r (free_subset_order r).
Lemma Exercise1_5e r X: total_order r -> (* 2 *)
  inc X (free_subsets r) -> small_set X.
```

If $X \subset Y$ then $X \leq Y$ (this is trivial). If E is totally ordered so is $\mathfrak{I}$. In fact, the empty set is the least element of $\mathfrak{I}$; two non-empty sets are singletons, and singletons are compared according to their elements. The converse is trivial. Bourbaki says that $\mathfrak{I}$ is canonically isomorphic to E in this case. This is obviously wrong: as noted above $x \mapsto \{x\}$ is an isomorphism between E and $\mathfrak{I} - \{\emptyset\}$.

```
Lemma Exercise1_5f r X Y: order r -> (* 2 *)
  inc X (free_subsets r) -> inc Y (free_subsets r) ->
  sub X Y ->  gle (free_subset_order r) X Y.
Lemma Exercise1_5g r: total_order r -> (* 16 *)
  total_order (free_subset_order r).
Lemma Exercise1_5h r: order r -> (* 4 *)
  total_order (free_subset_order r) -> total_order r.
```

---

6. Let E and F be two ordered sets, and let $\mathscr{A}(E,F)$ be the subset of the product ordered set F^E consisting of the *increasing* mappings of E into F.

(a) Show that if E, F, G are three ordered sets, then the ordered set $\mathscr{A}(E, F\times G)$ is isomorphic to the product ordered set $\mathscr{A}(E,F)\times\mathscr{A}(E,G)$.

(b) Show that if E, F, G are three ordered sets, then the ordered set $\mathscr{A}(E\times F, G)$ is isomorphic to the ordered set $\mathscr{A}(E,\mathscr{A}(F,G))$

(c) If $E \neq \emptyset$, then $\mathscr{A}(E,F)$ is a lattice if and only if F is a lattice.

(d) Suppose that E and F are both non-empty. Then $\mathscr{A}(E,F)$ is totally ordered if and only if one of the following conditions is satisfied:
(α) F consists in a single element;
(β) E consists in a single element and F is totally ordered;
(γ) E and F are both totally ordered and F has two elements.

Note. F^E is the set of functional graphs $E\to F$, it is canonically isomorphic to $\mathscr{F}(E;F)$, the set of mappings from E to F. In what follows, $\mathscr{A}(E,F)$ will be a subset of $\mathscr{F}(E;F)$. It will be ordered by: $f \leq g$ whenever $f(x) \leq g(x)$ for all x.

Solution. In what follows, r, r', r'' will denote orderings on set E, F, G. If $f : E \to F\times G$, we deduce two functions $\mathrm{pr}_1 f$ and $\mathrm{pr}_2 f$, thus a mapping $T : f \mapsto (\mathrm{pr}_1 f, \mathrm{pr}_2 f)$, and its restriction T' to $\mathscr{A}$. If $f : E\times F \to G$ is a function, we consider f_x defined by $f_x(y) = f((x,y))$, and $\tilde f : x \mapsto f_x$. The function $\Phi : f \mapsto \tilde f$ is the canonical function $\mathscr{F}(E\times F;G)$ onto $\mathscr{F}(E,\mathscr{F}(F;G))$, we consider its restriction Φ' to $\mathscr{A}$.

```
Definition increasing_mappings r r' :=
  Zo (functions (substrate r) (substrate r'))
  (fun z=> increasing_fun z r r').
Definition increasing_mappings_order r r' :=
  induced_order (order_function (substrate r) (substrate r') r')
  (increasing_mappings r r').
Definition first_projection f:= Lf (fun z=> P (Vf f z)).
Definition secnd_projection f:= Lf (fun z=> Q (Vf f z)).
Definition two_projections a b c :=
  Lf (fun z => (J (first_projection z a b)
    (secnd_projection z a c)))
  (functions a (b \times c))
```

```
  ((functions a b) \times (functions a c)).
Definition two_projections_increasing r r' r'' :=
  restriction2 (two_projections (substrate r) (substrate r')(substrate r''))
  (increasing_mappings r (order_product2 r' r''))
  ( (increasing_mappings r r') \times
    (increasing_mappings r r'')).
Definition second_partial_map2 r r' r'':=
  Lf (fun f=> restriction2
    (second_partial_function f)
    (substrate r) (increasing_mappings r' r''))
  (increasing_mappings (order_product2 r r') r'')
  (increasing_mappings r (increasing_mappings_order r' r'')).
```

We first show that T is a bijection $\mathscr{F}(\mathrm{E};\mathrm{F}\times\mathrm{G})$ onto $\mathscr{F}(\mathrm{E};\mathrm{F})\times\mathscr{F}(\mathrm{E};\mathrm{G})$.

```
Lemma Exercise1_6a f a b c: (* 10 *)
  function f -> source f = a ->
  target f = b \times c ->
   [/\ lf_axiom (fun z=> P (Vf f z)) a b,
     lf_axiom (fun z=> Q (Vf f z)) a c,
    function (first_projection f a b),
    function (secnd_projection f a c) &
    (forall x, inc x a -> Vf (first_projection f a b) x = P (Vf f x)) /\
    (forall x, inc x a -> Vf (secnd_projection f a c) x = Q (Vf f x))].
Lemma Exercise1_6b a b c: (* 4 *)
  lf_axiom
    (fun z => (J (first_projection z a b)
      (secnd_projection z a c)))
    (functions a (b \times c))
    ((functions a b) \times (functions a c)).
Lemma Exercise1_6c a b c: (* 22 *)
  bijection  (two_projections a b c).
```

We give here some trivial lemmas making the definitions explicit.

```
Section Exercise1_6a.
Variables r r': Set.
Hypotheses (or: order r)(or': order r').

Lemma soimP f:  (* 2 *)
  inc f (increasing_mappings r r') <->
    ((function_prop f (substrate r) (substrate r'))
    /\ increasing_fun f r r').
Lemma imo_osr: (* 2 *)
  order_on (increasing_mappings_order r r') (increasing_mappings r r').
Lemma imo_gleP f g: (* 6 *)
  gle (increasing_mappings_order r r') f g <->
  [/\ inc f (increasing_mappings r r'),
    inc g (increasing_mappings r r') &
    order_function_r (substrate r) (substrate r') r' f g].
Lemma imo_incr f g: (* 1 *)
   gle (increasing_mappings_order r r') f g ->
   forall i, inc i (substrate r) -> gle r' (Vf f i) (Vf g i).
End Exercise1_6a.
```

(a) Consider an increasing function $f : \mathrm{E} \to \mathrm{F} \times \mathrm{G}$. Both projections $\mathrm{pr}_1 f$ and $\mathrm{pr}_2 f$ are increasing; the converse is equally true. This means that $f \mapsto (\mathrm{pr}_1 f, \mathrm{pr}_2 f)$ induces a bijection $\mathscr{A}(\mathrm{E}, \mathrm{F} \times \mathrm{G})$ onto $\mathscr{A}(\mathrm{E}, \mathrm{F}) \times \mathscr{A}(\mathrm{E}, \mathrm{G})$. Finally, we show that it is an order isomorphism.

```
Section Exercise1_6.
Variables r r' r'': Set.
Hypotheses (or: order r)(or': order r')(or'': order r'').
Let E := substrate r. Let F := substrate r'. Let G := substrate r''.

Lemma Exercise1_6d f: (* 6 *)
  increasing_fun f r (order_product2 r' r'') ->
  (increasing_fun (first_projection f E F) r r' /\
    increasing_fun (secnd_projection f E G) r r'').

Lemma Exercie1_6e: (* 17 *)
  (restriction2_axioms
    (two_projections E F G)
    (increasing_mappings r (order_product2 r' r''))
    ((increasing_mappings r r') \times (increasing_mappings r r''))).
Lemma Exercise1_6f: (* 25 *)
  bijection (two_projections_increasing r r' r'').
Lemma Exercise1_6g: (* 56 *)
  order_isomorphism (two_projections_increasing r r' r'')
  (increasing_mappings_order r (order_product2 r' r''))
  (order_product2 (increasing_mappings_order r r')
    (increasing_mappings_order r r'')).
```

(b) Let's show that $\mathscr{A}(\mathrm{E} \times \mathrm{F}, \mathrm{G})$ is isomorphic to the ordered set $\mathrm{T} = \mathscr{A}(\mathrm{E}, \mathscr{A}(\mathrm{F}, \mathrm{G}))$. We first assume E and F non-empty.

In all our lemmas we consider orderings r, r' and r'', with substrate E, F and G. An increasing function $f : r \times r' \to r''$ is a function with source $\mathrm{E} \times \mathrm{F}$ and target G; the first lemma says, that if r and r' are orders, the substrate of $r \times r'$ is indeed the product of the substrates, namely $\mathrm{E} \times \mathrm{F}$, and if these two sets are non-empty, one can recover the sets from the product. The second lemma says that f_x is increasing for all $x \in \mathrm{E}$. The third lemma says that the range of $x \mapsto f_x$ is a subset of the set of increasing functions.

```
Lemma Exercise1_6h f: (* 3 *)
  nonempty E -> nonempty F -> increasing_fun f (order_product2 r r') r'' ->
 ( (domain (source f)) = E /\ (range (source f)) = F).
Lemma Exercise1_6i f x: (* 16 *)
  nonempty E -> nonempty F ->
  increasing_fun f (order_product2 r r') r'' ->
  inc x E -> increasing_fun (second_partial_fun f x) r' r''.
Lemma Exercise1_6j f: (* 16 *)
  nonempty E -> nonempty F ->
  increasing_fun f (order_product2 r r') r'' ->
  (restriction2_axioms (second_partial_function f) E
  (increasing_mappings r' r'')).
```

For each $a \in \mathrm{E}$, $\Phi(f)(a)$ is in $\mathscr{F}(\mathrm{F}; \mathrm{G})$; if f is increasing this is in $\mathscr{A}(\mathrm{F}, \mathrm{G})$. Thus we can restrict $\Phi(f)$ as a function $\Phi'(f)$ from E into $\mathscr{A}(\mathrm{F}, \mathrm{G})$. The mapping Φ' will be our isomorphism. It is clearly injective; proving surjectivity is a bit longer.

If f_a and $a \mapsto f_a$ are increasing, then $a \le a'$ and $b \le b'$ implies $f_a(b) \le f_a(b') \le f_{a'}(b')$ so that f is increasing. Conversely, if f is increasing, then f_a is increasing since if $b \le b'$ then

$f_a(b) \le f_a(b')$, using $a \le a$; and $a \mapsto f_a$ is increasing since if $a \le a'$ then $f_a(b) \le f_{a'}(b)$, using $b \le b$. These are the only properties of the order that are used here.

```
Lemma Exercise1_6k: (* 31 *)
 nonempty E -> nonempty F ->
  lf_axiom (fun f=> restriction2
    (second_partial_function f) E (increasing_mappings r' r''))
  (increasing_mappings (order_product2 r r') r'')
  (increasing_mappings r (increasing_mappings_order r' r'')).
Lemma Exercise1_6l: (* 73 *)
  nonempty (substrate r) -> nonempty (substrate r') ->
    bijection_prop (second_partial_map2 r r' r'' )
      (increasing_mappings (order_product2 r r') r'')
      (increasing_mappings r (increasing_mappings_order r' r'')).
Lemma Exercise1_6m:  (* 53 *)
  nonempty E -> nonempty F ->
  order_isomorphism (second_partial_map2 r r' r'')
  (increasing_mappings_order (order_product2 r r') r'')
  (increasing_mappings_order r (increasing_mappings_order r' r'')).
```

Let $\mathrm{S} = \mathscr{A}(\mathrm{E}\times\mathrm{F},\mathrm{G})$ and $\mathrm{T} = \mathscr{A}(\mathrm{E},\mathscr{A}(\mathrm{F},\mathrm{G}))$. Assume now one of E and F empty. Then $\mathrm{E}\times\mathrm{F}$ is empty, case where there is a single element in S (the empty function with target G). If E is empty, then T is a singleton (it contains the empty function). In the case F is empty, then $\mathscr{A}(\mathrm{F},\mathrm{G})$ is a singleton. Thus T is a singleton (it contains only the constant function). Thus S and T are singletons, there is a bijection $f : \mathrm{S} \to \mathrm{T}$. It is obviously an order isomorphism.

```
Lemma Exercise1_6m': (* 66 *)
  (increasing_mappings_order (order_product2 r r') r'') \Is
  (increasing_mappings_order r (increasing_mappings_order r' r'')).
```

(c) Assume $t \in \mathrm{E}$. For $a \in \mathrm{F}$, let C_a be the constant function with value a. Then $\mathrm{C}_a \in \mathscr{A}(\mathrm{E},\mathrm{F})$. Moreover evaluation at t shows that $\mathrm{C}_a \le \mathrm{C}_b$ is equivalent to $a \le b$.

```
Lemma constant_increasing  y: inc y F -> (* 4 *)
    (inc (constant_function E F y) (increasing_mappings r r')).
Lemma constant_increasing1: (* 6 *)
  nonempty E ->
  forall y y', inc y F -> inc y' F ->
    (gle r' y y' <->
    gle (increasing_mappings_order r r')
    (constant_function E F y)
    (constant_function E F y')).
```

Assume that F is a lattice; given two functions f and g we can consider the function $x \mapsto \sup(f(x), g(x))$. It is the least upper bound of f and g. Conversely, assume that $\mathscr{A}(\mathrm{E},\mathrm{F})$ is a lattice. Given two values a and b in F, the constant functions C_a and C_b with values a and b are in $\mathscr{A}(\mathrm{E},\mathrm{F})$. Let $t \in \mathrm{E}$ and f be the supremum of C_a and C_b, and $c = f(t)$. This is an upper bound of a and b. Let d be another upper bound. Then C_d is an upper bound of C_a and C_b, hence $f \le \mathrm{C}_d$. Evaluating at t gives $c \le d$. The proof is long but presents no difficulty. Thus, $\mathscr{A}(\mathrm{E},\mathrm{F})$ is a lattice if and only if F is a lattice.

```
Lemma Exercise1_6n:   (* 125 *)
  nonempty E ->
  (lattice r' <-> lattice (increasing_mappings_order r r')).
```

(d) We first show that if $\mathscr{A}(\mathrm{E},\mathrm{F})$ is totally ordered and E non-empty, then F is totally ordered, using constant functions as above. If F is a singleton, then $\mathscr{A}(\mathrm{E},\mathrm{F})$ has a single element, hence is totally ordered. If E is a singleton, all functions are constant and $\mathscr{A}(\mathrm{E},\mathrm{F})$ is isomorphic to F. Finally, assume that E and F are two totally ordered sets, and F has two elements. We may assume that these elements are distinct and satisfy $a < b$. Assume $f(u) < g(u)$. Then $f(u) = a$ and $g(u) = b$. If no such u exists, then $f \geq g$. Otherwise, consider v; if $f(v) > g(v)$ we get $f(v) = b$ and $g(v) = a$. Since u and v can be compared, this implies that one of f and g is non-increasing.

```
Lemma Exercise1_6o: (* 6 *)
  nonempty E ->
  total_order (increasing_mappings_order r r') ->
  total_order r'.
Lemma Exercise1_6p:  (* 8 *)
  singletonp F ->
  total_order (increasing_mappings_order r r').
Lemma Exercise1_6q: (* 10 *)
  singletonp E -> total_order r' ->
  total_order (increasing_mappings_order r r').
Lemma Exercise1_6r:  (* 36 *)
  total_order r -> total_order r' ->
  (exists a b, F = doubleton a b) ->
  total_order (increasing_mappings_order r r').
```

We have shown that each of the three conditions implies that $\mathscr{A}(\mathrm{E},\mathrm{F})$ is totally ordered. Let's consider the converse. We know that F is totally ordered. If the first two conditions are false, then E and F have at least two elements. Since F is totally ordered, we may assume $a < b$ in F.

For $u \in \mathrm{E}$ we consider the mapping f_u that associates a if $x \leq u$ and b otherwise. This is an increasing mapping thus an element of $\mathscr{A}(\mathrm{E},\mathrm{F})$. Consider two incomparable elements u and v of E. We have $a = f_u(u) = f_v(v)$ while $b = f_u(v) = f_v(u)$, so that f_u and f_v are non-comparable. Thus $\mathscr{A}(\mathrm{E},\mathrm{F})$ totally ordered implies that E is totally ordered.

Assume now that F has more than two elements. We can find three elements such that $a < c < b$. Consider $u < v$ in E. We have $a = f_u(u) < \mathrm{C}_c(u) = c = \mathrm{C}_c(v) < f_u(v) = b$. Thus the two functions f_u and C_c are non-comparable.

```
Lemma Exercise1_6s:  (* 90 *)
    [\/ singletonp (substrate r'),
      (singletonp (substrate r) /\ total_order r') |
      [/\ total_order r', total_order r &
         exists u v, substrate r' = doubleton u v]]).
  nonempty(substrate r) -> nonempty (substrate r') ->
End Exercise1_6.
```

---

7. In order that every mapping of an ordered set E into an ordered set F with at least two elements, which is both an increasing and a decreasing mapping, should be constant on E, it is necessary and sufficient that E should be connected with respect to the reflexive and symmetric relation "x and y are comparable" (Chapter II, § 6, Exercise 10). This condition is satisfied if E is either left or right directed.

Note. If F is empty or has a single element, then all functions with values in F are constant. This explains why F is assumed to have at least two elements.

Solution. We consider two orderings r and r', and let E and F be the substrates. Denote by $x \equiv y$ the relation "x and y are comparable" (according to r), and by $x \sim y$ the equivalence relation associated to it. This is an equivalence relation on E, and $x \equiv y$ implies $x \sim y$. The conclusion (C) is that all elements of E are related by $\equiv$.

```
Definition cr_equiv r :=
  chain_equivalence (ocomparable r) (substrate r).
Definition cr_component r :=
  connected_comp (ocomparable r) (substrate r).

Lemma cr_properties r: order r -> (* 17 *)
 [/\ equivalence (cr_equiv r) ,
    (forall x y, ocomparable r x y ->
       (inc x (substrate r)/\ inc y (substrate r))),
    substrate  (cr_equiv r) =  substrate r,
    (forall x, inc x (substrate r) -> class(cr_equiv r) x = cr_component r x) &
    (forall x y, ocomparable r x y -> related (cr_equiv r) x y)].
```

Assume E right directed. For any x and y in E, there is an upper bound z. Thus $x \equiv z$ and $y \equiv z$ so that $x \sim y$, and (C) holds.

```
Lemma Exercise1_7a r x: right_directed r -> (* 9 *)
  inc x (substrate r) -> cr_component r x = substrate r.
Lemma Exercise1_7b r x: left_directed r -> (* 9 *)
  inc x (substrate r) -> cr_component r x = substrate r.
```

If f is increasing and decreasing, the relation $x \equiv y$ implies $f(x) = f(y)$. Thus f is constant on chains. If E is the class of x for $\equiv$ then f must be constant.

```
Lemma Exercise1_7c r r' f x y: (* 2 *)
  increasing_fun f r r' -> decreasing_fun f r r' -> ocomparable r x y ->
  Vf f x = Vf f y.
Lemma Exercise1_7d r r' f: (* 11 *)
  increasing_fun f r r' ->  decreasing_fun f r r' ->
  (exists2 x, inc x (substrate r) & cr_component r x = substrate r) ->
  (constantfp f).
```

Converse. We assume that F is a set with at least two elements a and b, and that E is not connected; more precisely we assume that there is $c \in$ E so that the component C of c is not E. We consider the function g that maps x to a if $x \in$ C, and to b otherwise. This is a non-constant function. Assume $x \le y$. Then $x \in$ C and $y \in$ C are equivalent, so that $g(x) = g(y)$. As a consequence g is increasing and decreasing.

```
Lemma Exercise1_7e r r': (* 40 *)
  order r -> order r' ->
  (exists u v, [/\ inc u (substrate r'), inc v (substrate r')  & u <>v]) ->
  (exists2 x, inc x (substrate r) & cr_component r x <> substrate r) ->
  exists f, [/\ increasing_fun f r r',  decreasing_fun f r r' &
    ~(constantfp f)].
```

8. Let E and F be two ordered sets, let f be an increasing mapping of E into F, and g an increasing mapping of F into E. Let A (resp. B) be the set of all $x \in E$ (resp. $y \in F$) such that $g(f(x)) = x$ (resp. $f(g(y)) = y$). Show that the two ordered sets A and B are canonically isomorphic.

Solution. The restriction of the function f is a bijection from A onto B. It is clearly increasing.

```
Lemma Exercise1_8 r r' f g: (* 37 *)
  let A := Zo (substrate r) (fun z => Vf g (Vf f z) = z) in
   let B := Zo (substrate r') (fun z => Vf f (Vf g z) = z) in
     increasing_fun f r r' ->  increasing_fun g r' r ->
     (induced_order r A) \Is (induced_order r' B).
```

---

9. *If E is a lattice, prove that

$$\sup_j(\inf_i x_{ij}) \le \inf_i(\sup_j x_{ij})$$

for every finite "double" family (x_{ij}).*

Note. This exercise is surrounded by stars as Bourbaki has not yet defined "finite". If the double family is not finite, then the sup and inf are not always defined. The same holds if the family is empty.

Solution. We consider an order r and its substrate E. By induction, each finite non-empty set has a supremum and an infimum (see main text). The same holds for a functional graph whose domain is non-empty and finite.

```
Section Exercise1_9.
Variable r: Set.
Hypothesis lr: lattice r.
Let Hf := fun f => [/\ fgraph f, finite_set (domain f), nonempty (domain f) &
  sub (range f) (substrate r) ].

Lemma lattice_finite_sup_aux f: Hf f ->  (* 2 *)
  nonempty (range f) /\ finite_set (range f).
Lemma lattice_finite_sup4P f:  (Hf f) -> forall y, (* 6 *)
   (gle r(sup_graph r f) y <->
   (forall z, inc z (domain f) -> gle r (Vg f z) y)).
Lemma lattice_finite_inf4P f: (Hf f) -> forall y(* 6 *)
   (gle r y (inf_graph r f) <->
   (forall z, inc z (domain f) -> gle r y (Vg f z))).
Lemma lattice_finite_sup5 f: (Hf f) -> (* 6 *)
  inc (sup_graph r f) (substrate r).
Lemma lattice_finite_inf5 f: (Hf f) ->  (* 6 *)
  inc (inf_graph r f) (substrate r).
```

The result is obvious.

```
Lemma Exercise1_9 I1 I2 f: (* 33 *)
   fgraph f -> domain f = I1 \times I2 ->
  finite_set I1 -> finite_set I2 -> nonempty I1 -> nonempty I2 ->
```

```
  sub (range f) (substrate r) ->
  gle r
  (sup_graph r (Lg I2 (fun j => inf_graph r (Lg I1 (fun i => Vg f (J i j))))))
  (inf_graph r (Lg I1 (fun i => sup_graph r (Lg I2 (fun j => Vg f (J i j)))))).
End Exercise1_9.
```

General case; Assume that the sups and infs involved in the formula do exists. Then the result holds.

```
Lemma Exercise1_9_a r I1 I2 f  (* 28 *)
  (rf1 := fun j => Lg I1 (fun i => Vg f (J i j)))
  (rf2 := fun i => Lg I2 (fun j => Vg f (J i j)))
  (inf1 := (fun j => inf_graph r (rf1 j)))
  (sup1 := (fun i => sup_graph r (rf2 i))):
  fgraph f -> domain f = I1 \times I2 ->  sub (range f) (substrate r) ->
  order r ->
  (forall j, inc j I2 -> has_infimum r (range (rf1 j))) ->
  (forall i, inc i I1 -> has_supremum r (range (rf2 i))) ->
  has_supremum r (range (Lg I2 inf1)) ->
  has_infimum r (range (Lg I1 sup1)) ->
  gle r  (sup_graph r (Lg I2 inf1))   (inf_graph r (Lg I1 sup1)).
```

---

10. Let E and F be two lattices. Then a mapping f of E into F is increasing if and only if

$$f(\inf(x,y)) \le \inf(f(x), f(y))$$

for all $x \in \mathrm{E}$ and $y \in \mathrm{E}$.

*Give an example of an increasing mapping f of the product ordered set $\mathbf{N}\times\mathbf{N}$ into the ordered set $\mathbf{N}$ such that the relation

$$f(\inf(x,y)) = \inf(f(x), f(y))$$

is false for at least one pair $(x,y) \in \mathbf{N}\times\mathbf{N}$.*

Note. The stars indicate that a part of the exercise uses material not yet defined (the set **N** of natural numbers). We shall provide a second counter example. We shall also show that f is increasing if and only if

$$\sup(f(x), f(y)) \le f(\sup(x,y)).$$

Solution. The first part is easy. We show that the product of two lattices is a lattice.

```
Section Exercise1_10.
Variable r r': Set.
Hypothesis (lr: lattice r) (lr': lattice r').

Lemma Exercise1_10 f: function_prop f (substrate r)(substrate r') -> (* 14 *)
  ((increasing_fun f r r') <->
   (forall x y, inc x (substrate r) -> inc y (substrate r) ->
    gle r' (Vf f (inf r x y)) (inf r' (Vf f x) (Vf f y)))).
```

```
Lemma Exercise1_10b f: function_prop f (substrate r)(substrate r') -> (* 14 *)
  ((increasing_fun f r r') <->
   (forall x y, inc x (substrate r) -> inc y (substrate r) ->
    gle r' (sup r' (Vf f x) (Vf f y)) (Vf f (sup r x y)))).
Lemma product2_lattice: (* 27 *)
  lattice (order_product2 r r').

End Exercise1_10.
```

Let E be a lattice, with two elements a and b such that $a < b$. On the product $E\times E$ consider $x = (a, b)$ and $y = (b, a)$. Let $z = \inf(x, y)$. Then $z = (a, a)$. Assume $f(z) = a$, $f(x) = f(y) = b$. This gives a counter-example. We first consider $f((a, b)) = a + b$, where E is the set of natural numbers, $a = 0$ and $b = 1$. We consider a second example, where $E = \{a, b\}$, f maps (a, a) to a, all other elements to b. In both cases, f is increasing.

```
Definition Exercise1_10_counterexample r r' f:=
 [/\ lattice r, lattice r', function_prop f (substrate r)(substrate r'),
      (increasing_fun f r r') &
  exists x y, [/\ inc x (substrate r), inc y (substrate r) &
    (Vf f (inf r x y))  <> (inf r' (Vf f x) (Vf f y))]].

Lemma Exercise1_10_bis (* 37 *)
  (r := order_product2 Nat_order Nat_order)
  (r':= Nat_order)
  (f := Lf (fun z =>  (P z) +c (Q z)) (Nat \times  Nat) Nat):
   Exercise1_10_counterexample r r' f.
Lemma Exercise1_10_ter  (* 37 *)
  (r':= canonical_doubleton_order)
  (r := order_product2 r' r')
  (f := Lf (fun z => (Yo (z = J C0 C0)) C0 C1) (C2 \times C2) C2):
   Exercise1_10_counterexample r r' f.
```

---

11. A lattice E is said to be *complete* if every subset of E has a least upper bound and a greatest lower bound in E; this means in particular that E has a greatest and a least element.

(a) Show that if an ordered set E is such that every subset of E has a least upper bound in E, then E is a complete lattice.

(b) A product of ordered sets is a complete lattice if only if each of the factors is a complete lattice.

(c) An ordinal sum (Exercise 3) $\sum_{\iota\in I} E_\iota$ is a complete lattice if and only if the following conditions are satisfied:
(I) I is a complete lattice.
(II) If J is a subset of I which has no greatest element, and if $\sigma = \sup J$, then E_σ has a least element.
(III) For each $\iota \in I$ every subset of E_ι which has an upper bound in E_ι has a least upper bound in E_ι.
(IV) For each $\iota \in I$ such that E_ι has no greatest element, the set of all $\kappa > \iota$ has a least element α and E_α has a least element.

(d) The ordered set $\mathscr{A}(\mathrm{E},\mathrm{F})$ of increasing maps of an ordered set E into an ordered set F (Exercise 6) is a complete lattice if and only if F is a complete lattice.

Note. Condition (III) has to be replaced by: "every non-empty subset of E_ι which has an upper bound has a supremum". Example. Consider the sum of two sets, a singleton and the opposite order of **N**. The empty set is bounded in **N** but has no greatest lower bound.

In (d), if E is empty, then $\mathscr{A}(\mathrm{E},\mathrm{F})$ has a single element, thus is a complete lattice, whatever F.

We first show that the result of Exercise 9 holds in a complete lattice.

```
Definition complete_lattice r := order r /\
  forall X, sub X (substrate r) -> (has_supremum r X /\ has_infimum r X).

Lemma Exercise1_9_b I1 I2 r f:  (* 23 *)
  fgraph f -> domain f = I1 \times I2 ->
  sub (range f) (substrate r) ->
  complete_lattice r->
  gle r
  (sup_graph r (Lg I2 (fun j => inf_graph r (Lg I1 (fun i => Vg f (J i j))))))
  (inf_graph r (Lg I1 (fun i => sup_graph r (Lg I2 (fun j => Vg f (J i j)))))).
```

Solution. Here r denotes an ordering, and E its substrate. If we take the supremum and infimum of the empty set, we get a least and a greatest element. In particular E is non-empty and in (b) the factors are all non-empty.

(a) Let X be a subset of E and X' be the set of its lower bounds. If X' has a supremum, this is the infimum of X.

```
Lemma Exercise1_11a r: (* 6 *)
  complete_lattice r ->
  (has_greatest r /\ has_least r b).
Lemma Exercise1_11b r: order r ->  (* 10 *)
  (forall X, sub X (substrate r) -> has_supremum r X) ->
  complete_lattice r.
```

Extensions: If each subset of E has a greatest lower bound, then E is a complete lattice. If E is a complete lattice, so is the opposite ordering on E. If E is finite, totally ordered, non-empty, then E is a complete lattice.

```
Lemma finite_set_torder_least r: (* 3 *)
  total_order r ->finite_set (substrate r) -> nonempty (substrate r)
  -> exists x, least_element r x.
Lemma Exercise1_11h r: order r ->   (* 10 *)
  (forall X, sub X (substrate r) -> has_infimum r X) ->
  complete_lattice r.
Lemma Exercise1_11i r: (* 5 *)
  complete_lattice r -> complete_lattice (opp_order r).
Lemma Exercise1_11j r: (* 11 *)
  total_order r -> finite_set (substrate r) -> ne_substrate r ->
  complete_lattice r.
```

(b) Let X be a subset of $\prod \mathrm{E}_i$ and $\mathrm{X}_i = \mathrm{pr}_i \mathrm{X}$. If each E_i is a complete lattice, then $\sup \mathrm{X}_i$ is the supremum of the set X_i, and $i \mapsto \sup \mathrm{X}_i$ is the supremum of X. Conversely, if the product is a complete lattice, it is non-empty since it has a least element. If $\mathrm{X}_i \subset \mathrm{E}_i$ we can find a set X with $\mathrm{X}_i = \mathrm{pr}_i \mathrm{X}$. If x is the supremum of X then x_i is the supremum of X_i.

```
Lemma Exercise1_11c g: (* 45 *)
  order_fam g ->
  (allf g complete_lattice) ->
  complete_lattice (order_product g).
Lemma Exercise1_11d g:  (* 39 *)
  order_fam g -> complete_lattice (order_product g) ->
  (allf g complete_lattice).
```

(c) Consider a family g of orderings on E_i, indexed by I and an ordering r on I. Each E_i can be identified with a subset $\bar{E}_i$ of the disjoint union Σ. If $i \in I$ and $x \in E_i$, we denote by $\bar{x}$ the element x considered in Σ. Note that $\mathrm{pr}_2 \bar{x}$ is the index i such that $x \in E_i$. We compare two elements of Σ: if they are in the same E_i, we use the ordering of E_i, otherwise we compare pr_2 in I. We assume E_i non-empty, so that there is a function k defined on I such that $k(i) \in E_i$, and $\bar{k}(i) \in \Sigma$.

Assume first that the ordinal sum is a complete lattice. Let J be a subset of I, and consider the least upper bound $\bar{x}$ of $\bar{k}(J)$. This element is in some $\bar{E}_j$ for $j \in I$, so that j is the supremum of J, and condition (I) holds. Assume that J has no greatest element, so that $j \not\in J$. Every element in $\bar{E}_j$ is an upper bound of $\bar{k}(J)$. Thus x must be the least element of E_j. This shows condition (II).

Consider now a subset X of E_i. This can be identified with a subset $\bar{X}$ of Σ that has a least upper bound $\bar{x}$. Assume that $u \in X$ and X is bounded above by v. We have $\bar{u} \le \bar{x} \le \bar{v}$ so that $\mathrm{pr}_2 \bar{x} = i$. Then x is the least upper bound of X; this shows point (III).

Consider finally point (IV). Let $i \in I$. Consider J, the subset of I formed of indices $j > i$; consider E_i as a subset X of Σ. Let $\bar{x}$ be its supremum and $j = \mathrm{pr}_2 \bar{x}$. Since E_i is nonempty, we have $\bar{k}(i) \le \bar{x}$, hence $i \le j$. If $i = j$, then x is the greatest element of E_i. Let's assume that E_i has no greatest element. This implies $j \in J$. For every $j' \in J$, any element of $E_{j'}$ is an upper bound of X, thus $j \le j'$. This means that j is the least element of J. Moreover x is the least element of E_j. This proves (IV).

Conversely, assume the four assumptions true. Take a subset X of Σ. Let J be the set of all i such that at least one element of X is in $\bar{E}_i$. This set has a least upper bound, say i. Assume first that J has no greatest element. By assumption (II), the set E_i has a least element x. We pretend that $\bar{x}$ is the least upper bound of X. It is a strict upper bound of X since i is a strict upper bound of J. Consider another upper bound, say $\bar{z}$. We have $i \le \mathrm{pr}_2 \bar{z}$. If $i < \mathrm{pr}_2 \bar{z}$ it follows $x < z$. If $i = \mathrm{pr}_2 \bar{z}$, we have $x \le z$ in E_i hence $\bar{x} \le \bar{z}$.

Assume now that J has a greatest element j. Let X_j be the (nonempty) set of all $\bar{x}$ of X such that $\mathrm{pr}_2 \bar{x} = j$. Consider first the case where X_j has an upper bound and apply (III). There a least upper bound x of X_j. Then $\bar{x}$ is the supremum of X.

Assume that X_j has no upper bound in E_j. In particular E_j has no greatest element; by (IV) there is an index k, the least index such that $k > j$ and a least element x in E_k. Then $\bar{x}$ is the supremum of X.

```
Definition greatest_induced r X x := greatest (induced_order r X) x.
Definition least_induced r X x := least (induced_order r X) x.

Lemma Exercise1_11e r g: (* 206 *)
  orsum_ax r g -> orsum_ax2 g ->
  (complete_lattice (order_sum r g) <->
  [/\ complete_lattice r,
    (forall j, sub j (substrate r) ->
```

```
    ~ (exists u, greatest_induced r j u) ->
    has_least (Vg g (supremum r j))),
  (forall i x, inc i (substrate r) -> sub  x (substrate (Vg g i)) ->
    (exists u, upper_bound (Vg g i) x u) ->  nonempty x ->
    (exists u, least_upper_bound  (Vg g i) x u)) &
  (forall i, inc i (substrate r) ->
    ~ (has_greatest (Vg g i)) ->
    exists v, least_induced r (Zo (substrate r) (fun j =>
      glt r i j)) v /\ has_least (Vg g v))]).
```

(d) Assume that $\mathscr{A}(E,F)$ is a complete lattice. If E is non-empty, then F is a complete lattice, see Exercise 6 (c).

Let's show the converse. We consider a subset X of $\mathscr{A}(E,F)$, and for each $x \in E$ the set G_x of all $f(x)$ for $f \in X$. If F is a complete lattice, this set has a least upper bound, say f_x. This gives us a function $f : x \mapsto f_x$, which is the least upper bound. The function is increasing by the following argument: assume $a \le b$; we have $g(b) \le \sup_{h\in X} h(b)$ if $g \in X$; thus $g(a) \le g(b) \le f(b)$. Taking the supremum over g gives $f(a) \le f(b)$.

```
Lemma Exercise1_11f r r': (* 34 *)
  order r -> order r' ->  nonempty (substrate r) ->
  complete_lattice  (increasing_mappings_order r r') -> complete_lattice r'.
Lemma Exercise1_11g r r': (* 43 *)
  order r -> order r' ->
  complete_lattice r' -> complete_lattice  (increasing_mappings_order r r').
```

We prove now the following theorem of Tarski: let E be a complete lattice, and $f : E \to E$ an increasing function. Then the set of fixed points of f is a complete lattice.

Let Z be the set of fixed points of f and X a subset of Z. Consider w, the supremum of X in E, and A, the set of all t such that $w \le t$ and $f(t) \le t$, and z the infimum of A. The quantities w and z exist, since E is a complete lattice. We pretend that z is the supremum of X (considered as an ordered subset of Z). It is clear that $f(z)$ is a lower bound of A, so that $f(z) \le z$, and $f(f(z)) \le f(z)$. It is clear that w is a lower bound of A, so that $w \le z$, and $w \le f(w) \le f(z)$. It follows $f(z) \in A$, thus $z \le f(z)$; hence $f(z) = z$, and $z \in Z$. The result follows.

```
Lemma tarski1 r f: (* 58 *)
  complete_lattice r -> increasing_fun f r r ->
  complete_lattice (induced_order r (fixpoints f)).
```

---

12. Let Φ be a mapping of a set A into itself. Let $\mathfrak{F}$ be the subset of $\mathfrak{P}(A)$ consisting of all $X \subset A$ such that $f(X) \subset X$ for each $f \in \Phi$. Show that $\mathfrak{F}$ is a complete lattice with respect to the relation of inclusion.

Solution. Applying lemma `setU_sup1` shows that the union $\bigcup X$ of a family of sets is the least upper bound (The greatest lower bound is the intersection if non-empty, the set E otherwise).

```
Lemma Exercise1_12 E f: (* 13 *)
  function_prop f E E ->
  complete_lattice (sub_order (Zo (\Po E) (fun X =>
    sub (Vfs f X) X))).
```

---

13. Let E be an ordered set. A mapping f of E into itself is said to be a *closure* if it satisfies the following conditions: (1) f is increasing, (2) for each $x \in \mathrm{E}$, $f(x) \geq x$, (3) for each $x \in \mathrm{E}$, $f(f(x)) = f(x)$. Let F be the set of elements of E which are invariant under f.

(a) Show that for each $x \in \mathrm{E}$ the set F_x of elements $y \in \mathrm{F}$ such that $x \leq y$ is not empty and has a least element, namely $f(x)$. Conversely, if G is a subset of E such that, for each $x \in \mathrm{E}$, the set of all $y \in \mathrm{G}$ such that $x \leq y$ has a least element $g(x)$, then g is a closure and G is the set of elements of E that are invariant under g.

(b) Suppose that E is a complete lattice. Show that the greatest lower bound in E of any non-empty subset of F belongs to F.

(c) Show that if E is a lattice, then $f(\sup(x,y)) = f(\sup(f(x), f(y)))$ for each pair of elements x, y of E.

Note. The "non-empty" in (b) is unnecessary: the greatest lower bound of the empty set is the greatest element of E; it is clearly invariant by F.

(a) First part is trivial. Second part is a bit more complicated. Consider a subset G of E. For any $x \in \mathrm{E}$, let G_x be the set of upper bounds of x that are in G. Assume that this set has a least element. We deduce a function g such that $g(x)$ is the least element of G_x; in particular $x \leq g(x)$. The key relation is: if $x \in \mathrm{E}$ and $y \in \mathrm{G}$, if $x \leq y$ then $g(x) \leq y$. In particular, if $x \in \mathrm{G}$ then $g(x) = x$. The result follows.

(b) Assume that E is a complete lattice, let X be a subset of F, and y its greatest lower bound. For $x \in \mathrm{X}$ we have $y \leq x$, thus $f(y) \leq f(x) = x$, so that $f(y)$ is a lower bound and $f(y) \leq y$. Since $f(y) \leq y$, this shows that y is a fixed-point.

(c) Assume that E is a lattice, and consider two elements x, y of E. Let $z = \sup(x,y)$ and $\mathrm{T} = \sup(f(x), f(y))$. By Exercise 10 we have $\mathrm{T} \leq f(z)$. Since $x \leq f(x)$ and $y \leq f(y)$ we deduce $z \leq \mathrm{T}$. We deduce $f(z) \leq f(\mathrm{T}) \leq f(f(z)) = f(z)$. Thus $f(z) = f(\mathrm{T})$.

```
Definition closure  f r :=
  [/\ increasing_fun f r r,
  (forall x, inc x (substrate r) -> gle r x (Vf f x)) &
  (forall x, inc x (substrate r) -> Vf f (Vf f x) = Vf f x)].
Definition upper_bounds F r x := Zo F (fun y => gle x y).

Section Exercise1_13.
Variables r f: Set.
Hypothesis cf: closure f r.

Lemma Exercise1_13d x y: (* 13 *)
  lattice r ->
  inc x (substrate r) -> inc y (substrate r) ->
  Vf f (sup r x y)  = Vf f (sup r (Vf f x) (Vf f y)).
Lemma Exercise1_13c E (F := fixpoints f) : complete_lattice r ->
   sub E F -> inc (infimum r E) F.  (* 10 *)
Lemma Exercise1_13a x (F := fixpoints f):   (* 10 *)
  inc x (source f) -> least (induced_order r (upper_bounds F r x)) (Vf f x).
End Exercise1_13.

Lemma Exercise1_13b r G (* 40 *)
  (ir := fun x => induced_order r (upper_bounds G r x))
```

```
  (g := Lf (fun x => the_least (ir x)) (substrate r) (substrate r)):
  order r -> sub G (substrate r) ->
  (forall x, inc x (substrate r) ->  has_least (ir x)) ->
  (closure g r /\  (G = fixpoints  g)).
```

---

14. Let A and B be two sets, and let R be any subset of A × B. For each subset X of A (resp. each subset Y of B) let ρ(X) (resp. σ(Y)) denote the set of all $y \in B$ (resp. $x \in A$) such that $(x, y) \in R$ for all $x \in A$ (resp. $(x, y) \in R$ for all $y \in B$). Show that ρ and σ are decreasing mappings and that the mapping $X \to \sigma(\rho(X))$ and $Y \to \rho(\sigma(Y))$ are closures (Exercise 13) in $\mathfrak{P}(A)$ and $\mathfrak{P}(B)$ respectively (ordered by inclusion).

Note. The definition has to be corrected as: "For each subset X of A (resp. each subset Y of B) let ρ(X) (resp. σ(Y)) denote the set of all $y \in B$ (resp. $x \in A$) such that $(x, y) \in R$ for all $x \in X$ (resp. $(x, y) \in R$ for all $y \in Y$)."

Solution. The functions σ and ρ are trivially decreasing, so that there composition is increasing. We have $x \subset \sigma\rho x$ and $y \subset \rho\sigma x$. The same argument as in Proposition 2 of § 1, no. 5, shows $\rho\sigma\rho = \rho$ and $\sigma\rho\sigma = \sigma$.

```
Lemma Exercise1_14 A B R (* 86 *)
  (rho := fun X => Zo B (fun y => forall x,inc x X -> inc (J x y) R))
  (sigma := fun Y => Zo A (fun x => forall y, inc y Y -> inc (J x y) R))
  (fr:=Lf rho (\Po A) (\Po B))
  (fs:= Lf sigma (\Po B) (\Po A))
  (iA := subp_order A)
  (iB := subp_order B):
  sub R (A \times B) ->
  [/\ decreasing_fun fr iA iB, decreasing_fun fs iB iA,
      closure (compose fs fr) iA & closure (compose fr fs) iB].
```

---

15. (a) Let E be an ordered set, and for each subset X of E let ρ(X) (resp. σ(X)) denote the set of upper (resp. lower) bounds of X in E. Show that, in $\mathfrak{P}(E)$, the set $\tilde{E}$ of subsets X such that $X = \sigma(\rho(X))$ is a complete lattice, and that the mapping $i : x \to \sigma(\{x\})$ is an isomorphism (called *canonical*) of E onto an ordered subset E' of $\tilde{E}$ such that, if a family (x_ι) of elements of E has a least upper bound (resp. greatest lower bound) in E, the image of this least upper bound (resp. greatest lower bound) is the least upper bound (resp. greatest lower bound) in $\tilde{E}$ of the family of images of the x_ι. $\tilde{E}$ is called the *completion* of the ordered set E.

(b) Show that, for every subset X of E, σ(ρ(X)) is the least upper bound in $\tilde{E}$ of the subset $i(X)$ of $\tilde{E}$. If f is any increasing mapping of E into a complete lattice F, there exists a unique increasing mapping $\bar{f}$ of $\tilde{E}$ into F such that $f = \bar{f} \circ i$ and $\bar{f}(\sup Z) = \sup(\bar{f}(Z))$ for every subset Z of $\tilde{E}$.

(c) If E is totally ordered, show that $\tilde{E}$ is totally ordered.

Note. Statement (b) is wrong.

Solution. We start with some definitions. The ordering of E is denoted by r.

```
Definition up_bounds r X :=
  Zo (substrate r)(fun z => upper_bound r X z).
Definition lo_bounds r X :=
  Zo (substrate r)(fun z => lower_bound r X z).
Definition uplo_bounds r X := lo_bounds r (up_bounds r X).
Definition completion r:=
  Zo (\Po (substrate r)) (fun z => z = uplo_bounds r z).
Definition completion_order r := sub_order (completion r).
```

(a) Let $f = \sigma \circ \rho$. This is a closure (even when r is not an ordering), according to the previous exercise, but we give a direct proof. The function f is increasing, $x \subset f(x)$, $\sigma\rho\sigma = \sigma$, and $f(f(x)) = f(x)$. We deduce that, if $\mathrm{X} \subset \mathrm{E}$, then $\sigma(\mathrm{X})$ and $f(\mathrm{X})$ are in $\tilde{\mathrm{E}}$.

```
Lemma Exercise1_15a1 r A B: sub A B -> (* 2 *)
  sub (up_bounds r B) (up_bounds r A).
Lemma Exercise1_15a2 r A B: sub A B -> (* 2 *)
  sub (lo_bounds r B) (lo_bounds r A).
Lemma Exercise1_15a3 r A B: sub A B -> (* 1 *)
  sub (uplo_bounds r A) (uplo_bounds r B).
Lemma Exercise1_15a4 r A:  (* 2 *)
  sub A (substrate r) -> sub A (uplo_bounds r A).
Lemma Exercise1_15a5 r A: sub A (substrate r) -> (* 4 *)
  lo_bounds r (up_bounds r (lo_bounds r A)) = (lo_bounds r A).
Lemma Exercise1_15a6 r A: sub A (substrate r) -> (* 2 *)
  uplo_bounds r (uplo_bounds r A) = (uplo_bounds r A).
Lemma Exercise1_15a7 r A: sub A (substrate r) -> (* 2 *)
  inc (uplo_bounds r A) (completion r).
Lemma Exercise1_15a8 r A: sub A (substrate r) -> (* 2 *)
  inc (lo_bounds r A) (completion r).
```

Assume now that E is an ordered set, and assume that A has a least upper bound α. Then $\sigma(\mathrm{A})$ is the set of all x such that $x \le \alpha$. If B has a least upper bound β, then $\sigma(\mathrm{A}) = \sigma(\mathrm{B})$ is $\alpha = \beta$. In particular, if A and B are singletons we get $\mathrm{A} = \mathrm{B}$.

If E has a least element e, then $\{e\}$ is the least element of $\tilde{\mathrm{E}}$, otherwise $\emptyset$ is the least element of $\tilde{\mathrm{E}}$. In any case E is the greatest element.

```
Section Exercise1_15.
Variable r:Set.
Hypothesis or: order r.

Lemma Exercise1_15a9  x y: (* 7 *)
  inc x (substrate r) -> inc y (substrate r) ->
  (lo_bounds r (singleton x) = lo_bounds r (singleton y))
    -> x = y.
Lemma Exercise1_15a10 e: (* 14 *)
  least r e ->
  least (completion_order r) (singleton e).
Lemma Exercise1_15a11: (* 12 *)
  ~ (has_least r) ->
  least (completion_order r) emptyset.
Lemma Exercise1_15a12: (* 3*)
  has_least (completion_order r).
Lemma Exercise1_15a13: (* 5 *)
  greatest (completion_order r) (substrate r).
```

We show that the completion is a complete lattice. Given a family X_i, the least upper bound is $f(\bigcup X_i)$, the greatest lower bound is $f(\bigcap X_i)$. This is because f is increasing and $f(X_i) = X_i$. In the case of intersection, we have to consider the case where the family is empty.

```
Lemma Exercise1_15a14 X: (* 18 *)
  sub X (completion r) ->
  least_upper_bound (completion_order r) X (uplo_bounds r (union X)).
Lemma Exercise1_15a15 X: (* 17 *)
  sub X (completion r) -> nonempty X ->
  greatest_lower_bound (completion_order r) X
  (uplo_bounds r (intersection X)).
Lemma Exercise1_15a16:  (* 3 *)
  complete_lattice (completion_order r).
```

Let's study the property of $i(x) = \sigma(\{x\})$. We know that it is injective from E into $\tilde{E}$. It is an order isomorphism on its image.

```
Definition lobs z := lo_bounds r (singleton z).

Lemma Exercise1_15a17: (* 2 *)
  lf_axiom mobs (substrate r) (substrate (completion_order r)).
Lemma Exercise1_15a18a x: (* 1 *)
  inc x (substrate r) -> inc x (lobs x).
Lemma Exercise1_15a18: (* 11 *)
  order_morphism (Lf lobs (substrate r) (substrate (completion_order r)))
     r (completion_order r).
```

Let's study the links between i and bounds. In the case of the greatest lower bound, the empty family is an exception.

```
Lemma Exercise1_15a19 X x: (* 22 *)
  sub X (substrate r) -> least_upper_bound r X x ->
  least_upper_bound (completion_order r) (fun_image X lobs) (lobs x).
Lemma Exercise1_15a20  X x: (* 34 *)
  sub X (substrate r) -> greatest_lower_bound r X x ->
  greatest_lower_bound  (completion_order r)
  (fun_image X (fun z => lo_bounds r (singleton z)))
  (lo_bounds r (singleton x)).
```

(b) $\sigma(\rho(X)) = \sup i\langle X\rangle$ is easy.

```
Lemma Exercise1_15b1 X: (* 14 *)
  sub X (substrate r) ->
  least_upper_bound (completion_order r) (fun_image X lobs) (uplo_bounds r X).
End Exercise1_15.
```

Counter-example. Let $E = \{a, b, c\}$ ordered by $a < c$ and $b < c$ (the quantities a and b being non-comparable). There are four sets of upper bounds: E, $\{a, c\}$, $\{b, c\}$ and $\{c\}$. The associated sets of lower bounds are $\emptyset$, $\{a\}$, $\{b\}$, and E. The function i maps a, b and c to $\{a\}$, $\{b\}$ and E.

Let $F = \{0, 1, 2\}$ with its usual order, and $f : E \to F$, such that $f(a) = 0$, $f(b) = 1$ and $f(c) = 2$. This function is increasing.

Assume $f = g \circ i$. Then g maps $\{a\}$, $\{b\}$ and E onto 0, 1, and 2 respectively. Let $Z = \{\{a\}, \{b\}\}$. The supremum of Z in the completion is E; on the other hand, g maps the two elements of Z to 0 and 1, whose supremum is 1. Thus g does not satisfy (b).

In the code that follows, we shall use 0, 1 and 2, instead of a, b and c; so that f becomes the identity function.

```
Lemma Exercise1_15b2 (* 256 *)
  (E := tripleton \0c \1c \2c)
  (r1 := diagonal E \cup doubleton (J \0c \2c) (J \1c \2c))
  (r2 := Nint_cco \0c \2c)
  (f := identity E):
  [/\ [/\ order r1, substrate r1 = E &
     completion r1 =
     (doubleton (singleton \0c) (singleton \1c)) \cup (doubleton emptyset E)],
   complete_lattice r2, substrate r2 = E,
   increasing_fun f r1 r2 &
   ~(exists g,
     [/\  (increasing_fun g (completion_order r1) r2),
     (forall t, inc t E -> Vf f t = Vf g (lo_bounds r1 (singleton t)))&
     (forall Z, sub Z (completion r1) ->
       Vf g (supremum (completion_order r1) Z) =
       supremum r2 (Vfs g Z))])].
```

(c) If E is totally ordered, then $\tilde{E}$ is totally ordered. This follows from that fact that, if $X \in \tilde{E}$, $a \in X$ and $b \le a$ then $b \in X$.

```
Lemma Exercise1_15c r: total_order r -> (* 19 *)
  total_order (completion_order r).
```

¶ 16. A lattice E is said to be *distributive* if it satisfies the following two conditions

(D') $$\sup(x, \inf(y, z)) = \inf(\sup(x, y), \sup(x, z)),$$

(D") $$\inf(x, \sup(y, z)) = \sup(\inf(x, y), \inf(x, z))$$

for all x, y, z in E. A totally ordered set is a distributive lattice.

(a) Show that each of the conditions (D'), (D") separately implies the condition

(D) $$\sup(\inf(x, y), \inf(y, z), \inf(z, x)) = \inf(\sup(x, y), \sup(y, z), \sup(z, x))$$

for all x, y, z in E.

(b) Show that the condition (D) implies the condition

(M) $$\text{If } x \ge z, \text{ then } \sup(z, \inf(x, y)) = \inf(x, \sup(y, z))).$$

Deduce that (D) implies each of (D') and (D"), and hence that the three axioms (D), (D') and (D") are equivalent (to show, for example, that D implies D′, take the least upper bound of x and each side of (D) and use (M)).

(c) Show that each of the two conditions

(T') $$\inf(z, \sup(x, y)) \le \sup(x, \inf(y, z)),$$

(T") $$\inf(\sup(x, y), \sup(z, \inf(x, y))) = \sup(\inf(x, y), \inf(y, z), \inf(z, x)),$$

for all x, y, z in E is necessary and sufficient for E to be distributive. (To show that (T') implies (D"), consider the element

$$\inf(z, \sup(x, \inf(y, z))).)$$

Solution. The lemmas and the definitions are given in the main text.

Both conditions (D') and (D") imply (D): this is trivial.

If $z \le x$, then $\sup(x, z) = x$ and $\inf(x, z) = x$. Injecting these relations in (M) and simplifying further yields (D), so that (D) implies (M).

Let's show that that (D) implies (D'). Write (D) as $\alpha = \beta$. Set $a = \sup(x, \alpha)$. This the supremum of four terms, two of them being smaller than x. After simplification, we see that a is the LHS of (D'). Thus, we have to show $\sup(x, \beta) = \inf(\sup(x, y), \sup(x, z))$. Since β is a infimum we can apply (M) and get $\sup(x, \beta) = \inf(\sup(x, y), d)$. Applying (M) to d gives the result.

Exchanging inf and sup and applying (M) in the other way shows that (D) implies (D").

We prove here that (D") implies (T') and (T') implies (D'). The first claim is obvious. We first notice that in (D') the sup is smaller than the inf. Write (D') as $a = b$. Let $c = \inf(z, \sup(x, y))$. Applying (T') gives $a \le \sup(x, c)$ and $c \le b$, from which $a \le b$ follows. We also show that (D") is equivalent to (T"). One implication is easy. Conversely (T") (of the form $a = b$) implies (T') (of the form $a' \le b'$), since $a' \le a$ and $b \le b'$ are clear.

¶ 17. A lattice E which has a least element α is said to be *relatively complemented* if, for each pair of elements x, y of E such that $x \le y$, there exists an element x' such that $\sup(x, x') = y$ and $\inf(x, x') = \alpha$. Such an element x' is called a *relative complement* of x with respect to y.

*(a) Show that the set E of vector subspaces of a vector space of dimension ≥ 2, ordered by inclusion, is a relatively complemented lattice, but that if x, y are two elements of E such that $x \le y$, there exists in general several distinct complements of x with respect to y.$_*$

(b) If E is distributive and relatively complemented, show that if $x \le y$ in E, there exists a unique relative complement of x with respect to y. E is said to be a *Boolean lattice* if it is distributive and relatively complemented and if, moreover, it has a greatest element ω. For each $x \in$ E, let x^* be the complement of x with respect to ω. The mapping $x \to x^*$ is an isomorphism of E onto the ordered set obtained by endowing E with the opposite ordering, and we have $(x^*)^* = x$. If A is any set, then the set $\mathfrak{P}(A)$ of all subsets A, ordered by inclusion, is a Boolean lattice.

(c) If E is a *complete* Boolean lattice (Exercise 11), show that for each family (x_λ) of elements of E and each $y \in$ E we have

$$\inf(y, \sup_\lambda(x_\lambda)) = \sup_\lambda(\inf(y, x_\lambda)).$$

(Reduce to the case $y = \alpha$, and use the fact that if $\inf(z, x_\lambda) = \alpha$ for every index λ, then $z^* \ge x_\lambda$ for every λ).

Notes. (a) The stars surrounding part (a) express the fact this requires a notion not yet introduced, here the notion of a vector space. The complement of subspace X is any subspace Y such that the intersection is trivial and the union spans the whole space. It is easy to show that a complement exists and is not unique.

In (c), the hint "Reduce to the case $y = \alpha$" is strange.

Solution. We define now a relatively complemented set, a Boolean lattice, the complement and the standard complement (see below), and show that in a relatively complemented set, a complement does exists. Assume that α is the least element and ω the greatest element. We have $\sup(x,\alpha) = x$, $\inf(x,\omega) = x$, $\inf(x,\alpha) = \alpha$ and $\sup(x,\omega) = \omega$.

```
Definition complement_pr r x y x' :=
 [/\ inc x' (substrate r),sup r x x' = y & inf r x x' = the_least r].

Definition relatively_complemented r:=
  [/\ lattice r, has_least r &
  (forall x y, gle r x y -> exists x', complement_pr r x y x')].

Definition boolean_lattice r:=
  [/\ relatively_complemented r, has_greatest r &
  distributive_lattice3 r].

Definition the_complement r x y:=
  select (complement_pr r x y) (substrate r).

Definition standard_completion r x :=
  the_complement r x (the_greatest r).
Lemma least_greatest_pr1 r a: boolean_lattice r -> (* 2 *)
  inc a (substrate r) ->
  ( sup r (the_least r) a = a /\
    inf r a (the_greatest r) = a /\
    inf r (the_least r) a = (the_least r) /\
    sup r a (the_greatest r) = (the_greatest r)).
```

(b) Let's show that in a distributive and relatively complemented set, there exists a unique complement. Consider x, complemented by x' and x'', and apply relation (D) to these three quantities. We get

$$\sup(\alpha, \sup(\alpha, \inf(x', x''))) = \inf(y, \inf(y, \sup(x', x''));$$

This simplifies to $\inf(x', x'') = \sup(x', x'')$ and to $x' = x''$.

Let's call "standard completion" and denote by x^* the completion with the greatest element of E. By uniqueness of completion and commutativity of supremum and infimum, we have $(x^*)^* = x$.

```
Lemma Exercise1_17a r x y:    (* 18 *)
  relatively_complemented r ->
  distributive_lattice3 r -> gle r x y ->
  exists! x', complement_pr r x y x'.
Lemma the_complement_pr r x y: (* 7 *)
  relatively_complemented r -> distributive_lattice3 r -> gle r x y ->
  complement_pr r x y (the_complement r x y).
Lemma scompl_pr r x: (* 2 *)
  boolean_lattice r -> inc x (substrate r) ->
  complement_pr r x (the_greatest r) (standard_completion r x).
Lemma scompl_unique r x y: (* 6 *)
  boolean_lattice r -> inc x (substrate r) ->
  complement_pr r x (the_greatest r) y ->
  y = standard_completion r x.
Lemma scomplI r x: (* 4 *)
  boolean_lattice r -> inc x (substrate r) ->
  standard_completion r (standard_completion r x) = x.
```

Consider two elements x and y, their standard completion a and b. Let $c = \inf(a,b)$. We have $\inf(y,c) = \alpha$. We have $\sup(y,c) = \sup(y,a)$ (we use (D′)). Assume $x \le y$ so that $\sup(x,a) \le \sup(y,a)$. Since $\sup(x,a) = \omega$ we deduce $\sup(y,c) = \omega$. As a consequence, c is the standard completion of y, hence $b = c = \inf(a,b)$. This shows $b \le a$.

```
Lemma scompl_mon r x y: (* 25 *)
  boolean_lattice r -> gle r x y ->
  gle r (standard_completion r y) (standard_completion r x).
```

We show that $x \mapsto x^*$ is an isomorphism. This is obvious since it is increasing (for the order on E and its reverse) and involutive.

```
Lemma Exercise1_17b r: boolean_lattice r -> (* 14 *)
  order_isomorphism (Lf (standard_completion r) (substrate r)(substrate r))
  r (opp_order r).
```

We know that $\mathfrak{P}(A)$ is a lattice, where inf and sup are intersection and union. It is a boolean lattice, where x^* is the complement of x in A. We first show that $\emptyset$ is the least element, and A the greatest. We show distributivity via (T′).

```
Lemma Exercise1_17c A: (* 56 *)
  boolean_lattice (subp_order A) /\
    (forall x, inc x (\Po A) ->
      standard_completion (subp_order A) x = complement A x).
```

(c) We start with four formulas: $\inf(y, \sup(y^*, x)) = \inf(y, x)$, $\sup(y, \inf(y^*, x)) = \sup(y, x)$, and the same with y and y^* exchanged.

Consider a family x_λ, let $u = \sup_\lambda(\inf(y, x_\lambda))$ and $v = \inf(y, \sup_\lambda(x_\lambda)$. We have to show $v = u$. Let X be the range of the family x_λ and Y the set of all $\inf(y, x_\lambda)$. Then $u = \sup Y$ and $v = \inf(y, \sup X)$. Since E is a complete lattice, the two quantities $\sup X$ and $\sup Y$ are defined and are in E. The relation $u \le v$ is clear, and $v \le y$ is obvious. It follows $u \le y$.

Define $z = \sup(y^*, u)$. We have $\inf(y,z) = \inf(y, \sup(y^*, u)) = \inf(y,u) = u$. Similarly, if $z' = \sup(y^*, \sup X)$ we have $\inf(y, z') = v$. It suffices to show $z = z'$. The relation $z \le z'$ is clear.

For each λ, $\inf(y, x_\lambda) \in Y$, thus $\inf(y, x_\lambda) \le \sup Y$. Taking the supremum with y^* and simplifying gives $\sup(y^*, x_\lambda) \le z$, thus $x_\lambda \le z$ and $z' \le z$.

```
Lemma Exercise1_17d r x y: boolean_lattice r -> (* 12 *)
  inc x (substrate r) ->  inc y (substrate r) ->
  let ys := (standard_completion r y) in
  [/\ inf r y (sup r ys x) = inf r y x,
  sup r y (inf r ys x) = sup r y x,
  inf r ys (sup r y x) = inf r ys x &
  sup r ys  (inf r y x) = sup r ys x].
Lemma Exercise1_17e r x y: (* 42 *)
  boolean_lattice r -> complete_lattice r ->
  inc y (substrate r) -> sub x (substrate r)->
  inf r y (supremum r x)
   = supremum r (fun_image x (fun z => inf r y z)).
```

¶ 18. *Let A be a set with at least three elements, let $\mathscr{P}$ be the set of all partitions of A, ordered by the relation "ϖ is finer than ϖ'" between ϖ and ϖ' (no 1, Example 4). Show that $\mathscr{P}$ is a complete lattice (Exercise 11), is not distributive (Exercise 17), but is relatively complemented (To prove the last assertion, well-order the sets belonging to a partition.)*

Note. The stars indicate that this exercise requires material not yet introduced, here the existence of a well-ordering. This is in fact not necessary.

In what follows $a \le b$ will denote the relation "a is coarser than b"; this is the opposite relation of "a is finer than b". In all three statements of the Exercise, $\le$ can be replaced by $\ge$.

Given a partition A, we can consider the equivalence $\sim_A$ associated to A ($x \sim_A y$ if and only if x and y are in some z with $z \in A$). We have $A \le B$ if $x \sim_A y \implies x \sim_B y$. Then sup(A, B) is the partition associated to "$x \sim_A y$ and $x \sim_B y$". More generally, the supremum of a set of partitions is the partition associated to the disjunction of the associated equivalences. The infimum is a bit more complex to analyze. In the case of two arguments, inf(A, B) is the set of connected components of the relation "$x \sim_A y$ or $x \sim_B y$".

Solution. Consider the set of all $A \cap B$ for $A \in \varpi$ and $B \in \varpi'$. We have shown (in Part I of this report) that this is the least upper bound (for the "coarser" ordering for coverings). When we remove the empty set, we get the least upper bound of two partitions. We first recall the property of the covering intersection. We show here that this defines the supremum of two elements.

```
(*
  Lemma setI_covering2_P x y z:
    inc z (intersection_covering2 x y) <->
    exists a b, [/\ inc a x, inc b y &  a \cap  b = z].
*)

Definition intersection_partition2 u v :=
  (intersection_covering2 u v) -s1 emptyset.

Lemma disjoint_pr1 a b: (* 2 *)
  (forall x, inc x a -> inc x b -> a = b) ->
  disjointVeq a b.
Lemma intersection_is_partition2 u v x: (* 18 *)
  partition_s u x -> partition_s v  x ->
  partition_s (intersection_partition2 u v) x.
Lemma intersection_p2_comm:  u v:  (* 3 *)
  (intersection_partition2 u v) = (intersection_partition2 v u).
Lemma intersection_is_sup2_a u v x: (* 4 *)
  partition_s u x -> partition_s v  x ->
  gle (coarser x) u (intersection_partition2 u v).
Lemma intersection_is_sup2 u v x: (* 11 *)
  partition_s u x -> partition_s v  x ->
  least_upper_bound (coarser x)(doubleton u v)(intersection_partition2 u v).
Lemma intersection_is_sup2_b E u v: (* 7 *)
   partition_s u E -> partition_s v E ->
   sup (coarser E) u v = (intersection_partition2 u v).
```

Consider now a nonempty family F_i of partitions of X. For any element $x = (x_i)_i$ of $\prod F_i$, we can consider the subset $\bar{x} = \bigcap x_i$ of E. The set of all $\bar{x}$ (minus the empty set) is a partition of E. It is the least upper bound of the family. If follows that any non-empty set of partitions has a least upper bound. It follows that E is a complete lattice (it has a least element, which is empty if E is empty, {E} otherwise).

```
Definition intersection_partition f :=
  (fun_image (productb f) intersectionb) -s1 emptyset.

Lemma intersection_is_partition f x: (* 37 *)
  fgraph f ->  (allf f (partition_s ^~ x))  ->
  nonempty (domain f) ->
  partition_s (intersection_partition f) x.
Lemma intersection_is_sup_a f x y: (* 6 *)
  fgraph f ->  (allf f (partition_s ^~ x))  ->
  inc y (range f) ->
  gle (coarser x) y (intersection_partition f).
Lemma intersection_is_sup f x: (* 25 *)
  fgraph f ->  (allf f (partition_s ^~ x))  ->
  nonempty (domain f) ->
  least_upper_bound (coarser x) (range f) (intersection_partition f).
Lemma Exercise1_18a E:  complete_lattice (coarser E). (* 29 *)
```

Let's now show that $\mathscr{P}$ is not distributive. It is clear that we can use the coarser or finer order indifferently. If E is empty, or is a singleton, there is a unique partition. If E has two elements, there are two partitions, the least and the greatest ones. In these cases, the set is distributive.

Assume that E has at least three elements, x, y and z, and let $\mathrm{T} = \{x,y,z\}$ and $\mathrm{F} = \mathrm{E} - \mathrm{T}$. If U is a set, we denote by $\bar{\mathrm{U}}$ the quantity $\mathrm{U} \cup \{\mathrm{F}\} - \{\emptyset\}$. The empty set is not an element of $\bar{\mathrm{U}}$, and if the empty set is not an element of U we have $\bar{\mathrm{U}} = \mathrm{U}$ if F is empty and $\bar{\mathrm{U}} = \mathrm{U} \cup \{\mathrm{F}\}$ otherwise. If U is a partition of T, then $\bar{\mathrm{U}}$ is a partition of E. Let p_x be the partition of T formed of $\{x\}$ and $\{y,z\}$ and $\mathrm{P}_x = \bar{p}_x$. Define similarly P_y and P_z. Let $\alpha = \bar{a}$ and $\omega = \bar{o}$ where a is the greatest partition of T and o the least (a is a set containing three singletons and $\bar{o} = \{\mathrm{T}\}$). This gives five partitions of E. We show that $\mathscr{P}$ is not distributive by considering the application of (D′) to P_x, P_y and P_z. It says

$$\sup(\mathrm{P}_x, \inf(\mathrm{P}_y, \mathrm{P}_z)) = \inf(\sup(\mathrm{P}_x, \mathrm{P}_y), \sup(\mathrm{P}_x, \mathrm{P}_z)).$$

Since o is the greatest partition of T we get that that $\omega \le \bar{\mathrm{P}}$ whenever P is a partition of T. Assume now a is a partition such that $a \le \mathrm{P}_y$ and $a \le \mathrm{P}_z$; then a contains two sets b and b' such that $\{x,y\} \subset b$ and $\{x,z\} \subset b'$. These sets are equal, since they cannot be disjoint. We deduce $a \le \omega$ so that $\inf(\mathrm{P}_y, \mathrm{P}_z) = \omega$. Since $\omega \le \mathrm{P}_x$ we get

$$\mathrm{P}_x = \inf(\sup(\mathrm{P}_x, \mathrm{P}_y), \sup(\mathrm{P}_x, \mathrm{P}_z)).$$

Assume $\{a,b\} = \{x,y\}$. Let p_t be the set containing $\{a\}$ and $\{b,z\}$, and let $\mathrm{P}_t = \bar{p}_t$. Then P_t is one of P_y and P_z, thus is a partition. Since P_x and P_t have three elements each the intersection has nine elements, some being empty. A tedious study shows that the intersection is α. We deduce $\sup(\mathrm{P}_x, \mathrm{P}_y) = \sup(\mathrm{P}_x, \mathrm{P}_z) = \alpha$ hence $\mathrm{P}_x = \alpha$. This is absurd.

```
Lemma Exercise1_18b E: \3c <=c (cardinal E) -> (* 195 *)
  ~ (distributive_lattice2 (coarser E)).
```

Let's characterize the finest partition (it is α such that for all x, x is coarser than α, or α is finer than x).

```
Lemma Exercise1_18c E: (* 9 *)
  greatest_partition E =  the_greatest (coarser E).
```

Let's show that the set is relatively complemented for the "finer" ordering. If $\le$ denotes "coarser" we have to show: for all X and Y such that $Y \le X$, there exists X' such that $\inf(X, X') = Y$ and $\sup(X, X') = \alpha$.

Bourbaki hints to well-order the elements; this is not needed; however, we use the axiom of choice that asserts that for any $a \in X$ there is an element e_a such that $e_a \in a$ (recall that a in non-empty). Define P_v to be the set of all e_a for $a \in X$ and $a \subset v$. We consider the set Z formed of all P_v for $v \in y$ and all singletons $S_b = \{b\}$, with $b \in E$, that are not of the form e_a for $a \in X$.

Obviously, $S_b \subset E$ and $P_v \subset E$. Consider $x \in E$. There is a such that $x \in a \in X$, and $b \in Y$ such that $a \subset b$. Assume $x = e_c$ for some $c \in X$. We deduce $x \in c \in X$, hence $a = c$ and $x \in P_b$. Otherwise $x \in S_x$, and $S_x \in Z$. Thus $\bigcup Z = E$.

Assume $v \in Y$. There is x such that $x \in v \in Y$, and some u such that $x \in u \in X$. There is also some $w \in Y$ such that $u \subset w$. This gives $x \in w \in Y$ hence $v = w$, thus $u \subset v$. We have $e_u \in P_v$, so that P_v is non-empty.

The elements of Z are mutually disjoint. This is obvious if they have the form S_b, since S_b is a singleton. By construction if $S_b \in Z$ then b is in no P_v. Assume $x \in P_u \cap P_v$. Then $x = e_a = e_b$, with $a \subset u$ and $b \subset v$. The two sets u and v contain x, thus cannot be disjoint, thus are equal and $P_u = P_v$.

We have $Y \le Z$. Given a singleton $S_b \in Z$, there is u such that $b \in u \subset Y$, thus $S_b \subset u$. On the other hand, $P_v \subset v$. Since $Y \le X$, we deduce $Y \le W$, where $W = \inf(X, Z)$.

Conversely, we have $W \le Y$. Consider an element a of Y. We have $P_a \in Z$. Since $W \subset Z$ there exists $b \in W$ such that $P_a \subset b$. Consider $t \in a$. Since $t \in E$, there is c such that $t \in c \in X$, and d_1 such that $c \subset d_1 \in Y$. It follows $d_1 = a$, $c \subset a$. We deduce $e_c \in P_a$, thus $e_c \in b$. Since $c \in X$ and $W \le X$ there exists $d \in W$ such that $c \subset d$. Obviously, $e_c \in d$, so that b and d are not disjoint. It follows $b = d$, thus $c \subset b$, and $t \in b$.

Let's show $\sup(X, Z) = \alpha$. We have to show that the intersection of X and Z contains all singletons and nothing else. Consider the intersection of an element a of X and an element b of Z. The result is obvious if b is a singleton. Assume $b = P_v$. If $t \in a \cap b$ then $t = e_c$ for some $c \in X$ such that $c \subset v$. Since t is in a and c, these two sets cannot be disjoint, thus are equal, and $t = e_a$. Thus $a \cap b$ is empty or is a singleton. Conversely, consider an element x in E. There is a such that $x \in a \in X$, and b such that $a \subset b \in Y$. If $x = e_a$, the previous argument says that $a \cap P_b = \{x\}$. Otherwise we know $S_x \in Z$, $x \cap S_x = \{x\}$.

```
Lemma Exercise1_18d E x y: (r := coarser E); (* 130 *)
  gle r y x -> exists x',
  [/\ inc X'(substrate r), inf r X X' = Y & sup r X X' = greatest_partition E].
```

---

19. An ordered set E is said to be *without gaps* if it contains two distinct comparable elements and if, for each pair of elements x, y such that $x < y$, the open interval $]x, y[$ is not empty. Show that the ordinal sum $\sum_{\iota \in I} E_\iota$ (Exercise 3) is without gaps if and only if the following conditions are satisfied:

(I) Either I contains two distinct comparable elements, or else there exists $\iota \in I$ such that E_ι contains two distinct comparable elements.

(II) Each E_ι which contains at least two distinct comparable elements is without gaps.

(III) If α, β are two elements of I such that α < β and if the interval]α,β[in I is empty, then either E_α has no maximal element or else E_β has no minimal element.

In particular, every ordinal sum $\sum_{\iota\in I} E_\iota$ of sets without gaps is itself without gaps, provided that no E_ι has a maximal element (or provided that no E_ι has a minimal element). If I is without gaps, and if each E_ι is either without gaps or contains no two distinct comparable elements, then $\sum_{\iota\in I} E_\iota$ is without gaps.

Note. We assume that no set E_ι is empty.

Solution. Assume the sum without gaps. Condition (I) says that the sum has two distinct comparable elements. Condition (II) is immediate. Consider now condition (III). Consider a maximal element x of E_α, a minimal element y of E_β, where α < β. We have $x < y$ (considered as elements of the sum); hence there is z such that $x < y < z$. It index cannot be α nor β, hence there is a γ such that α < γ < β.

Converse. We use part (I) to show that there are at least two comparable elements in the sum. Consider two elements x and y such that $x < y$. Assume $x \in E_\alpha$ and $y \in E_\beta$. There are two cases to consider: if α < β, we conclude by (III), otherwise, α = β and $x < y$ in E_α and we conclude by (II).

```
Definition without_gaps r :=
  [/\ order r, (exists x y, glt r x y) &
  (forall x y, glt r x y -> exists2 z, glt r x z & glt r z y)].

Section Exercise1_19.
Variables (r g: Set).
Hypotheses  (ax:orsum_ax r g) (ax2: orsum_ax2 g).

Lemma Exercise1_19a: (* 114 *)
  (without_gaps (order_sum r  g) <->
   [/\ (exists i j, glt r i j) \/
        (exists i x y, inc i (substrate r) /\ glt (Vg g i) x y),
    (forall i x y, inc i (substrate r) ->  glt (Vg g i) x y ->
      without_gaps (Vg g i))
    & (forall i j, glt r i j ->
      [\/ (exists2 k, glt r i k & glt r k j),
        (forall u, ~ (maximal (Vg g i) u)) |
        (forall u, ~ (minimal (Vg g j) u))]])].
```

Second claim. We assume the index set non-empty (otherwise the sum is empty). Last claim. The condition "Each E_ι is either without gaps or contains no two distinct comparable elements", is nothing else than (II).

```
Lemma Exercise1_19b: (* 5 *)
  ne_substrate r ->
  (forall i u, ~ (maximal (Vg g i) u)) ->
  (forall i, inc i (substrate r) -> without_gaps (Vg g i)) ->
  without_gaps (order_sum r g).
Lemma Exercise1_19c: (* 5 *)
  ne_substrate r ->
  (forall i u, ~ (minimal (Vg g i) u)) ->
  (forall i, inc i (substrate r) -> without_gaps (Vg g i)) ->
  without_gaps (order_sum r g).
Lemma Exercise1_19d: (* 6 *)
  without_gaps r ->
  (forall i, inc i (substrate r) ->
```

```
      (without_gaps (Vg g i) \/
        (forall x y, inc x (substrate (Vg g i)) -> inc y (substrate (Vg g i))
            -> ~ (glt (Vg g i) x y))))
    ->  without_gaps (order_sum r g).
End  Exercise1_19.
```

¶ 20. An ordered set E is said to be *scattered* if no ordered subset of E is without gaps (Exercise 19). Every subset of a scattered set is scattered. *Every well-ordered set of more than one element is scattered.*

(a) Suppose that E is scattered. Then if x, y are any two elements of E such that $x < y$, there exists two elements x', y' of E such that $x \le x' < y' \le y$, and such that the interval $]x', y'[$ is empty. *Give an example of a totally ordered set which satisfies this condition and is not scattered (consider Cantor's triadic set).*

(b) An ordinal sum $\sum_{\iota\in \mathrm{I}} \mathrm{E}_\iota$ (where neither I nor any E_ι is empty) is scattered if and only if I and each E_ι is scattered. (Note that E contains a subset isomorphic to I and that every subset F of E is the ordinal sum of those sets $\mathrm{F} \cap \mathrm{E}_\iota$ which are non-empty; finally use Exercise 19.)

Note. The stars indicates that the exercise uses properties not yet defined, for instance the notion of well-ordered sets, and the Cantor set. We shall use an ad-hoc replacement here. In (a) the condition "of more than one element" is unnecessary (since the empty set, or singletons are not without gaps). In (b), the condition I non-empty is unnecessary.

Solution. By definition, an ordering $\le$ is scattered, if whenever F is a subset of its substrate, then $\le_{\mathrm{F}}$, the ordering induced by r on F is without gaps. If we express $\le_{\mathrm{F}}$ in terms of $\le$ we get: whenever F has at least two elements such that $x < y$, then there are two elements x and y, such that $x < y$ and no z in F satisfies $x < z < y$. Clearly, if $\le$ is scattered, so is $\le_{\mathrm{F}}$. If F is well-ordered, $a < b$, then a has a successor c and the interval $]a, c[$ is empty. If E is well-ordered, every subset is well-ordered, so E is scattered.

```
Definition scattered r := order r /\
  (forall x, sub x (substrate r) -> ~ (without_gaps (induced_order r x))).

Lemma Exercise1_20a r x: (* 3 *)
  sub x (substrate r) -> scattered r -> scattered (induced_order r x).
Lemma Exercise1_20b r: worder r -> scattered r. (* 15 *)
```

(a) Assume $\le$ scattered and $x < y$. Take for F the interval $[x, y]$. We deduce the existence of x' and y' such that $x \le x' < y' \le y$ and t there is no z such that $x' < z < y'$.

```
Definition Exercise1_20_prop r:=
forall x y, glt r x y ->
  exists x', exists y',
   gle r x x' /\ glt r x' y'/\  gle r y' y /\
    (forall z, ~ (glt r x' z /\ glt r z y')).

Lemma Exercise1_20c  r: (* 19 *)
  scattered r -> Exercise1_20_prop r.
```

Recall that the Cantor Set is the subset of the set of real numbers that can be written as $\sum_i x_i/3^{i+1}$, with $x_i = 0$ or $x_i = 2$, thus is order isomorphic to $\{0,2\}^{\mathbf{N}}$, where functional graphs are lexicographically ordered. Instead of $\{0,2\}$ we shall use the canonical doubleton, and denote its elements by α and β. The important property is that $f < g$ if and only if there is an index i such that $f_i = \alpha$, $g_i = \beta$, and $f_j = g_j$ for $j < i$.

```
Definition cantor_tri_aux := cst_graph Nat canonical_doubleton_order.
Definition cantor_tri_order:= order_prod Nat_order cantor_tri_aux.
Definition cantor_tri_sub:= productb (cst_graph Nat C2).

Lemma cantor_tri_order_axioms: orprod_ax Nat_order cantor_tri_aux.  (* 4 *)
Lemma cantor_tri_order_total : total_order cantor_tri_order. (* 3 *)
Lemma cantor_tri_order_sr1 : (* 2 *)
  prod_of_substrates cantor_tri_aux = cantor_tri_sub.
Lemma cantor_tri_order_sr : (* 3 *)
 substrate cantor_tri_order = cantor_tri_sub.
Lemma cantor_tri_order_gltP x x': (* 22 *)
  glt cantor_tri_order x x' <->
  [/\ inc x cantor_tri_sub,  inc x' cantor_tri_sub &
    exists j, [/\ natp j,
      (forall i, natp i -> i <c  j -> Vg x i = Vg x' i),
      Vg x j = C0 & Vg x' j = C1]].
```

Define $f_{k\gamma}$ to be the function that maps i to f_i if $i \le k$ and to γ otherwise. Assume $f < g$, let i be as above. Then $f \le f_{i\beta} < g_{i\alpha} \le g$, and there is no z between $f_{i\beta}$ and $g_{i\alpha}$. [note: consider f, g, etc, as real numbers; there are real numbers x, y and an integer t such that $3^i f = t + x$, $3^i g = t + y$, $0 \le x \le 1/3$ and $2/3 \le y \le 1$; we have $3^i f_{i\beta} = t + 1/3$, $3^i g_{i\alpha} = t + 2/3$]

The Cantor set is not scattered. Let F be the set of elements not of the form $f_{i\alpha}$. This set is without gaps. It contains for instance the sequence associated to the real numbers 1/3 and 1, which are distinct and comparable. Assume $f < g$, and let i be as above. Since $g \in F$ there is an index k such that $i < k$ and $g(k) = \beta$. Let h be like g except $h(k) = \alpha$. Then $f < h < g$ and $h \in F$.

```
Lemma  Exercise1_20d: Exercise1_20_prop cantor_tri_order. (* 67 *)
Lemma  Exercise1_20e: ~ (scattered cantor_tri_order).  (* 66 *)
```

(b) Consider an ordinal sum $\Sigma = \sum E_\iota$. We identify $x \in E_\iota$ with $\bar{x} \in \Sigma$; note that $\iota = \mathrm{pr}_2 \bar{x}$. Since E_ι is non-empty, we may assume $e_\iota \in E_\iota$.

Assume Σ scattered. We consider a subset without gaps X of I, and let $\bar{W}$ the set of all $\bar{e}_i$ for $i \in X$. Since X has two distinct comparable elements, the same holds for $\bar{W}$. Assume $\bar{x} < \bar{y}$ in $\bar{W}$, with indices ι and κ (in X). We have $\iota < \kappa$ so that there is $\lambda \in X$ with $\iota < \lambda < \kappa$. Then $\bar{x} < \bar{e}_\lambda < \kappa$. Thus $\bar{W}$ is without gaps, absurd. The set E_ι is obviously scattered.

Converse. Assume I and each E_ι scattered, and let Σ' be a subset without gaps of Σ. Let I' be the set of $\iota \in I$ such that $\Sigma' \cap E_\iota$ is nonempty, ordered by the ordering of I. Let E'_ι be the set of $x \in E_\iota$ such that $\bar{x} \in \Sigma'$. It is non-empty if $\iota \in I'$. We order it by the ordering induced from E_ι. Consider now the ordinal sum $\sum_{I'} E'_\iota$. Its substrate is Σ', and its ordering is that of Σ. We apply Exercise 19. By condition (II), no two distinct elements of E_ι are comparable. Thus all elements are maximal and minimal; this contradicts the conclusion of (III). Thus (III) reads: if $\alpha < \beta$ in I', then the interval $]\alpha, \beta[$ is non-empty; together with (I), this says that I' is without gaps and I is not scattered.

```
Lemma  Exercise1_20f r g: (* 145 *)
  orsum_ax r g ->  orsum_ax2  g ->
```

```
  (scattered (order_sum r g) <->
    (scattered r /\ forall i, inc i (domain g) -> scattered (Vg g i))).
```

---

21. Let E be a non-empty totally ordered set, and let $S\{x, y\}$ be the relation "the closed interval with endpoints x, y is scattered" (Exercise 20). Show that S is an equivalence relation which is weakly compatible (Exercise 2) in x and y with the order relation on E, that the equivalence classes with respect to S are scattered sets, and that the quotient ordered set E/S is either without gaps or else consists of a single element. Deduce that E is isomorphic to an ordinal sum of scattered sets whose index set is either without gaps or else consists of a single element.

Note. There is no need to assume E non-empty.

Complements to Exercise 1.2. Assume that both conditions (C) and (C') are satisfied and that $\le$ is a total order on E. Then the quotient order E/S is totally ordered, and $X \le Y$ in the quotient is equivalent to $x \le y$ whenever $x \in X$ and $y \in Y$, where x and y are the representatives of X and Y for x and y. If x and y are in E, X and Y are their classes, then $x \le y$ implies $X \le Y$ and $X < Y$ implies $x < y$.

```
Lemma Exercise1_2g r s: (* 21 *)
  weak_order_compatibility r s->
  Ex1_2_hC' r s -> total_order r ->
  let r' := (quotient_order r s) in
    forall x y, gle r' x y <-> [/\ inc x (quotient s) , inc y (quotient s)&
       gle r (rep x) (rep y)].
Lemma Exercise1_2h r s: (* 9*)
  weak_order_compatibility r s->
  Ex1_2_hC' r s -> total_order r ->
  total_order (quotient_order r s).
Lemma Exercise1_2j r s: (* 12 *)
  weak_order_compatibility r s->
  Ex1_2_hC' r s -> total_order r ->
  let r' := (quotient_order r s) in
    forall x y, inc x (substrate r ) -> inc y (substrate r) ->
      ((gle r x y -> gle r' (class s x) (class s y))
      /\ (glt r' (class s x) (class s y) -> glt r x y)).
```

Let I be the quotient set E/S with its ordering. Consider the identity function on I, and write it $i \mapsto E_i$. Each E_i is ordered by the ordering induced from E. Let f be the canonical mapping $E \to \sum E_i$ (it associates to $x \in E$ the pair (x, i) where $x \in E_i$, obviously i is the class of x for S). This is an order isomorphism.

```
Lemma Exercise1_2i r s (* 71 *)
  (q := quotient s)
  (r' := quotient_order r s)
  (f' := diagonal q)
  (g' := Lg q (fun z => induced_order r z))
  (du := disjointU f')
  (f :=  Lf (fun x => J x (class s x)) (substrate r) du):
  weak_order_compatibility r s->
  Ex1_2_hC' r s -> total_order r ->
```

```
  [/\ orsum_ax r' g',
    (forall i, inc i (domain g') -> nonempty (substrate (Vg g' i))),
    substrate (order_sum r'  g') = du,
    (forall x y, inc x (substrate r) -> inc y (substrate r) ->
      (related s x y <->
        related (equivalence_associated (second_proj du))  (Vf f x) (Vf f y)))&
     order_isomorphism f r (order_sum r' g')].
```

Solution. The condition "without gaps", simplified to WG, is the conjunction of three conditions: WG0 that says that $(E, \leq)$ is an ordered set, WG1 that says that there are x and y such that $x < y$ and WG2 that says that if $x < y$, then there is z such that $x < z < y$. The relation S is either $x \leq y$ and $[x, y]$ is scattered, or the same with x and y exchanged. Note that V scattered means that, if $U \subset V$, it is not without gaps for the ordering of E (which is the ordering of V).

In a totally ordered set, condition WG1 is: "there are two distinct elements in U", and its negation as "there is at most one element in U". We can now say: U is not without gaps if and only if either U is a small set, or it contains $a < b$ such that $]a, b[$ is empty.

```
Definition scattered_rel r x y :=
  (gle r  x y /\  scattered (induced_order r (interval_cc r x y)))
  \/ (gle r  y x /\  scattered (induced_order r (interval_cc r y x))).

Definition scattered_equiv r := graph_on (scattered_rel r) (substrate r).

Lemma Exercise1_21aP r v: order r -> (* 3 *)
  sub v (substrate r) ->
  (scattered (induced_order r v) <->
    (forall u, sub u v ->  ~ without_gaps (induced_order r u))).
Lemma Exercise1_21bP r u: total_order r -> (* 5 *)
  sub u  (substrate r) ->
  ((exists x y, glt (induced_order r u) x y) <->
    (exists x y, [/\ inc x u, inc y u & x<> y])).
Lemma Exercise1_21cP r u: total_order r -> (* 5 *)
  sub u  (substrate r) ->
  ((forall a b, inc a u -> inc b u -> a = b) <->
    ~ (exists x y, glt (induced_order r u) x y)).
Lemma Exercise1_21dP r u: total_order r -> (* 23 *)
  sub u (substrate r) ->
  ((~ without_gaps (induced_order r u)) <->
  ((forall a b, inc a u -> inc b u -> a = b)
      \/ (exists a b, [/\ inc a u, inc b u, glt r a b &
        (forall z, inc z u -> gle r z a \/ gle r b z)]))).
```

Assume that U is without gaps, and consider two elements a, b of U such that $a < b$. The intersection $U \cap [a, b]$ is without gaps.

```
Lemma Exercise1_21e r u a b: total_order r -> (* 19 *)
  let v:= u \cap (interval_cc r a b) in
  sub u (substrate r) -> without_gaps (induced_order r u) ->
  (exists x y,  [/\ inc x v, inc y v & glt r x y]) ->
  without_gaps (induced_order r v).
Lemma Exercise1_21f r a b u: total_order r -> (* 3 *)
  sub u (substrate r) ->
  inc a u -> inc b u -> glt r a b -> without_gaps (induced_order r u) ->
  without_gaps (induced_order r (u \cap (interval_cc r a b))).
```

The union of two scattered intervals $[x, y]$ and $[y, z]$ is scattered. Proof by contradiction. Let u be a subset without gaps of the union, consider the intersection u_1 and u_2 with the two intervals. They are not without gaps.

Let's show that S is an equivalence. Symmetry is true by definition; reflexivity is a consequence of the fact that singletons are scattered. Transitivity is a consequence of the previous result if $x \le y \le z$, and if $x \le z \le y$ it is a consequence of 20a.

```
Lemma Exercise1_21g r x y z: total_order r -> (* 39 *)
  gle r x y ->  gle r y z->
  scattered (induced_order r (interval_cc r x y)) ->
  scattered (induced_order r (interval_cc r y z)) ->
  scattered (induced_order r (interval_cc r x z)).
Lemma Exercise1_21h r: total_order r -> (* 41 *)
  equivalence_re (scattered_rel r) (substrate r).
Lemma Exercise1_21i r: total_order r -> (* 2 *)
  equivalence (scattered_equiv r).
Lemma Exercise1_21j r: total_order r -> (* 2 *)
  substrate (scattered_equiv r) = substrate r.
```

We simplify a bit the definition of being related by this relation. We first consider a mix of 21a and 21d.

```
Definition scattered_aux1 r x y :=
  (forall u, sub u (interval_cc r x y) ->
    ((forall a b, inc a u -> inc b u -> a = b)
      \/ (exists a b, [/\ inc a u, inc b u, glt r a b &
        (forall z, inc z u -> gle r z a \/ gle r b z)]))).
Definition scattered_aux r x y :=
  gle r x y /\ scattered_aux1 r x y.

Lemma Exercise1_21kP r x y: total_order r -> (* 11 *)
  gle r  x y ->
  (scattered (induced_order r (interval_cc r x y)) <->
   scattered_aux1 r x y).
Lemma Exercise1_21l r x y: total_order r -> (* 7 *)
  (related (scattered_equiv r) x y <->
    (scattered_aux r x y \/ scattered_aux r y x)).
```

Let's show weak compatibility. Assume that x and x' and related, and $x \le y$. We want to find y such that $x' \le y'$ and y is related to y'. If $x' \le y$ we may choose $y = y'$. Otherwise we have $x \le y \le x'$, and we chose $y = x'$. The interval $[y, x']$ is scattered, as a subset of a scattered set.

```
Lemma Exercise1_21m r: total_order r -> (* 15 *)
  weak_order_compatibility r (scattered_equiv r).
```

Let's show that equivalence classes are scattered. Let X be a subset of an equivalence class, assumed without gaps. It has two elements a and b such that $a < b$; and for every $c < d$, there is e such that $c < e < d$. These elements a and b are related so that the interval $[a, b]$ is scattered. Let Y be the intersection of X and the interval $[a, b]$. It is not without gaps, but has two comparable elements, namely a and b, hence there exists c, d such that $]c, d[\cap Y$ is empty. But there is an $e \in]c, d[\cap X$. This element is clearly in $[a, b]$ hence in Y, absurd.

```
Lemma Exercise1_21n r x: (* 37 *)
  total_order r -> inc x (substrate r) ->
  scattered (induced_order r (class (scattered_equiv r) x)).
```

Since the equivalence S is weakly compatible with the order, it induces a preorder on the quotient. We show here that it is an order (cf. Exercise 1.2).

```
Lemma Exercise1_21o r: total_order r -> (* 10 *)
  Ex1_2_hC' r (scattered_equiv r).
Lemma Exercise1_21p r: total_order r ->  (* 3 *)
  order (quotient_order r (scattered_equiv r)).
```

Let's show that the quotient ordered set is either without gaps or a small set (empty or containing a single element). All we need to show is that if $X < Y$ in the quotient, there is Z such that $X < Z < Y$. Let x and y be the representatives of X and Y. They are not related by the equivalence S, hence $[x, y]$ is not scattered. We use contradiction, assume there is a set U with at least two elements, such that no interval is empty. Let $a \in U$. We have $x \le a \le y$, so that classes are in the same order. Let A be the class of a, so that $X \le A \le Y$. By assumption, one $\le$ is equality. This implies $a \in X$ or $a \in Y$.

Let X_1 and X_2 be the intersections of U with X and Y. These sets have at most one element, for otherwise, we could find an empty interval $]a, b[$. This interval has a point c in U, which is in X or in Y. If we consider X_1, $c \le b$, $b \in X$ and $c \in Y$ would imply $Y \le X$, absurd.

Now U has three points, and is the union of two sets with at most one point, absurd.

```
Lemma Exercise1_21q r: total_order r -> (* 100 *)
  let r' := quotient_order r  (scattered_equiv r) in
    small_set (substrate r') \/ without_gaps r'.
```

The last result is trivial.

```
Lemma Exercise1_21r r: total_order r -> (* 13 *)
  exists r' g',
  [/\ orsum_ax r' g', orsum_ax2 g',
    r \Is  (order_sum r' g'),
    (small_set (substrate r') \/ without_gaps r') &
    allf g' scattered].
```

¶ 22. (a) Let E be an ordered set; A subset U of E is said to be *open* if for each $x \in U$, U contains the interval $[x, \rightarrow[$. An open set U is said to be *regular* if there exists no open set $V \supset U$, distinct from U such that U is cofinal in V. Show that every open set U is cofinal in exactly one regular open set $\bar{U}$. The mapping $U \to \bar{U}$ is increasing. If U, V are two open sets such that $U \cap V = \emptyset$, then also $\bar{U} \cap \bar{V} = \emptyset$.

(b) Show that the set R(E) of regular open sets of E, ordered by inclusion, is a complete Boolean lattice (Exercise 17). For R(E) to consist of two elements, it is necessary and sufficient that E should be non-empty and *right directed*.

(c) If F is a cofinal subset of E, show that the mapping $U \to U \cap F$ is an isomorphism of R(E) onto R(F).

(d) If E_1, E_2 are two ordered sets, then every open set in $E_1 \times E_2$ is of the form $U_1 \times U_2$, where U_i is open in E_i ($i = 1,2$). The set $R(E_1 \times E_2)$ is isomorphic to $R(E_1) \times R(E_2)$.

Note. Both claims in (d) are wrong, see below.

Solution. We start with some definitions and a trivial lemma. Note that $V \supset U$ follows from U is cofinal in V so is omitted in the definition of regular.

```
Definition open_o r u:=
  sub u (substrate r) /\ forall x y, inc x u -> gle r x y -> inc y u.
Definition open_r r u:=
  open_o r u /\
  forall v, open_o r v ->  cofinal (induced_order r v) u -> u = v.

Definition open_bar r u :=
  union (Zo (\Po(substrate r))
    (fun z => open_o r z /\ cofinal (induced_order r z) u)).

Definition reg_opens r := Zo (\Po (substrate r)) (open_r r).
Definition reg_open_order r := sub_order (reg_opens r).

Lemma inf_pr2 r x y z: (* 3 *)
  order r -> gle r z x  -> gle r z y ->
  (forall t, gle r t x -> gle r t y -> gle r t z) ->
  inf r x y = z.
```

(a) We start by showing the the union and intersection of open sets is open (note that the empty intersection is empty, hence open).

```
Section Exercise1_22.
Variable r:Set.
Hypothesis or: order r.

Lemma reg_opens_i x: open_r r x -> inc x (reg_opens r).
Lemma Exercise1_22a u1 u2: (* 4 *)
  open_o r u1 -> open_o r u2 -> open_o r (u1 \cup u2).
Lemma Exercise1_22b u:  (* 12 *)
  alls u (open_o r) ->  open_o r (intersection u).
Lemma Exercise1_22c u: (* 4 *)
  alls u (open_o r) ->  open_o r (union u).
```

Assume U is cofinal in U_1 and U_2, which are regular open. if $U_3 = U_1 \cup U_2$, regularity of U_1 shows $U_3 = U_1$. Similarly $U_3 = U_2$.

```
Lemma cofinal_inducedP u:  sub u (substrate r) -> forall v,
  ( cofinal (induced_order r u) v <->
    (sub v u /\ (forall x, inc x u -> exists2 y, inc y v & gle r x y))).
Lemma Exercise1_22d x u1 u2: (* 19 *)
  open_o r x -> open_r r u1 -> open_r r u2 ->
  cofinal (induced_order r u1) x -> cofinal (induced_order r u2) x  ->
  u1 = u2.
```

Let $\bar{U}$ be the union of all open sets V containing U in which U is cofinal (since $V = U$ is possible, we have $U \subset \bar{U}$). U is cofinal in $\bar{U}$.

```
Lemma Exercise1_22e u: (* 3 *)
  open_o r u ->  sub u (open_bar r u).
Lemma Exercise1_22f u: (* 2 *)
  open_o r u ->  sub (open_bar r u) (substrate r).
Lemma Exercise1_22g u: (* 4 *)
  open_o r u ->
  cofinal (induced_order r (open_bar r u)) u.
```

Consider $x \in \mathrm{E}$, such that whenever $x \le y$, y is bounded above by an element of U. Then $x \in \bar{\mathrm{U}}$ (consider $\mathrm{U} \cup [x, \to[$). This criterion will be used a lot.

Assume $\bar{\mathrm{U}}$ cofinal in V. If $x \in \mathrm{V}$ and $x \le y$, then $y \in \mathrm{V}$ and is bounded by $z \in \bar{\mathrm{U}}$, which is bounded by an element of U, hence $x \in \bar{\mathrm{U}}$, and $\bar{\mathrm{U}}$ is a regular open set.

```
Exercise1_22h u x: (* 17 *)
  open_o r u -> inc x (substrate r) ->
  (forall y, gle r x y -> exists2 z, inc z u &  gle r y z) ->
  inc x (open_bar r u).
Lemma Exercise1_22i u: (* 9 *)
  open_o r u ->  open_r r (open_bar r u).
```

Assume $\mathrm{U} \subset \mathrm{V}$, and $x \in \bar{\mathrm{U}}$, and $x \le y$. Then $y \in \bar{\mathrm{U}}$, hence is bounded by an element of U (thus V), and x is in $\bar{\mathrm{V}}$. Hence $\bar{\mathrm{U}} \subset \bar{\mathrm{V}}$.

Assume that U and V are open sets. Consider an element $a \in \bar{\mathrm{U}} \cap \bar{\mathrm{V}}$, say $a \in \mathrm{K}_1$ and $a \in \mathrm{K}_2$. There is $x \in \mathrm{U}$, with $a \le x$. Since K_2 is open, we have $x \in \mathrm{K}_2$. Thus, there is $y \in \mathrm{V}$ such that $x \le y$. Since U is open, we have $y \in \mathrm{U} \cap \mathrm{V}$. Thus, if $\mathrm{U} \cap \mathrm{V} = \emptyset$ then $\bar{\mathrm{U}} \cap \bar{\mathrm{V}} = \emptyset$.

```
Lemma Exercise1_22j u v: (* 6 *)
  open_o r u -> open_o r v -> sub u v ->
  sub (open_bar r u)  (open_bar r v).
Lemma Exercise1_22k  u v: (* 9 *)
  open_o r u -> open_o r v -> disjoint u v  ->
  disjoint (open_bar r u)  (open_bar r v).
```

(b) The set R(E) has a least and a greatest element, namely $\emptyset$ and E. We have $\bar{\mathrm{X}} = \mathrm{X}$ whenever X is regular.

```
Lemma Exercise1_22m: open_r r emptyset. (* 5 *)
Lemma Exercise1_22n: open_r r (substrate r). (* 2 *)
Lemma Exercise1_22p x:  -> (* 2 *)
  open_r r x -> x = (open_bar r x) .
```

Let I_x denote the interval $[x, \to[$ and $\mathrm{U}_x = \bar{\mathrm{I}}_x$. Assume that E is not right directed. This means that there exists x and y such that $\mathrm{I}_x \cap \mathrm{I}_y = \emptyset$, thus $\mathrm{U}_x \cap \mathrm{U}_y = \emptyset$.

```
Lemma Exercise1_22o: (* 25 *)
  ~ (right_directed r) ->
  exists a b, [/\ open_r r a, open_r r b, nonempty a, nonempty b & a <> b].
```

Let's show that R(E) has two elements if and only if E is non-empty and right directed. If E is empty, then R(E) contains only E. If E is not right directed it contains two distinct non-empty sets (by the argument above), plus the empty set; thus has at least three elements. Conversely, if E is non-empty, R(E) contains at least $\emptyset$ and E, which are distinct. Assume E right directed, U in R(E). Suppose U non-empty, $x \in \mathrm{U}$, $y \in \mathrm{E}$. For any z with $y \le z$ there is an upper bound for x and z; this bound is in U. By 22h, it follows $y \in \bar{\mathrm{U}} = \mathrm{U}$. Thus $\mathrm{U} = \mathrm{E}$.

```
Lemma Exercise1_22qP: (* 35 *)
  (doubletonp (reg_opens r)) <->
  (nonempty (substrate r) /\ (right_directed r)).
```

Let's show that R(E) is a complete lattice. We consider some properties of the ordering induced by inclusion on R(E). We first show that E and $\emptyset$ are the greatest and least elements. This is a trivial consequence of the fact that these sets are in R(E)

```
Lemma Exercise1_22rP u v: (* 2 *)
  gle (reg_open_order r) u v <->
  [/\ open_r r u, open_r r v &sub u v].
Lemma Exercise1_22s1P x: (* 2 *)
  inc x (substrate (reg_open_order r)) <-> open_r r x.
Lemma Exercise1_22s:  (* 4 *)
  greatest (reg_open_order r) (substrate r).
Lemma Exercise1_22t:  (* 4 *)
  least (reg_open_order r) (emptyset).
```

The function $\mathrm{U} \mapsto \bar{\mathrm{U}}$ is a closure. As a consequence, the bar of the union of a family of elements of R(E) is the least upper bound. This shows that the set is a complete lattice, thus is a lattice. The sup and inf of two elements are $\overline{\mathrm{U} \cup \mathrm{V}}$ and $\overline{\mathrm{U} \cap \mathrm{V}}$.

```
Lemma Exercise1_22u u v: (* 16 *)
  open_r r u -> open_r r v ->
  inf (reg_open_order r) u v = open_bar r (u \cap v).
Lemma Exercise1_22v X: (* 16 *)
  sub X (substrate (reg_open_order r)) ->
  least_upper_bound (reg_open_order r) X (open_bar r (union X)).
Lemma Exercise1_22w u v:  (* 5 *)
  open_r r u -> open_r r v ->
  sup (reg_open_order r) u v = open_bar r (u \cup v).
Lemma Exercise1_22x: (* 4 *)
  complete_lattice (reg_open_order r).
Lemma Exercise1_22y:  (* 3 *)
  lattice (reg_open_order r).
```

Assume $\mathrm{X} \subset \mathrm{Y}$, where X and Y are regular open sets. Let Z be the set of all elements of Y not bounded by an element of X. This is an open set. Every element of Y is bounded by an element of X or Z. This implies that Y is a subset of $\bar{\mathrm{Z}} \cup \mathrm{X}$, hence $\mathrm{Y} = \sup(\bar{\mathrm{Z}}, \mathrm{X})$. Obviously, $\mathrm{Z} \cap \mathrm{X} = \emptyset$, thus $\bar{\mathrm{Z}} \cap \bar{\mathrm{X}} = \emptyset$. Since $\mathrm{X} = \bar{\mathrm{X}}$, we get $\inf(\bar{\mathrm{Z}}, \mathrm{X}) = \emptyset$. As a consequence, our set is relatively complemented.

```
Lemma Exercise1_22z: (* 42 *)
  relatively_complemented (reg_open_order r).
```

We have to show that our set is distributive. Condition (T') reads $\overline{\mathrm{Z} \cap \overline{\mathrm{X} \cup \mathrm{Y}}} \subset \overline{\mathrm{X} \cup \overline{\mathrm{Y} \cap \mathrm{Z}}}$. Write this as $\bar{\mathrm{A}} \subset \bar{\mathrm{B}}$. By 1.22h, we have to show that for any $x \in \bar{\mathrm{A}}$, and $x \le x'$ there is $y \in \mathrm{B}$ such that $x' \le y$. We have $x' \in \bar{\mathrm{A}}$, hence there is $x'' \in \mathrm{A}$ such that $x' \le x''$. Since A has the form $\mathrm{Z} \cap \overline{\mathrm{X} \cup \mathrm{Y}}$ there is $y \in \mathrm{X} \cup \mathrm{Y}$ such that $x'' \le y$. We have also $y \in \mathrm{Z}$. We have $\mathrm{Z} \cap (\mathrm{X} \cup \mathrm{Y}) \subset \mathrm{X} \cup (\mathrm{Y} \cap \mathrm{Z})$ (the relation we want to prove, without the bars). Hence $y \in \mathrm{X} \cup (\mathrm{Y} \cap \mathrm{Z}) \subset \bar{\mathrm{A}}$.

```
Lemma Exercise1_22A: (* 40 *)
  boolean_lattice (reg_open_order r).
```

(c) Let F be a cofinal subset of E.

Assume U regular in E. Then $U \cap F$ is regular. In fact, assume $U \cap F$ cofinal in an open set V of F. Let $x \in V$. We must show $x \in U \cap F$. We have obviously $x \in F$. Assume $x \le y$. Since F is cofinal, there is $z \in F$ such that $y \le z$. Since V is open, we have $z \in V$, so there is $t \in U \cap F$ with $z \le t$. Thus, there is $t \in U$ such that $y \le t$. By 1.22h, this says $x \in \bar{U}$, hence $x \in U$.

```
Lemma Exercise1_22B F x: (* 24 *)
  cofinal r F -> open_r r x ->
  open_r (induced_order r F) (x \cap F).
```

Thus $U \mapsto U \cap F$ maps R(E) into R(F). Assume $U \cap F \subset U' \cap F$. Let $x \in U$, and $x \le y$. There exists $z \in F$ such that $y \le z$. We have $z \in U'$, and by 1.22h, this says $x \in \bar{U}'$, hence $U \subset U'$.

```
Lemma Exercise1_22C F U U': (* 11 *)
  cofinal r F -> open_r r U -> open_r r U' ->
    sub (U \cap F) (U' \cap F) -> sub U U'.
End Exercise1_22.
```

Let g be the mapping $X \mapsto X \cap F$. We have shown that if $g(x) \subset g(y)$ then $x \subset y$. The converse is obvious. As a consequence, g is increasing and injective. It is also surjective. We use the same argument as before. If X is a regular open subset of F, Y the set of elements $y \in E$ such that there is $x \in X$ with $x \le y$, then $g(\bar{Y}) = X$.

```
Lemma Exercise1_22D r F: order r -> (* 55 *)
  cofinal r F ->
  order_isomorphism (Lf (fun z => z \cap F) (reg_opens r)
    (reg_opens (induced_order r F)))
  (reg_open_order r)(reg_open_order (induced_order r F)).
```

(d) We consider the product of two orderings. The product of two open sets is obviously open. Converse is false: consider the trivial ordering ($x \le y$ is $x = y$). Then any set is open, and the product is trivial. But not any subset of the product is a product.

The assertion $R(E_1 \times E_2)$ is isomorphic to $R(E_1) \times R(E_2)$ is false. Assume E_1 and E_2 are singletons; then the product is a singleton; each set is non-empty and right directed. Thus the three sets $R(E_1 \times E_2)$, $R(E_1)$ and $R(E_2)$ have two elements each and the product has four elements.

```
Lemma Exercise1_22E r r' X X': (* 8 *)
  order r -> order r' ->
  open_o r X -> open_o r' X' -> open_o  (order_product2 r r') (X \times X').
Lemma Exercise1_22F E X: (* 11 *)
  sub X E -> open_r (diagonal E) X.
Lemma Exercise1_22G E: (* 13 *)
  let r := diagonal E in
    (order_product2 r r = diagonal (E \times E)).
Lemma Exercise1_22H: exists r, (* 14 *)
  let r' := order_product2 r r in
   order r /\   (exists2 t, open_o r' t & forall a b, t <> a \times b).
Lemma Exercise1_22I: exists r,   (* 37 *)
   let r' := order_product2 r r in
     let R := reg_open_order r in
       order r /\ ~(reg_open_order r' \Is order_product2 R R).
```

¶ 23. Let E be an ordered set and let $R_0(E) = R(E) - \{\emptyset\}$ (Exercise 22). For each $x \in E$, let $r(x)$ denote the unique regular open set in which the interval $[x, \rightarrow[$ (which is an open set) is cofinal. The mapping r so defined is called the canonical mapping of E into $R_0(E)$. Endow $R_0(E)$ with the order relation *opposite* to the relation of inclusion.

(a) Show that the mapping r is increasing and that $r(E)$ is cofinal in $R_0(E)$.

(b) An ordered set E is said to be *antidirected* if the canonical mapping $r : E \to R_0(E)$ is injective. For this to be so it is necessary and sufficient that the following two conditions should be satisfied.

(I) If x and y are two elements of E such that $x < y$, there exists $z \in E$ such that $x < z$ and such that the intervals $[y, \rightarrow[$ and $[z, \rightarrow[$ do not intersect.

(II) If x and y are two non-comparable elements of E then either there exists $x' \geq x$ such that the intervals $[x', \rightarrow[$ and $[y, \rightarrow[$ do not intersect, or else there exists $y' \geq y$ such that the intervals $[x, \rightarrow[$ and $[y', \rightarrow[$ do not intersect.

(c) Show that, for every ordered set E, $R_0(E)$ is antidirected and that the canonical mapping of $R_0(E)$ into $R_0(R_0(E))$ is bijective (use Exercise 22(a)).

Note. part (c) not yet done.

Solution. We start with the definition of E_0 and its ordering.

```
Definition nreg_opens r :=
  (reg_opens r) -s1 emptyset.
Definition nregs_order r :=
  opp_order (sub_order (nreg_opens r)).
Definition canonical_reg_open r x :=
  open_bar r (Zo (substrate r) (fun z => gle r x z)).

Lemma Exercise1_23aP r X: (* 5 *)
  inc X (nreg_opens r) <-> (open_r r X /\ nonempty X).

Lemma Exercise1_23bP r: order r -> forall X Y, (* 5 *)
  (gle  (nregs_order r) X Y <->
  [/\ nonempty X, nonempty Y, open_r r X, open_r r Y & sub Y X]).
```

(a) We have the following interesting property: $y \in r(x)$ if and only if, whenever $y \leq z$, there is a common upper bound to x and z. The mapping r is increasing, as the composition of two increasing functions. In general, it is not strictly increasing (if E is right directed, it is constant).

```
Lemma Exercise1_23c r x: order r -> (* 2 *)
  open_o r  (Zo (substrate r) (fun z => gle r x z)).
Lemma Exercise1_23d1 r x: (* 2 *)
   order r -> inc x (substrate r) ->
  inc x (canonical_reg_open r x).
Lemma  Exercise1_23d2  r x:  (* 3 *)
  order r -> inc x (substrate r) ->
  inc (canonical_reg_open r x) (nreg_opens r).
Lemma Exercise1_23e r x y: order r -> (* 10 *)
  inc x (substrate r) -> inc y (substrate r) ->
```

```
  (inc y (canonical_reg_open r x) <->
    forall z, gle r y z -> exists2 t, gle r z t & gle r x t).
Lemma  Exercise1_23f r x y: order r -> (* 7 *)
  gle r x y -> gle  (nreg_orders r)
  (canonical_reg_open r x)  (canonical_reg_open r y).
```

If X is regular, then $\mathrm{X} = \bar{\mathrm{X}}$. If $x \in \mathrm{X}$, then $[x, \rightarrow[\subset \mathrm{X}$, thus $r(x) \subset \mathrm{X}$. As a consequence, the image of r is cofinal in R_0.

```
Lemma  Exercise1_23g r: order r -> (* 14 *)
  cofinal (nreg_orders r)
   (fun_image (substrate r) (canonical_reg_open r)).
```

(b) Let $\mathrm{A}(x, y)$ be the property that the intervals $[x, \rightarrow[$ and $[y, \rightarrow[$ do not intersect. We might replace $x < z$ in (I) by $x \leq z$, for $\mathrm{A}(y, x)$ is false (since y is in the intersection). Our criterion 1.23e says: $a \in r(x)$ if and only if, for all b such that $a \leq b$, $\mathrm{A}(b, x)$ is false.

Thus (I) can be written as: if $x < y$ then $x \notin r(y)$, and (II) as if x and y are non-comparable, then $x \notin r(y)$ or $y \notin r(x)$. Write this as "not ($x \in r(y)$ and $y \in r(x)$)". Since $x \in r(x)$ and $y \in r(y)$ this is $r(x) \neq r(y)$. Condition (II) becomes: if $r(x) = r(y)$, then x and y are comparable. But (I) excludes $x < y$ and $y < x$, so that (I) and (II) imply injectivity of r. Conversely, injectivity implies (II). Assume now $x < y$. There is z such that $x \leq z$ and $\mathrm{A}(y, z)$, since otherwise we would have $r(x) \subset r(y)$. But $x \leq y$ implies $r(x) \subset r(y)$, thus $r(x) = r(y)$. If the set is antidirected, we get $x = y$, absurd.

```
Definition anti_directed r:=
    {inc (substrate r) &, injective (canonical_reg_open r) }.

Lemma  Exercise1_23hP r: order r -> (* 58 *)
  let aux := (fun x y => forall z, gle r x z -> gle r y z -> False) in
  (anti_directed r) <->
  ((forall x y, glt r x y -> exists2 z, glt r x z & aux y z)
    /\ forall x y, inc x (substrate r) -> inc y (substrate r) ->
     [\/ gle r x y, gle r y x, (exists2 x', gle r x x' & aux x' y) |
        (exists2 y', gle r y y' & aux x y')]).
```

(c) We are assumed to show that, for every ordered set E, $\mathrm{R}_0(\mathrm{E})$ is antidirected and that the canonical mapping of $\mathrm{R}_0(\mathrm{E})$ into $\mathrm{R}_0(\mathrm{R}_0(\mathrm{E}))$ is bijective.

The condition $\mathrm{A}(\mathrm{X}, \mathrm{Y})$ in $\mathrm{R}_0(\mathrm{E})$ says that there is no upper bound for X and Y. We know that R(E) is a complete lattice, so that the condition becomes: there is only one upper bound, namely the empty set (that is not in R_0). Since $x \in \mathrm{X}$ implies $\mathrm{X} \leq r(x)$, the condition becomes X and Y are disjoint.

```
Lemma Exercise1_23i r x y: order r -> (* 10 *)
  inc x y -> inc y  (nreg_opens r) ->
    gle (nregs_order r) y (canonical_reg_open r x).
Lemma Exercise1_23j r: order r -> (* 11 *)
  let r':= nregs_order r in
    (forall x y, inc x (substrate r') -> inc y (substrate r') ->
      ((forall z, gle r' x z ->  gle r' y z -> False) <->
       (disjoint  x y))).
```

We know that R(E) is a Boolean lattice. Given X and Y we construct a regular open set Z, a subset of X that is disjoint from Y. (This is the complement (for the lattice) of Y in X whenever

$Y \subset X$). If Z is empty, then $X \subset Y$. This can be restated as: if $X \subset Y$ is false, then $Z \in R_0(E)$. It follows that $R_0(E)$ is antidirected. The canonical mapping of $R_0(E)$ into $R_0(R_0(E))$ is hence injective.

```
Lemma Exercise1_23k r: order r -> (* 63 *)
  anti_directed (nregs_order r).
```

The mapping $r : R_0(E) \to R_0(R_0(E))$ is injective by the previous lemma. We can rewrite 1.23i as: $y \in r(x)$ if and only if for all t, $y \le t$ implies $t \cap x$ is non-empty. Proof: Let $a \in y$. We have $y \le r(a)$, hence $r(a) \cap x$ is non-empty. This implies that there is $b \in x$, such that $a \le b$. Conversely, assume every $a \in y$ bounded by an element of x; assume $y \le t$. There is $a \in t \subset y$. If $a \le b$ then $b \in t$. If moreover $a \in x$ the set $t \cap x$ is non-empty.

Hence $y \in r(x)$ if and only if every element of y is bounded by an element of x. It follows that if x is non-empty and open, then $y \in r(\bar{x})$ if and only if every element of y is bounded by an element of x.

Let Y be in $R_0(R_0(E))$. Proving surjectivity of r means showing existence of a set X such that every element of the union of the elements of Y is bounded by an element of X. Take $X = \bigcup Y$. The set $r(\bar{X})$ is the set of all $z \in E$, such that each element of z is bounded by an element of $\bigcup Y$. It follows $Y \subset r(\bar{X})$. Assume Y cofinal. Since Y is regular, this will imply $Y = r(\bar{X})$. Let X be a regular open set, such that each element of X is bounded by an element of the union of Y. We must show: There is $Z \in Y$ such that $Z \subset X$. Is this true?

```
Lemma Exercise1_23l r y: order r ->
  let r' := nreg_orders r in
    inc y (nreg_opens r') ->
    exists ! x, (inc x (nreg_opens r) /\
      y =  canonical_reg_open r' x).
Proof. Abort.
```

---

24. * (a) An ordered set E is said to be *branched* (on the right) if for each $x \in E$ there exist y, z in E such that $x \le y$, $x \le z$ and the intervals $[y, \to[$ and $[z, \to[$ do not intersect. An antidirected set with no maximal element (Exercise 23) is branched.

(b) Let E be the set of intervals in **R** of the form $[k.2^{-n}, (k+1).2^{-n}]$ $(0 \le k < 2^n)$, ordered by the relation $\supset$. Show that E is antidirected and has no maximal elements.

(c) Give an example of a branched set in which there exists no antidirected cofinal subset (Take the product of the set E defined in (b) with a well-ordered set which contains no countable cofinal subset, and use Exercise 22.)

(d) Give an example of an ordered set E which is not antidirected, but which has an antidirected cofinal subset (Note that an ordinal sum $\sum_{\xi \in E} F_\xi$ contains a cofinal subset isomorphic to E).*

Note Points (c) and (d) remain to do. The indications are a bit strange.

The stars indicate that the exercise requires material not defined yet, namely the set of real numbers. We shall use rational numbers instead, this gives a different set, but an isomorphic ordering.

(a) is obvious.

```
Definition branched r :=
  order r /\ (forall x, inc x (substrate r) ->
    exists y  z, [/\ gle r x y, gle r x z &
      (forall t, gle r y t -> gle r z t -> False)]).

Lemma Exercise1_24a r: (* 8 *)
  order r -> anti_directed r ->
  (forall x, inc x (substrate r) -> ~ maximal r x) ->
  branched r.
```

(b) Consider the elements of **Q** of the form $a/2^n$, denoted a_n. They are in Q^+, and $a_n \leq b_n$ if and only if $a \leq b$.

```
Definition Qpair k n :=
  BQ_of_pair (BZ_of_nat k) (BZ_of_nat (\2c ^c n)).

Lemma zpow2_pos n: natp n -> inc (BZ_of_nat (\2c ^c n)) BZps.  (* 4 *)
Lemma Qpair_q k n: natp k ->  natp n -> ratp (Qpair k n).  (* 1 *)
Lemma Qpair_eq k n m: natp k -> natp n -> natp m ->
  Qpair k n = Qpair (k *c \2c ^c m) (m +c n).  (* 7 *)
Lemma Qpair_le0P a b c d: (* 6 *)
  natp a -> natp b -> natp c -> natp d ->
  let f:= fun k n =>  k *c (\2c ^c n) in
   (Qpair a b) <=q (Qpair c d) <-> (f a d) <=c (f c b).
Lemma Qpair_leP k k' n: natp k -> natp k' -> natp n ->  (* 4 *)
 (k <=c k' <-> (Qpair k n) <=q (Qpair k' n)).
```

Let E be the set of all intervals of the form $A_{kn} = [k_n, (k+1)_n]$, ordered by $\supset$. Note that k_n and $(k+1)_n$ are in the interval. Since, for any intervals I_1 and I_2, we have $I_1 \subset I_2$ if the endpoints of I_1 are in I_2 we get that $A_{kn} \leq A_{lm}$ is equivalent to

$$k.2^m \leq l.2^n \qquad (l+1).2^n \leq (k+1)2^m.$$

This relation implies $n \leq m$. If $m = n + p$ it reduces to

$$k2^p \leq l \qquad (l+1) \leq (k+1)2^p$$

In particular, if $p = 0$, it implies $k = l$, so that k and l are uniquely defined from A_{kn}.

```
Definition Qpairi k n :=
 interval_cc BQ_ordering (Qpair k n) (Qpair (csucc k) n).
Definition Qpairis :=
  fun_image (Nat \times Nat) (fun z => Qpairi (P z) (Q z)).
Definition Qpairi_o := opp_order (sub_order Qpairis).

Lemma Qpairis_prP x: (* 4 *)
  inc x Qpairis <->
  exists k n, [/\ natp k, natp n & x =  Qpairi k n].

Lemma Qpairio_osr: order_on Qpairi_o Qpairis. (* 2 *)
Lemma Qpairio_gleP x y:  (* 1 *)
   gle Qpairi_o x y <-> [/\ inc x  Qpairis, inc y  Qpairis & sub y x].
Lemma Qpairis_pr1P n k x: inc x (Qpairi k n) (* 2 *)
  <-> (Qpair k n) <=q x /\  x <=q (Qpair (csucc k) n).
Lemma Qpairis_pr2 k n: natp k -> natp n -> (* 6 *)
```

```
  (inc (Qpair k n)  (Qpairi k n) /\ inc  (Qpair (csucc k) n) (Qpairi k n)).
Lemma Qpairio_gle1P k n l m: (* 18 *)
  natp k -> natp n -> natp l -> natp m ->
  let f:= fun k n => (k *c (\2c ^c n)) in
  gle Qpairi_o (Qpairi k n) (Qpairi l m) <->
    ((f k m) <=c  (f l n)  /\ (f (csucc l) n) <=c (f (csucc k) m)).
Lemma Qpairio_gle2P k n l m: (* 37 *)
  natp k -> natp n ->  natp l -> natp m ->
  let f:= fun k n => (k *c (\2c ^c n)) in
  gle Qpairi_o (Qpairi k n) (Qpairi l m) <->
  (exists p, [/\ natp p, m = n +c p,
    (f k p) <=c l & (csucc l) <=c (f (csucc k) p)]).
Lemma Qpairio_eq k n l m: (* 17 *)
  natp k -> natp n -> natp l -> natp m  ->
   (Qpairi k n) = (Qpairi l m) -> (k = l /\ n = m).
```

The set E has no maximal element, since if $x = \mathrm{A}_{kn}$, then $y = \mathrm{A}_{2k,n+1}$ satisfies $x < y$. Let $x = \mathrm{A}_{kn}$ and $y = \mathrm{A}_{lm}$. Assume $x \le z$ and $y \le z$. Evaluating the conditions above shows that if $n \le m$ then $x \le y$. In particular, x and y are comparable. Thus condition (II) of the previous exercise is obviously satisfied. Condition (I) is also true: Assume first $x < y$. Write $x = \mathrm{A}_{kn}$ and $y = \mathrm{A}_{k.2^p+l,n+p}$. We have $p > 0$ and $0 \le l < 2^p$. Take $z = \mathrm{A}_{k.2^p+m,n+p}$, where $m = 1$ (if $l = 0$) and $m = 0$ (otherwise). Then y and z are non-comparable.

As a consequence, E is antidirected and branched.

```
Lemma Qpairio_gle3 k n l m z: (* 31 *)
  natp k -> natp n -> natp l -> natp m ->
  n <=c m  ->
  gle Qpairi_o (Qpairi k n) z -> gle Qpairi_o(Qpairi l m) z ->
  gle Qpairi_o (Qpairi k n) (Qpairi l m).
Lemma Qpairio_gle4 x y:  (* 9 *)
   inc x (substrate Qpairi_o) -> inc y (substrate Qpairi_o) ->
   [\/ gle Qpairi_o x y,
   gle Qpairi_o y x |
   (forall z : Set, gle Qpairi_o x z -> gle Qpairi_o y z -> False)].

Lemma Exercise1_24b x: (* 22 *)
  inc x (substrate Qpairi_o) -> ~ (maximal Qpairi_o x).
Lemma Exercise1_24c: anti_directed Qpairi_o.  (* 78 *)
Lemma Exercise1_24d: branched Qpairi_o. (* 1 *)
```

(c) Consider now points (c). A product is branched if one factor is branched. If the product is E × F, where E is the set studied above, if F is totally ordered, if X is a cofinal subset of the product, then $[x, \to[\, \cap\, [y, \to[$ is non-empty if and only if x and y are comparable.

Consider an ordinal sum $\sum \mathrm{E}_i$. It is wrong that the sum has a cofinal set isomorphic to the index set. Assume for instance that the index set has a single element α. Then the sum has a cofinal set reduced to a single element, thus has a greatest element. But the sum is isomorphic to E_α. However, if for any maximal index α the set E_α has a greatest element x_α we can proceed as follows. Consider the element y_i of the sum, which is x_α in the previous case, any element of E_i otherwise. The set Y of all y_i is isomorphic to the index set. Consider x_i in the sum. If i is maximal, we have $x_i \le x_\alpha = y_i$. Otherwise there is j such that $i < j$ and $x_i < y_j$. Thus Y is cofinal.

```
Lemma Exercise1_24e r r': (* 14 *)
```

```
  branched r -> order r' ->
  branched (order_product2 r r').
```

13.3 Section 2

Consider an ordered set E, such that, whenever a and b are in E, $a \leq b$ is equivalent to $a \subset b$. Consider a family A_i that has an upper bound S. Then S contains the union U of the family. If this union is in E, it is the least upper bound of the family.

```
Lemma induced_sub_pr1 r X x: (* 2 *)
  (forall a b, gle r a b -> sub a b) ->
  upper_bound r X x -> sub (union X) x.
Lemma induced_sub_pr2 r X:  (* 5 *)
  order r ->
  (forall a b, inc a (substrate r) -> inc b (substrate r) ->
    (gle r a b <-> sub a b)) ->
  sub X (substrate r) -> inc (union X) (substrate r) ->
  least_upper_bound r X (union X).
Lemma inc_coarse a b E: (* 1 *)
  inc a E -> inc b E -> inc (J a b) (coarse E).
```

1. Show that, in the set of orderings on a set E, the minimal elements (with respect to the ordered relation "Γ is coarser than Γ'" between Γ and Γ') are the total orderings on E, and that if Γ is any ordering on E, the graph of Γ is the intersection of the graphs of the total orderings on E which are coarser than Γ (apply Theorem 2 of no. 4). Deduce that every ordered set is isomorphic to a subset of a product of totally ordered sets.

Note. For Bourbaki, an ordering Γ is a triple (G, E, E) satisfying some conditions, and its graph is G. We consider only graphs; moreover we shall consider the opposite ordering of "coarser". The first point becomes: the maximal elements for $\subset$ are total ordering, and we use Zorn's Lemma for the second point.

Solution. We start with some properties of $\subset$ as a ordering.

```
Definition orders x := Zo (\Po (coarse x))(order_on ^~ x).
Definition finer_order x :=  sub_order (orders x).

Lemma ordersP x z: (* 3 *)
  inc z (orders x) <-> (order_on z x).
Lemma fo_osr x: (* 1 *)
  order_on (finer_order x)(orders x).
Lemma fo_gleP x u v: (* 5 *)
  gle (finer_order x) u v <->
  [/\ order u, order v, substrate u = x, substrate v = x & sub u v].
Lemma fo_gle1P x u v: (* 11 *)
  gle (finer_order x) u v <->
  [/\ order u, order v, substrate u = x, substrate v = x &
  forall a b, inc a x -> inc b x -> gle u a b -> gle v a b].
```

Let $\leq$ be an ordering on E, x and y two elements of E. Let $a \leq' b$ be the relation "$a \leq b$ or ($a \leq x$ and $y \leq b$)". This is an ordering if $y \leq x$ is false, and then $x \leq' y$, so that $\leq$ is not maximal. Thus a maximal ordering is total. The converse is immediate.

```
Lemma Exercise2_1a r x y  (* 44 *)
  (E :=  substrate r)
  (r':= r \cup (Zo (coarse E)(fun z=>  gle r (P z) x /\ gle r y (Q z)))):
  order r -> inc x (substrate r) -> inc y (substrate r) ->
  ~ gle r y x ->
  (gle (finer_order E) r r' /\ inc (J x y) r').
Lemma Exercise2_1bP E r: (* 12 *)
  maximal (finer_order E) r <-> (total_order r /\ substrate r = E).
```

The set of orderings is inductive (the union of a non-empty totally ordered family of orderings is an ordering). First Corollary to Zorn's lemma says that any ordering can be extended to a total ordering.

```
Lemma fo_inductive: (* 48 *)
  forall E, inductive (finer_order E).
Lemma order_total_extension r: order r -> (* 6 *)
  exists r', [/\ total_order r', substrate r' = substrate r & sub r r'].
```

Let G be an ordering, F the set of all total orderings that extend G, and G' the intersection of F. Since F is non-empty, G' is an ordering and $G \subset G'$. For any pair (x, y) of non-comparable elements of E, we can extend G such that $x < y$ or $y < x$, and extend these two a total ordering. Thus x and y are non-comparable in the intersection, i.e., $G = G'$.

```
Lemma Exercise2_1c r: (* 27 *)
 order r -> r = intersection (Zo (orders (substrate r))
   (fun r' => total_order r' /\ sub r r')).
```

Let E and F be as above. Consider the order product of the elements of F (indexed by some set I). This set has substrate F^E, and the mapping that associates to x the constant graph x is an order isomorphism.

```
Lemma Exercise2_1d r:   (* 27 *)
  order r -> exists g h,
 [/\ order_fam g,
   (allf g total_order) &
  order_morphism h r (order_product g)].
```

---

2. Let E be an ordered set and let $\mathfrak{B}$ be the set of subsets of E which are well-ordered by the induced ordering. Show that the relation "X is a segment of Y" on $\mathfrak{B}$ is an order relation between X and Y and that $\mathfrak{B}$ is inductive with respect to this order relation. Deduce that there exist well-ordered subsets of E which have no strict upper bound in E.

Solution. Let's show that the set $\mathfrak{B}$ is inductive. We consider a totally ordered family X_i. We pretend that its union X is an upper bound. The non trivial point is to show that the union is well-ordered. Consider Y a nonempty subset of X. Let $a \in Y$, say $a \in X_i$. Let Z be the intersection of Y and X_i and b its least element. Let c be in Y, say $c \in X_j$. We have to show $b \le c$. This is clear if $c \in X_i$, in particular when $X_j \subset X_i$. If this relation is false, then $X_i \subset X_j$, so that b and c are in X_j, thus are comparable. If $c \le b$ we get $c \in X_i$, because X_i is a segment of X_j.

```
Lemma Exercise2_2a r: (* 77 *)
 (B:= Zo (\Po (substrate r)) (fun z=> worder (induced_order r z)))
 (sso := Zo (coarse B)(fun z=> segmentp (induced_order r (Q z)) (P z))):
 order r ->  (order_on sso B /\ inductive sso).
```

Let E be an ordered set. We apply Zorn's lemma to the ordering defined above. It says that we have a maximal element, i.e., a well-ordered subset X of E. Assume that X has an upper bound x and $Y = X \cup \{x\}$. This is a well-ordered set (adding of a greatest element to a well-ordered set), hence $Y \in \mathfrak{B}$, thus $x \in X$.

```
Lemma Exercise2_2b r: order r -> (* 45 *)
  exists x, [/\ sub x (substrate r), worder (induced_order r x) &
    forall z, upper_bound r x z -> inc z x].
```

---

3. Let E be an ordered set. Show that there exist two subsets A, B, of E such that $A \cup B = E$ and $A \cap B = \emptyset$ and such that A is well-ordered and B has no least element (for example, take B to be the union of those subsets of E which have no least element). *Give an example in which there are several partitions of E into two subsets having these properties.$_*$

Note. The stars indicate the exercise requires material not yet defined, namely existence of an infinite set (if E is finite, and totally ordered, we have $B = \emptyset$). We have an example where E is finite.

Solution. We first rewrite the condition "$A \cup B = E$ and $A \cap B = \emptyset$" as A is the complementary of a subset B of E. The result is obvious.

Example 1: Take **N**, with the opposite order of its natural ordering; any A of the form $[0, n[$ is well ordered (being finite and totally ordered), while the complement has no least element.

Example 2: Take for E a set with three elements, with the trivial ordering. Let A be empty or a singleton. Then A is well-ordered, and its complement has at least two elements, thus no least element.

```
Lemma complement_p1 C A B: (* 7 *)
  A \cup B = C -> A \cap B = emptyset ->
  (sub A C /\ A = C -s B).
Lemma complement_p2 C A B: (* 1 *)
  A \cup B = C -> A \cap B = emptyset ->
  (sub B C /\ B = C -s A).
Lemma complement_p3 C B (A := C -s B): (* 5 *)
   sub B C -> (A \cup B = C /\ A \cap B = emptyset).
Lemma complement_p4 C A (B:= C -s A) :  (* 1*)
  sub A C -> (A \cup B = C /\ A \cap B = emptyset).
Lemma Nat_greatest A: (* 7 *)
  sub A Nat -> nonempty A ->
  (exists x, upper_bound Nat_order A x) ->
  (has_greatest (induced_order Nat_order A)).

Definition ex23_prop r A B:=
  [/\ A \cup B = substrate r, A \cap B = emptyset,
  worder (induced_order r A) &
  ~ (has_least (induced_order r B) y)].
```

```
Lemma Exercise2_3a r: (* 20 *)
  order r -> exists A B,  ex23_prop r A B.
Lemma  Exercise2_3b n (r := opp_order Nat_order)   (* 35 *)
  (A := Nint n) (B:= Nat -s A) :
   natp n -> (order r /\ ex23_prop r A B).
Lemma Exercise2_3c   (* 39 *)
  (r := diagonal C3) (A1:= C0) (A2:= C1):
  [/\ order r,
     (ex23_prop r A1 ((substrate r) -s A1)) &
     (ex23_prop r A2 ((substrate r) -s A2)) ].
```

¶ 4. An ordered set F is said to be *partially well-ordered* if every totally ordered subset of F is well-ordered. Show that in every ordered set E there exists a partially well-ordered subset which is cofinal in E (Consider the set $\mathfrak{F}$ of partially well-ordered subsets of E, and the order relation "$\mathrm{X} \subset \mathrm{Y}$ and no element of $\mathrm{Y} - \mathrm{X}$ is bounded above by any element of X" between X and Y of $\mathfrak{F}$. Show that $\mathfrak{F}$ is inductive with respect to this order relation).

Solution. Consider an ordered set $(\mathrm{E}, \le)$. We denote by $\le_{\mathrm{X}}$ the ordering induced by $\le$ on X, and by $p(\mathrm{F})$ the property that, for any subset X of F, if $\le_{\mathrm{X}}$ is total, then $\le_{\mathrm{X}}$ is a well-ordering. This is equivalent to: every nonempty subset X of F such that $\le_{\mathrm{X}}$ is total has a least element (for $\le_{\mathrm{X}}$).

Assume $\mathrm{F} \in \mathfrak{F}$ and $t \in \mathrm{E}$. Then $\mathrm{F} \cup \{t\} \in \mathfrak{F}$. In fact, consider a non-empty totally ordered subset X of $\mathrm{F} \cup \{t\}$. This set has a least element if it does not contain t, or contains only t. Consider otherwise the set $\mathrm{X} - \{t\}$. It has a least element y, and $\inf(y, t)$ is the least element of X.

Let $f(x, y)$ be the property that, for any a in x and b in $y - x$, the relation $b \le a$ is false. Consider the relation: $x \prec y$ if x and y are two elements of $\mathfrak{F}$ such that $x \subset y$ and $f(x, y)$. The first condition says that the relation is antisymmetric. It is in fact an ordering.

Consider a family X_i of elements of $\mathfrak{F}$, its union X, and a non-empty totally ordered subset Y of X. We have $x \in \mathrm{Y} \cap \mathrm{X}_i$ for some x and i. Let $\mathrm{K} = \mathrm{Y} \cap \mathrm{X}_i$. This is a non-empty totally ordered subset of X_i, thus has a least element y. We pretend that this is the least element of Y, provided that the family X_i is totally ordered by $\prec$. Consider $y' \in \mathrm{Y}$. It is in some X_j. If $\mathrm{X}_j \subset \mathrm{X}_j$, then $y' \in \mathrm{K}$ and the conclusion follows. Otherwise we have $y \in \mathrm{X}_i$, $y' \in \mathrm{X}_j - \mathrm{X}_i$, this implies that $y' \le y$ is false; but since the ordering on Y is total, it implies $y \le y'$. This argument shows $\mathrm{X} \in \mathfrak{F}$. From this, it is easy to deduce that X is an upper bound for X_i.

This shows that $\mathfrak{F}$ is inductive, thus has a maximal element F. For no $t \notin \mathrm{F}$ we have $\mathrm{F} \le \mathrm{F} \cup \{t\}$. This says that F is cofinal in E.

```
Lemma Exercise2_4 r   (* 131 *)
  (pworder :=  fun F => forall X,
    sub X F -> total_order (induced_order r X) -> worder (induced_order r X)):
 order r ->  exists2 F, pworder F /\ cofinal r F.
```

---

5. Let E be an ordered set and let $\mathfrak{I}$ be the set of *free* subsets of E, ordered by the relation defined in § 1, Exercise 5. Show that, if E is inductive, then $\mathfrak{I}$ has a greatest element.

Solution. This is a direct consequence of the first corollary to Zorn's lemma.

```
Lemma Exercise2_5 r: order r -> inductive r -> (* 12 *)
  has_greatest (free_subset_order r).
```

¶ 6. Let E be an ordered set and let f be a mapping from E into E such that $f(x) \geq x$ for all $x \in \mathrm{E}$.

(a) Let $\mathfrak{S}$ be the set of subsets M of E with the following properties: (1) the relation $x \in \mathrm{M}$ implies $f(x) \in \mathrm{M}$; (2) if a non-empty subset of M has a least upper bound in E, then this least upper bound belongs to M. For each $a \in \mathrm{E}$, show that the intersection C_a of the sets of $\mathfrak{S}$ which contain a also belongs to $\mathfrak{S}$; that C_a is well-ordered; and that if C_a has an upper bound b in E, then $b \in \mathrm{C}_a$ and $f(b) = b$. C_a is said to be the *chain* of a (with respect to the function f). (Consider the set $\mathfrak{M}$ whose elements are the empty set and the subsets X of E which contain a and have a least upper bound m in E such that $m \notin \mathrm{X}$ or $f(m) > m$, and apply Lemma 3 of no. 3 to the set $\mathfrak{M}$.)

(b) Deduce from (a) that if E is inductive, then there exists $b \in \mathrm{E}$ such that $f(b) = b$.

Note. Lemma 3 of no. 3 is the one that shows the theorem of Zermelo. The third claim of (a) should be corrected (as in the French edition of Bourbaki) as: "if C_a has a least upper bound b, then $b \in \mathrm{C}_a$ and $f(b) = b$". This implies that f has a fixed-point. However, if E is inductive, it has a maximal element, which a fixed point of f. Thus (b) is trivial.

Solution. Consider a nonempty well-ordered set E. If e is the least element, the segment with end-point e is empty. If the interval $]\leftarrow, a]$ is not the whole set, it is a segment with end-point b (this is called the successor of a).

```
Lemma worder_has_empty_seg r: worder r -> (* 5 *)
  nonempty (substrate r) -> exists2 x,
    inc x (substrate r) & segment r x = emptyset.

Lemma Exercise2_6g r a (m := Zo (substrate r) (fun z => glt r a z)) : (* 12 *)
  worder r -> inc a (substrate r) ->
  nonempty m -> exists2 b,
      inc b (substrate r) & (segmentc r a = segment r b).
```

We start with some definitions and show (b), which is trivial.

```
Section Exercise2_6.
Variables r f : Set.
Hypothesis or: order r.
Hypothesis ff: function f.
Hypothesis sf: substrate r = source f.
Hypothesis tf: substrate r = target f.
Hypothesis fxx: forall x, inc x (substrate r) -> gle r x (Vf f x).

Definition bigS :=
  Zo (\Po (substrate r))
  (fun M => (forall x, inc x M -> inc (Vf f x) M) /\
```

```
    (forall N x, sub N M -> nonempty N -> least_upper_bound r N x -> inc x M)).

Definition chainx a := intersection (Zo bigS (inc a)).

Lemma Exercise2_6i: inductive r -> (* 2 *)
  exists2 a, inc a (source f) & Vf f a = a.
```

We start with some trivial lemmas. We deduce $C_a \in \mathfrak{S}$, and if $b = \sup(C_a)$, then $b \in C_a$ and $f(b) = b$.

```
Lemma Exercise2_6a: inc (substrate r) bigS.  (* 3 *)
Lemma Exercise2_6b a:  (* 1 *)
  inc a (source f) -> nonempty (Zo bigS (inc a)).
Lemma Exercise2_6c a: (* 2 *)
  inc a (source f) -> inc a (chainx a).

Lemma Exercise2_6d a:  (* 13 *)
  inc a (source f) -> inc (chainx a) bigS.
Lemma Exercise2_6e a b: (* 7 *)
  inc a (source f) -> least_upper_bound r  (chainx a) b ->
  (inc b  (chainx a) /\ Vf f b = b).
```

Consider now the second claim: C_a is well-ordered. For this purpose we consider some sets. $\mathfrak{M}_0$ is the set all subsets of E that contain a and have a least upper bound. If x is such a set, we have $\sup x \in E$ and $f(\sup x) \in E$. The set $\mathfrak{M}_1$ contains those x for which $\sup x \not\in x$, and the set $\mathfrak{M}_2$ contains those x for which $\sup x < f(\sup x)$. The set $\mathfrak{M}$ will be the union of $\mathfrak{M}_1$ and $\mathfrak{M}_2$, to which we adjoin the empty set. Define p by: $p(\emptyset) = a$; if $x \in \mathfrak{M}_1$ then $p(x) = \sup(x)$; otherwise $p(x) = f(\sup x)$. By construction $p(x) \in E - x$, whenever $x \in \mathfrak{M}$.

We now apply Lemma 3.3. It asserts the existence of a set M, well-ordered by $\le_M$, such that (3): for any $x \in M$, if S_x is the segment (for $\le_M$) with endpoint x, then $S_x \in \mathfrak{M}$ and $p(S_x) = x$. Moreover (4): $M \not\in \mathfrak{M}$.

Since M is non-empty by (4), it has a least element, say x. We have $S_x = \emptyset$. Applying $p(S_x) = x$ gives $x = a$. Thus $a \in M$. Assume that M has a least upper bound m. Thus $M \in \mathfrak{M}_0$. Now (4) says $m \in M$ and $m = f(m)$.

We notice that $\le_M$ and $\le$ coincide on M (so that M will be totally ordered by $\le$). In fact, assume $y <_M x$ so that $y \in S_x$. The segment S_x, being non-empty in $\mathfrak{M}$, has a least upper bound c (hence $y \le c$). We have $c \le p(S(x))$ by construction, hence $c \le x$, thus $y \le x$.

Assume that M has a greatest element x (for $\le_M$). This is a greatest element for $\le$, hence is sup M. We deduce $x = m$, thus $x = f(x) \in M$.

Let's show that M satisfies (1). Fix $y \in M$. Consider the set of all x such that $y <_M x$. If it is empty then y is the greatest element of M for $\le_M$, thus is the greatest element for $\le$, thus is m, hence $y = f(y) \in M$. Otherwise, there is $z \in M$ such that S_z is the set of all t such that $t \le_M y$. The same argument as above shows that $y = \sup S_z$. Since S_z is non-empty and not in $\mathfrak{M}_1$, but in $\mathfrak{M}$, it is in $\mathfrak{M}_2$, so that $z = p(S_z) = f(\sup S_z) = f(y)$. This shows $f(y) \in M$.

Let's show that M satisfies (2). Take a non-empty subset N of M that has a least upper bound y, and let Q be the set of elements $z \in M$ such that $y \le z$. Assume first Q empty. Take $t \in N$. If for some $u \in N$ we have $t \le u$ then $t \le u \le y$. Otherwise, since M is totally ordered, t is an upper bound of N and $t \le y$. Thus $y = \sup M$, hence $y \in M$. Assume Q non-empty; so that it has a least element z. In particular $y \le z$. If $z \in N$, then $z \le y$, and $y = z \in M$. Otherwise $N \subset S_z$. The segment has a least upper bound α and $y \le \alpha \le z$. Note that $\alpha \not\in S_z$ (since $\alpha \in M$

implies $z \leq \alpha$). Thus $S_z \in \mathfrak{M}_1$. Evaluating p yields $\alpha = z$. It suffices to show $\alpha \leq y$ since it implies $y = \alpha = z \in M$. We must show that y is an upper bound of S_y. Thus assume $t <_M z$. Assume that for some $v \in N$ we have $t < v$. Then $v \leq y$, thus $t \leq y$. Otherwise, t is an upper bound for N and $y \leq t$. This implies $y \leq z$, absurd.

The properties shown above can be summarized as: M contains the chain of a. The chain is well-ordered as a subset of a well-ordered sets, with compatible orderings.

```
Lemma Exercise2_6h a: (* 172 *)
  inc a (source f) -> worder (induced_order r  (chainx a)).
End Exercise2_6.
```

¶ 7. Let E be an ordered set and let F be the set of all closures (§ 1, Exercise 13) in E. Order F by putting $u \leq v$ whenever $u(x) \leq v(x)$ for all $x \in E$. Then F has a least element e, the identity mapping of E onto itself. For each $u \in F$, let $I(u)$ denote the set of elements of E which are invariant under u.

(a) Show that $u \leq v$ in F if and only if $I(v) \subset I(u)$.

(b) Show that if every pair of elements of E has a greatest lower bound in E, then every pair of elements of F has a greatest lower bound in F. If E is a complete lattice, then so is F (§ 1, Exercise 11).

(c) Show that if E is inductive (with respect to the relation $\leq$), then every pair u, v of elements of F has a least upper bound in F (Show that if $f(x) = v(u(x))$ and if $w(x)$ denotes the greatest element of the chain of x, relative to f (Exercise 6), then w is a closure in E and is the least upper bound of u and v.)

Note. Point (c) is wrong.

Let's start with the definitions.

```
Section Exercise27.
Variable r: Set.
Hypothesis or: order r.
Let E := substrate r.

Definition closures  :=
  Zo (functions E E) (fun z=> closure z r).

Definition closure_ordering  :=
  induced_order (order_function E E r) (closures).
```

Let's show some trivial properties.

```
Lemma Exercise2_7aP f g: (* 9 *)
  gle (closure_ordering) f g <->
  [/\ inc f (closures), inc g (closures) &
    forall i, inc i (substrate r) -> gle r (Vf f i) (Vf g i)].
Lemma Exercise2_7b: (* 3 *)
  order_on closure_ordering closures.
Lemma Exercise2_7c: (* 11 *)
  least (closure_ordering) (identity (substrate r)).
```

(a) Assume $\mathrm{I}(g) \subset \mathrm{I}(f)$. Fix a; let $b = g(a)$. We have $a \le b$, thus $f(a) \le f(b)$. But $b \in \mathrm{I}(g)$ so that $f(b) = b$, thus $f(a) \le g(a)$. This shows $f \le g$. Conversely, assume $a \in \mathrm{I}(g)$. If $f \le g$, we have $a \le f(a) \le g(a) = a$. This shows $a \in \mathrm{I}(f)$.

```
Lemma Exercise2_7dP f g: (* 12 *)
  inc f closures -> inc g closures ->
  (gle (closure_ordering) f g <->
     sub (fixpoints g) (fixpoints f)).
```

(b) The infimum of a family of closures, if it exists, is a closure. We start with a family of two elements.

```
Lemma Exercise2_7e f g: (* 58 *)
  (forall x y, inc x E -> inc y E ->
   has_infimum r (doubleton x y)) ->
  inc f (closures) -> inc g (closures)  ->
  has_infimum (closure_ordering) (doubleton f g).
Lemma Exercise2_7f: complete_lattice r -> (* 65 *)
  complete_lattice (closure_ordering).
```

(c) Let u and v be two closures. The question is: under which condition is there a least upper bound. Define $f = u \circ v$. We have $x \le u(x) \le u(v(x))$ and $x \le v(x) \le u(v(x))$, so that $\mathrm{I}(f) = \mathrm{I}(u) \cap \mathrm{I}(v)$. Let w be an upper bound of u and v. Then $u(v(x)) \le u(w(x)) \le w(w(x))$, so $f(x) \le w(x)$. It follows: if w is a closure such that $\mathrm{I}(w) = \mathrm{I}(f)$ then $w = \sup(u, v)$.

Let $\mathrm{I}_x(f)$ be the set of all elements y such that $x \le y$ and $f(y) = y$. Assume that $\mathrm{I}_x(f)$ has a least element $g(x)$). Then $g = \sup(u, v)$. Assume E inductive, so that $\mathrm{I}_x(f)$ is non-empty. Assume E well-ordered, so that $\mathrm{I}_x(f)$ has a least element. In this case $\sup(u, v)$. exists.

```
Definition Ixf x f :=  Zo E (fun z => gle r x z /\ Vf f z = z).
Definition Jf f := (forall x, inc x E -> exists y,
        least (induced_order r (Ixf x f)) y).

Lemma Exercise2_7g u v: (* 89 *)
  inc u (closures) -> inc v (closures) ->
   Jf (u \co v) ->
   has_supremum (closure_ordering) (doubleton u v).
Lemma Exercise2_7h u v: (* 15 *)
  inductive r -> worder r ->
  inc u (closures) -> inc v (closures) ->
   has_supremum (closure_ordering) (doubleton u v).
```

Discussion. We may assume E non-empty, since otherwise the identity function is the only closure. If w is an upper bound of u and v, then $\mathrm{I}(w) \subset \mathrm{I}(u) \cap \mathrm{I}(v)$ so that $\mathrm{I}(u) \cap \mathrm{I}(v)$ must be non-empty. In order to achieve this, Bourbaki assumes E inductive. If M is the set of maximal elements, then $\mathrm{M} \subset \mathrm{I}(u)$, whatever u. If E is inductive, then M is a non-empty subset of $\mathrm{I}(u) \cap \mathrm{I}(v)$. Moreover, the chain of x (with respect to f) being well-ordered has an upper bound. But nothing says that it has a least upper bound (i.e., a greatest element) and the proposed construction fails.

If u is a closure, then $\mathrm{I}(u)$ is cofinal in E. Conversely, assume I cofinal and well-ordered. Define $u(x)$ to be the least element of I that is $\ge x$. Then u is a closure, and I is the set of its fixed points. Assume that E has no greatest element and I is as above; then (c) fails. Proof. Since I is well-ordered it is isomorphic to an ordinal α via a function ϕ. Let's says that $i \in \mathrm{I}$

is even (respectively odd) if the remainder of the division of $\phi(i)$ by 2 is zero (resp. non-zero). The sets of even and odd elements I_1 and I_2 of I are cofinal (note that I has no greatest element). This gives two closures u_1 and u_2 such that $I(u_1) \cap I(u_2) = \emptyset$.

If E is totally ordered then E inductive just says that E has a greatest element. If E has a greatest element a, then the constant function a is the greatest closure. In what follows, we shall assume E totally ordered, with a greatest element.

```
Lemma Exercise2_7B1 u: (* 4 *)
  inc u closures  -> cofinal r (fixpoints u).
Lemma Exercise2_7B2 F: (* 3 8 *-
  cofinal r F -> worder (induced_order r F) ->
  exists2 u, inc u closures & (fixpoints u) = F.
Lemma Exercise2_7B3 F: (* 100 *)
  cofinal r F -> worder (induced_order r F) ->
  ~ (has_greatest r) -> exists F1 F2,
   [/\ cofinal r F1, worder (induced_order r F1),
       cofinal r F2, worder (induced_order r F2) & disjoint F1 F2].
Lemma Exercise2_7B4 F: cofinal r F -> worder (induced_order r F) ->
  ~ (has_greatest r) -> nonempty E ->
  exists u v, [/\ inc u closures, inc v closures &
   ~  has_supremum closure_ordering (doubleton u v)]. (* 16 *)

Lemma Exercise2_7B5 : (* 12 *)
  has_greatest r -> has_greatest (closure_ordering).

End Exercise27.

Lemma Exercise2_7B6 r:
  has_greatest r -> inductive r.
Lemma Exercise2_7B7 r: total_order r -> inductive r -> has_greatest r.
```

Consider the ordinal sum $E = N_1 + N_2$, where N_1 is the set of natural integers, and N_2 the set of natural integers with the reverse ordering. This is is an inductive set, as N_2 has a greatest element. Note: N_2 is cofinal in E, but there is no closure u such that $I(u) = E_2$. In fact, there is no greatest subset of (N_2) which is the set of fixed points of a closure.

```
Definition NNstar :=
  order_sum2 Nat_order (opp_order Nat_order).

Lemma Exercise2_7A2 r r': (* 2 *)
  order r  -> order r' ->
  has_greatest r' -> inductive (order_sum2 r r').
Lemma Exercise2_7A3:  (* 11 *)
  [/\ order_on NNstar (canonical_du2 Nat Nat) &
    (forall x x', gle NNstar x x' <->
    [/\ inc x (canonical_du2 Nat Nat),
      inc x' (canonical_du2 Nat Nat) &
     [\/  [/\ Q x = C0, Q x' = C0 & (P x) <=c (P x')],
          [/\ Q x <> C0, Q x' <> C0 & (P x') <=c (P x)] |
         (Q x = C0 /\ Q x' <> C0)]])].
Lemma Exercise2_7A4: inductive NNstar. (* 7 *)
```

If f is a function $N_1 \mapsto N_1$ it can be extended to E by putting $f(x) = x$ on N_2. If f is a closure, its extension is a closure as well.

```
Definition extension_to_NNstar f :=
  Lf (fun z=> Yo (Q z = C0) (J (Vf f (P z)) C0) z)
  (substrate NNstar) (substrate NNstar).
Lemma Exercise2_7A5 f (g:= extension_to_NNstar f) (E:= substrate NNstar):
 function_prop f Nat Nat ->
   [/\ (forall x, natp x -> Vf g (J x C0) = (J (Vf f x) C0)),
       (forall x, inc x E -> Q x = C0 ->
         (P (Vf g x)  = Vf f (P x) /\ Q (Vf g x) = C0)),
    (forall x, inc x E -> Q x <> C0 ->  Vf g x = x) &
    function_prop g E E]. (* 14 *)
Lemma Exercise2_7A6 f: closure f Nat_order -> (* 49 *)
  closure  (extension_to_NNstar f) NNstar.
```

Let $v(x)$ be the function defined on **N** by: if x is even, then $v(x) = x$, otherwise $v(x) = x+1$, and u the same function with "even" replaced by "odd". These are closures, and there is no upper bound. In fact, if w is an upper bound, for any x, we have $u(x) \le w(x)$ and $v(x) \le w(x)$. We show that this implies $x \ne w(x)$ (which is false for $x = w(0)$).

```
Lemma Exercise2_7A7: (* 1 *)
  (lf_axiom (fun z => Yo (evenp z) z (csucc z)) Nat Nat).
Lemma Exercise2_7A8:  (* 1 *)
  (lf_axiom (fun z => Yo (evenp z) (csucc z) z) Nat Nat).
Lemma Exercise2_7A9:  (* 16 *)
  closure (Lf (fun z => Yo (evenp z) z (csucc z)) Nat Nat) Nat_order.
Lemma Exercise2_7A10:  (* 14 *)
  closure (Lf (fun z => Yo (evenp z) (csucc z) z) Nat Nat) Nat_order.

Lemma Exercise2_7A11 x w  (* 4 *)
    (u :=Lf (fun z => Yo (evenp z) (csucc z) z) Nat Nat)
    (v :=Lf (fun z => Yo (evenp z) z (csucc z)) Nat Nat):
   natp x -> (Vf u x) <=c w -> (Vf v x) <=c w ->
   x <> w.
```

The two closures u and v can be extended as u' and v' to the ordinal sum $\mathrm{E} = \mathrm{N}_1 + \mathrm{N}_2$. Every upper bound w satisfies: if $w(y) = y$, then $y \in \mathrm{N}_2$, because of `Exercise2_7A11`. It is easy to construct such functions: consider for instance the function f_y that maps x to $\sup(x, y)$. This is a closure, and is an upper bound if $y \in \mathrm{N}_2$.

Consider now any upper bound w. Let $k = w(0)$, where $0 \in \mathrm{N}_1$ is the least element of E. We have $k \in \mathrm{N}_2$, and, for $x \le k$ we have $w(x) = k$. Let $k' = k+1$ (in N_2). We have $k' < k$. Define $f(x)$ to be k' if $x \le k'$ and $w(x)$ otherwise. This is an upper bound and shows that w is not the least upper bound.

```
Lemma Exercise2_7A12: exists r u v,  (* 164 *)
  [/\ order r, inductive r,
  inc u (closures r), inc v (closures r) &
   ~  has_supremum (closure_ordering r) (doubleton u v)].
```

¶ 8. An ordered set E is said to be *ramified* (on the right) if, for each pair of elements x, y of E such that $x < y$, there exists $z > x$ such that y and z are not comparable. E is said to be *completely ramified* (on the right) if it is ramified and has no maximal elements. Every antidirected set (§ 1, Exercise 22) is ramified.

(a) Let E be an ordered set and let a be an element of E. Let $\mathfrak{R}_a$ denote the set of ramified subsets of E which have a as least element. Show that $\mathfrak{R}_a$, ordered by inclusion, has a maximal element.

(b) If E is branched (§ 1, Exercise 24), show that every maximal element of $\mathfrak{R}_a$ is completely ramified.

(c) Give an example of a branched set which is not ramified. The branched set defined in § 1, Exercise 24 (c) is completely ramified.

(d) Let E be a set in which each interval $]\leftarrow, x]$ is totally ordered. Show that E has an antidirected cofinal subset (§ 1, Exercise 22) (use (b)).

Notes. Points (b) (c) and (d) not yet done.

Solution. The first point is trivial. Assume $x < y$ in an antidirected set. There exists z such that $x < z$ and the intervals $[y, \rightarrow[$ and $[z, \rightarrow[$ do not intersect. This implies that y and z are non-comparable and the set is ramified.

```
Definition ramified r :=
 forall x y, glt r x y -> exists z, [/\ glt r x z, ~ gle r y z &  ~ gle r z y].

Definition ramifiedc r :=
  ramified r /\ not (exists x, maximal r x).

Lemma Exercise2_8a r:  (* 5 *)
  order r -> anti_directed r -> ramified r.
```

(a) Let $\mathfrak{R}_a$ be the set of all subsets Z of E such that the induced ordering is ramified and has a as least element. We first rewrite this condition in terms of the ordering on E. This set has a maximal element, thanks to Zorn's Lemma: consider a totally ordered family A_i of elements of $\mathfrak{R}_a$. If the family is empty, it has $\{a\}$ as upper bound, otherwise it has $\bigcup A_i$ as upper bound.

```
Definition  Exercise2_8a_R r a :=
  Zo (\Po (substrate r))
  (fun z => ramified (induced_order r z) /\
    least  (induced_order r z) a).

Lemma Exercise2_8b r a F: order r ->  (* 16 *)
 (inc F (Exercise2_8a_R r a) <->
  [/\ sub F (substrate r),
      forall x y, glt r x y -> inc x F -> inc y F ->
       exists z, [/\ glt r x z, ~ gle r y z,  ~ gle  r z y & inc z F],
     inc a F &
     (forall z, inc z F -> gle r a z)]).
Lemma Exercise2_8c r a: (* 31 *)
  order r -> inc a (substrate r) ->
  exists A, maximal (sub_order  (Exercise2_8a_R r a)) A.
```

9. An ordinal sum $\sum_{\iota\in I} E_\iota$ (§ 1, Exercise 3) is well-ordered if and only if I and each E_ι is well-ordered.

Solution. One implication is `orsum_wor`. The other one is easy. We assume E_I non-empty, so that I can be identified with a subset

```
Lemma orsum_wor_aux r g: (* 16 *)
  orsum_ax r g  -> worder (order_sum r g) -> (allf g worder).
Lemma orsum_wo_P r g:  (* 18 *)
  orsum_ax r g -> allf g ne_substrate  ->
  (worder (order_sum r g) <-> worder r /\   allf^~ worder g).
```

---

10. Let I be an ordered set and let $(E_\iota)_{\iota\in I}$ be a family of ordered sets, all equal to the same ordered set E. Show that the ordinal sum $\sum_{\iota\in I} E_\iota$ (§ 1, Exercise 3) is isomorphic to the lexicographic product of the sequence $(F_\lambda)_{\lambda\in\{\alpha,\beta\}}$, where the set $\{\alpha,\beta\}$ of two distinct elements is well-ordered by the relation whose graph is $\{(\alpha,\alpha),(\alpha,\beta),(\beta,\beta)\}$, and where $F_\alpha = I$ and $F_\beta = E$. This product is called the *lexicographic product* of E by I and is written E.I.

Solution. This is lemma `order_prod_pr` of Chapter 11.

¶ 11. *Let I be a well-ordered set and let $(E_\iota)_{\iota\in I}$ be a family of ordered sets, each of which contains at least two distinct comparable elements. Then the lexicographic product of the E_ι is well-ordered if and only if each of the E_ι is well-ordered and I is *finite* (if I is infinite, construct a strictly decreasing infinite sequence in the lexicographic product of the E_ι).*

Note. The exercise is surrounded by stars because the term “finite” is not yet defined. The assumptions can be weakened.

Auxiliary result. (now lemma `worder_decreasing_finite` in the main text). If $f : I \to J$ is strictly decreasing, both sets I and J are well-ordered, then I is finite. Proof: Let K be the image of f, it is well-ordered for $\le$ and $\ge$ (note that f is an order isomorphism for $\ge$), thus is finite (cf Exercise 4.3 below), thus I is finite. We present here an alternate proof. Since the set of integers is well-ordered, there is an order isomorphism $g : \mathbf{N} \to s(I)$ or $g : I \to s(\mathbf{N})$, where $s(K)$ means “segment of K”. We may assume $s(\mathbf{N}) \ne \mathbf{N}$, for this reduces to the first case; and then $s(\mathbf{N})$ is finite as well as I. The first case is absurd: the set of $f(g(i))$ has a least element (in J), and this gives a greatest element of $\mathbf{N}$, absurd.

Solution. Let’s consider a family of orderings E_i, indexed by a well-ordered set I. Let (H) the condition that no E_i is empty. If (H) fails, then the product is empty, thus total and well-ordered, and nothing can be said for those sets E_i that are non-empty. So we assume (H).

By (H), there is an element f in the product, and for any $i \in I$, any $x \in E_i$, there is $\bar{x}$ in the product, such that $\bar{x}_i = x$ and $\bar{x}_j = f_j$ for $j \ne i$. This means that E_i is isomorphic to a subset of the product, so that, if the product is totally ordered or well-ordered, so is each factor.

If I is finite and each factor is well-ordered, then the product is well-ordered. Proof. Let i be the least element of I, X be a non-empty subset of the product, and X_i the i-th projection.

This is a non-empty subset of E_i and has a least element a. Let $\bar{X}$ the set of elements of X for which the i-th component is equal to a. If $x \in \bar{X}$ and $y \in X - \bar{X}$ then $x < y$. Denote by f' the restriction of f to $I - \{i\}$. We denote by $\bar{X}'$ the set of restrictions. By induction, the restriction product is well-ordered and this set has a least element x', that is the restriction of some element $x \in \bar{X}$. If $y \in \bar{X}$ then $x \le y$ if and only if $x' \le y'$ in the restriction product. This x is the least element of $\bar{X}$, hence of X. This is lemma `orprod_wor` of the main text.

Converse. Let's say that a set is big if it is not small; this means that it has at least two elements. Consider a big family. By the axiom of choice, the product contains two elements x and y such that $x_i \neq y_i$ for every index i. If E_i is totally ordered, then $\inf(x_i, y_i) < \sup(x_i, y_i)$. Hence the product contains two elements x and y such that $x_i < y_i$ for every index i. Define $z(k)$ by $z(k)_i = x_i$ if $i < k$ and $z(k)_i = y_i$ otherwise. This is a decreasing function of k, and is strictly decreasing on J, the set of all indices such that E_i is not a singleton. It follows J finite.

```
Definition all_big_substrate g :=
  (forall i, inc i (domain g) -> ~ (small_set (substrate (Vg g i)))).

Lemma all_big_substrate_prop  g : all_big_substrate g ->  (* 18 *)
  exists f1 f2, [/\ inc f1 (prod_of_substrates g),
   inc f2 (prod_of_substrates g) &
   forall i, inc i (domain g) -> (Vg f1 i) <> (Vg f2 i)].

Section Exercise2_11.
Variables (r g: Set).
Hypothesis oa:  orprod_ax r g.

Lemma orprod_total2: allf g ne_substrate -> (* 37 *)
  ((total_order (order_prod r g) -> (allf g total_order))
  /\
   (worder (order_prod r g) -> (allf g  worder)).)

Lemma orprod_total3P: allf g ne_substrate  -> (* 2 *)
  ( (allf g total_order) <-> total_order (order_prod r g)).

Lemma orprod_total4: (* 22 *)
  total_order (order_prod r g) ->
  all_big_substrate g  ->
  exists f1 f2,
   [/\ inc f1 (substrate (order_prod r g)),
       inc f2 (substrate (order_prod r g)) &
       forall i, inc i (domain g) -> glt (Vg g i) (Vg f1 i) (Vg f2 i)].
Lemma orprod_worder_bisP:  (* 33 *)
  all_big_substrate g ->
  ( ( allf g  worder /\ finite_set (substrate r))
    <-> worder (order_prod r g)).
```

¶ 12. Let I be a totally ordered set and let $(E_\iota)_{\iota\in I}$ be a family of ordered sets indexed by I. Let $R\{x, y\}$ denote the following relation on $E = \prod_{\iota\in I} E_\iota$: "the set of indices $\iota \in I$ such that $\mathrm{pr}_\iota x \neq \mathrm{pr}_\iota y$ is well-ordered, and if κ is the least element of this subset of I, we have $\mathrm{pr}_\kappa x < \mathrm{pr}_\kappa y$". Show that $R\{x, y\}$ is an order relation between x and y on E. If the E_ι are totally ordered, show that the connected components of E with respect to the relation "x and y are comparable"

(Chapter II, § 6, Exercise 10) are totally ordered sets. Suppose that each E_ι has at least two elements. Then E is totally ordered if and only if I is well-ordered and each E_ι is totally ordered (use Exercise 3); and E is then the lexicographic product of the E_ι.

Solution. Assume that E is is a totally ordered set, A and B are two well-ordered subsets of E. Let C be a subset of $A \cup B$; it is well-ordered: consider a non-empty subset X. If X does not meet both A and B it is contained in one of them, so X has a least element. Otherwise, the intersections have a least element a and b. Then $\inf(a, b)$ is the least element of X.

```
Definition olex_nsv r x y:=
  Zo (substrate r) (fun i =>  (Vg x i <> Vg y i)).
Definition olex_io r x y:=   (induced_order r (olex_nsv r x y)).

Definition olex_comp1_r r g x y :=
  worder (olex_io r x y) /\
  let i := the_least (olex_io r x y) in glt (Vg g i) (Vg x i) (Vg y i).

Definition olex_comp2_r r g x y :=
  [/\ (inc x (prod_of_substrates g)),
      (inc y (prod_of_substrates g)) &
      (x = y \/ olex_comp1_r r g x y) ].

Definition olex r g := graph_on  (olex_comp2_r r g) (prod_of_substrates g).
Definition olex_ax r g:=
  [/\ total_order r, substrate r = domain g & order_fam g].

Lemma union2_wor r A B C:  (* 33 *)
  total_order r -> sub A (substrate r) ->  sub B (substrate r) ->
  sub C (A \cup B) ->
  worder (induced_order r A) ->worder (induced_order r B) ->
  worder (induced_order r C).
Lemma olex_nsvS r x y: olex_nsv r x y = olex_nsv r y x.  (* 1 *)
Lemma olex_ioS r x y: olex_io r x y = olex_io r y x.  (* 1 *)
```

Let T_{xy} be the set of indices i such that $x_i \neq y_i$. If T_{xy} and T_{yz} are well-ordered so is T_{yz}. Let a, b and c be the least element of these sets. We have $c = \inf(a, b)$, and this shows that $R\{x, y\}$ is transitive, thus is an order relation.

```
Section Olex_basic.
Variables (r g: Set).
Hypothesis ax: olex_ax r g.

Lemma olex_R x:  (* 1 *)
  inc x (prod_of_substrates g) -> olex_comp2_r r g x x.
Lemma olex_gleP x y:  (* 2 *)
  gle (olex r g) x y <-> olex_comp2_r r g x y.
Lemma olex_nsve x y:  (* 8 *)
   inc x (prod_of_substrates g) -> inc y (prod_of_substrates g) ->
   ((x = y) <-> (olex_nsv r x y = emptyset)).
Lemma olex_nsve1 x y: (* 9 *)
  inc x (prod_of_substrates g) -> inc y (prod_of_substrates g) ->
  let r' := (olex_io r x y) in let i := the_least r' in
    x <> y -> worder r' -> least r' i.

Lemma olex_glt_aux x y (r' := olex_io r x y) (i := the_least r'): (* 13 *)
```

```
  glt (olex r g) x y  ->
  (inc i (substrate r) /\ forall j, glt r j i -> Vg x j = Vg y j).
Lemma olex_osr: order_on  (olex r g) (prod_of_substrates g).  (* 78 *)
```

Consider now the second point. Let $x \sim y$ be "$x \le y$ or $y \le x$". This relation is equivalent to: the set of indices such that $x_i \neq y_i$ is well-ordered, and, if this set is non-empty and has a least element j, then x_j and y_j are comparable. If all E_ι are totally ordered, this second condition becomes trivial, $x \sim y$ simplifies to T_{xy} is well-ordered, so that $\sim$ is transitive. If x and y are two elements in the same connected component for $\sim$, by transitivity, we get $x \sim y$.

Assume I well-ordered. The condition that T_{xy} is well-ordered becomes trivial. In this case, the ordering on E is the lexicographic ordering, so that E is totally ordered provided that each E_i is totally ordered. Conversely, assume E totally ordered and each E_i has at least two elements. Consider two elements x and y in E such that $x_i \neq y_i$ whatever i. If $x = y$, then I is empty. Otherwise, to say that x and y are comparable implies I well-ordered. Then the ordering on E is the lexicographic ordering and each E_i is totally ordered.

```
Lemma olex_cc_comparable1  (r':= olex r g):  (* 41 *)
  (allf g total_order) ->
  forall x y,
      ocomparable r' x y <-> [/\ inc x (substrate r'), inc y (substrate r') &
          worder (olex_io r x y)].
Lemma olex_cc_comparable2  (r':= olex r g): (* 11 *)
  (allf g total_order)  -> transitive_r (ocomparable r').

Lemma olex_cc_tor (r':= olex r g):  (* 29 *)
  (allf g total_order)  ->
  forall x, inc x (substrate r') ->
   total_order (induced_order r'
        (connected_comp  (ocomparable r') (substrate r') x)).

Lemma olex_lex: worder r -> olex r g = order_prod r g. (* 25 *)
Lemma olex_total1: worder r -> (* 2 *)
  (allf g total_order)  -> total_order (olex r g).
Lemma olex_total2:  (* 28 *)
  total_order (olex r g) ->
  all_big_substrate g ->
  (worder r /\ (allf g total_order)).
End Olex_basic.
```

---

13. (a) Let $\mathrm{Is}(\Gamma, \Gamma')$ be the relation "Γ is an ordering (on E), and Γ' is an ordering (on E$'$), and there exists an isomorphism of E, ordered by Γ, onto E$'$, ordered by Γ'". Show that $\mathrm{Is}(\Gamma, \Gamma')$ is an equivalence relation on every set whose elements are orderings. The term $\tau_\Delta(\mathrm{Is}(\Gamma, \Delta))$ is an ordering called the *order-type* of Γ and denoted by $\mathrm{Ord}(\Gamma)$, or $\mathrm{Ord}(\mathrm{E})$ by abuse of notations. Two ordered sets are isomorphic if and only if their order-types are equal.

(b) Let $\mathrm{R}\wr\lambda, \mu\wr$ be the relation: "λ is an order-type, and μ is an order-type and there exists an isomorphism of the set ordered by λ onto a subset of the set ordered by μ". Show that $\mathrm{R}\wr\lambda, \mu\wr$ is a preorder relation between λ and μ. It will be denoted by $\lambda \prec \mu$.

(c) Let I be an ordered set and let $(\lambda_\iota)_{\iota \in \mathrm{I}}$ be a family of order-types indexed by I. The order-type of the ordinal sum (§ 1, Exercise 3) of the family of sets ordered by the λ_ι $(\iota \in \mathrm{I})$ is called the *ordinal sum* of the order-types λ_ι ($\iota \in I$) and is denoted by $\sum_{\iota\in I}\lambda_\iota$. If $(E_\iota)_{\iota\in I}$ is a family of ordered sets, the order type of $\sum_{\iota\in I}E_\iota$ is $\sum_{\iota\in I}\mathrm{Ord}(E_\iota)$. If I is the ordinal sum of a family $(J_\kappa)_{\kappa\in K}$, show that

$$\sum_{\kappa\in K}\Big(\sum_{\iota\in J_\kappa}\lambda_\iota\Big) = \sum_{\iota\in I}\lambda_\iota.$$

(d) Let I be a well-ordered set and $(\lambda_\iota)_{\iota\in I}$ be a family of order-types indexed by I. The order-type of the lexicographic product of the family of sets indexed by by the λ_ι ($\iota \in I$) is called the *ordinal product* of the order-types λ_ι ($\iota \in I$) and is denoted by $\mathrm{P}_{\iota\in I}\lambda_\iota$. If $(E_\iota)_{\iota\in I}$ is a family of ordered sets, the order type of the lexicographic product of the family $(E_\iota)_{\iota\in I}$ is $\mathrm{P}_{\iota\in I}\mathrm{Ord}(E_\iota)$. If I is the ordinal sum of a family of well-ordered sets $(J_\kappa)_{\kappa\in K}$ indexed by a well-ordered set K, show that

$$\mathrm{P}_{\kappa\in K}\Big(\mathrm{P}_{\iota\in J_\kappa}\lambda_\iota\Big) = \mathrm{P}_{\iota\in I}\lambda_\iota.$$

(e) We denote by $\lambda+\mu$ (resp. $\mu\lambda$) the ordinal sum (resp. ordinal product) of the family $(\xi_\iota)_{\iota\in J}$ where $J = \{\alpha,\beta\}$ is a set with two distinct elements, ordered by the relation whose graph is $\{(\alpha,\alpha),(\alpha,\beta),(\beta,\beta)\}$, and where $\xi_\alpha = \lambda$ and $\xi_\beta = \mu$. Show that if I is a well-ordered set of order-type λ and if $(\mu_\iota)_{\iota\in I}$ is a family of order-types such that $\mu_\iota = \mu$ for each $\iota \in I$ then $\sum_{\iota\in I}\mu_\iota = \mu\lambda$. We have $(\lambda+\mu)+\nu = \lambda+(\mu+\nu)$, $(\lambda\mu)\nu = \lambda(\mu\nu)$, and $\lambda(\mu+\nu) = \lambda\mu+\lambda\nu$ (but in general $\lambda+\mu \neq \mu+\lambda$, $\lambda\mu \neq \mu\lambda$ and $(\lambda+\mu)\nu \neq \lambda\nu+\mu\nu$).

(f) Let $(\lambda_\iota)_{\iota\in I}$ and $(\mu_\iota)_{\iota\in I}$ be two families of order-types indexed by the same ordered set I. Show that if $\lambda_\iota \prec \mu_\iota$ for each $\iota \in I$, then $\sum_{\iota\in I}\lambda_\iota \prec \sum_{\iota\in I}\mu_\iota$ and (if I is well-ordered) $\mathrm{P}_{\iota\in I}\lambda_\iota \prec \mathrm{P}_{\iota\in I}\mu_\iota$. If J is a subset of I, show that $\sum_{\iota\in J}\lambda_\iota \prec \sum_{\iota\in I}\lambda_\iota$ and (if I is well-ordered and the λ_ι are non-empty) $\mathrm{P}_{\iota\in J}\lambda_\iota \prec \mathrm{P}_{\iota\in I}\lambda_\iota$.

(g) Let λ^* denote the order-type of the set ordered by the opposite of the ordering λ. Then we have

$$(\lambda^*)^* = \lambda \qquad \text{and} \qquad \Big(\sum_{\iota\in I}\lambda_\iota\Big)^* = \sum_{\iota\in I^*}\lambda_\iota^*$$

where I^* denotes the set I endowed with the opposite of the ordering given on I.

Discussion.

(a) Lemmas `orderIR`, `orderIS` and `orderIT` show that the relation “Is” is an equivalence relation (on every set whose elements are orders). Axiom scheme S7 of Bourbaki says that two isomorphic sets has the same order type. Since this axiom scheme does not hold in our framework, we add, in section 11.28, an axiom, that says that $\mathrm{Ord}(\Gamma)$ is order-isomorphic to Γ whenever Γ is an ordering, and $\mathrm{Ord}(\Gamma) = \mathrm{Ord}(\Gamma')$ whenever Γ and Γ' are order isomorphic. If Γ and Γ' have the same-order type, they are isomorphic to their order-type, thus order-isomorphic. We have defined $\mathrm{ord}(\Gamma)$, that behaves exactly as $\mathrm{Ord}(\Gamma)$, when Γ is a well-ordering. Each property shown here (except (g)) is also proved for well-orderings, named `ofoo` instead of `OT_foo`.

(b) Lemmas `OT_order_le_reflexive` and `OT_order_le_transitive` show that $\prec$ is a preorder.

(c) is `OT_sum_invariant3` and `OT_sum_assoc1`.

(d) is `OT_prod_invariant3` and `OT_prod_assoc1`.

(e) is `OT_prod_pr1`, `OT_sum_assoc3`, `OT_prod_assoc3`, and `OT_sum_distributive3`.

(f) The first result is `OT_sum_increasing2`; there are three other results.

(g) is implemented as `OT_double_opposite` and `OT_opposite_sum`.

¶ 14. An *ordinal* is the order-type of a well-ordered set (Exercise 13).

(a) Show that, if $(\lambda_\iota)_{\iota\in I}$ is a family of ordinals indexed by a well-ordered set I, then the ordinal sum $\sum_{\iota\in I}\lambda_\iota$ is an ordinal; *and that, if moreover, I is finite, then the ordinal product $\mathsf{P}_{\iota\in I}\lambda_\iota$ is an ordinal (Exercise 11).* The order-type of the empty set is denoted by 0, and that of a set with one element by 1 (by abuse of language, cf. Show that

$$\alpha + 0 = 0 + \alpha = \alpha \qquad \text{and} \qquad \alpha.1 = 1.\alpha = \alpha$$

for every ordinal α.

(b) Show that the relation "λ is an ordinal and μ is an ordinal and $\lambda \prec \mu$" is a *well-ordering* relation, denoted by $\lambda \leq \mu$ (Note that, if λ are μ are ordinals, the relation $\lambda \prec \mu$ is equivalent to "λ is equal to the order-type of a segment of μ" (no. 5, Theorem 3, Corollary 3): given a family $(\lambda_\iota)_{\iota\in I}$ of ordinals, consider a well-ordering in I and take the ordinal sum of the family of sets ordered by the λ_ι; finally use Proposition 2 of no. 1.)

(c) Let α be an ordinal. Show that the relation "ξ is an ordinal and $\xi \leq \alpha$" is collectivizing in ξ, and that the set O_α of ordinals $< \alpha$ is a well-ordered set such that $\mathrm{Ord}(O_\alpha) = \alpha$. We shall often identify O_α with α.

(d) Show that for every family of ordinals $(\xi_\iota)_{\iota\in I}$ there exists a unique ordinal α such that the relation "λ is an ordinal and $\xi_\iota \leq \lambda$ for all $\iota \in I$" is equivalent to $\alpha \leq \lambda$. By abuse of language, α is called the *least upper bound* of the family of ordinals $(\xi_\iota)_{\iota\in I}$, and we write $\alpha = \sup_{\iota\in I}\xi_\iota$ (it is the greatest element of the union of $\{\alpha\}$ and the set of the ξ_ι). The least upper bound of the set of ordinals $\xi < \alpha$ is either α or an ordinal β such that $\alpha = \beta + 1$. In the latter case β is said to be the *predecessor* of α.

Note. Bourbaki signals an abuse of language, but there are many of them here. In Exercise 2.1, an ordering is a correspondence with a graph, so that 0 is really $(\emptyset, \emptyset, \emptyset)$ and might be confused with the cardinal 0. If an ordering is a graph, then $0 = \emptyset$, and this is the same as the cardinal 0. "The order-type of the empty set" should be replaced by "the order-type of the unique ordering of the empty set". It happens that (if an ordering is a graph), that this ordering is really $\emptyset$. In the same fashion, "that of a set with one element" should be replaced by: any singleton has a unique ordering, which is a well-ordering, all these orderings are isomorphic, thus have the same order-type. This ordinal has the form $\{(\alpha,\alpha)\}$ and is an ordering on $\{\alpha\}$. In $\alpha.1 = 1.\alpha$, the dot has to be understood as the ordinal product. Exercise 2.11 says that a finite ordinal product is well-ordered; it relies on the property of being "finite" which is not yet defined, thus the stars.

Discussion. To each well-ordering Γ, one may associate its von Neumann ordinal $\mathrm{ord}(\Gamma)$. This set has a natural ordering $o(\mathrm{ord}(\Gamma))$, which order-isomorphic to Γ. These two orderings have the same order-type by lemma `OT_ordinal_compat`. This allows us to use "ord" whenever Bourbaki uses "Ord". This means that, whenever Bourbaki considers an ordinal γ (that orders E), we consider a von Neumann ordinal X, with its ordering $o(X)$.

(a) The first two properties are `OS_sum2` and `OS_prod2`. We have also `osum0r`, and variants.

(b) is `wordering_ole`, and is trivial. This is because, if λ and μ are ordinals, then $\lambda \prec \mu$ is the same as $\lambda \subset \mu$; lemma `ordinal_le_P1` says that it is equivalent to "λ is equal to the order-type of a segment of μ". Consider now a family λ_i of ordinals. There is a well-ordered set E and an element μ_i of E such that $\lambda_i \to \mu_i$ is an order isomorphism. Bourbaki hints to consider the ordinal sum. In the case of von Neumann ordinals, one could consider the union; however, the intersection of all ordinals is the least element of the family.

(c) The lemmas `oleP` and `oltP` say that for any ordinal α, the relations $x \in \alpha^+$ and $x \in \alpha$ are equivalent to $x \leq \alpha$ and $x < \alpha$. Lemma `set_ord_lt_prop3` says that the ordinal of the set of $x < \alpha$, ordered by $\leq$ is α. This $O_\alpha = \alpha$ and $O'_\alpha = \alpha^+$.

(d) As mentioned above, the union is the least upper bound. We show below that "it is the greatest element of the union of $\{\alpha\}$ and the set of the ξ_ι)" (this property characterizes the upper bound, thus is not very interesting). Lemma `ord_sup_pr7` says that the supremum β of O_α is α or $\alpha = \beta^+$ (this is the ordinal obtained from β by adjoining a greatest element), lemma `osucc_pr` shows that it is $\beta + 1$.

```
Lemma ord_sup_pr6 E: ordinal_set E -> (* 9 *)
  greatest (ole_on (E +s1 (\osup E))) (\osup E).
```

---

15. (a) Let α and β be two ordinals. Show that the inequality $\alpha < \beta$ is equivalent to $\alpha + 1 \leq \beta$, and that it implies the inequalities $\xi + \alpha < \xi + \beta$, $\alpha + \xi \leq \beta + \xi$, $\alpha\xi \leq \beta\xi$ for all ordinals ξ, and $\xi\alpha < \xi\beta$ if $\xi > 0$.

(b) Deduce from (a) that there exists no set to which every ordinal belongs (use Exercise 14 (d)).

(c) Let α, β, μ be three ordinals. Show that each of the relations $\mu + \alpha < \mu + \beta$, $\alpha + \mu < \beta + \mu$ implies $\alpha < \beta$; and that each of the relations $\mu\alpha < \mu\beta$, $\alpha\mu < \beta\mu$ implies $\alpha < \beta$ provided that $\mu > 0$.

(d) Show that the relation $\mu + \alpha = \mu + \beta$ implies $\alpha = \beta$, and that $\mu\alpha = \mu\beta$ implies $\alpha = \beta$ provided that $\mu > 0$.

(e) Two ordinals α and β are such that $\alpha \leq \beta$ if and only if there exists an ordinal ξ such that $\beta = \alpha + \xi$. This ordinal is then unique and is such that $\xi \leq \beta$; it is written $(-\alpha) + \beta$.

(f) Let α, β, ζ be three ordinals such that $\zeta < \alpha\beta$. Show that there exist two ordinals ξ, η such that $\zeta = \alpha\eta + \xi$ and $\xi < \alpha$, $\eta < \beta$ (cf. No. 5, Theorem 3, Corollary 3). Moreover, ξ and η are uniquely determined by these conditions.

Note. Corollary 3 of Theorem 3 of No 5 says that every subset of a well-ordered set is isomorphic to a segment.

Solution

(a) The first claim is `ord_succ_lt2`. See relations (11.8) and followings in Chapter 11.

(b) is `ordinal_not_collectivizing`.

(c) is implemented in (11.12); The "provided $\mu > 0$" is obviously superfluous.

(d) is `osum2_simpl` and `oprod2_simpl`.

(e) is `odiff_pr`.

(f) is `odivision_exists` and `odivision_unique`.

¶ 16. An ordinal $\rho > 0$ is said to be *indecomposable* if there exists no pair of ordinals ξ, η such that $\xi < \rho$, $\eta < \rho$, and $\xi + \eta = \rho$.

(a) An ordinal ρ is indecomposable if and only if $\xi + \rho = \rho$ for every ordinal ξ such that $\xi < \rho$.

(b) If $\rho > 1$ is an indecomposable ordinal and if α is any ordinal > 0, then $\alpha\rho$ is indecomposable, and conversely (use Exercise 15(f)).

(c) If ρ is indecomposable and if $0 < \alpha < \rho$, then $\rho = \alpha\xi$, where ξ is an indecomposable ordinal (use Exercise 15(f)).

(d) Let α be an ordinal > 0. Show that there exists a greatest indecomposable ordinal among the indecomposable ordinals $\leq \alpha$ (consider the decomposition $\alpha = \rho + \xi$, where ρ is indecomposable).

(e) If E is a set of indecomposable ordinals, deduce from (d) that the least upper bound of E (Exercise 14(d)) is an indecomposable ordinal.

Solution. Point (a) (b) and (c) are lemmas `indecompP`, `indecomp_prodP`, and `indecomp_div`. Points (d) and (e) are `indecomp_sup1` and `indecomp_sup`.

Point (d). The hint is strange. One might consider the least ξ such that there is an indecomposable ρ such that $\alpha = \rho + \xi$. However, this does not give the greatest ρ, since we cannot simplify on the right.

Solution[1]. Let n be the exponent of the Cantor Normal Form of α. Then ω^n is the solution (ordinal power and CNF are defined below).

Alternate solution. The key relation is: if ρ is indecomposable, and $\xi < \rho$, $\eta < \rho$, then $\xi + \eta < \rho$ (see main text). Let ρ be the supremum of the set of indecomposable ordinals $\leq \alpha$, $\xi < \rho$, $\eta < \rho$. There is an indecomposable β such that $\xi < \beta$ and $\eta < \beta$, and moreover $\beta \leq \alpha$, so that $\xi + \eta < \beta \leq \rho$.

¶ 17. Given an ordinal α_0, a term $f(\xi)$ is said to be an *ordinal functional symbol (with respect to* ξ*) defined for* $\xi \geq \alpha_0$ if the relation "ξ is an ordinal and $\xi \geq \alpha_0$" implies the relation "$f(\xi)$ is an ordinal"; $f(\xi)$ is said to be *normal* if the relation $\alpha_0 \leq \xi < \eta$ implies $f(\xi) < f(\eta)$ and if for each family $(\xi_\iota)_{\iota\in I}$ of ordinals $\geq \alpha_0$ we have $\sup_{\iota\in I} f(\xi_\iota) = f(\sup_{\iota\in I} \xi_\iota)$ (cf. Exercise 14(d)).

(a) Show that for each ordinal $\alpha > 0$, $\alpha + \xi$ and $\alpha\xi$ are ordinal functional symbols defined for $\xi \geq 0$ (use Exercise 15(f)).

(b) Let $w(\xi)$ be an ordinal functional symbol defined for $\xi \geq \alpha_0$ such that $w(\xi) \geq \xi$ and such that $\alpha_0 \leq \xi < \eta$ implies $w(\xi) < w(\eta)$. Also let $g(\xi,\eta)$ be a term such that the relation "ξ and η are ordinals $\geq \alpha_0$" implies the relation "$g(\xi,\eta)$ is an ordinal such that $g(\xi,\eta) > \xi$". Define a term $f(\xi,\eta)$ with the following properties: (1) for each ordinal $\xi \geq \alpha_0$, $f(\xi,1) = w(\xi)$; (2) for each ordinal $\xi \geq \alpha_0$ and each ordinal $\eta > 1$, $f(\xi,\eta) = \sup_{0<\zeta<\eta} g(f(\xi,\zeta),\xi)$ (use Criterion C60 of no. 2). Show that if $f_1(\xi,\eta)$ is another term with these properties, then $f(\xi,\eta) = f_1(\xi,\eta)$ for all

[1] This was our first implementation

$\xi \geq \alpha_0$ and all $\eta \geq 1$. Prove that, for each ordinal $\xi \geq \alpha_0$, $f(\xi,\eta)$ is a normal functional symbol with respect to η (defined for all $\eta \geq 1$). Show that $f(\xi,\eta) \geq \xi$ for all $\eta \geq 1$ and $\xi \geq \alpha_0$ and that $f(\xi,\eta) \geq \eta$ for all $\xi \geq \sup(\alpha_0, 1)$ and $\eta \geq 1$. Furthermore, for each pair (α,β) of ordinals such that $\alpha > 0$, $\alpha \geq \alpha_0$, and $\beta \geq w(\alpha)$ there exists a unique ordinal ξ such that

$$f(\alpha,\xi) \leq \beta < f(\alpha,\xi+1),$$

and we have $\xi \leq \beta$.

(c) If we take $\alpha_0 = 0$, $w(\xi) = \xi + 1$, $g(\xi,\eta) = \xi + 1$ then $f(\xi,\eta) = \xi + \eta$. If we take $\alpha_0 = 1$, $w(\xi) = \xi$, $g(\xi,\eta) = \xi + \eta$ then $f(\xi,\eta) = \xi\eta$.

(d) Show that if the relations $\alpha_0 \leq \xi \leq \xi'$, $\alpha_0 \leq \eta \leq \eta'$ imply $g(\xi,\eta) \leq g(\xi',\eta')$, then the relations $\alpha_0 \leq \xi \leq \xi'$, $1 \leq \eta \leq \eta'$ imply $f(\xi,\eta) \leq f(\xi',\eta')$. If the relations $\alpha_0 \leq \xi \leq \xi'$, $\alpha_0 \leq \eta < \eta'$ imply $g(\xi,\eta) < g(\xi,\eta')$ and $g(\xi,\eta) \leq g(\xi',\eta)$, then the relations $\alpha_0 \leq \xi < \xi'$ and $\eta \geq 0$ imply $f(\xi,\eta+1) < f(\xi',\eta+1)$.

(e) Suppose that $w(\xi) = \xi$ and that the relations $\alpha_0 \leq \xi \leq \xi'$, $\alpha_0 \leq \eta < \eta'$ imply $g(\xi,\eta) < g(\xi,\eta')$ and $g(\xi,\eta) \leq g(\xi',\eta)$. Suppose, moreover, that for each $\xi \geq \alpha_0$, $g(\xi,\eta)$ is a normal functional symbol with respect to η (defined for $\eta \geq \alpha_0$), and that, whenever $\xi \geq \alpha_0$, $\eta \geq \alpha_0$, and $\zeta \geq \alpha_0$, we have the associativity relation

$$g(g(\xi,\eta),\zeta) = g(\xi, g(\eta,\zeta)).$$

Show that, if $\xi \geq \alpha_0$, $\eta \geq 1$, and $\zeta \geq 1$, then

$$g(f(\xi,\eta), f(\xi,\zeta)) = f(\xi,\eta+\zeta)$$

("distributivity" of g with respect to f) and

$$f(f(\xi,\eta),\zeta) = f(\xi,\eta\zeta)$$

("associativity" of f).

Note. Instead of "let $f(\xi)$ be an ordinal functional symbol (with respect to ξ)", we say "let f be an OFS", or let $\xi \mapsto f(\xi)$ be an OFS. Defining $f(\xi,0) = w_0(\xi)$ and allowing $\zeta = 0$ in (2) simplifies some formulas.

Solution.

(a) is `osum_normal` and `oprod_normal`.

(b): Lemmas `ord_induction_exists` and `ord_induction_unique`. assert existence and uniqueness of f. It is a normal function by `ord_induction_p10`. Existence and uniqueness of β is given by `ord_induction_p11` and `ord_induction_p12`.

(c) is `ord_induction_p13` and `ord_induction_p14`.

(d) is `ord_induction_p15` and `ord_induction_p17`.

(e) is `ord_induction_p18` and `ord_induction_p19`.

¶ 18. In the definition procedure defined in Exercise 17 (b), take $\alpha_0 = 1 + 1$ (denoted by 2 by abuse of language),

$$w(\xi) = \xi, \qquad g(\xi,\eta) = \xi\eta.$$

Denote $f(\xi,\eta)$ by ξ^η and define α^0 to be 1 for all ordinals α. Also define 0^β to be 0 and 1^β to be 1 for all ordinals $\beta \geq 1$.

(a) Show that if $\alpha > 1$ and $\beta < \beta'$, we have $\alpha^\beta < \alpha^{\beta'}$, and that, for each ordinal $\alpha > 1$, α^ξ is a normal functional symbol with respect to ξ. Moreover, if $0 < \alpha \leq \alpha'$, we have $\alpha^\beta \leq \alpha'^\beta$.

(b) Show that $\alpha^\xi.\alpha^\eta = \alpha^{\xi+\eta}$ and $(\alpha^\xi)^\eta = \alpha^{\xi\eta}$.

(c) Show that, if $\alpha \geq 2$ and $\beta \geq 1$, $\alpha^\beta \geq \alpha\beta$.

(d) For each pair of ordinals $\beta \geq 1$ and $\alpha \geq 2$, there exists three ordinals ξ, γ, δ such that $\beta = \alpha^\xi\gamma + \delta$, where $0 < \gamma < \alpha$ and $\delta < \alpha^\xi$, and these ordinals are uniquely by these conditions.

Solution.

(a) is `opow_increasing2`, `opow_normal`, `opow_increasing4`.

(b) is `opow_sum` and `opow_prod`.

(c) is `opow_increasing5`.

(d) is `ord_ext_div_unique` and `ord_ext_div_exists`.

---

19. * Let α and β be two ordinals and let E and F be two well-ordered sets such that Ord(E) = α and Ord(F) = β. In the set E^F of mappings of F into E, consider the subset G of mappings g such that $g(y)$ is equal to the least element of E for all but a *finite* number of elements $y \in F$. If F^* is the ordered set obtained by endowing F with the opposite order, show that G is a connected component with respect to the relation "x and y are comparable" (Chapter II, § 6, Exercise 10) in the product E^{F^*} endowed with the ordering defined in Exercise 12, and show that G is well-ordered. Furthermore, prove that Ord(G) = α^β (use the uniqueness property of Exercise17 (b)).$_*$

Note. The star is missing at the end of the exercise. These stars indicate that the exercise requires notions not yet define (here "finite").

Solution. We consider two orderings r and r' corresponding to E and F respectively. Let $\bar{r}'$ be the opposite ordering of r'. Consider the constant function defined on F that maps any element to r. If we apply the construction of Exercise 12, we get an ordering on E^F, the set of graphs of functions from F to E, that satisfies some properties. Let m be the least element of E, and $\bar{m}$ the constant function F → E with value m, and I_x the set of indices i such that x_i is not m. Let G be the set of functions $x : F \to E$ for which I_x is finite. We consider the ordering induced on G by that of E^F.

```
Section OlexPowBasic.
Variables (r r': Set).
Hypotheses (wor: worder r) (wor':worder r').
Definition olexp_g := cst_graph (substrate r') r.
Definition olexp_1E := the_least r.
Definition olexp' := olex (opp_order r') olexp_g.
Definition olexp_I x := Zo (substrate r') (fun i => (Vg x i <> olexp_1E)).
Definition olexp_G:= Zo (gfunctions (substrate r') (substrate r))
  (fun x => finite_set (olexp_I x)).
Definition olexp := induced_order olexp'  olexp_G.

Lemma olexp_ax: olex_ax (opp_order r') olexp_g.  (* 6 *)
Lemma olexp'_osr:  (* 4 *)
```

```
  order_on olexp' (gfunctions (substrate r') (substrate r)).
Lemma olexp_gleh x y:  (* 5 *)
 (induced_order (opp_order r') (olex_nsv r' x y)) =
  olex_io (opp_order r')  x y.

Lemma olexp'_gleP x y: (* 19 *)
  gle olexp' x y <->
  [/\ inc x (gfunctions (substrate r') (substrate r)),
     inc y (gfunctions (substrate r') (substrate r)) &
     (x = y \/
     let T := olex_nsv r' x y in
     let r'' := induced_order (opp_order r') T in
       worder r'' /\
       let i := the_least r'' in glt r (Vg x i) (Vg y i))].
```

We show here some trivial properties. If x and y are in G then T_{xy} is finite, hence well-ordered. We can then simplify the order relation on G. In fact $x < y$ means that there is j such that $x_j < y_j$, and $x_i = y_i$ whenever $j < i$. The set G is ordered by `olexp`. This is a total order (see `olex_cc_comparable`).

Assume F empty. Then G has a single element, the empty graph. Assume F non-empty; if E is empty, then G is empty. Assume E non-empty, so that E has a least element m, and $\bar m$ is the least element of G. Let's show that G is the connected component of $\bar m$. Note that $g \in$ G is comparable with m since $m \le g$. On the other hand, consider an element in the connected component, and a chain x_i. We prove, by induction on the length of the chain, that each x_i is in G. All we need to show is that, if $x \in$ G, x and y are comparable, then $y \in$ G. This holds because T_{xy} is well-ordered for the restriction of the opposite order of a well-order, thus is finite.

```
Lemma olexp_Gs: sub olexp_G (substrate olexp'). (* 1 *)
Lemma olexp_lEp: nonempty (substrate r) ->  (* 3 *)
  (inc olexp_lE (substrate r)
  /\ (forall x, inc x (substrate r) -> gle r olexp_lE x)).

Lemma olexp_Gxy x y: inc x olexp_G -> inc y olexp_G -> (* 8 *)
  finite_set (olex_nsv (opp_order r') x y).
Lemma olexp_Gxy1 x y: inc x olexp_G -> inc y olexp_G -> (* 8 *)
  worder (olex_io (opp_order r') x y).
Lemma olexp_gle1P x y: inc x olexp_G -> inc y olexp_G -> (* 44 *)
  (gle olexp' x y <->
   ( x = y \/
     (exists j,
        [/\ inc j (substrate r'),
            glt r (Vg x j) (Vg y j) &
            forall i, glt r' j i -> Vg x i = Vg y i]))).

Lemma olexp_osr: order_on olexp  olexp_G.  (* 1 *)
Lemma olexp_gleP x y: (* 3 *)
 gle olexp x y <-> [/\ inc x olexp_G, inc y olexp_G &
     ( x = y \/
     (exists j,
        [/\ inc j (substrate r'),
            glt r (Vg x j) (Vg y j) &
            forall i, glt r' j i -> Vg x i = Vg y i]))].
Lemma olexp_total: total_order olexp. (* 17 *)
```

```
Lemma olex_Fe: (substrate r' = emptyset) ->  singletonp olexp_G. (* 7 *)
Lemma olex_nFe_Ee: (* 2 *)
  (substrate r' <> emptyset) -> (substrate r = emptyset) ->
   olexp_G = emptyset.
Lemma olexp_G_least  (m:= cst_graph (substrate r') olexp_1E):  (* 18 *)
  nonempty (substrate r) -> least olexp m.
Lemma olex_G_cc (* 69 *)
  (m:= cst_graph (substrate r') olexp_1E)
  (comp:= ocomparable olexp')
  (G := (connected_comp comp (substrate olexp') m)) :
  nonempty (substrate r) ->
  olexp_G = G.
```

Let's show that G is well-ordered. Consider a non-empty subset of X of G. We proceed by contradiction, assuming X has no least element. Let Y be any subset of X such that (a) $Y \subset X$, (b) Y is non-empty and (c) if $x \in Y$ and $y \in X - Y$ then $x < y$. In particular, Y has no least element. Consider a subset A of F such that (d) whenever x and y are in Y and $i \in A$, then $x_i = y_i$ and (e) that if $i \in I_x$ then either $i \in A$ or $i < j$ for all $j \in A$. We define J_x to be the set $I_x - A$. This set cannot be empty, for otherwise x would be the least element of Y. Since J_x is finite, it has a greatest element M_x. Let α be the least element of the form M_x for $x \in Y$. Let B be the set of all $x \in Y$ for which M_x is α, and C the set of all x_α for $x \in B$. This is a non-empty set, thus has a least element, β. Let Y' be the set of all elements $x \in B$ such that $x_\alpha = \beta$, and $A' = A \cup \{\alpha\}$. If $x \in Y'$ and $y \in Y - Y'$ then $x < y$. It follows that Y' and A' satisfy conditions (a) to (e). We construct by induction a sequence (Y_k, A_k) starting with $Y_0 = X$ and $A_0 = \emptyset$. This gives us a sequence α_k. We have $\alpha_k \in A_{k+1}$, and the relation "$\alpha < i$ for all $i \in A$" implies $\alpha_k < \alpha_n$ for $n < k$. Thus, the sequence α_k is strictly decreasing, in a well-ordered set, absurd.

```
Lemma olexp_worder: worder olexp. ( * 251 *)
End OlexPowBasic.
```

By abuse of notations, this ordering will be denoted E^F. if we replace E and F by isomorphic set, we get isomorphic powers.

```
Lemma image_of_inf r r' f: worder r -> worder r' ->  (* 13 *)
  order_isomorphism f r r' -> nonempty (substrate r) ->
  Vf f (the_least r) = the_least r'.
Lemma fct_co_simpl_right f1 f2 g: (* 7 *)
  f1 \coP g -> f2 \coP g -> bijection g -> f1 \co g = f2 \co g -> f1 = f2.
Lemma fct_co_simpl_left f1 f2 g: (* 7 *)
  g \coP f1 -> g \coP f2 -> bijection g ->g \co f1 = g \co f2 -> f1 = f2.

Lemma opowa_invariant r1 r2 r3 r4:  (* 202 *)
   worder r1 -> worder r2 -> worder r3 -> worder r4 ->
   r1 \Is r2 -> r3 \Is r4 ->
   (olexp r1 r3) \Is (olexp r2 r4).
```

If a and b are two ordinals, $E = o(a)$ and $F = o(b)$, then we may consider the ordinal $\mathrm{ord}(E^F)$. We shall denote it a^b by abuse of notations. We show here that this coincides with the ordinal power for $a = 0$, $a = 1$, $b = 0$ and $b = 1$.

```
Definition ord_powa x y := ordinal (olexp (ordinal_o x) (ordinal_o y)).
Notation "x ^O y" := (ord_powa x y) (at level 30).
```

```
Lemma OS_ord_powa a b: ordinalp a -> ordinalp b -> (* 1 *)
  ordinalp (a ^O b).

Lemma ord_powa_pr0 r r': worder r -> worder r' -> (* 11 *)
   ordinal (olexp r r') = (ordinal r) ^O (ordinal r').
Lemma ord_powa_pr1 x : ordinalp x -> x ^O \0o = x ^o \0o. (* 5 *)
Lemma ord_powa_pr2 y :
  ordinalp y -> y <> \0o -> \0o ^O y = \0o ^o y. (* 5 *)
Lemma ord_powa_pr3 y : ordinalp y -> \0o ^O y = \0o ^o y. (* 3 *)
Lemma ord_powa_pr4 y : ordinalp y -> \1o ^O y = \1o ^o y.  (* 18 *)
Lemma ord_powa_pr5 x : ordinalp x -> x ^O \1c = x ^o \1c. (* 42 *)
```

Let's show an auxiliary result: assume that a and b are two ordinals such that $a < b$, and let $f : a \to b$ be an order isomorphism onto some segment of b. Then f is the canonical inclusion. Proof: we know that there a unique morphism whose range is a segment, either in the direction $a \to b$, or in the direction $b \to a$. The assumption $a < b$ excludes one case; Since $a \subset b$, the canonical inclusion is a solution, by uniqueness, it has to be f.

```
Lemma inclusion_morphism_a a b (f:= Lf id a b) : a <=o b -> (* 11 *)
   (segmentp (ordinal_o b) (Imf f)
    /\  order_morphism f  (ordinal_o a) (ordinal_o b)).
Lemma inclusion_morphism_b a b f : a <o b -> (* 10 *)
  segmentp (ordinal_o b) (Imf f) ->
  order_morphism f (ordinal_o a) (ordinal_o b) ->
  (forall x, inc x a -> Vf f x = x).
```

We have $a^{b+c} = a^b \cdot a^c$. Using order invariance, we are reduced to show that, if A, B and C are well-ordered, $\mathrm{B}+\mathrm{C}$ is the disjoint union, if $f : \mathrm{B}+\mathrm{C} \to \mathrm{A}$ has restrictions $f_\mathrm{B} : \mathrm{B} \to \mathrm{A}$ and $f_\mathrm{C} : \mathrm{C} \to \mathrm{A}$, then $f \to (f_\mathrm{C}, f_\mathrm{B})$ is an order isomorphism. Note that f takes almost always the value inf(A) if and only if both f_B and f_C do. The proof is long but straightforward.

We deduce: if $y \le z$ and $(x \neq 0)$ then $x^y \le x^z$ (write $z = y+t$, and use $x^t \neq 0$). We also have $a^{b+1} = a^b \cdot a$.

We deduce that a^b is the the usual ordinal power. Proof. We may assume $a \ge 2$. Consider the least ordinal b for which this is false. By the previous considerations, this has to be a limit ordinal. By normality of the power function, and minimality, it suffices to show $a^b = \sup_{c<b}(a^c)$. Let s be the supremum; we know $s \le a^b$.

Assume by contradiction $s < a^b$. Let $\mathrm{G}(a,b)$ denote the set of functional graphs $b \to a$ that take almost everywhere the least element of a (i.e., zero). This is a well-ordered set, and a^b is its ordinal. Let α be the order isomorphism that maps $\mathrm{G}(a,b)$ to a^b. Since $s \in a^b$, there is some x such that $s = \alpha(x)$. There is c_1 such that $t > c_1$ implies $x(t) = 0$ (if x is zero, take $c_1 = 0$, otherwise, take for c_1 the greatest t such that $c(t)$ is non-zero). Since b is limit $c_1+1 < b$. Let $c = c_1+1$. We have thus: if $t \ge c$ then $x(t) = 0$. Let γ be the order isomorphism $a^c \to \mathrm{G}(a,c)$. Let $\beta : \mathrm{G}(a,c) \to \mathrm{G}(a,b)$ be the map that associates to each u its extension $\bar{u}$ (if $t \ge c$, then $\bar{u}(t) = 0$). This is an order morphism. Let $\mathrm{P}(u)$ be the condition "$u \in \mathrm{G}(a,b)$ and if $t \ge c$ then $u(t) = 0$". Then $u = \beta(u')$, where u' is the restriction of u to c and is in $\mathrm{G}(a,c)$. Note that $\mathrm{P}(x)$ holds. Note that $\mathrm{P}(u)$ holds if $u \le v$ and $\mathrm{P}(v)$ holds. This can be restated as: the image of β is a segment.

By composition of these three morphisms, we get an order morphism $a^c \to a^b$, and the image is a segment. Thus, this is the canonical inclusion. Since s is in the image, we get $s < a^c$. This contradicts the definition of the supremum.

```
(*
Lemma inclusion_morphism_a a b (f:= Lf id a b) : a <=o b -> (* 14 *)
   (segmentp (ordinal_o b) (range (graph f))
    /\  order_morphism f  (ordinal_o a) (ordinal_o b)).
Lemma inclusion_morphism_b a b f : a <o b -> (* 10 *)
  segmentp (ordinal_o b) (range (graph f)) ->
  order_morphism f (ordinal_o a) (ordinal_o b) ->
  (forall x, inc x a -> Vf x f = x).
*)

Lemma power_of_suma a b c: (* 188 *)
   ordinalp a -> ordinalp b -> ordinalp c ->
  (a ^O (b +o c)) = (a ^O b) *o (a ^O c).

Lemma ord_powa_M_eqle a b c: (* 12 *)
   ordinalp a -> b <=o c -> a <> \0o -> a ^O b <=o a ^O c.
Lemma ord_powa_succ a b: ordinalp a -> ordinalp b -> (* 2*)
   a ^O (osucc b) = (a ^O b) *o a.
Lemma ord_powa_same a b: ordinalp a -> ordinalp b ->
   a ^O b = a ^o b. (* 199 *)
```

¶ 20. A set X is said to be *transitive* if the relation $x \in \mathrm{X}$ implies $x \subset \mathrm{X}$.

(a) If Y is a transitive set, then so is $\mathrm{Y} \cup \{\mathrm{Y}\}$. If $(\mathrm{Y}_\iota)_{\iota\in\mathrm{I}}$ is a family of transitive sets, then $\bigcup_{\iota\in\mathrm{I}} \mathrm{Y}_\iota$ and $\bigcap_{\iota\in\mathrm{I}} \mathrm{Y}_\iota$ are transitive.

(b) A set X is a *pseudo-ordinal* if every transitive set Y such that $\mathrm{Y} \subset \mathrm{X}$ and $\mathrm{Y} \neq \mathrm{X}$ is an element of X. A set S is said to be *decent* if the relation $x \in \mathrm{S}$ implies $x \notin x$. Show that every pseudo-ordinal is transitive and decent (consider the union of decent transitive subsets of X and use (a)). If X is a pseudo-ordinal, so is $\mathrm{X} \cup \{\mathrm{X}\}$.

(c) Let X be a transitive set and suppose that each $x \in \mathrm{X}$ is a pseudo-ordinal. Then X is a pseudo-ordinal (note that, for each $x \in \mathrm{X}$, $x \cup \{x\}$ is a pseudo-ordinal contained in X).

(d) Show that $\emptyset$ is a pseudo-ordinal and that every element of a pseudo-ordinal X is a pseudo-ordinal (Consider the union of the transitive subsets of X whose elements are pseudo-ordinals).

(e) If $(\mathrm{X}_\iota)_{\iota\in\mathrm{I}}$ is a family of pseudo-ordinals then $\bigcap_{\iota\in\mathrm{I}} \mathrm{X}_\iota$ is the least element of this family (with respect to the relation of inclusion). (Use (b).) Deduce that, if E is a pseudo-ordinal, the relation $x \subset y$ between elements x, y of E is a well-ordering relation.

(f) Show that for each ordinal α there exists a unique pseudo-ordinal E_α such that $\mathrm{Ord}(\mathrm{E}_\alpha) = \alpha$ (use (e) and Criterion C60). In particular the pseudo-ordinals whose order-type are 0, 1, $2 = 1+1$, and $3 = 2+1$ are respectively

$$\emptyset, \quad \{\emptyset\}, \quad \{\emptyset, \{\emptyset\}\}, \quad \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$$

Note. The French version of the exercise has one more item. It says: if X and Y are two pseudo-ordinals, then either $\mathrm{X} \in \mathrm{Y}$ or $\mathrm{Y} \in \mathrm{X}$ or $\mathrm{X} = \mathrm{Y}$. The hint for item (c) is: “note that if $\mathrm{Y} \neq \mathrm{X}$ is transitive, then $\mathrm{Y} \subset \mathrm{X}$”.

This exercise is implemented in section 4.

13.4 Section 3

¶ 1. Let E and F be two sets, let f be an injection of E into F, and let g be a mapping of F into E. Show that there exist two subsets A, B of E such that $B = E - A$ and two subsets A', B' of F such that $B' = F' - A'$ for which $A' = f(A)$ and $B = g(B')$. (Let $R = E - g(F)$) and put $h = g \circ f$; take A to be the intersection of the subsets M of E such that $M \supset R \cup h(M)$.)

Note. The 1956 Edition of Bourbaki has: f and g injective. In this case, f induces a bijection $A \to A'$ and g induces a bijection $B' \to B$, thus E and F are equipotent. This is the Cantor-Bernstein theorem. The result is true if one of f or g is injective (by symmetry, one case implies the other). The hint works only if g is injective.

```
Lemma Exercise3_1 E F f g: (* 63 *)
  function_prop f E F -> function_prop g F E -> injection f ->
  exists A A',
    [/\ sub A E, sub A' F, Vfs f A = A' &  Vfs g (F -s A') = E -s A].

Lemma Exercise3_1b E F f g: (* 4 *)
  function_prop f E F -> function_prop g F E -> injection g ->
  exists A A',
    [/\ sub A E, sub A' F, Vfs f A = A' & Vfs g (F -s A') = E -s A].
```

---

2. If E and F are two distinct sets, show that $E^F \neq F^E$. Deduce that if E and F are the cardinals 2 and 4 $(= 2+2)$, then at least one of the sets E^F, F^E is not a cardinal.

Note. Remember that $C = F^E$ (respectively $D = E^F$) is the set of functional graphs $E \to F$ (respectively $F \to E$). If F is non-empty, the set C is non-empty, and its elements are graphs with domain E. Thus, if both sets E and F are non-empty, $C = D$ implies $E = F$. The same is true if exactly one of E, F is empty (since exactly one of C, D is empty), and if $E = F = \emptyset$.

The lemma `power_2_4` says that, if $E = 2$ and $F = 4$, the two sets $\mathscr{F}(E;F)$ and $\mathscr{F}(F;E)$ are equipotent. In particular, C and D are equipotent, and have the same cardinal: $\text{card}(C) = \text{card}(D)$. This implies that one of "$C = \text{card}(C)$" or "$D = \text{card}(D)$" must be false.

In our implementation a cardinal is a von Neumann ordinal. Assume that F^E is an ordinal. If this ordinal is non-zero, it contains the empty set as least element. Since the domain of the empty set is empty, we get $E = \emptyset$, case where $F^E = \{\emptyset\}$ is an ordinal. Otherwise F is non-empty. Thus, in our implementation, E^F is an ordinal (thus a cardinal) if and only if it is zero or one.

```
Lemma Exercise3_2a: forall E F, (* 12 *)
  gfunctions E F = gfunctions F E -> E = F.
Lemma Exercise3_2b: (* 8 *)
  let E := \2c in let F := card_four in
  let s1 := gfunctions E F in let s2 :=gfunctions F E in
  ~ (cardinalp s1) \/ ~ (cardinalp s2).
Lemma Exercise3_2c: forall E F, (* 12 *)
  ordinalp (gfunctions E F) -> E = emptyset \/ F = emptyset.
```

¶ 3. Let $(\mathfrak{a}_\iota)_{\iota\in I}$ and $(\mathfrak{b}_\iota)_{\iota\in I}$ be two families of cardinals such that $\mathfrak{b}_\iota \geq 2$ for each $\iota \in I$.

(a) Show that, if $\mathfrak{a}_\iota \leq \mathfrak{b}_\iota$ for each $\iota \in I$, then

$$\sum_{\iota\in I} \mathfrak{a}_\iota \leq \prod_{\iota\in I} \mathfrak{b}_\iota.$$

(b) Show that, if $\mathfrak{a}_\iota < \mathfrak{b}_\iota$ for each $\iota \in I$, then

$$\sum_{\iota\in I} \mathfrak{a}_\iota < \prod_{\iota\in I} \mathfrak{b}_\iota.$$

(Note that a product $\prod_{\iota\in I} E_\iota$ cannot be the union of a family $(A_\iota)_{\iota\in I}$ such that $\mathrm{Card}(A_\iota) < \mathrm{Card}(E_\iota)$ for all $\iota \in I$, by observing that $\mathrm{Card}(\mathrm{pr}_\iota(A_\iota)) < \mathrm{Card}(E_\iota)$.).

Note. This is known as the Koenig theorem and implemented as `compare_sum_prod2` and `compare_sum_prod3` in section 11.20.

Assume that E_i is a family of cardinals, with $E_i \geq 2$. This means that E_i has at least two distinct elements, say a_i and b_i. We pretend that $\sum E_i \leq \prod E_i$ (this shows (a)). This is obvious if I is empty or reduced to a singleton. Assume that I has two elements. We consider the following function that maps the disjoint union of E_1 and E_2 to the product: elements a_1, a_2, b_1, b_2 are mapped to (a_1,a_2), (a_1,b_2), (b_1,a_2), and (b_1,b_2). Let's maps remaining elements x of E_1 to (x,a_2) and remaining elements y of E_2 to (a_1,y). This function is an injection.

(**Note**; the proof can be simplified by taking $a_i = 0$ and $b_i = 1$. If I has two elements, we must show $a+b \leq ab$. Write $a = c+2$, this gives $2+b+c \leq b+b+bc$ which holds when $2 \leq b$).

Assume now that I has at least three elements. To $x \in E_j$ we associate the sequence $(x_i)_i$ as follows. We have $x_j = x$, and if $i \neq j$ we take $x_i = b_i$ (if $x = a_j$) and $x_i = b_i$ (otherwise). This function is injective: assume $(x_i)_i = (y_i)_i$. If $y \in E_j$, we have $x = y$ from $x_j = y_j$. Assume $y \in E_k$ with $k \neq j$. We have either $x = a_j$ and $y = b_k$ or $x \neq a_j$ and $y \neq b_k$ (consider otherwise an index distinct from i and j), and these two cases are impossible.

Assume that a_i is a family of cardinals, $a_i < b_i$. Let J be the set of indices such that $b_i \geq 2$. Note that $j \notin J$ implies $b_i = 1$ and $a_i = 0$. Thus $\sum_I a_i = \sum_J b_j$ and $\prod_I a_i = \prod_J b_j$. Thus we may assume $J = I$, and apply (a). We pretend $\sum_J a_j \neq \prod_J b_j$, proof by contradiction. For otherwise, there would exist a family of sets A_i such that $\mathrm{Card}(A_i) < \mathrm{Card}(E_i)$, and a bijection that maps the disjoint union of A_i onto the product. Let B_i be the image of A_i, so that $\mathrm{Card}(A_i) < \mathrm{Card}(E_i)$, and $\bigcup B_i = \prod E_i$. Let $C_i = \mathrm{pr}_i\langle B_i\rangle$ (the set of i-th components of elements of B_i, this is a subset of E_i). We have $\mathrm{Card}(C_i) \leq \mathrm{Card}(A_i)$. This implies $C_i \neq E_i$. In particular, there is $x_i \in E_i$, not in C_i. The family $(x_i)_i$ is in the product, thus in some B_i; its i-th component is in C_i, absurd.

---

4. Let E be a set and let f be a mapping of $\mathfrak{P}(E) - \emptyset$ into E such that for each non-empty subset X of E we have $f(X) \in X$ ("Choice function").

(a) Let $\mathfrak{b}$ be a cardinal and let A be the set of all $x \in E$ such that $\mathrm{Card}(\overset{-1}{f}(x)) \leq \mathfrak{b}$. Show that if $\mathfrak{a} = \mathrm{Card}(A)$, then $2^{\mathfrak{a}} \leq 1 + \mathfrak{a}\mathfrak{b}$ (note that if $Y \subset A$ and $Y \neq \emptyset$ then $f(Y) \in A$).

(b) Let B be the set of all $x \in E$ such that, for each non-empty subset X of $\overset{-1}{f}(x)$, $\mathrm{Card}(X) \leq \mathfrak{b}$. Show that $\mathrm{Card}(B) \leq \mathfrak{b}$.

Solution. Let A_x be the set of all non-empty subsets Y of A such that $f(Y) = x$. This is a disjoint family of subsets of $A' := \mathfrak{P}(A) - \{\emptyset\}$. Assume $Y \in A'$. Then $f(Y) \in Y \subset A$, so that

$Y \in A_{f(Y)}$. We have hence a partition of A'. If $Y \in A_x$ then $Y \in f^{-1}(x)$ so $A_x \subset f^{-1}(x)$. It follows that A_x has cardinal $\le \mathfrak{b}$. This shows (a).

(b) The French version defines B as the set of all $x \in E$ such that for every subset $X \neq \emptyset$ that belongs to $f^{-1}(x)$, one has $\mathrm{Card}(X) \le \mathfrak{b}$. The English text considers the set of all x such that $\mathrm{Card}(f^{-1}(x)) \le \mathfrak{b}$, which is quite different.

The result is obvious when B is empty. Otherwise let $x = f(B)$ so that $x \in B$. We have $B \in f^{-1}(x)$, so the result follows by definition of B (with $X = B$).

```
Section Exercise34.
Variable E: Set.
Variable f: Set -> Set.
Hypothesis choice: forall X, sub X E -> nonempty X -> inc (f X) X.
Definition finv x := Zo (\Po E) (fun z => f z = x).

Lemma Exercise34a b: cardinalp b -> (* 24 *)
  let A := Zo E (fun x => (cardinal (finv x)) <=c b) in
  let a := cardinal A in
  (\2c ^c a) <=c (csucc (a *c b)).

Lemma Exercise34b b: cardinalp b -> (* 7 *)
  let B := Zo E (fun x => forall X, inc X (finv x) -> nonempty X ->
        cardinal X <=c b) in
  cardinal B <=c b.
End Exercise34.
```

---

5. Let $(\lambda_\iota)_{\iota\in I}$ be a family of order-types (§2, Exercise 13), indexed by an ordered set I. Show that

$$\mathrm{Card}\Big(\sum_{\iota\in I}\lambda_i\Big) = \sum_{\iota\in I}\mathrm{Card}(\lambda_i)$$

and that, if I is well-ordered,

$$\mathrm{Card}\Big(\mathsf{P}_{\iota\in I}\lambda_i\Big) = \mathsf{P}_{\iota\in I}\mathrm{Card}(\lambda_i).$$

Solution. According to Exercise 2.13, an order type λ_i is some particular ordered set $(E_i, \le_i)$. The quantity $\mathrm{Card}(\lambda_i)$ is the cardinal of E_i. The quantity $\sum\lambda_i$ is some ordering on a set E order-isomorphic to the ordinal sum $S = \sum E_i$. We must show that Card(E) is the sum $s = \sum \mathrm{Card}(E_i)$. We show here that $\mathrm{Card}(\sum E_i) = t$. Since E is order-isomorphic to S, it has the same cardinal. We then show that, if I is well-ordered and if each λ_i is an ordinal, so that $\sum\lambda_i$ is an ordinal, then $\mathrm{Card}(\sum\lambda_i) = s$.

```
Definition fam_card_sub f :=
  (Lg (domain f) (fun z => cardinal (substrate (Vg f z)))).

Lemma Exercise3_5a r f: orsum_ax r f -> (* 3 *)
  cardinal (substrate (order_sum r f)) = csum (fam_card_sub f).
Lemma Exercise3_5b r f: orprod_ax r f -> (* 3 *)
  cardinal (substrate (order_prod r f)) = cprod (fam_card_sub f).
Lemma Exercise3_5c r f h: orsum_ax r f -> (* 2 *)
  h \Is (order_sum r f) ->
  cardinal (substrate h) =  csum (fam_card_sub f).
```

```
Lemma Exercise3_5d: r f h: ordprod_ax  r f -> (* 2 *)
  h \Is (order_prod r f) ->
  cardinal (substrate h) =  cprod (fam_card_sub f).

Lemma Exercise3_5e r g:  (* 15 *)
  worder_on r (domain g) -> ordinal_fam g ->
  cardinal (osum r g) =
    csumb (domain g) (fun z => cardinal (Vg g z)).
Lemma Exercise3_5f r g:  (* 15 *)
  worder_on r (domain g) -> ordinal_fam g ->
  finite_set (substrate r) ->
  cardinal (oprod r g) =
    cprodb (domain g) (fun z => cardinal (Vg g z)).
```

---

6. Show that for every set E there exists $X \subset E$ such that $X \notin E$ (use Theorem 2 of no. 6).

Discussion. If this property were false, we would have: for all X, $X \subset E \implies X \in E$, this is $\mathfrak{P}(E) \subset E$, and implies $\mathrm{Card}(\mathfrak{P}(E)) \leq \mathrm{Card}(E)$. This contradicts Cantor. Note that we can choose for X the set of all $t \in E$ such that $t \notin t$, so that the result is trivial.

```
Lemma Exercise3_6 E: exists X, sub X E /\ ~ (inc X E). (* 7 *)
Lemma Exercise3_6b E:  (* 4 *)
  let X:=  Zo E (fun t => ~(inc t t)) in sub X E /\ ~ (inc X E).
```

13.5 Section 4

1. (a) Let E be a set and let $\mathfrak{F}(E)$ be the set of finite subsets of E. Show that $\mathfrak{F}(E)$ is the smallest subset $\mathfrak{G}$ of $\mathfrak{P}(E)$ satisfying the following conditions: (i) $\emptyset \in \mathfrak{G}$; (ii) the relation $X \in \mathfrak{G}$ and $x \in E$ imply $X \cup \{x\} \in \mathfrak{G}$.

(b) Deduce from (a) that the union of two finite subsets A and B is finite (consider the set of subsets X of E such that $X \cup A$ is finite; cf § 5; no 1 Proposition 1, Corollary 1).

(c) Deduce from (a) and (b) that for every finite set E the set $\mathfrak{P}(E)$ is finite (consider the set of subsets X of E such that $\mathfrak{P}(X)$ is finite; cf § 5; no 1 Proposition 1, Corollary 4).

Note. The two corollaries say that the union of a finite family of finite sets is finite, and that the power set of a finite set is finite.

Solution. If we denote by $\mathfrak{F}(E)$ the set of finite subsets of E, and by $P(E, \mathfrak{G})$ the property: (i) $\emptyset \in \mathfrak{G}$; (ii) the relation $X \in \mathfrak{G}$ and $x \in E$ imply $X \cup \{x\} \in \mathfrak{G}$, then $P(E, \mathfrak{F})$ is true; by induction, $P(E, \mathfrak{G})$ implies $\mathfrak{F} \subset \mathfrak{G}$. As a consequence, the union of two finite sets is finite. The power set of a finite set is finite (proof by induction: $\mathfrak{P}(X \cup \{x\})$ is the union of $\mathfrak{P}(X)$ and the image of $\mathfrak{P}(X)$ by the mapping $Y \mapsto (Y \cup \{x\})$; if $x \notin X$, the union is disjoint, and the cardinal is twice the cardinal of $\mathfrak{P}(X)$).

```
Definition finite_subsets E := Zo(\Po E) finite_set.

Definition finite_subsets_prop E F:=
  inc emptyset  F /\ forall x X, inc x E -> inc X F -> inc (X +s1 x) F.
```

```
Lemma finite_subsets_pr E: (* 12 *)
  finite_subsets_prop E (finite_subsets E) /\
  (forall F, finite_subsets_prop E F -> sub(finite_subsets E) F).
Lemma finite_union2 x y: (* 18 *)
  finite_set x -> finite_set y -> finite_set (x \cup y).
Lemma finite_powerset x:  (* 50 *)
  finite_set x -> finite_set (\Po x).
```

---

2. Show that a set E is finite if and only if every non-empty subset of $\mathfrak{P}(E)$ has a maximal element (with respect to inclusion). (To show that the condition is sufficient, apply it to the set $\mathfrak{F}(E)$ of finite subsets of E).

Solution. Notice that if F is a finite subset of an infinite set E, there exists another finite subset F' of E, such that F is a strict subset of F'. Conversely, assume E finite, and proceed by induction. Say $E = F \cup \{a\}$, and let Y be a subset. If no element of Y contains a, the result is obvious by induction. Otherwise, let Z be the set of elements of Y that do contain a. By induction, Z has a maximal element.

```
Lemma finite_is_maximal_inclusion x: (* 45 *)
  finite_set x <->
  (forall y, sub y (\Po x) -> nonempty y -> exists2 z,
     inc z y & forall t, inc t y -> sub z t -> z = t).
```

---

3. Show that if a well-ordered set E is such that the ordered set obtained by endowing E with the opposite ordering is also well-ordered, then E is finite (consider the greatest element x of E such that the segment S_x is finite).

Solution. Let S be the subset of E formed of all x such that $]\leftarrow, x[$ is finite. If E is empty, there is nothing to do; otherwise E has a least element, which is in S so that S has a greatest element y. Let $F =]\leftarrow, y[\cup \{y\}$. This is a finite segment. Since it cannot be of the form $]\leftarrow, x[$, it has to be E itself. Thus E is finite.

```
Lemma worder_has_empty_seg r: worder r -> (* 7 *)
  nonempty (substrate r) -> exists x,
    (inc x (substrate r) /\ segment r x = emptyset).
Lemma well_ordered_opposite r: (* 20 *)
  worder r -> worder (opp_order r) -> finite_set (substrate r).
```

---

4. Let E be a finite set with $n \geq 2$ elements, and let C be a subset of $E \times E$ such that, for each pair x, y of distinct elements of E, exactly one of the two elements (x, y), (y, x) of $E \times E$ belongs to C. Show that there is a mapping f of the interval $[1, n]$ onto E such that $(f(i), f(i+1)) \in C$ for $1 \leq i \leq n-1$ (use induction on n).

Discussion. We consider the assumption H(C,E) that says that for each pair x, y of distinct elements of E, exactly one of the two pairs (x, y), (y, x) belongs to C. The Exercise assumes further that $\mathrm{C} \subset \mathrm{E} \times \mathrm{E}$, but this is of no help. We consider the conclusion: there is a bijection $f : [1, \mathrm{Card(E)}] \to \mathrm{E}$ such that $(f(i), f(i+1)) \in \mathrm{E}$ (the Exercise requires f surjective, but $[1, \mathrm{Card(E)}]$ and E are two finite sets with the same cardinal, so bijectivity is equivalent to surjectivity). We show that, if the assumption implies the conclusion whenever E has cardinal $< n$, then the result holds for cardinal n as well, and conclude via `Nat_induction1`.

Solution. The result is obvious if E is empty. Otherwise, fix $a \in \mathrm{E}$, let E_1 be the set of all x such that $(x, a) \in \mathrm{C}$, and E_2 the set of all x such that $(a, x) \in \mathrm{C}$. If these sets have cardinal n_1 and n_2, we have $n_1 + n_2 + 1 = n$, and by induction we get two functions f_1 and f_2. Define f by $f_1(i)$ if $i \le n_1$, $f_2(i - n_1 - 1)$ if $i > n_1 + 1$, and $f(n_1 + 1) = a$. This is the desired function.

```
Lemma Exercise4_4 n E C: (* 144 * )
  natp n -> cardinal E = n -> sub C (coarse E) ->
  (forall x y, inc x E -> inc y E -> x <> y ->
    (exactly_one (inc (J x y) C) (inc (J y x) C))) ->
    exists2 f,  bijection_prop f (Nintcc \1c n) E &
      (forall i, \1c <=c i -> i <c n ->
        inc (J (Vf f i) (Vf f (csucc i))) C).
```

¶ 5. Let E be an ordered set for which there exists an integer k such that k is the greatest number of elements in a free subset X of E (§ 1, Exercise 5). Show that E can be partitioned into k totally ordered subsets (with respect to the induced ordering)[2]. The proof is in two steps:

(a) If E is finite and has n elements, use induction on n; let a be a minimal element of E and let $\mathrm{E}' = \mathrm{E} - \{a\}$. If there exists a partition of E′ into k totally ordered sets C_i $(1 \le i \le k)$, let U_i be the set of all $x \in \mathrm{C}_i$ which are $\ge a$. Show that there is at least one index i for which a free subset $\mathrm{E}' - \mathrm{U}_i$ has at most $k - 1$ elements. The proof of this is by *reduction ad absurdum*. For each i, let S_i be a free subset of $\mathrm{E}' - \mathrm{U}_i$ which has k elements, let S be the union of the sets S_i, and let s_j be the least element of $\mathrm{S} \cap \mathrm{C}_j$ for each index $j \le k$; show that the $k + 1$ elements a, $s_1, \dots, s_k$ form a free subset of E.

(b) If E is arbitrary, the proof is by induction on k, as follows. A subset C of E is said to be *strongly related* in E if for each finite subset F of E there exists a partition of F into at most k totally ordered sets such that $\mathrm{C} \cap \mathrm{F}$ is contained in one of them. Show that there exists a maximal strongly related subset C_0, and that every free subset of $\mathrm{E} - \mathrm{C}_0$ has at most $k - 1$ elements (Argue by contradiction, and suppose that there is a free subset $\{a_1, \dots, a_k\}$ of k elements in $\mathrm{E} - \mathrm{C}_0$. Consider each set $\mathrm{C}_0 \cup \{a_i\}$ $(1 \le i \le k)$, and express the fact that it is not strongly related, thus introducing a finite subset F_i of E for each index i. Then consider the union F of the sets F_i and use the fact that C_0 is strongly related to obtain a contradiction).

Preliminaries. If X is a set, we define $\max_{\mathrm{C}}(\mathrm{X})$ to be the largest integer k such that k is the number of elements of X. This integer exists provided that X is non-empty and there is an integer n such that each element of X has cardinal $\le n$ (in general, there is a least upper bound, but not always a greatest element).

```
Definition max_card_on_set X k :=
```

[2] This is called Dilworth's theorem in the French edition

```
  (exists2 x, inc x X & cardinal x = k) /\
  (forall x, inc x X -> cardinal x <=c k).
Definition the_max_card_on_set X := select (max_card_on_set X) Nat.

Lemma max_card_on_set_unique X k1 k2:
  max_card_on_set X k1 -> max_card_on_set X k2 ->
  k1 = k2.
Lemma max_card_on_set_exists X n: (* 12 *)
  natp n -> nonempty X ->
  (forall x, inc x X -> cardinal x <=c n) ->
  exists2 k, natp k & max_card_on_set X k.
Lemma the_max_card_on_set_prop X n (k := the_max_card_on_set X): (* 3 *)
  natp n -> nonempty X ->
  (forall x, inc x X -> cardinal x <=c n) ->
  max_card_on_set X k /\ natp k.
Lemma the_max_card_on_set_prop2 X n: natp n -> max_card_on_set X  n ->
   the_max_card_on_set X = n. (* 6 *)
```

Consider an order on a set E. We say that X is a *chain* if X is a subset of E totally ordered by the induced order. We say that X is an anti-chain if X is a free subset. We denote by $\mathscr{C}(E)$ and $\mathscr{A}(E)$ the set of chains and anti-chains. We define the *width* of the order as $\max_C(\mathscr{A})$ and the *length* of the order as $\max_C(\mathscr{A})$. Since the empty set belongs to $\mathscr{A}(E)$ and $\mathscr{C}(E)$, these quantities are defined when E is finite. The length is the largest integer k such that there is a sequence $x_1 < x_2 < \ldots < x_k$; the width is the largest integer k such that there is a there is a free subset with k elements. Note that a singleton belongs to both $\mathscr{A}$ and $\mathscr{C}$.

```
Definition total_suborder r x := total_order (induced_order r x).
Definition total_suborders r := Zo (\Po (substrate r)) (total_suborder r).

Lemma free_subsetsP r x: (* 2 *)
  inc x (free_subsets r) <-> (sub x (substrate r) /\ free_subset r x).
Lemma total_subordersP r x: (* 2 *)
  inc x (total_suborders r) <->
   (sub x (substrate r) /\ total_order (induced_order r x)).
Lemma sub_total_suborders1 r F: (* 3 *)
   order r -> sub F (substrate r) ->
   sub (total_suborders (induced_order r F)) (total_suborders r).
Lemma sub_total_suborders2 r X Y: order r -> (* 4 *)
    inc X (total_suborders r) -> sub Y X -> inc Y (total_suborders r).
Lemma total_suborder_prop r order r -> sub X (substrate r) -> (* 4 *)
   {inc X &, forall x y, ocomparable r x y} ->
   inc X  (total_suborders r).
Lemma sub_free_subsets r X Y: (* 3 *)
    inc X (free_subsets r) -> sub Y X -> inc Y (free_subsets r).
Lemma torder_set1 r x: order r -> inc x (substrate r) ->
  inc (singleton x) (total_suborders r).
Lemma empty_total_suborders r: inc emptyset (total_suborders r). (* 4 *)
Lemma nonempty_total_suborders r: nonempty (total_suborders r). (* 1 *)
Lemma nonempty_free_subsets r: nonempty (free_subsets r). (* 1 *)
```

We introduce the length and the width and consider some special cases. If E is empty, the width and the length are zero, the converse holds since singletons are free and chains. Assume E non-empty; The width is one if and only if the set is totally ordered; the length is one if and only if the set is trivial (i.e. the order is the diagonal of E). On the other hand, if E is finite and has n elements; then the length is n if and only if the set is totally ordered; the width is b if and only if the set is trivial.

```
Definition total_suborder r x := total_order (induced_order r x).
Definition total_suborders r := Zo (\Po (substrate r)) (total_suborder r).

Definition order_width  r := max_card_on_set (free_subsets r).
Definition order_length  r := max_card_on_set (total_suborders r).
Definition the_order_width r := the_max_card_on_set (free_subsets r).
Definition the_order_length  r := the_max_card_on_set (total_suborders r).

Lemma order_width_exists r (k := the_order_width r): (* 5 *)
  finite_set (substrate r) ->
  [/\ natp k, k <=c (cardinal (substrate r))  & order_width r k].
Lemma order_length_exists r (k := the_order_length r):  (* 6 *)
  finite_set (substrate r) ->
  [/\ natp k, k <=c (cardinal (substrate r))  & order_length r k].


Lemma cardinal_small_set x: small_set x -> cardinal x <=c \1c.
Lemma set0_width:
  the_order_width emptyset = \0c /\  the_order_length emptyset = \0c.
Lemma set0_width_1 r:  (* 2 *)
  order r -> order_length r \0c -> substrate r = emptyset.
Lemma set0_width_2 r: (* 2 *)
  order r -> order_width r \0c -> substrate r = emptyset.
Lemma total_order_width r: total_order r ->  (* 5 *)
  nonempty (substrate r)  -> the_order_width r = \1c.
Lemma total_order_width_contra r: order r ->
  order_width r \1c -> total_order r. (* 10 *)
Lemma diagonal_length E: nonempty E  -> (* 9 *)
  the_order_length (diagonal E) = \1c.
Lemma diagonal_length_contra r: order r -> (* 14 *)
  order_length r \1c -> r = diagonal (substrate r).
Lemma diagonal_width r: order r -> finite_set (substrate r) -> (* 16 *)
    (order_width r (cardinal (substrate r)) <-> r = diagonal (substrate r)).
Lemma total_order_length r: order r -> finite_set (substrate r) -> (* 8 *)
    (order_length r (cardinal (substrate r)) <-> total_order r).
```

We consider here the dual of theorem. Let E be an ordered set with a finite length k. There is a partition of E into k free subsets. Proof. The assumption is that there is an integer n such that every chain is of cardinal $\le n$. We deduce the existence of k and of a chain C with k elements. Let's define a chain with endpoint x as a totally ordered subset with greatest element x. Let $f(x)$ be maximum length of such a chain. We have obviously $1 \le f(x) \le k$. In the case where E is empty, the result holds. Otherwise $1 \le k$, C is non-empty and the greatest element x of C has $f(x) = k$. Note that f is strictly increasing: assume $x < y$, and consider a maximal chain with endpoint x, add y to the chain. Let A_i be the set of all x such that $f(x) = i$. This is the desired partition. Obviously, the sets are mutually disjoint, and E is the union of the A_i for $1 \le i \le k$. Note that A_i is free (if $x < y$ in A_i then $i = f(x) < f(y) = i$, absurd). It suffices to show that A_i is non-empty; proof by decreasing induction on i. We know that A_k is non-empty. Assume $f(y) = i+1$. There is a chain C of length $i+1$ with end-point y. Let $\mathrm{C}' = \mathrm{C} - \{y\}$. This is a totally ordered subset of E with i elements. We assume $1 \le i$ so that C' is non-empty, so has a greatest element x. Thus says $f(x) \ge i$. Obviously $x < y$ so that $f(x) < f(y) = i+1$. These two inequalities give $f(x) = i$, so that $x \in \mathrm{A}_i$.

Note. Let m be the width of E. so that each A_i has at most m elements. Since the k sets A_i form a partition of E, it follows that E has at most km elements (in order to get this result, we do not need to show that A_i, is non-empty). Variant (sometimes called Dilworth's lemma) is the following. Assume E finite of cardinal $nm+1$. Then either the width is $> n$ or the length is $> m$. So we can either find a big free set or a big chain. From this, we can extract a smaller one (whatever the size); hence either there is a free subset with $n+1$ or there is a chain with $m+1$ elements.

As a consequence (Erdös Szerkeres 1935), if E is totally ordered, every sequence of length $nm+1$ has either a strictly increasing subsequence of length $n+1$ or a strictly decreasing subsequence of length $m+1$. [let x_i be a sequence where the index set is I_p with $p = 1+nm$. Let $i \prec j$ if $i \le j$ and $x_i \le x_j$. This is an order on I_p. Let K be a totally ordered subset, i and j in K, $i < j$. These elements are comparable for $\prec$, and $j \prec i$ is absurd, hence $x_i \le x_j$. If we assume the elements distinct, we get $x_i < x_j$. Assume K free, $i < j$. Since $i \prec j$ is false, $x_i \le x_j$ is false; since E is totally ordered, $x_i > x_j$ holds.

```
Lemma Dilworth_dual r (k := the_order_length r): order r -> (* 117 *)
  (exists2 n, natp n &
     forall x, inc x (total_suborders r) -> cardinal x <=c n) ->
  exists A,
  [/\ natp k, order_length r k
    & [/\ partition_s A (substrate r),
    cardinal A = k & sub A (free_subsets r)]].

Lemma Dilworth_lemma_v1 r : order r -> finite_set (substrate r) -> (* 10 *)
  cardinal (substrate r) <=c (the_order_length r) *c (the_order_width r).
Lemma Dilworth_lemma_v2 r n m:
  order r -> natp n -> natp m -> cardinal (substrate r) = csucc (n *c m) ->
  (exists2 B, inc B (free_subsets r) & cardinal B = csucc m) \/
  exists2 A, inc A (total_suborders r) & cardinal A = csucc n.

Lemma Erdos_Szerkeres r n m f (D := csucc (n *c m)) : (* 38 *)
  total_order r -> natp n -> natp m -> fgraph f ->
  domain f = D -> sub (range  f) (substrate r) ->
  {inc D &, injective (Vg f)} ->
  (exists B, [/\ sub B D, cardinal B = csucc m &
     forall i j, inc i B -> inc j B -> i <c j -> glt r (Vg f j) (Vg f i)])
  \/ exists A, [/\ sub A D,  cardinal A = csucc n &
    forall i j, inc i A -> inc j A -> i <c j -> glt r (Vg f i) (Vg f j)].
```

Example. We shall prove the theorem of Sperner that considers the width of the inclusion order. We start with some technical lemmas. In what follows n will be an integer, $B(k) = \binom{n}{k}$ and $c_n = B(n/2)$. We know that the function B has a maximum at $n/2$. We deduce, for $k \le n$, that $n! \le c_n k!(n-k)!$. In case of equality, we have $B(k) = B(n/2)$, hence $k = n/2$ or $n-k = n/2$ (if $n = 2q$ is even, this is $k = q$; if $n = 2q+1$ is odd this is $k = q$ or $k = q+1$).

```
Definition is_half_n n k := (k = (chalf n) \/ n -c k = (chalf n)).

Lemma Sperner_b1 n k: natp n -> k <=c n -> (* 4* )
   (factorial n) <=c
   (binom n (chalf n)) *c ((factorial k) *c (factorial (n -c k))).
Lemma Sperner_b2 n k: natp n -> k <=c n -> (* 7 *)
   (factorial n) =
     (binom n (chalf n)) *c ((factorial k) *c (factorial (n -c k))) ->
   is_half_n n k.
```

We consider a set E and we are interested in $\mathfrak{P}(E)$ ordered by inclusion. Let F_k be the set of subsets of E with k elements. if A and B are in F_k, if k is finite, then $A \subset B$ implies $A = B$. So F_k is free.

```
Lemma Sperner_1 A B k: natp k -> cardinal A = k -> cardinal B = k -> (* 8 *)
  sub A B -> A = B.
Lemma Sperner_2 E k (F := subsets_with_p_elements k E): natp k -> (* 2 *)
  free_subset (subp_order E) F.
```

In what follows, we shall assume E finite with n elements. Since F_k has cardinal $B_(k)$ we deduce that the width of the order is $\geq c_n$. Sperner (1928) proved that we have actually equality. Moreover, if A is maximal and free, it has the form F_k where $k = n/2$ (in the case n odd k can be $(n+1)/2$).

```
Section Sperner.
Variable E: Set.
Hypothesis fsE: finite_set E.
Let r := subp_order E.
Let n := cardinal E.
Let cn := binom n (n %/c \2c).

Lemma perner_p0: natp n. (* 1 *)
Lemma Sperner_p1 k: k <=c n -> (* 2 *)
  free_subset r (subsets_with_p_elements k E).
Lemma Sperner_p2 k: k <=c n -> (* 2 *)
  cardinal (subsets_with_p_elements k E) = binom n k.
Lemma Sperner_p3 k: k <=c n -> binom n k <=c cn.  (* 1 *)
Lemma Sperner_p4: (* 2 *)
  inc A (free_subsets  r) <-> sub A (\Po E) /\ free_subset r A.
Lemma Sperner_p5 k: k <=c n -> (* 2 *)
  inc (subsets_with_p_elements k E) (free_subsets r).
```

Let C a chain; by the property given above, the cardinal function is injective on C. Since the cardinal is $\leq n$, there are at most $n+1$ elements in C. Denote by $\mathscr{M}$ the set of chains of length $n+1$. Let g be a bijection $I_n \to E$. Denote by $C(g)$ the set of all $g\langle i\rangle$ for $i \leq n$. (Recall that $g\langle i\rangle$ is the set of all $g(j)$ for $j \in i$ (i.e., $j < i$). This is a chain of length $n+1$. So $C(g) \in \mathscr{M}$ and the length of the order is $n+1$.

```
Definition Sperner_mx_chain :=
  (Zo (total_suborders r) (fun C =>  cardinal C = csucc n)).
Definition chain_of_fun  g := fun_image (csucc n) (Vfs g).

Lemma Sperner_3 C: (* 7 *)
   inc C (total_suborders r) -> {inc C &, injective cardinal}.
Lemma Sperner_4 C: (* 6 *)
  inc C (total_suborders r) -> cardinal C <=c (csucc n).
Lemma Sperner_5 g: bijection_prop g n E  -> (* 33 *)
 inc (chain_of_fun g) Sperner_mx_chain.
Lemma Sperner_6: order_length r (csucc n). (* 3 *)
```

Let $C \in \mathscr{M}$. Now the cardinal function is bijective, this says that for $i \leq n$ there is a set X_i of cardinal i in C. If $i < n$ there is x_i such that $X_{i+1} = X_i \cup \{x_i\}$. Define $g(i) = x_i$. This is a bijection, and $C(g) = g$.

```
Lemma Sperner_7 C: inc C Sperner_mx_chain -> (* 11 *)
   lf_axiom  cardinal C (csucc n) /\ bijection (Lf cardinal C (csucc n)).
Lemma Sperner_8 C (F := Vf (inverse_fun (Lf cardinal C (csucc n)))):
   inc C Sperner_mx_chain ->
   forall i, i <=c n -> inc (F i) C /\ cardinal (F i) = i. (* 4 *)
Lemma Sperner_9 C (F := Vf (inverse_fun (Lf cardinal C (csucc n))))
         (xi := fun i =>  union (F (csucc i) -s (F i))): (* 22 *)
   inc C Sperner_mx_chain ->
   forall i, i<c n -> F (csucc i) = (F i) +s1  (xi i).
Lemma Sperner_10 C (F := Vf (inverse_fun (Lf cardinal C (csucc n))))
         (g := Lf (fun i =>  union (F (csucc i) -s (F i))) n E):
   inc C Sperner_mx_chain ->
   bijection_prop g n E /\ C = chain_of_fun g. (* 59 *)
```

Let g be a bijection $\mathrm{I}_n \to \mathrm{E}$. Let X_i be as above for the chain $\mathrm{C}(g)$. Obviously $\mathrm{X}_i = g\langle i\rangle$. Now $g(i)$ is the only element of X_{i+1} not in X_i. This says that g depends only on $\mathrm{C}(g)$, or that $g \mapsto \mathrm{C}(g)$ is injective. This implies that the set of all g and the set of all $\mathrm{C}(g)$ have the same cardinal. Hence $\mathrm{card}(\mathscr{M}) = n!$.

```
Lemma Sperner_11 w i (C :=  chain_of_fun w)   (* 9 *)
     (F := Vf (inverse_fun (Lf cardinal C (csucc n)))):
  bijection_prop w n E -> i <=c n -> Vfs w i = F i.
Lemma Sperner_12 w i: bijection_prop w n E -> i <c n ->
   (Vf w i) = union (Vfs w (csucc i) -s Vfs w i). (* 18 *)
Lemma Sperner_13 w1 w2: bijection_prop w1 n E -> bijection_prop w2 n E ->
   chain_of_fun w1=  chain_of_fun w2 -> w1 = w2. (* 9 *)
Lemma Sperner_14: cardinal Sperner_mx_sub = factorial n (* 20 *)
```

We state now a technical result: Assume that U and V are two disjoint sets with cardinal a and b. There is a bijection $a+b \to \mathrm{U}\cup\mathrm{V}$ that maps a to U. In particular, if V is the complement of U in V, we get: There is a bijection $n \to \mathrm{E}$ that maps card(U) to U. Assume $\mathrm{U} \subset \mathrm{V} \subset \mathrm{E}$. Let g be a bijection $n \to \mathrm{E}$ that maps $a = \mathrm{card}(\mathrm{U})$ to U. Let U' and V' be the complement of U in V and of V in E. Let $c = \mathrm{card}(\mathrm{U}')$. We have a bijection $g' : n - a \to \mathrm{U}' \cup \mathrm{V}'$ that maps c to U". Define h by: if $i < c$ then $h(i) = g(i)$, otherwise $h(i) = g'(i - c)$. What we get is a bijection $n \to \mathrm{E}$ that maps a to U and $c + a$ to V. Consider $\mathrm{C}(h)$. This is an element of $\mathscr{M}$ that contains U and V.

```
Lemma Sperner_15 U V (a:= cardinal U) (b := cardinal V): (¨47 *)
  natp a -> natp b -> disjoint U V ->
    exists2 g, bijection_prop g (a+c b) (U \cup V)  & Vfs g a = U.
Lemma Sperner_16 U: sub U E ->
   exists2 g, bijection_prop g n E & Vfs g (cardinal U) = U. (* 8 *)
Lemma Sperner_17 U V: sub U V -> sub V E ->
  exists2 C, inc C Sperner_mx_chain & (inc U C /\ inc V C). (* 104 *)
```

Assume $\mathrm{A} \subset \mathrm{E}$ and A has k elements. Let B be the set of bijections $n \to \mathrm{E}$ that map k to A. We have shown above that there is g in this set. Now $f \in \mathrm{B}$ if and only if $h = g^{-1} \circ f$ is a bijection $n \to n$ that maps k to k. This is equivalent to; the restrictions of h to k and the complement of k of n are permutations. The number of these h (or of these f is $k!(n-k)!$. We deduce: The number of elements of $\mathscr{M}$ that contain A is $k!(n-k)!$.

```
Lemma Sperner_18 A (k := cardinal A): sub A E -> (* 122 *)
```

```
  cardinal (Zo (bijections n E) (fun g =>  Vfs g k = A)) =
     factorial k *c (factorial (n -c k)).
Lemma Sperner_19 A (k := cardinal A): (* 19 *)
  sub A E ->
  cardinal (Zo Sperner_mx_chain (inc A))
  = factorial k *c (factorial (n -c k)).
```

Consider now a free subset $\{A_1, A_2, \ldots, A_p\}$. Technically, there is an injection f such that $f(i) = A_{i+1}$. Define $\phi : \mathscr{M} \to p+1$ as follows: $\phi(C)$ is the union of the set I_C of all i such that $A_i \in C$. If no A_i is in C, then $I_C = \emptyset$ and $\phi(C) = 0$. If $A_i \in C$, then $I_C = \{i\}$ (if $A_J \in C$ then A_j is comparable to A_i hence equal), so that $\phi(C) = i$. Let M_i be the set of elements C of $\mathscr{M}$ such that $\phi(C) = i$. and m_i its cardinal:. We have $n! = \text{card}(\mathscr{M}) = \sum_i m_i$.

```
Definition free_rep f :=
  [/\ bijection f, natp (source f) & inc (target f) (free_subsets r)].
Definition chain_free_meet f C :=
  union(Zo (csucc (source f)) (fun i => \0c <c i /\ inc (Vf f (cpred i)) C)).
Definition chain_free_meet_i f i :=
  Zo Sperner_mx_chain (fun C => chain_free_meet f C  = i).

Lemma free_rep_prop1 A: inc A  (free_subsets r) ->
   exists2 f, free_rep f & target f = A.
Lemma free_rep_prop2 f : free_rep f ->
 cardinal (source f) = cardinal (target f).
Lemma Sperner_cf0 f i: free_rep f -> inc i (source f) -> (* 6 *)
   [/\  sub (Vf f i) E, cardinal  (Vf f i) <=c n & natp (cardinal (Vf f i))].
Lemma Sperner_cf1 f C i: free_rep f -> inc C Sperner_mx_chain ->  (* 23 *)
  inc i (source f) -> inc (Vf f i) C -> chain_free_meet f C = csucc i.
Lemma Sperner_cf2 f C: free_rep f -> inc C Sperner_mx_chain ->
  (forall  i, inc i (source f) -> ~inc (Vf f i) C) ->
  chain_free_meet f C = \0c. (* 9 *)
Lemma Sperner_cf3 f C: free_rep f -> inc C Sperner_mx_chain -> (* 7 *)
  chain_free_meet f C = \0c \/
  exists i,  [/\ inc i (source f), inc (Vf f i) C &
    chain_free_meet f C = (csucc i)].
Lemma Sperner_cf4 f (p := source f): free_rep f ->  (* 13 *)
   factorial n = csumb (csucc p) (fun i => cardinal (chain_free_meet_i f i)).
```

We rewrite the previous equality as $c_n n! = c_n m_0 + \sum_i c_n m_{i+1} = c_n m_0 + s$. Note that (for $i > 0$) M_i is the set of all chains C that contain A_i. Let a_i be the cardinal of A_i. We deduce $m_i = a_i!(n - a_i)!$. Now, $c_n m_i \geq n!$. Hence $s \leq pn!$, and if $s = pn!$ then $c_n m_i = n!$ for $i > 0$. It follows $p \leq c_n$.

```
Lemma Sperner_cf5 f (p := source f)  (* 7 *)
      (m := fun i => cardinal (chain_free_meet_i f i)):
  free_rep f ->
   cn *c factorial n = cn *c (m \0c) +c
     csumb p (fun i => cn *c (m (csucc i))).
Lemma Sperner_cf6 f i: free_rep f -> inc i (source f) ->  (* 7 *)
   chain_free_meet_i f (csucc i) = (Zo Sperner_mx_chain (inc (Vf f i))).
Lemma Sperner_cf7 f i (a := cardinal (Vf f i))  (* 3 *)
      (m := cardinal (chain_free_meet_i f (csucc i))):
  free_rep f -> inc i (source f) ->
  natp m /\ m = factorial a *c factorial (n -c a).
Lemma Sperner_cf8 f (p := source f)  (* 25 *)
```

```
      (m := fun i => cardinal (chain_free_meet_i f i))
      (s := csumb p (fun i => cn *c (m (csucc i)))):
  free_rep f ->
   [/\ natp s, p *c factorial n <=c s &
   (p *c factorial n = s ->
         forall i, inc i p -> cn *c m (csucc i) = factorial n)].
Lemma Sperner_cf9 f : free_rep f -> (source f) <=c cn. (* 7 *)
```

The previous result is also: if A is free, then its cardinal is $\leq c_n$. This prove that the width of the order is c_n.

```
Lemma Sperner_20 A: inc A  (free_subsets r) -> cardinal A <=c cn.
Lemma Sperner_21 : order_width r  cn. (* 5 *)
```

Consider a maximal free set, represented by a function as above. From the previous inequalities we get $m_0 = 0$ and $c_n m_i = n!$ for $i > 0$. The first equality says that M_0 is empty; so that every element of $\mathscr{M}$ contains an A_i; the second equality says that a_i is $n/2$ or $n - n/2$. Let k be $n/2$ or $n-n/2$; now F_k has cardinal c_n. If $\mathrm{A} \subset \mathrm{F}_k$ then $\mathrm{A} = \mathrm{F}_k$ since A is maximal. Assume n even. Here $n/2 = n - n/2$. It follows that for all i; $a_i = n/2$ and $\mathrm{A} = \mathrm{F}_{n/2}$.

Assume $n = 2q+1$ is odd. Now $a_i = q$ or $a_i = q+1$. We pretend $\mathrm{A} = \mathrm{F}$ or $\mathrm{A} = \mathrm{F}_{q+1}$. In case no a_i is q, then the second alternative holds. Assume some a_i is q. We pretend that all a_i are q so that the first alternative holds. Assume that U is in A with cardinal q; assume that V is in A not of cardinal q. It is hence of cardinal $q+1$ and of the form $\mathrm{W} \cup \{x\}$ where x is not in W and W has cardinal q. We prove $\mathrm{W} \in \mathrm{A}$, which contradicts $\mathrm{V} \in \mathrm{A}$ since A is free and $\mathrm{W} < \mathrm{V}$. The proof is by induction on the number of elements in U and not in W. Base case: there is none, hence $\mathrm{U} = \mathrm{V}$. Otherwise, there is x in V not in U, hence y in U not in V. Let V' be V with x replaced by y. By induction $\mathrm{V}' \in \mathrm{A}$. Now if we add y to V or x to V' we obtain the same set T. As $\mathrm{V} \subset \mathrm{T}$, there is a chain $\mathrm{C} \in \mathscr{M}$ that contains V and T. Now comes the trick. Since M_0 is empty, C contains an A_j, which is of cardinal q or $q+1$, and by injectivity of the cardinal C must be V or T so that one of these sets is in A; if it's V we win. Otherwise it is T, the set V' to which we added x; But A is free and $\mathrm{V}' \in \mathrm{A}$; absurd.

```
Lemma Sperner_cf11 f : free_rep f -> (source f) = cn ->  (* 25 *)
  (forall C, inc C Sperner_mx_chain ->
     exists2 i, inc i (source f) & inc (Vf f i) C) /\
  (forall i, inc i (source f) ->  is_half_n n  (cardinal (Vf f i))).
Lemma Sperner_cf12 f k (F  := (subsets_with_p_elements k E)):  (* 9 *)
 free_rep f -> (source f) = cn -> k <=c n ->  is_half_n n k ->
  sub (target f)  F -> target f =  F.
Lemma Sperner_cf13 f (F  := (subsets_with_p_elements (chalf n) E)):
 free_rep f -> source f = cn -> evenp n -> target f =  F. (* 10 *)
Lemma Sperner_cf14 f (F := fun k => (subsets_with_p_elements k E)):
 free_rep f -> source f = cn -> oddp n ->
 target f =  F (chalf n) \/ target f =  F (csucc (chalf n)). (* 127 *-
```

In summary: if A is maximal free, it is F_k, where k is $n/2$ or $(n+1)/2$ if n is odd.

```
Lemma Sperner_cf15 f (F := fun k => (subsets_with_p_elements k E)):
  free_rep f -> source f = cn ->
  exists k, [/\ natp k, is_half_n n k &  target f = F k]. ( (* 11 *)
Lemma Sperner_22 A: inc A  (free_subsets r) -> cardinal A = cn ->
  exists k, [/\ natp k, is_half_n n k & A = subsets_with_p_elements k E].
```

To finish, let's state the inequality of Lubell-Yamamoto-Meshkalin. If A is free, a_i is the number of elements of A of cardinal i then

$$\sum_{k\le n} \frac{a_k}{\binom{n}{a_k}} \le 1$$

Proof. Let's rewrite b_i instead of a_i. Represent A by a function as above. We have $n! = \sum_i m_i$ and $m_i = a_i!(n-a_i)!$ (for $i>0$). If we omit m_0 we get $\sum_i a_i!(n-a_i)! \le n!$. Let B_k be the set of indices i such that $a_i = k$. By associativity $\sum_i f(a_i)$ becomes $\sum_k b_k f(k)$. The result follows as $f(k) = n!/\mathrm{B}(k)$.

```
Lemma Lubell_Yamamoto_Meshalkin A   (* 40 *)
   (b := fun k => cardinal (Zo A (fun z => cardinal z = k))):
  (inc A (free_subsets r)) ->
  csumb (csucc n) (fun k => (b k) *c ((factorial k) *c (factorial (n-c k))))
   <=c factorial n.

End Sperner.
```

Alternate Proof of Dilworth's theorem in the finite case by induction on the cardinal. We assume that for every ordered set E with less than n elements, there is a partition into k chains, where k is the width of E, and we show that the result holds when E has n elements. Since E is finite there is a chain C of maximal length p. Assume first $p<2$. This says that E is free. The desired partition is the set of singletons. So, assume that C has at least two elements. Let α and ω be the least and greatest element of C. We have $\alpha<\omega$. Let E' be the complement of $\{\alpha,\omega\}$ in E, and k' its width. We have a maximal free subset X in E with k elements and a maximal free subset X' in E' with k' elements. Since X' is free in E we have $k'\le k$. Note that α and ω cannot be both in X. So $\mathrm{X}\cap\mathrm{X}'$ is free in E' with k or $k-1$ elements, so $k=k'$ or $k=k'+1$. Assume first $k=k'+1$. By the induction hypothesis, there is a partition of E' into k' chains. Since $\{\alpha,\omega\}$ is a chain, we can partition E into $k=k'+1$ chains. Assume now $k=k'$. Let E^+ and E^- be the sets of elements in E that are $\ge a$ (resp. $\le a$) for some $a\in\mathrm{X}'$. If x is neither in E^+ nor E^-, then $\mathrm{X}'\cup\{x\}$ is free, absurd. Obviously $\mathrm{E}^+\cap\mathrm{E}^-=\mathrm{X}'$. Note that $\alpha\not\in\mathrm{E}^+$ (otherwise we could add a least element to the chain C). So E^+ is smaller than E and we can apply the induction hypothesis. Since X' is a free subset of E^+, the width of E^+ is k and there is a partition P^+ into k chains. For $x\in\mathrm{E}^+$, let $i_+(x)$ be such that $x\in i_+(x)$ and $i_+(x)\in\mathrm{P}^+$. Assume x and y in X' (hence in P^+) and $i_+(x)=i_+(y)$. So, there is z in P^+ that has x and y as elements; hence x and y are comparable; since X' is free, this says $x=y$, so i_+ is an injection $\mathrm{X}'\to\mathrm{P}^+$; since the two sets have the same cardinal, this is a bijection. This means that every $y\in\mathrm{E}^+$ belongs to some $i_+(x)$ for $x\in\mathrm{X}'$. Similarly $\omega\not\in\mathrm{E}^-$, and we can define P^- and i_-. Let $i(x)=i_+(x)\cup i_-(x)$, and U be the set of all $i(x)$ for $x\in\mathrm{X}'$. Assume $a\in\mathrm{E}^+$, there is $x\in\mathrm{X}'$ such that $a\in i_+(x)$, hence $a\in i(x)$. The same holds if $a\in\mathrm{E}^-$, so that U is a covering of E. Assume $a\in i(x)\cap i(y)$. Then $x=y$; this is obvious if $a\in i_+(x)\cap i_+(y)$. or if $a\in i_-(x)\cap i_-(y)$. Assume $a\in i_+(x)\cap i_-(y)$. This implies $a\in\mathrm{E}^+\cap\mathrm{E}^-$, hence $a\in\mathrm{X}'$. Now a and x are comparable, since they belong to $i_+(x)$; they are equal because X' is free. So $x=a=y$. This says that U is a partition, it also says that U has k elements. That every element of U is totally ordered is easy: assume a and b in $i_+(x)\cup i_-(x)$. If they belong to both $i_+(x)$ or to both $i_-(x)$ the result is true. Assume $a\in i_+(x)$ and $b\in i_-(x)$. Since x and a belong to $i_+(x)$ they are comparable. We pretend $x\le a$. For otherwise $a\le x$. Since $a\in\mathrm{E}^+$, there is $c\in\mathrm{X}'$ such that $c\le a$; hence $c\le x$. Now X' is free so that $c=x$. Similarly, $b\le x$, so that $b\le a$. Qed.

```
Definition Exercise4_5_conc r k :=
  exists X, [/\ partition_s X (substrate r), cardinal X = k &
```

```
    sub X (total_suborders r)].

Lemma Exercise4_5b_alt r k: finite_set (substrate r) ->   (* 357 *)
  order r -> natp k ->  order_width r k -> Exercise4_5_conc r k.
```

Solution. We start with a lemma. Assume that Y and T are two sets with k elements. Assume $T \subset \bigcup Y$, the elements of Y mutually disjoint. Assume moreover that for all $Z \in Y$ the set $Z \cap T$ has at most one element. Then it has exactly one element (there is a function $f : T \to Y$ such that $t \in f(t)$; it is injective because $Z \cap T$ is small; it is bijective because Y and T have the same cardinal).

```
Lemma Exercise4_5a Y T k:   (* 22 *)
  cardinal Y = k -> cardinal T = k -> natp k ->
  sub T (union Y) -> disjoint_set Y ->
  (forall Z, inc Z Y -> small_set (T \cap Z)) ->
  (forall Z, inc Z Y -> singletonp (T \cap Z)).
```

(a): case E finite by induction on the cardinal of E. As before, we assume that E has n elements and that the result holds for sets with less than n elements whatever k. The result is obvious when E is empty. So, we may assume E non-empty, consider a minimal element a, define $E' = E - \{a\}$ with width k'.

Clearly $k' \le k$ (a free subset in E' is free in E) and $k \le k' + 1$ (the intersection of E' and a free subset of E with k elements is free in E' and has at least $k-1$ elements). Case 1: $k = k'+1$. By induction E' has a chain decomposition with $k-1$ elements. We complete it with $\{a\}$. This gives a decomposition with k sets. (This requires 100 lines of proof).

Case 2: $k' = k$, we get a decomposition with k elements. For each C_i in the partition we consider U_i, the set of element of C_i that are $\ge a$, and $\bar{U}_i = U_i \cup \{a\}$. This is a totally ordered set. Let $V_i = E - \bar{U}_i$. Assume that for some i, all free subsets of V_i have less than k elements. Let T be a free subset of E with k elements. It has at most one element in U'_i, thus has $k-1$ elements in V_i. Thus, there is a partition of V_i into $k-1$ sets, and it suffices to add $\bar{U}_i$ to get a partition of E into k parts. (This requires 100 lines of proof).

We assume now that for each i there is a free subset S_i with k elements such that $S_i \subset V_i$. [Note: we use here the axiom of choice]. Write $S_i \cap C_j = \{s_{ij}\}$. Note that s_{ij} is never a so that $s_{ij} \le a$ is false, since a is minimal. On the other hand $a \le s_{ii}$ is equally false, since s_{ii} is in C_i and V_i. Let W_j be the set of all s_{ij}. This set is non-empty, totally ordered and finite, thus has a least element, say s_j. We have $s_j \le s_{ij}$. Assume $s_i \le s_j$. Then s_i is some s_{li} and $s_{li} \le s_{lj}$. Since S_l is free, we get $s_{li} = s_{lj}$. We deduce $i = j$, for otherwise C_i and C_j are disjoint. This implies in particular that $l \mapsto s_l$ is injective. Let A be the set of all s_l. It has k elements. Let $B = A \cup \{a\}$. It has more than k elements, yet is free, absurd. (This requires 150 lines of proof).

```
Lemma Exercise4_5b r k: finite_set (substrate r) -> (* 353 *)
  order r -> natp k ->  Exercise4_5_hyp r k -> Exercise4_5_conc r k.
```

(b): general case by induction on k.

Since singletons are free, $k = 0$ says that E is empty, and the conclusion is trivial. Consider a free subset X_0 of maximal cardinal $k+1$. Assume (H): there is a chain C, such that each free subset of $E - C$ has at most k elements. Note that $X_0 \cap (E - C)$ has k elements, since X_0 cannot have two elements in the totally ordered set C. Note also that C is non-empty. We can apply the induction assumption to $X - C$, partition it into k parts and add C as a $k+1$-th set of the partition (This requires 100 lines of proof).

We say that C is strongly related if $C \subset E$, and for every finite subset F of E, there is a chain decomposition X of F and for some $x \in X$ we have $C \cap F \subset x$. [we assume that X has at most $k+1$ elements]. By (a), the empty set is strongly related. The set of these C is inductive for inclusion. All we need to do is prove that if $\mathfrak{C}$ is a totally ordered set of strongly related sets, then $C = \bigcup \mathfrak{C}$ is strongly related. Consider a finite set F. For each $x \in C \cap F$ we choose an element C_x of $\mathfrak{C}$ such that $x \in C_x$. The set of all $G = \{C_x, x \in C \cap F\}$ is finite and totally ordered, thus has a greatest element C_0 (for all x we have $C_x \subset C_0$). We have $C \cap F = C_0 \cap F$, and the conclusion holds since C_0 is strongly related. (This requires 80 lines of proof).

We assume now that C_0 is a maximal strongly related set. Let a and b be two elements of C_0, and $F = \{a, b\}$. Since C_0 is strongly related, there exists a totally ordered set X such that $F \subset X$. Thus F is totally ordered, and C_0 is totally ordered. Assume that every free subset of X_0 has $\leq k$ elements. Then C_0 satisfies assumption (H) and the theorem is proved. On the contrary, assume that there is a free set with $k+1$ elements $a_1, \dots, a_{k+1}$ not in C_0. Let $C_i = C_0 \cup \{a_i\}$. By maximality of C_0, this is not a strongly related set. Thus, there exists a finite set F_i, such that, for every chain decomposition X of F, $C_i \cap F_i$ is not a subset of any element of X. Consider the union G of these F_i and the set of all a_i. This is a finite set. Since C_0 is a strongly related set, there exists a chain decomposition X of G such that $C_0 \cap G$ is a subset of some element Y_0 of X. Fix i. Consider T_i, the set of all $F_i \cap Y$ for $Y \in X$. These sets are totally ordered, and the union is F_i. There are at most $k+1$ distinct elements in T; so that t_i is a chain decomposition of F_i. Thus $C_i \cap F_i$ is not a subset of $F_i \cap Y$, whatever Y. Take $Y = Y_0$. We know that $C_0 \cap F_i$ is a subset of $F_i \cap Y$. This implies $a_i \not\in Y$. Finally, each a_i (there are $k+1$ such elements) belongs to at most one element X_j of X (there are $\leq k+1$ such elements), since the set of a_i is free, the sets X_j are totally ordered. This implies that each X_j (included Y) contains exactly one a_i. Contradiction. (This requires 110 lines of proof).

```
Lemma Exercise4_5d r k: (* 306 *)
  order r -> natp k -> order_width r k -> Exercise4_5_conc r k.
```

¶ 6. (a) Let A be a set and let $(X_i)_{1 \leq i \leq m}$, $(Y_j)_{m+1 \leq j \leq m+n}$ be two finite families of subsets of A. Let h be the least integer such that, for each integer $r \leq m-h$ and each subset $\{i_1, \dots, i_{r+h}\}$ of $r+h$ elements of $[1, m]$, there exists a subset $\{j_1, \dots, j_r\}$ of r elements of $[m+1, m+n]$ for which the union of the sets X_{i_α} $(1 \leq \alpha \leq r+h)$ meets each of the sets Y_{j_β} $(1 \leq \beta \leq r)$ (which implies that $m \leq n+h$). Show that there exists a finite subset B of A with at most $n+h$ elements such that every X_i $(1 \leq i \leq m)$ and every Y_j $(m+1 \leq j \leq m+n)$ meets B. (Consider the order relation on the interval $[1, m+n]$ whose graph is the union of the diagonal and the set of pairs (i, j) such that $1 \leq i \leq m$ and $m+1 \leq j \leq m+n$ and $X_i \cap Y_j \neq \emptyset$, and apply Exercise 5 to this ordered set.)

(b) Let E and F be two finite sets and let $x \to A(x)$ be a mapping of E into $\mathfrak{P}(F)$. Then there exists an injection f of E into F such that $f(x) \in A(x)$ for each $x \in E$ if and only if for each subset H of E, we have $\mathrm{Card}\big(\bigcup_{x \in H} A(x)\big) \geq \mathrm{Card}(H)$ (the method of proof is analogous to that of (a), with $h = 0$).

(c) With the hypotheses of (b), let G be a subset of F. Then there exists an injection f of E into F such that $f(x) \in A(x)$ for each $x \in E$ and such that $f(E) \supset G$ if and only if f satisfies the condition of (b) and for each subset L of G the cardinal of the set of all $x \in E$ such that $A(x) \cap L \neq \emptyset$ is $\geq \mathrm{Card}(L)$. (Let $(a_i)_{1 \leq i \leq p}$ be the sequence of distinct elements of G, arranged in some order; let $(b_j)_{p+1\le j\le p+m}$ be the sequence of distinct elements of F, arranged in some order; and let $(c_k)_{p+m+1\le k\le p+m+n}$ be the sequence of distinct elements of E, arranged in some order. Consider the order relation on the set $[1, p+m+n]$ whose graph is the union of the diagonal and the set of pairs (i, j) such that either

$$1 \le i \le p \text{ and } p+1 \le j \le p+m \text{ and } a_i = b_j,$$

$$\text{or } 1 \le i \le p \text{ and } p+m+1 \le j \le p+m+n \text{ and } a_i \in \mathrm{A}(c_j),$$

$$\text{or } p+1 \le i \le p+m \text{ and } p+m+1 \le j \le p+m+n \text{ and } b_i \in \mathrm{A}(c_j);$$

then apply Exercise 5.)

Note: (a) The sets X_i and Y_j must be non-empty, since otherwise no set can meet them all. (c) is non-trivial, and the indication is false.

(a) Let I be the interval $[1, m]$ and J the interval $[m+1, m+n]$. All we need to know is that I and J have m and n elements, and are disjoint.

```
Section Exercise46.
Variables A X Y m n :Set.
Hypothesis (nN: natp n) (mN: natp m).
Hypothesis Xpr:
  [/\ fgraph X, cardinal (domain X) = m, sub (range X) (\Po A) &
   nonempty_fam X].
Hypothesis Ypr:
  [/\ fgraph Y, cardinal (domain Y) = n, sub (range Y) (\Po A) &
   nonempty_fam Y].
Hypothesis disdom: disjoint (domain X) (domain Y).

Definition E46_hprop h := forall r  Z, r <=c (m -c h) ->
   sub Z (domain X) -> cardinal Z = r +c h  ->
   exists T, [/\ sub T (domain Y),  cardinal T = r &
   forall j, inc j T -> meet (Vg Y j) (unionb (restr X Z))].

Definition E46_hp h := [/\ natp h, h <=c m, E46_hprop h &
 forall l, natp l ->  l <=c m -> E46_hprop l -> h <=c l].

Definition E46_conc h := exists B, [/\ (cardinal B) <=c (n +c h),
  finite_set B, allf X (meet B) &allf Y (meet B)].
```

Note that p_m holds, so that there is a least h satisfying p_h. Note also that if $h \le m$, taking $r = m-h$ and $\mathrm{Z} = \mathrm{I}$ yields a subset with r elements of J. Thus $m \le n+h$.

Write $i < j$ if X_i meets Y_j. We can convert this into an order relation on $\mathrm{E} = \mathrm{I} \cup \mathrm{J}$ by adding reflexivity. The condition the union of X_{i_α} $(1 \le \alpha \le r+h)$ meets $\mathrm{Y}_{\mathrm{J}_\beta}$ can be restated as: there is at least one α such that $i_\alpha \le j_\beta$. The assumptions of (a) can be restated as h is the least integer satisfying property p_h: for any $r \le m-h$ and any $\mathrm{Z} \subset \mathrm{I}$ of cardinal $r+h$, there is a subset T of J with r elements such that, if $j \in \mathrm{T}$, there is $i \in \mathrm{Z}$ such that $i \le j$.

```
Definition E46_u := (domain X) \cup (domain Y).
Definition E46_order_rel x y :=
  x = y \/ [/\ inc x (domain X), inc y (domain Y) & meet (Vg X x) (Vg Y y)].
Definition E46_order_r :=  graph_on E46_order_rel  E46_u.

Lemma Exercise4_6a:  (* 12 *)
  order_on  E46_order_r E46_u) /\
```

```
  (forall x y, gle E46_order_r x y <->
    [/\ inc x E46_u, inc y E46_u & E46_order_rel x y]).
Lemma Exercise4_6b h: h <=c m ->  (* 10 *)
  E46_hprop h -> m <=c (n +c h).
Lemma Exercise4_6c: exists h,  E46_hp h. (* 8 *)
Lemma Exercise4_6d h: E46_hprop h <->  (* 14 *)
  forall r Z, r <=c (m -c h) ->
   sub Z (domain X) -> cardinal Z = r +c h  ->
   exists T, [/\ sub T (domain Y),  cardinal T = r &
   forall j, inc j T -> exists2 i, inc i Z & gle E46_order_r i j].
```

Consider a free subset K of E and write $Z = K \cap I$. Then K is the disjoint union of K and a subset L of J. If Z has at most h elements, then K has at most $h+n$ elements since L has at most n elements. But if Z has $r+h$ elements, there is a set T with r elements such that $L \cap T = \emptyset$, so that L has at most $n-h$ elements, and we get the same conclusion. Note that J is a free subset with n elements, so that the with of the order is $n+k$ for some k, with $k \le h$. By Dilworth, there is a partition on k chains.

```
Lemma Exercise4_6e h K:  (* 39 *)
  natp h -> E46_hprop h -> inc K (free_subsets E46_order_r) ->
  (cardinal K) <=c (n +c h).

Lemma Exercise4_6f h: natp h -> E46_hprop h -> (* 8 *)
  exists k, [/\ natp k, k <=c h & order_width E46_order_r  (n +c k)].
Lemma Exercise4_6g h: E46_hp h -> (* 4 *)
  exists k, [/\ natp k, k <=c h & Exercise4_5_conc E46_order_r  (n +c k)].
```

Assume that U is a non-empty totally ordered subset of E. If a and b are in U we have $a = b$, $a < b$ or $b < a$. In the case $a < b$ we have $a \in I$ and $b \in J$ so that U cannot have more than two elements. Consider the following quantity x_U. If U has two elements, say a and b with $a < b$, then x_U is an element of the intersection $X_a \cap Y_b$ (which is nonempty by definition of <). Otherwise U is a singleton, say $U = \{i\}$ or $U = \{j\}$ with $i \in I$ and $j \in J$. We choose for x_U some element of X_i or Y_j (remember that we assume these sets to be nonempty). In any case, we have $x_U \in A$.

Assume that E is the union of $n+k$ totally ordered non-empty subsets. Let B be the set of all x_U for U in the union. This is a finite subset of A with at most $n+k$ elements. We have shown that there exists an index k with $k \le h$, thus B has at most $n+h$ elements. By construction B meets any element of X_i and any Y_j.

```
Lemma Exercise4_6h h: E46_hp h ->  E46_conc h. (* 109 *)
End Exercise46.
```

(b) Assume that E and F are two finite sets, $A : E \to \mathfrak{P}(F)$ some function, and G a subset with p elements of F. Let (P) the assumption that there is an injective function $f : E \to F$ whose range contains G, such that $f(x) \in A(x)$ and let (Q) be the assumption that for any $H \subset E$, the union of all $A(x)$, for $x \in H$ has at least as many elements as H, and let (R) be the assumption that for any subset L of G, the number of elements x such that $A(x)$ meets L is at least the cardinal of L.

Point (c) consists in proving that (P) is equivalent to (Q) and (R), point (b) considers the case $G = \emptyset$. Note that (R) holds trivially in this case.

```
Definition E46b_hyp E F A :=
  exists2 f, injection_prop f E F &
  (forall x, inc x E -> inc (Vf f x) (A x)).
Definition E46b_conc E A :=
  forall H, sub H E ->
  (cardinal H) <=c (cardinal (union (fun_image H A))).
Definition E46c_hyp E F A G :=
  exists f, [/\ injection_prop f E F,
  sub G (Imf f) &  (forall x, inc x E -> inc (Vf f x) (A x))].
Definition E46c_conc E A G :=
  (cardinal L) <=c (cardinal (Zo E (fun z => meet (A z) L)))
```

Assume that (P) holds, so that we have an injection f. We have $f\langle \mathrm{H}\rangle \subset \bigcup_{x\in \mathrm{H}} \mathrm{A}(x)$ and $\mathrm{card}(\mathrm{H}) = \mathrm{card}(f\langle \mathrm{H}\rangle)$.

Assume $\mathrm{L} \subset \mathrm{G} \subset f\langle \mathrm{E}\rangle$. There is K such that $\mathrm{L} = f\langle \mathrm{K}\rangle$, and K has the same cardinal as L. If $x \in \mathrm{K}$, then $f(x) \in \mathrm{A}(x) \cap \mathrm{L}$. Thus, (P) implies (Q) and (R).

```
Lemma Exercise4_6i E F A:  E46b_hyp E F A -> E46b_conc E A. (* 10 *)
Lemma Exercise4_6j E F A G: E46c_hyp E F A G -> (* 18 *)
  (E46b_conc E A /\  E46c_conc E A G).
```

Discussion. Let's try to prove that (Q) and (R) imply (P) using the hint of Bourbaki for (c). Let $\mathrm{D} = \mathrm{G}_1 \cup \mathrm{F}_2 \cup \mathrm{E}_3$. This means that an element of D is either a y_1 with $y \in \mathrm{G}$, a y_2 with $y \in \mathrm{F}$, or a x_3 with $x \in \mathrm{E}$. If $y \in \mathrm{G}$ then y_1 and y_2 belong to D. If z is y_1 we define $\bar{z}$ to be y_2.

Consider the order relation such that if $y \in \mathrm{A}(x)$ then $y_2 < x_3$, and if $y \in \mathrm{G}$ then $y_1 < y_2$ (to these rules we add: if y is in $\mathrm{A}(x)$ and G, then $y_1 < x_3$, and reflexivity). Note that $z < \bar{z}$, and if $x < y$ the index of x is less than the index of y. So, if U is a totally ordered subset of D it has at most one element of E, at most one element of F, at most one element of G. If it has x_3 and y_1 or y_2, then $y \in \mathrm{A}(x)$; if it has y_1 and z_2, then $y = z$. Let f be as above. We have $f(x)_2 < x_3$. Define $\mathrm{U}(y)$ as follows: it contains y_2; if $y \in \mathrm{G}$ it contains y_1; if $y = f(x)$ it contains x_3. This is a totally ordered subset of D. The set of these U is a partition of D with m elements (where m is the cardinal of F).

We show here the converse: there is a partition of D into m totally ordered subsets. This follows from Exercise 5; all we have to do is show that every free subset has at most m elements (note that F is free). Let J be a free subset. Replace every z by $\bar{z}$ when $y \in \mathrm{G}$ (this means: replace y_1 by y_2, so that every element has index 2 or 3). This yields a free subset with the same number of elements. Let J_2 and J_3 be the elements of J with index 2 and 3, K and H the same sets where we forget the indices. Now J has $\mathrm{card}(\mathrm{H}) + \mathrm{card}(\mathrm{K})$ elements. Let H' be the union of the $\mathrm{A}(x)$ for $x \in \mathrm{H}$. The sets H' and K are disjoint (if $y \in \mathrm{H}'$ there is $x \in \mathrm{H}$ such that $y_2 < x_3$, but $x_3 \in \mathrm{J}_3$ so y_2 cannot be in J_2). Since they are subsets of F we have $\mathrm{card}(\mathrm{H}') + \mathrm{card}(\mathrm{K}) \le m$. Now assumption (Q) says $\mathrm{card}(\mathrm{H}) \le \mathrm{card}(\mathrm{H}')$ and the conclusion follows.

Let $(\mathrm{U}_i)_i$ be the partition; each U_i contains at most one y_2; each y_2 is in some U_i so each U_i contains exactly one y_2. Assume $x \in \mathrm{E}$. Assume x_3 in U_i and U_i contains y_2. Define $f(x) = y$. Since $y_2 < x_3$ we have $y \in \mathrm{A}(x)$. Since x_2 is in at most one U_i, the function f is injective.

This shows point (b). Assumption (R) has not been used, so that we cannot deduce (c). Example. Let $\mathrm{E} = \{x\}$, $\mathrm{F} = \{a, b\}$, $\mathrm{G} = \{b\}$, $\mathrm{A}(x) = \mathrm{F}$. We have two choices for $f(x)$ it can be a or b. The relation $\mathrm{G} \subset f\langle \mathrm{E}\rangle$ holds in only one case. We can solve the problem by removing one of the partitions; this means decide that one of the sets is not totally ordered; this means remove a relation $y_2 < x_3$; this means: remove a particular y from a particular $A(x)$. The following works= form every $A(x)$ that contains an element of G remove all elements not in G. Does this work in general?

```
Lemma Exercise4_6k E F A G: (* 225 *)
  finite_set F -> sub G F ->
  (forall x, inc x E -> sub (A x) F) ->
   E46b_conc E A ->  E46c_conc E A G -> E46b_hyp E F A.

Lemma Exercise4_6l E F A: (* 8 *)
  finite_set F -> (forall x, inc x E -> sub (A x) F) ->
   E46b_conc E A ->  E46b_hyp E F A.
```

---

7. An element a of a lattice E is said to *irreducible* if the relation $\sup(x, y) = a$ implies either $x = a$ or $y = a$.

(a) Show that in a finite lattice E every element a can be written as $\sup(e_1, \ldots, e_k)$, where the e_i $(1 \le i \le k)$ are irreducible.

(b) Let E be a finite lattice and let J be the set of its irreducible elements. For each $x \in E$ let $S(x)$ be the set of all $y \in J$ which are $\le x$. Show that the mapping $x \to S(x)$ is an isomorphism of E onto a subset of $\mathfrak{P}(J)$, ordered by inclusion, and that $S(\inf(x, y)) = S(x) \cap S(y)$.

Solution. We start with a lemma that will be useful for the next exercise. Let A and B be two subsets of E, ξ_A and ξ_B the characteristic functions. We have $A \subset B$ if and only if, for each $x \in E$ we have $\xi_A(x) \le \xi_B(x)$. In a lattice, $\sup(A \cup \{b\}) = \sup(\sup A, b)$ provided that $\sup A$ exists.

We introduce some definitions, and show that if x is not irreducible, it can be written as $\sup(a, b)$ where $a < x$ and $b < x$.

```
Lemma char_fun_sub A A' B:  sub A B -> sub A' B -> (* 8 *)
  ((sub A A') <-> (forall x, inc x B ->
     (Vf (char_fun A B) x) <=c (Vf (char_fun A' B) x))).
Lemma supremum_singleton r x:  (* 1 *)
   order r -> inc x (substrate r) -> supremum r (singleton x) = x.

Definition sup_irred r x:=
  forall a b, inc a (substrate r) -> inc b (substrate r) ->
   x = sup r a b -> (x = a \/ x = b).

Definition irreds r := Zo (substrate r)(sup_irred r).

Definition E47S r x := Zo (substrate r)
   (fun z => (sup_irred r z) /\ (gle r z x)).

Section Irred_latice.

Variable r:Set.
Hypothesis lr:lattice r.

Lemma Exercise4_7a x: inc x (substrate r) -> (* 6 *)
 sup_irred r x \/ (exists a b, [/\ glt r a x, glt r b x & x = sup r a b]).
Lemma Exercise3_8a a: distributive_lattice3 r -> (* 10 *)
```

```
  sup_irred r a ->
  forall x y, inc x (substrate r) -> inc y (substrate r) ->
   gle r a (sup r x y) -> (gle r a x \/ gle r a y).
```

Consider now what happens when E is empty. Point (a) is trivial. In (b) we have to show there is some subset F of $\mathfrak{P}(\mathrm{J})$ isomorphic to E, and we can choose the empty set. In Exercise 8(c), we have to show that F is isomorphic to some set A (see discussion below); this set is empty if E is empty. Exercise 8(c) is trivial. In Exercise 8(d), the case $k = 0$ is trivial, and $k > 0$ implies that E has at least two elements. Therefore, we shall from now on (until the end of the next exercise) assume E non-empty.

We assume from now on that E is finite. Then every non-empty subset of E has a maximal and minimal element. In particular, E has a least element a, which is in J, and in every $\mathrm{S}(x)$. Every subset of E has a supremum (which is a if the set is empty).

It follows that $\mathrm{S}(x)$ is non-empty, and thus has a least upper bound. Obviously $\sup(\mathrm{S}(x)) \leq x$, and $a \leq b$ implies $\mathrm{S}(a) \subset \mathrm{S}(b)$. We pretend that $x = \sup(\mathrm{S}(x))$, proof by contradiction. Let V be the set of elements x such that $\sup(\mathrm{S}(x)) \neq x$. Assume that this set is non-empty, and consider a minimal element. It cannot be an irreducible element. Thus, assume $x = \sup(a,b)$, with $a < x$ and $b < x$. We cannot have $a \in \mathrm{V}$ so that $a \leq \sup(\mathrm{S}(a))$, and $a \leq \mathrm{S}(x)$, similarly, $b \leq \mathrm{S}(x)$ so that $x \leq \sup(\mathrm{S}(x))$. We deduce: $a \leq b$ is equivalent to $\mathrm{S}(a) \subset \mathrm{S}(b)$.

```
Hypothesis fs: finite_set (substrate r).
Hypothesis nes: nonempty (substrate r).

Definition E48P := (irreds r) -s1 (the_least r).

Lemma Exercise4_7b: order r. (* 1 *)
Lemma finite_set_maximal1 U:  (* 8 *)
  sub U (substrate r) -> nonempty U ->
  exists2 x, inc x U & (forall y, inc y U -> gle r x y -> x = y).
Lemma finite_set_minimal1 U: (* 10 *)
  sub U (substrate r) -> nonempty U ->
  exists2 x, inc x U & (forall y, inc y U -> gle r y x -> y = x).

Lemma Exercise4_7c: has_least r.  (* 6 *)
Lemma Exercise4_7d:  (* 6 *)
  inc (the_least r) (irreds r).
Lemma Exercise4_7e: sub E48P (substrate r).  (* 1 *)
Lemma Exercise4_7f: (* 1 *)
  irreds r = E48P +s1 (the_least r).
Lemma Exercise4_7g a:  inc a (substrate r) -> (* 4 *)
  inc (the_least r) (E47S r a).
Lemma Exercise4_7h a: (* 1 *)
   inc a (substrate r) -> nonempty  (E47S r a).
Lemma Exercise4_7i x: sub x (substrate r) -> (* 5 *)
  has_supremum r x.
Lemma Exercise4_7j a: inc a (substrate r) -> (* 4 *)
  gle r (supremum r (E47S r a)) a.
Lemma Exercise4_7k a b:  (* 2 *)
  gle r a b -> sub (E47S r a)  (E47S r b).
Lemma Exercise4_7l a: inc a (substrate r) -> (* 31 *)
  (supremum r (E47S r a)) = a.
Lemma Exercise4_7m a b:  (* 7 *)
  inc a (substrate r) -> inc b (substrate r) ->
  sub (E47S r a)  (E47S r b) -> gle r a b.
```

Statement (a) is equivalent to $x = \sup \mathrm{S}(x)$. Statement (b) is trivial: we have already shown that S is strictly increasing. The inclusion $\mathrm{S}(\inf(a,b)) \subset \mathrm{S}(a) \cap \mathrm{S}(b)$ follows. Conversely; if t is in $\mathrm{S}(a)$ and $\mathrm{S}(b)$, it is a lower bound of a and b, so that $t \le \inf(a,b)$ and $t \in \mathrm{S}(\inf(a,b))$.

```
Lemma Exercise4_7n a: inc a (substrate r) -> (* 6 *)
  exists S, [/\ finite_set S, nonempty S, sub S (substrate r),
  (forall x, inc x S -> sup_irred r x) &
  supremum r S = a].
Lemma Exercise4_7o a b: (* 6 *)
  inc a (substrate r) -> inc b (substrate r) ->
  (E47S r (inf r a b)) = (E47S r a) \cap (E47S r b).
Lemma Exercise4_7p: (* 12 *)
  let tg := (irreds r) in
   order_morphism (Lf (E47S r) (substrate r) (\Po tg))
   r (subp_order tg).
Lemma Exercise4_7q a: inc a (substrate r)  (* 1 *)
 -> sub (E47S r a) (irreds r).
```

¶ 8. (a) Let E be a distributive lattice (§ 1, Exercise 16). If a is irreducible in E (Exercise 7), show that the relation $a \le \sup(x,y)$ implies $a \le x$ or $a \le y$.

(b) Let E be a finite distributive lattice and let J be the set of its irreducible elements, ordered by the induced ordering. Show that the isomorphism $x \to \mathrm{S}(x)$ of E onto a subset of $\mathfrak{P}(\mathrm{J})$ defined in Exercise 7 (b) is such that $\mathrm{S}(\sup(x,y)) = \mathrm{S}(x) \cup \mathrm{S}(y)$. Deduce that if J^* is the ordered set obtained by endowing J with the opposite ordering, then E is isomorphic to the set $\mathscr{A}(\mathrm{J}^*, \mathrm{I})$ of increasing mappings of J^* into $\mathrm{I} = \{0,1\}$ (§ 1, Exercise 6).

(c) With the hypotheses of (b), let P be the set of elements of J other than the least element of E. For each $x \in \mathrm{E}$, let $y_1, \dots, y_k$ be the distinct minimal elements of the interval $]x, \rightarrow[$ in E; for each index i, let q_i be an element of P such that $q_i \not\in \mathrm{S}(x)$ and $q \in \mathrm{S}(y_i)$. Show that no two elements $q_1, \dots, q_k$ are comparable.

(d) Conversely, let $q_1, \dots, q_k$ be k elements of P, no two of which are comparable. Let $u = \sup(q_1, \dots, q_k)$ and let

$$v_i = \sup_{1 \le j \le k, j \ne i} (q_j) \qquad (1 \le i \le k).$$

Show that $v_i < u$ for $1 \le i \le k$. Let $x = \inf(v_1, \dots, v_k)$ and let

$$y_i = \inf_{1 \le j \le k, j \ne i} (v_j)$$

Show that $x < y_i$ for each index i, and deduce that the interval $]x, \rightarrow[$ has at least k distinct minimal elements.

Comment. The statement "E is isomorphic to the $\mathscr{A}(\mathrm{J}^*, \mathrm{I})$" is wrong, and we show two ways to correct it.

Solution. We continue to assume that r is the ordering of a non-empty finite set, and is a lattice. We add the assumption that E is distributive.

Assume $t \le \sup(x,y)$. By distributivity, $t = \inf(t, \sup(x,y)) = \sup(\inf(t,x), \inf(t,y))$; if t is irreducible, t is one of $\inf(t,x)$, $\inf(t,y)$, so that $t \le x$ or $t \le y$. Thus $t \in \mathrm{S}(x) \cup \mathrm{S}(y)$ and

$S(\sup(x,y)) = S(x) \cup S(y)$. By induction, if X is a finite set and t irreducible, then $t \le \sup(X)$ implies $t \le x$ for some $x \in X$.

Let a be the least element of E, denote by $\bar{X}$ the set $X - \{a\}$. Let $P = \bar{J}$. Let $p(X)$ be the property: X is a non-empty subset of J, and $x \in X$, $y \in J$ and $y \le x$ imply $y \in X$. Let $q(X)$ be the property: X is a subset of P, and $x \in X$, $y \in P$ and $y \le x$ imply $y \in X$. It is easily seen that $p(X)$ holds if and only if X has the form $S(x)$, and that $q(X)$ holds if and only if X has the form $\bar{S}(x)$.

```
Hypothesis dl3: distributive_lattice3 r.

Lemma Exercise4_8b a b: (* 8 *)
  inc a (substrate r) -> inc b (substrate r) ->
  (E47S r (sup r a b)) = (E47S r a) \cup (E47S r b).
Lemma Exercise4_8c1 t U: inc t (substrate r) -> (* 23 *)
  sup_irred r t ->  sub U (irreds r) ->  gle r t (supremum r U) ->
  nonempty U ->  exists2 x, inc x U & gle r t x.
Lemma Exercise4_8c U: sub U (irreds r) -> (* 12 *)
  (forall x y, inc y U -> sup_irred r x -> gle r x y -> inc x U) ->
   nonempty U ->
  U = (E47S r (supremum r U)).
Lemma Exercise4_8d:  (* 8 *)
  let p:= fun U => [/\ sub U (irreds r), nonempty U &
    (forall x y, inc y U -> sup_irred r x -> gle r x y -> inc x U)] in
  (forall x, inc x (substrate r) -> p (E47S r x)) /\
  (forall U, p U -> exists2 x, (inc x (substrate r)) & U = (E47S r x)).
Lemma Exercise4_8e: (* 26 *)
  let comp:= fun X => X -s1 (the_least r) in
  let p:= fun U => (sub U E48P /\
    (forall x y, inc y U -> inc x E48P -> gle r x y -> inc x U)) in
  (forall x, inc x (substrate r) -> p (comp (E47S r x))) /\
  (forall U, p U -> exists2 x, (inc x (substrate r)) & U = comp (E47S r x)).
```

(b) The first claim is obvious. The second is wrong. In fact, both functions S and $\bar{S}$ are isomorphisms of E into a subset of $\mathfrak{P}(J)$ or $\mathfrak{P}(P)$. Define $A_1 = \mathscr{A}(J^*, I)$ and $A_2 = \mathscr{A}(P^*, I)$. If a set satisfies property p or q, its characteristic function is in A_1 or A_2. By composition, we get a morphism $E \to A_1$ and $E \to A_2$ (its is injective and order-preserving, not necessarily surjective). If $f \in A_1$, then either f is the zero function, or $f(a) = 1$. Let A_3 be the subset of A_1 formed of all functions that are not constantly zero. The mapping that associates to each function f its restriction to P is an order isomorphism of A_3 to A_2. We show that E is isomorphic to A_2 and to A_3.

Note the special case $E = \emptyset$. Here A_1 has a single element, and $A_1 - \{0\}$ is empty, so that there is nothing to prove. If E is non-empty, then it has a least element, P and A_2 are well-defined.

We start with a bunch of definitions: the ordered sets J^* and P^*, the sets A_1 and A_2, the zero function and the set A_3. Note that I, considered as an ordered set, is the interval $[0,1]$ of **N**.

```
Definition E48I := doubleton \0c \1c.
Definition E48z := Lf (fun z => \0c) (irreds r) E48I.
Definition E48Ps := opp_order (induced_order r E48P).
Definition E48Js := opp_order (induced_order r (irreds r)).
Definition E48Io :=  Nint_cco \0c \1c.
Definition E48AJIo := increasing_mappings_order E48Js E48Io.
```

```
Definition E48APIo := increasing_mappings_order E48Ps E48Io.
Definition E48AJI := increasing_mappings E48Js E48Io.
Definition E48API := increasing_mappings E48Ps E48Io.
Definition E48AJImo:=
  induced_order E48AJIo ((substrate E48AJIo) -s1 E48z).
```

The following lemmas are trivial (except for the characterization of $\mathscr{A}(\mathrm{J}^*,\mathrm{I})$) and that of $\mathscr{A}(\mathrm{J}^*,\mathrm{I}) - \{0\}$.

```
Lemma Exercise4_8f K: sub K (substrate r) ->  (* 3 *)
   order_on (opp_order (induced_order r K)) K.

Lemma Exercise4_8g:   (* 2 *)
  order_on E48Js (irreds r) /\ order_on E48Ps E48P.
Lemma Exercise4_8h: order_on E48Io E48I. (* 7 *)
Lemma Exercise4_8iP K  (* 20 *)
   (o :=  opp_order (induced_order r K))
   (A:= increasing_mappings o E48Io):
  sub K (substrate r) ->
 forall f, inc f A <->
  [/\ function f, source f = K, target f =  E48I &
  (forall i j, inc i K -> inc j K -> gle r i j ->
     Vf f j = \1c -> Vf f i = \1c)].
Lemma Exercise4_8j K   (* 2 *)
  (o :=  opp_order (induced_order r K))
  (A:= increasing_mappings o E48Io)
  (no := increasing_mappings_order o E48Io):
  sub K (substrate r) ->order_on no A.
Lemma Exercise4_8k: (* 2 *)
  order_on E48AJIo E48AJI /\ order_on E48APIo E48API.
Lemma Exercise4_8l: inc E48z (substrate E48AJIo). (* 7 *)
Lemma Exercise4_8m:  (* 23 *)
 order_on E48AJImo (E48AJI -s1 E48z) /\
 forall f, inc f (substrate E48AJImo) <->
    (inc f E48AJI /\ Vf f (the_least r) = \1c).
```

We now show that the characteristic functions of $\mathrm{S}(x)$ and $\bar{\mathrm{S}}(x)$ are in A_3 and A_2.

```
Lemma  Exercise4_8n x: inc x (substrate r) ->  (* 13 *)
  inc (char_fun (E47S r x) (irreds r)) (substrate E48AJImo).
Lemma Exercise4_8o x (comp:= fun X => X -s1 (the_least r)):  (* 10 *)
  inc x (substrate r) ->
  inc (char_fun (comp (E47S r x)) E48P) E48API.
Lemma Exercise4_8p f: (* 8 *)
   function f -> target f = E48I ->
   char_fun (Vfi1 f \1c) (source f) = f.
Lemma Exercise4_8qP X Y Z:  sub X Z -> sub Y Z -> (* 16 *)
  ((sub X Y) <-> (forall x, inc x Z ->
    gle E48Io (Vf (char_fun X Z) x) (Vf (char_fun Y Z) x))).
Lemma Exercise4_8r: r \Is E48APIo.  (* 52 *)
Lemma Exercise4_8s: r \Is E48AJImo. (* 57 *)
```

For each $x \in \mathrm{E}$, we consider the set $\mathrm{A}(x)$ of all $a > x$ such that, whenever b satisfies $x < b \leq a$ we have $b = a$. We consider the property $p(\mathrm{K})$ that says that K is a subset of P and the conditions "$a \in \mathrm{K}$, $b \in \mathrm{K}$, $a \leq b$" imply $a = b$.

(c) We show: for each x, there is a set $K(x)$ satisfying p, and a bijection $f : A(x) \to K(x)$. We choose $f(y)$ to be some z such that $z \in S(y) - S(x)$. It exists since S is strictly increasing. Note that z cannot be the least element of E, thus is in P. We have $\sup(x, f(y)) = y$ since y is minimal. Thus f is the desired condition.

(d) We show: for any K that satisfies p, there is an x, such that $A(x)$ has at least as many elements as K. Consider $f : K \to E$ that maps x to $\sup(K - \{x\})$. Let $u = \sup K$. We have $\sup(x, f(x)) = u$. This implies $f(x) \le u$. We have $f(x) < u$, for otherwise we would have $x \le f(x)$. If K is not $\{x\}$, there is an element y of $K - \{x\}$ such that $x \le y$, contradicting $p(K)$. Otherwise, we get $u = x$. If $x \neq y$ we have $x \le f(y)$; we deduce $\sup(f(x), f(y)) = u$. This implies injectivity of f. Let L be the image of f. This set has the same number of elements as K.

We define now $g(x) = \inf(L - \{x\})$, $v = \inf(L)$. For any $x \in L$, we have $v = \inf(x, g(x))$. From this we deduce $v < g(x)$, for otherwise we would have $g(x) \le x$. This says $\sup(g(x), x) = x$. Using distributivity, we get $\inf\{k(y), y \neq x\} = x$, where $k(y) = \sup(y, x)$. Since x and y are distinct in L, we have $k(y) = v$, $\{k(y), y \neq x\} = \{v\}$, and $v = x$, absurd. Since the set of all z such that $v < z \le g(x)$ is non-empty, it has a minimal element, call it $h(x)$. We have $h(x) \in A(v)$. Note that h is injective; if $h(x) = h(y)$ we have $h(x) \le g(x)$ and $h(x) \le g(y)$, thus $h(x) \le \inf(g(x), g(y))$. But $x \neq y$ implies $\inf(g(x), g(y)) = \inf(L) = v$. Thus, $A(v)$ has at least as many elements as L.

```
Definition all_uncomp_inP K :=
  sub K E48P /\ forall x y, inc x K -> inc y K -> gle r x y -> x = y.

Definition minimal_in_int x a :=
   glt r x a /\  (forall b, glt r x b -> gle r b a -> b = a).

Definition minimals x :=  Zo (substrate r) (minimal_in_int x).

Lemma Exercise4_7c1:  exists a, greatest r a. (* 6 *)
Lemma Exercise4_7i1 x: sub x (substrate r) -> (* 5 *)
  has_infimum r x.
Lemma Exercise4_8u: infimum r emptyset = the_greatest r. (* 6 *)
Lemma distributive_rec x T: (* 57 *)
  inc x (substrate r) -> sub T (substrate r) ->
  sup r x (infimum r T) = infimum r (fun_image T (fun z => sup r z x)).

Lemma Exercise4_8t x: inc x (substrate r) -> (* 40 *)
  exists K, all_uncomp_inP K /\ K \Eq (minimalq x).
Lemma Exercise4_8v K: all_uncomp_inP K -> exists x, (* 151 *)
  inc x (substrate r) /\
   (cardinal K) <=c (cardinal (minimals x)).

End Irred_lattice.
```

¶ 9. A subset A of a lattice E is said to be a *sublattice* if for each pair (x, y) of elements of A, $\sup_E(x, y)$ and $\inf_E(x, y)$ belong to A.

(a) Let $(C_i)_{1 \le i \le n}$ be a finite family of totally ordered sets and let $E = \prod_{i=1}^{n} C_i$ be their product. Let A be a sublattice of E. Show that A cannot have more than n irreducible elements (Exercise 7) no two of which are comparable. (The proof is by *reductio ad absurdum.* Suppose that there exist $r > n$ irreducible elements $a_1, \dots, a_r$ in A, no two of which are comparable. Consider the elements $u = \sup(a_1, \dots, a_r)$ and

$$v_j = \inf_{1 \le j \le r, j \ne i} (a_j)$$

of A. By projecting onto the factors, show that $u = v_i$ for some index i, and hence that two of the a_i are comparable).

(b) Conversely, let F be a finite distributive lattice, let P be the set of irreducible elements of F other than the least element of F, and suppose that n is the greatest number of elements in a free subset of P (§ 1, Exercise 5). Show that F is isomorphic to a sublattice of a product of n totally ordered sets (Apply Exercise 5, which shows that P is the union of n totally ordered sets P_i with no elements in common. Let C_i be the totally ordered set obtained by adjoining a least element to P_i ($1 \le i \le n$). With each $x \in \mathrm{F}$ associate the family $(x_i)_{1 \le i \le n}$ where x_i is the least upper bound in C_i of the sets of elements of P_i which are $\le x$.)

Solution. We know that a product of lattices is a lattice. We show here that we can compute a sup and an inf, by taking the sup and inf components by components. The product of distributive lattices is distributive. The product of totally ordered sets is a distributive lattice.

```
Lemma setX_lattice_sup g (r :=  order_product g) x y (* 37 *)
   (z := sup r x y) (t := inf r x y):
  order_fam g -> (allf g lattice) ->
  inc x (substrate r) -> inc y (substrate r) ->
  [/\ inc z (substrate r),
    inc t (substrate r),
    (forall i, inc i (domain g) -> Vg z i = sup (Vg g i)(Vg x i) (Vg y i)) &
    (forall i, inc i (domain g) -> Vg t i = inf (Vg g i)(Vg x i) (Vg y i))].
Lemma setX_lattice_finite_sup (* 17 *)
   g (r :=  order_product g) E (z := supremum r E):
  order_fam g -> (allf g lattice) ->
  finite_set E -> sub E (substrate r) -> nonempty E ->
  (inc z (substrate r) /\
    forall i, inc i (domain g) -> Vg z i = supremum (Vg g i)(fun_image E (Vg^~ i))).
Lemma setX_dlattice g:  (* 26 *)
  order_fam g -> (allf g lattice) ->
   (allf  g  distributive_lattice1) ->
  distributive_lattice1 (order_product g).
Lemma setX_torder_dlattice g: (* 6 *)
  order_fam g -> (allf g total_order) ->
  (lattice (order_product g) /\  distributive_lattice1 (order_product g)).
```

(a) Let E be a product on n totally ordered sets. If $n = 0$, the product has a single element, which is obviously irreducible. Thus, we assume $n > 0$. Let B be a free subset formed of irreducible elements. We want to show Card(B) $\le n$; the proof is by contradiction. Thus, we assume that C is a subset of B with $n+1$ elements. In particular C is finite and nonempty. For each index i, there is an element $x \in \mathrm{C}$ such that x_i is maximal. Call this $f(i)$. There is at least one element x in C not of the form $f(i)$. Let $\bar{x}$ be the supremum of $\mathrm{C} - \{x\}$. We have $x \le \bar{x}$. Assume $x \le \sup(a, b)$ where a and b are in A; since x is irreducible, we have $x \le a$ or $x \le b$. Assume $b \in \mathrm{C}$ and $b \ne x$. Then the second case is excluded.

Assume $\mathrm{C} = \{y_0, y_1, \dots y_n\}$ where $y_0 = x$. Define x_k by $x_1 = y_1$ and $x_{k+1} = \sup(x_k, y_{k_1})$. We have $y_i \in \mathrm{C} - \{x\}$ and $x_i \in \mathrm{A}$. We have $\bar{x} = x_n$. By induction (starting with n) we have $x \le x_i$, thus $x \le y_1$; absurd.

```
Definition sublattice r A :=
   forall x y, inc x A -> inc y A -> (inc (sup r x y) A /\ inc (inf r x y) A).

Lemma sublattice_pr r A (rA:= induced_order r A): (* 32 *)
  lattice r -> sublattice r A -> sub A (substrate r)->
  [/\ lattice rA, substrate rA = A,
   (forall a b, inc a A -> inc b A -> sup r a b = sup rA a b) &
   (forall a b, inc a A -> inc b A -> inf r a b = inf rA a b) ].
Lemma sublattice_dr r A : (* 10 *)
  lattice r -> sublattice r A -> sub A (substrate r)->
  distributive_lattice1 r ->
  distributive_lattice3 (induced_order r A).
Lemma Exercise4_9a g n A  (r := order_product g): (* 205 *)
   order_fam g -> (allf g total_order) ->
   natp n -> cardinal (domain g) = n -> n <> \0c ->
   sub A (substrate r) -> sublattice r A ->
   forall B, free_subset r B ->
      (forall x, inc x B -> sup_irred (induced_order r A) x) ->
      sub B A -> cardinal B <=c n.
```

(b) We consider here a finite distributive lattice F, such that the maximum number of elements in a free subset of P has n elements. We assume F non empty, so that F has a least element e. If $n = 0$, it implies that P is empty. Thus $\mathrm{S}(x)$ is always $\{e\}$. This implies $\mathrm{F} = \{e\}$. The result is trivial in this case.

By exercise 5, we can partition P into n sets; to each set, we add e, this gives some totally ordered sets C_i. Let E be the product of these sets. To each $x \in \mathrm{F}$ we associate $\mathrm{F}_i(x)$ the set of all $y \in \mathrm{C}_i$ such that $y \le x$. This is a non-empty finite set in a totally ordered set, thus has a greatest element $f_i(x)$. Let $f(x)$ be the product of the $f_i(x)$. This gives us a function $f : \mathrm{F} \to \prod \mathrm{C}_i$.

We have $\mathrm{S}(x) = \bigcup \mathrm{F}_i(x)$ (we use here the fact that $n > 0$, since $e \in \mathrm{S}(x)$). Thus $x = \sup \bigcup(\mathrm{F}_i(x))$. By associativity, x is the supremum of the $\sup(\mathrm{F}_i(x))$. Note that the supremum of $\mathrm{F}_i(x)$ is $f_i(x)$, so that $x = \sup f_i(x)$. This shows in particular that f is injective. Let A be the image of f. Then f is a bijection $\mathrm{F} \to \mathrm{A}$; it is clearly an order isomorphism. It is compatible with inf and sup. More precisely, let $\bar{x}$ and $\bar{y}$ be two elements of A, and say $\bar{x} = f(x)$, $\bar{y} = f(y)$. Set $z = \sup(x, y)$. We have $f(z) = \sup(f(x), f(y))$. This relation says $\sup(\bar{x}, \bar{y}) \in \mathrm{A}$, and shows that A is a sublattice. Denote by z_i the quantity $f_i(z)$. By definition of the product, we have to show $z_i = \sup(x_i, y_i)$. We have obviously $x_i \le z_i$ and $y_i \le z_i$. Note that $z_i \le \sup(x, y)$. Since z_i is irreducible, it follows that $z_i \le x$ or $z_i \le y$. The case of inf is similar.

Discussion Let x be an element of the product, with components x_i. We say that x is i-small if $x_j = e$ for $j \neq i$. Note that two i-small elements are comparable. Moreover, if $x = \sup(a, b)$ and if x is i-small, so are a and b, so that x is the greatest of them. Thus x is irreducible. Let y_i (resp., z_i) be the element of the product whose j-th component is x_j if $j = i$ (resp., if $j \neq i$) and e otherwise. We have $x = \sup(y_i, z_i)$. Assume that x is not small. This means that there exists i such that $e_i \neq e$ and x is not i-small. In this case, none of y_i and z_i is x and x is not irreducible. Thus, the irreducible elements are the small ones. Given $n+1$ small elements, there exists i and two elements in the list which are i-small, thus comparable. Thus n is the least number of elements in a free subset of the irreducible elements of E. This argument shows that E is not too big. Assume x small, and not the least element of E. Then there is a unique x_i not equal to e, call it $g(x)$. This gives an injection from the set of irreducible elements of E into P. (is it bijective? how are f and g related?).

```
Lemma Exercise4_9b r (P := E48P r) n: (* 333 *)
  lattice r -> distributive_lattice1 r -> finite_set (substrate r) ->
  nonempty (substrate r) ->
  natp n -> order_width (induced_order r P) n ->
  exists g A f,
    let r' := (order_product g) in
    [/\ order_fam g, (allf g total_order), cardinal (domain g) = n,
    sub A (substrate r') & sublattice r' A /\
    order_isomorphism f r (induced_order r' A)].
```

¶ 10. (a) An ordered set E is isomorphic to a subset of a product of n totally ordered sets if and only if the graph of the ordering on E is the intersection of the graphs of n total orderings on E. (To show that the condition is necessary, show that if $\mathrm{F} = \prod_{i=1}^{n} \mathrm{F}_i$ is a product of n totally ordered sets, then the graph of the product ordering on F is the intersection of n graphs of lexicographic orderings on F.)

(b) An ordered set E is isomorphic to a subset of the product of two totally ordered sets if and only if the ordering Γ on E is such that there exists another ordering Γ' on E with the property that any two distinct elements of E are comparable with respect to exactly one of the orderings Γ and Γ'.

(c) Let A be a finite set of n elements. Let E be the subset of $\mathfrak{P}(\mathrm{A})$ consisting of all subsets $\{x\}$ and $\mathrm{A}-\{x\}$ as x runs through A. Show that n is the smallest integer m such that E, ordered by inclusion, is isomorphic to a subset of a product of m totally ordered sets (use (a)).

Solution.

(a) Let $\mathrm{T}_n(\mathrm{P})$ be the property that P is a product of n totally ordered sets, let $\mathrm{H}_n(\mathrm{E})$ be the property that E is isomorphic to a subset of some P such that $\mathrm{T}_n(\mathrm{P})$ holds. If $n = 0$, this says that E is small (empty or a singleton). Note that, if we add m singletons to P, we obtain some Q, isomorphic to P, and $\mathrm{T}_{n+m}(\mathrm{Q})$ holds, so that H_{n+m} holds as well. Let $\mathrm{C}_n(\mathrm{E})$ the the assumption that E is the intersection of n total orders; if we want this to be equivalent to $\mathrm{H}_n(\mathrm{E})$, it is important to allow some of these orders to be the same. Note that $\mathrm{C}_0(\mathrm{E})$ says that E is empty, this is not equivalent to $\mathrm{H}_0(\mathrm{E})$.

```
Definition Ex4_10_hyp r n:=
   exists g A f,
     [/\ order_fam g, (allf g total_order), cardinal (domain g) = n ,
     sub A (substrate (order_product g)) &
     order_isomorphism f r (induced_order (order_product g) A)].
Definition Ex4_10_conc r n :=
   exists g, [/\ order_fam g, (allf g total_order), cardinal (domain g) = n ,
   (forall i, inc i (domain g) -> substrate (Vg g i) = substrate r) &
   r = intersectionb g].
```

Fix $k < n$. Let g_k be the function that associates $i + k$ (modulo n) to i. This is a bijection.

```
Definition shift_mod_n n k :=
   Lf (fun i => ((i +c k) %%c n)) (Nint n)(Nint n).
```

```
Lemma shift_mod_n_ax n k: (* 3 *)
 natp n -> n <> \0c -> natp k ->
 forall i, inc i (Nint n) -> inc ((i +c k) %%c n) (Nint n).
Lemma shift_mod_n_vl n k i: (* 15 *)
  natp n -> n <> \0c -> k <c n ->  i<c n ->
  ((i +c k) %%c n) = Yo (i +c k <c n) (i +c k) ((i +c k) -c n).
Lemma shift_mod_n_fb n k:  (* 46 *)
  natp n -> n <> \0c -> k <c n ->
  bijection (shift_mod_n n k).
```

Assume E isomorphic to a subset Q of $\mathrm{P} = \prod \mathrm{E}_i$. We may assume that the index set I is the interval $[0, n-1]$. Let $k < n$. Consider $g_k : [0, n[\to \mathrm{I}$ as an order isomorphism. This makes I a well-ordered set, and k is the least element. Consider the lexicographic ordering P_k induced on P. This is a total ordering. Two elements comparable in P are comparable for this product. Assume now $x < y$ for any P_k. Assume $x_i \le y_i$ false. In particular $x_i \neq y_i$. The least index (for P_i) for which this holds is i, thus $x < y$ is false for P_i. We deduce that the ordering of P is the intersection of n total orderings. Note that these are all distinct if each P_i has at least two elements. It follows that the ordering of Q is the intersection of n total orderings. It suffices to transport these orderings back to E.

Conversely, consider now a family of total orders Γ_i and their intersection Γ. We assume that all these ordering have a common substrate E. Let $f : \mathrm{E} \to \mathrm{E}^n$ be the mapping such that $f(x)_i = x$, for any i. Let A be the image of f. Then f is obviously an order isomorphism (where the i-the factor is ordered by Γ_i).

```
Lemma Exercise4_10a r n: (* 180 *)
   order r -> natp n -> n <> \0c -> Ex4_10_hyp r n ->  Ex4_10_conc r n.
Lemma Exercise4_10b r n: (* 36 *)
  order r -> natp n -> n <> \0c -> Ex4_10_conc r n -> Ex4_10_hyp r n.
```

(b) We say that r' is orthogonal to r, if it has the same substrate and each pair of distinct elements is comparable by exactly one of the two orderings. In this case, it is plain that the union $r_1 = r \cup r'$ is a total ordering on the common substrate. Let $r_2 = r \cup \bar{r}'$ where $\bar{r}'$ is the opposite ordering of r_2. This is a second total ordering, and $r = r_1 \cap r_2$.

Assume now that E is isomorphic to a subset A of a product of two totally ordered sets. If $x \in \mathrm{E}$, we denote by x_1 and x_2 the two components in the product and consider "$x = y$ or $x_1 <_1 y_1$ and $y_2 <_2 x_2$". This is an orthogonal ordering.

```
Definition orthogonal_order r r' :=
   forall x y, inc x (substrate r) -> inc y (substrate r) -> x <> y ->
   exactly_one (ocomparable r x y) (ocomparable r' x y).

Lemma orthogonal_union_order r r': (* 62 *)
  order r -> order r' -> substrate r = substrate r' -> orthogonal_order r r' ->
  (total_order (r \cup r') /\ substrate (r \cup r') = substrate r).
Lemma orthogonal_union_inter r r'   (* 20 *)
  (r1 := r \cup r') (r2 := r \cup opp_order r'):
  order r -> order r' -> substrate r = substrate r' -> orthogonal_order r r' ->
   [/\ total_order r1, total_order r2,
    substrate r1 = substrate r, substrate r2 = substrate r & r = r1 \cap r2].
Lemma Exercise4_10c r: order r -> (Ex4_10_hyp r \2c <-> (* 87 *)
  exists r', [/\ substrate r' = substrate r, order r' & orthogonal_order r r']).
```

(c) Let E a finite set with n elements x_i. Set $y_i = \{x_i\}$ and $z_i = \mathrm{E} - y_i$. Let F_1 be the set of all singletons, F_2 the set of all complements of singletons and $\mathrm{F} = \mathrm{F}_1 \cup \mathrm{F}_2$. Elements of F_1 are non-comparable, as well as elements of F_2. In general y_i and z_i are non-comparable (the exception is when E is a singleton). If $i \neq j$ then $y_i \leq z_j$ (inequality is strict when E has at most three elements).

Assume that F is isomorphic to a subset of a product of m totally ordered sets. We want to show $n \leq m$, so that may assume $n \neq 0$. We may also assume $m \neq 0$ (if $x_i \in \mathrm{E}$, then y_i and z_i are two distinct elements of F). Thus we assume that F is the intersection of m total orderings $\leq_k$. Since F_1 is finite and non-empty, there is a greatest element i_k for $\leq_k$. If $n \leq m$ is false, there is some $x \in \mathrm{E}$ such that $\{x\}$ is not of the form i_k, whatever k. We have $\{x\} \leq_k \{i_k\}$ by definition of i_k. We have $\{i_k\} \leq \mathrm{E} - \{x\}$. Thus the same holds with $\leq_k$. Thus for all k, $\{x\} \leq_k \mathrm{E} - \{x\}$. Thus the same holds with $\leq$. Absurd.

Conversely, F is isomorphic to a subset of a product of n totally ordered sets. This is obvious if $n = 0$. Thus, we have to show that F is the intersection of n total orderings. We use the same argument as above: we use a bijection $[0, n[\to \mathrm{E}$, and compose with $i \mapsto i + k$ modulo n. This gives k total orderings on E. Fix k, and define $x \leq_k y$ by: if x and y are singletons, say $x = \{u\}$ and $y = \{v\}$, then x and y compare the same as u and v. If x and y are complements of singletons, say $x = \mathrm{E} - \{u\}$ and $y = \mathrm{E} - \{v\}$, then x and y compare the same as v and u. Finally, if $x = \{u\}$ and $y = \mathrm{E} - \{v\}$, then $x \leq y$ whenever $u \neq v$. If $u = v$, then $x \leq y$ unless u is the greatest element of E, case where $y \leq x$.

This is easily seen to be an ordering. It is obviously total. It clearly extends the ordering of F. Assume now that x and y are related by every $\leq_k$. If x and y are distinct singletons, then $x <_k y$ for some k (for which y is the greatest element), and $y <_l x$ for some l. If x and y and distinct complements of singletons, we have similar relations. If $x = \{u\}$ and $y = \mathrm{E} - \{u\}$, then $x < y$ in general, and $y < x$ is a particular case.

Note that F has $2n$ elements, unless $n = 2$; the previous construction works also in the case $n = 2$. The proof is long: 70 lines for the first part, 150 lines to show that the relation defined in the second part is a total ordering. and 120 lines to compute the intersection.

```
Lemma Exercise4_10d E n (* 500 *)
  (F := (fun_image E singleton) \cup (fun_image E (fun z => E -s1 z)))
  (r := sub_order F):
  natp n -> cardinal E = n ->
  ( Ex4_10_hyp r n /\
   forall m, natp m -> Ex4_10_hyp r m -> n <=c m).
```

¶ 11. Let A be a set and let $\mathfrak{R}$ be a subset of the set $\mathfrak{F}(\mathrm{A})$ of finite subsets of A. $\mathfrak{R}$ is set to be *mobile* if it satisfies the following condition:
(MO) If X, Y are two distinct elements of $\mathfrak{R}$ and if $z \in \mathrm{X} \cap \mathrm{Y}$, then there exists $\mathrm{Z} \subset \mathrm{X} \cap \mathrm{Y}$ belonging to $\mathfrak{R}$ such that $z \not\in \mathrm{Z}$.
A subset P of A is then said to be *pure* if it contains no set belonging to $\mathfrak{R}$.

(a) Show that every pure subset of A is contained in a maximal pure subset of A.

(b) Let M be a maximal pure subset of A. Show that for each $x \in \complement\mathrm{M}$ there exists a unique finite subset $\mathrm{E}_\mathrm{M}(x)$ of M such that $\mathrm{E}_\mathrm{M}(x) \cup \{x\} \in \mathfrak{R}$. Moreover, if $y \in \mathrm{E}_\mathrm{M}(x)$, the set $(\mathrm{M} \cup \{x\}) - \{y\}$ is a maximal pure subset of A.

(c) Let M, N be two maximal pure subsets of A, such that $N \cap \complement M$ is finite. Show that $\mathrm{Card}(M) = \mathrm{Card}(N)$. (Proof by induction on the cardinal of $N \cap \complement M$, using (b).)

(d) Let M, N be two maximal pure subsets of A, and put $N' = N \cap \complement M$, $M' = M \cap \complement N$. Show that $M' \subset \bigcap_{x \in N'} E_M(x)$. * Deduce that $\mathrm{Card}(M) = \mathrm{Card}(N)$ (by virtue of (c), we are reduced to the case where N′ and M′ are infinite; show then that $\mathrm{Card}(M') \le \mathrm{Card}(N')$). *

Notes. Assume $\emptyset \in \mathfrak{R}$. Then (MO) holds, for it suffices to take $M = \emptyset$, and there is no pure set. One could make this case interesting by saying that M is pure if it has no non-empty subset in $\mathfrak{R}$. This is the position taken by the authors of [17], they also impose M non-empty.

Example 1. Let $\mathfrak{R}$ be the set of all finite subsets of A with cardinal $> n$. Then M is pure if it has cardinal $\le n$ and maximal pure if and only its cardinal is n.

Example 2. Let $\mathfrak{R}$ be the set of all doubletons. It is mobile and non-empty free subsets are the singletons.

Example 3. If $\mathfrak{R}$ contains all singletons (for instance, $\mathfrak{R}$ could be the set of all non-empty subsets of A with cardinal $\le n$, where n is a non-zero integer). There is only one pure set, the empty set. According to [17], there is no pure set.

In [1], the author considers a variant, where each element of $\mathfrak{R}$ is minimal (for inclusion). Elements of $\mathfrak{R}$ are called *dependent sets*. This means, in example 1, that we consider only sets with $n+1$ elements, and in the case of example 3, of all singletons. This does not change the notion of "pure" and $E_M(x)$ is minimal.

We introduce here (MO) and the set $\overline{\mathfrak{R}}$ of minimal elements (for inclusion).

```
Definition mobile_r R :=  forall X Y, inc X R -> inc Y R -> X <> Y ->
  forall z, inc z (X\cap Y)
   ->  exists Z, [/\ inc Z R, sub Z (X \cup Y) & ~ (inc z Z)].
Definition min_incl_r R :=
   Zo R (fun z => forall x, inc x R -> sub x z -> z = x).
```

We start with examples and show that they are mobile.

```
Lemma Ex4_11_ex0 R: inc emptyset R -> mobile_r R. (* 2 *)
Lemma Ex4_11_ex1 A n (* 23 *)
   (R := Zo (\Po A) (fun z => finite_set z /\ n <c cardinal z)):
  natp n -> mobile_r R.
Lemma Ex4_11_ex2 A (* 20 *)
   (R := Zo (\Po A) (fun z => cardinal z = \2c)):
  mobile_r R.
Lemma Ex4_11_ex3a A R: (* 14 *)
  (forall x, inc x R -> sub x A) ->
  (forall x, inc x A -> inc (singleton x) R) ->
  mobile_r R.
Lemma Ex4_11_ex3b A n (* 6 *)
   (R := Zo (\Po A) (fun z => nonempty z /\ cardinal z <=c n)):
  natp n -> n <> \0c -> mobile_r R.
```

Assume all elements of $\mathfrak{R}$ finite. Then any element X of $\mathfrak{R}$ is a superset of an element of $\overline{\mathfrak{R}}$ (since X is finite, there is a subset of X in $\mathfrak{R}$ with least cardinal). We deduce that $\overline{\mathfrak{R}}$ is mobile if $\mathfrak{R}$ is mobile. The converse may be false: let $\mathfrak{R}$ be the set of all singletons, but $\{a\}$ is replaced by $\{a,b\}$; let X and Y be these two sets. Then $\overline{\mathfrak{R}}$ is the set of all singletons but $\{a\}$, thus is mobile. Let $X = \{b\}$ and $Y = \{a,b\}$; these sets are in $\mathfrak{R}$, and $b \in X \cap Y$. Assume $Z \subset X \cup Y$ and $b \notin Z$ says $Z = \emptyset$ or $Z = \{a\}$, thus $Z \notin \mathfrak{R}$, and $\mathfrak{R}$ is not mobile.

```
Lemma Ex4_11_minR_pr R:  (* 21 *)
   (forall x, inc x R -> finite_set x) ->
   (forall x, inc x R -> exists2 y, sub y x & inc y (min_incl_r R)).
Lemma Ex4_11_minR_mb R: (* 5 *)
   (forall x, inc x R -> finite_set x) ->
   mobile_r R -> mobile_r (min_incl_r R).
```

We introduce now the notion of "pure" and "maximal pure", and characterize them in the examples given above. In the case of example 1, if Card(A) ≤ n, then all subsets are pure and A is the only maximal pure set, otherwise pure set are those of cardinal n.

```
Definition pure R P:= forall x, sub x P -> ~(inc x R).
    pure R P /\ forall p, pure R p -> sub P p -> sub p A -> P = p.
Definition set_of_pure A R:= Zo (\Po A) (pure R).
Definition max_pure A R P:=
 [/\ sub P A, pure R P & forall p, pure R p -> sub P p -> sub p A -> P = p].

Lemma set_of_pureP A R x: (* 2 *)
   inc x (set_of_pure A R) <-> (sub x A /\ pure R x).
Lemma Ex4_11_ex0_pure A R : (* 2 *)
    inc emptyset R -> set_of_pure A R = emptyset.
Lemma Ex4_11_ex1_pure A n (* 53 *)
  (R := Zo (\Po A) (fun z => finite_set z /\ n <c cardinal z)):
  inc n Nat ->
  [/\  set_of_pure A R = Zo (\Po A) (fun z => cardinal z <=c n),
    (n <=c cardinal A ->
     forall M, sub M A -> (max_pure A R M <-> cardinal M = n)) &
    (cardinal A  <=c n ->
      (set_of_pure A R = \Po A
      /\ (forall M, max_pure A R M <-> M = A)))].
Lemma Ex4_11_ex2_pure A (* 27 *)
  (R := Zo (\Po A) (fun z => cardinal z = \2c)):
  (set_of_pure A R = Zo (\Po A) small_set
    /\ ( nonempty A ->
         forall M, max_pure A R M <-> (sub M A /\ singletonp M))).
Lemma Ex4_11_ex3_pure A R: (* 3 *)
  (forall x, inc x A -> inc (singleton x) R) ->
  (forall M, sub M A -> pure R M -> M = emptyset).
```

Let (MF) be the property that $\mathfrak{R}$ is mobile, and all its elements are finite subsets of A (we also assume $\emptyset \not\in \mathfrak{R}$, otherwise there is no pure set). If this property holds, then $\overline{\mathfrak{R}}$ satisfies (MF), and both sets have the same pure sets.

```
Definition mobile_ext R A:=
  [/\ (forall x, inc x R -> sub x A),
      (forall x, inc x R -> finite_set x),
       mobile_r R  &
      ~ (inc emptyset R)].

Lemma Ex4_11_minR_P2 R: (* 3 *)
   (forall x, inc x R -> finite_set x) ->
   (forall P, pure R P <-> pure (min_incl_r R) P).
Lemma Ex4_11_minR_pr3 A R (R' := min_incl_r R): (* 6 *)
  mobile_ext R A ->
  (mobile_ext R' A /\ set_of_pure A R =  set_of_pure A R').
```

Consider the following properties (1) $\mathscr{U}$ is a non-empty subset of $\mathfrak{P}(A)$, (2) $\mathscr{U}$ is inductive for inclusion, (3) if M and N are not in $\mathscr{U}$, but the intersection is, then for any x in the union, $M \cup N - \{x\}$ is not in $\mathscr{U}$, (4) any subset of an element of $\mathscr{U}$ is in $\mathscr{U}$ and (5) any subset of A not in $\mathscr{U}$ has a finite subset not in $\mathscr{U}$.

```
Definition pure_prop1 A S :=
  (nonempty S) /\ (forall x, inc x S -> sub x A).
Definition pure_prop2 S :=
  (inductive (sub_order S))
Definition pure_prop3 A S :=
  (forall M N, sub M A -> sub N A -> ~ inc M S ->  ~ inc N S ->
    inc (M \cap N) S ->
    forall x, inc x (M \cup N) -> ~ (inc ((M \cup N) -s1 x) S)).
Definition pure_prop4  S :=
  (forall x y, inc x S -> sub y x -> inc y S).
Definition pure_prop5 A S :=
  forall x, sub x A -> ~ (inc x S) ->
     exists y, [/\ sub y x, finite_set y & ~ (inc y S)].
```

Consider a mobile set $\mathfrak{R}$, and its set of pure elements $\mathscr{U}$. It is inductive for inclusion. In fact, consider a totally ordered family U of $\mathscr{U}$. Let's show that its union is pure; so assume that the union contains a set F in $\mathfrak{R}$. Each element of F is in some $V \in U$, the set of these V has a greatest element (since F is finite), so that V is pure, absurd.

Properties (1), (4), (5) are trivial. Consider (3). Let M and N not in $\mathscr{U}$, so that there are subsets M' and N' in $\mathfrak{R}$. Assume $M \cap N \in \mathscr{U}$ and $x \in M \cup N$. If x fails to belong to M' and N', the result is clear. The result is also clear if $M' = N'$; otherwise it follows from (MO).

```
Lemma pure_properties_res1 A R: (* 34 *)
  mobile_ext R A -> (pure_prop2 (set_of_pure A R)).
Lemma pure_properties_res2 A R (S:= set_of_pure A R):  (* 40 *)
  mobile_ext R A ->
  [/\ pure_prop1 A S, pure_prop2 S, pure_prop3 A S,
  pure_prop4 S &  pure_prop5 A S].
```

The author of [1] claims that "(1), (2) and (3)" is equivalent to (DMF): $\mathscr{U}$ is the set of pure sets for some minimal $\mathscr{R}$ satisfying (MF). This is not quite exact: we know that (4) is necessary and we give an example where (1), (2), (3) and (5) holds, while (4) is false (take for $\mathscr{U}$, the set of all doubletons).

Note that (5) is a consequence of (2) and (4). Let M be a subset of A not in $\mathscr{U}$. Let c be the least cardinal such that there is a subset N of M of cardinal c not in $\mathscr{U}$. Assume by contradiction that c is infinite, and let $f : c \to N$ be a bijection. To each ordinal $x < c$ we associate the set T_x of all $f(y)$ such that $y < x$. Since c is infinite, it is a limit ordinal, so that N is the union of all T_x. On the other hand, T_x has cardinal x, and this is strictly less than c. Thus $T_x \in \mathscr{U}$. The set of all T_x is obviously totally ordered by inclusion. Now (2) says that there is an upper bound $T \in \mathscr{U}$. It follows $N \subset T$; by (4), we get $N \in \mathscr{U}$, absurd.

Assume that (1), (2), (3) and (4) hold. By the previous argument, we may assume (5). Let $\mathscr{R}$ be the set of all subsets M of A not in $\mathscr{U}$ such that if N is a subset of M, and $N \notin S$, then $N = M$. Obviously, if $M \in \mathscr{R}$ and $N \in \mathscr{R}$ and $N \subset M$, then $N = R$; thus all elements of $\mathscr{R}$ are minimal for inclusion. From (1) and (4), it follows $\emptyset \in \mathscr{U}$, thus $\emptyset \notin \mathscr{R}$. Assume $M \subset A$ and $M \notin \mathscr{U}$. From (5), there is a finite subset N of M not in $\mathscr{U}$; if we choose N of minimal cardinal, we get $N \in \mathscr{R}$. Assume $M \in \mathscr{R}$, the finite subset of M not in $\mathscr{U}$ has to be M, so that M is finite.

Assume $M \in \mathscr{U}$, $N \subset M$; by (4), $N \in \mathscr{U}$ and $N \not\in \mathscr{R}$, so that M is pure. Assume M pure, $M \not\in \mathscr{U}$; there is a subset N of M in $\mathscr{R}$, absurd. Thus $\mathscr{U}$ is the set of pure sets of $\mathscr{R}$. Finally (3) says that $\mathscr{R}$ is mobile.

```
Lemma pure_properties_res3: exists A S, (* 14 *)
  [/\ pure_prop1 A S, pure_prop2 S, pure_prop3 A S, pure_prop5 A S &
   ~  (pure_prop4 S)].
Lemma pure_properties_res4 A S: (* 74 *)
  (pure_prop2 S) -> (pure_prop4 S) ->  (pure_prop5 A S).
Lemma pure_properties_res5 A S: (* 50 *)
  pure_prop1 A S -> pure_prop3 A S -> pure_prop4 S -> pure_prop5 A S ->
  exists R,
    [/\ mobile_ext R A,
        S = (set_of_pure A R) &
        (forall x z, inc x R -> inc z R -> sub x z -> z = x)].
Lemma pure_properties_res6 A S: (* 2 *)
 pure_prop1 A S -> pure_prop2 S -> pure_prop3 A S -> pure_prop4 S ->
 exists R,
    [/\ mobile_ext R A,
        S = (set_of_pure A R) &
      (forall x z, inc x R -> inc z R -> sub x z -> z = x)].
```

In [1], it is said that, if $\mathscr{R}$ is a set of non-empty finite subsets of A, minimal for inclusion, then (MO) is equivalent to (MO'): if E and F are distinct members of $\mathscr{R}$, if $x \in E \cap F$ and $y \in E - F$, then $E \cup F$ has a subset belonging to $\mathscr{U}$, which contains y but fails to contain x.

It is clear that (MO') implies (MO). Conversely, assume (MO) holds, (MO') fails. There are E and F, distinct members of $\mathscr{R}$, $x \in E \cap F$ and $y \in E - F$, such that $E \cup F$ has no subset belonging to $\mathscr{U}$, which contains y but fails to contain x, and $\text{Card}(E \cup F)$ is minimal (note that $E \cup F$ is finite). By (MO), there is some $G \in \mathscr{R}$, a subset of $E \cup F$ that fails to contain x. If it contains y, we win. Otherwise $G - E$ is nonempty (since elements of $\mathscr{U}$ are minimal, $G \subset E$ says $G = E$, thus $y \in G$). If $z \in G - E$, by minimality, there is H, a subset of $G \cup F$ in $\mathscr{R}$, containing x but not z. Again, by minimality, there is J, a subset of $E \cup H$ in $\mathscr{R}$, containing y but not x.

```
Definition mobile_alt R :=
  forall E F, inc E R -> inc F R -> E <> F ->
  forall x y, inc x (E \cap F) -> inc y (E -s F) ->
  exists G, [/\ inc G R, sub G (E \cup F), inc y G & ~ (inc x G)].

Lemma pure_properties_res7 A R: (* 93 *)
 (forall x, inc x R -> sub x A) ->
  (forall x, inc x R -> finite_set x)  -> ~ (inc emptyset R) ->
  (forall x z, inc x R -> inc z R -> sub x z -> z = x) ->
  (mobile_r R <-> mobile_alt R).
```

The following theorem is proved in [1]. Consider n sets $A_i \in \mathscr{R}$, and a subset B of A with r elements ($r < n$). We can eliminate elements of B, as follows. Let $m = n - r$. We shall assume that no A_i is a subset of the union of the A_j with lower index. There are m sets C_i, such that no C_i is a subset of the union of the other C_j, and C_i is a subset of the complement of B in the union of A_i. The proof is by induction on r.

Assume first $r = 0$, thus B empty. We chose x_i in A_i, not in A_j for $j < i$. We define C_{ij} by induction, to be A_j for $i = 0$, and $C_{i+1,j}$ is some subset of $C_{ii} \cup C_{ij}$ containing x_j but not x_i (if $x_i \not\in C_{ij}$ we take C_{ij}, otherwise apply (MO')). We prove (by induction on i) that C_{ij} is a subset of $A_j \cup \bigcup_{k<i} A_k$ containing x_j, but not x_k for $k < i$ or $k > j$. Thus C_{ii} is a subset of $\bigcup A_k$ in $\mathscr{R}$, contains x_i but no other x_k, thus is the desired family.

So we assume that B has $r' = r-1$ elements, x is an additional element, and for any family A_i (with $m+r'$ elements), there is a family C_i (with m elements), satisfying some conditions. Consider now a family A_i with r elements. Apply to $(A_i)_i$ the result of the case $r = 0$. We get a sequence A'_i of sets satisfying a stronger condition, whose union is a subset of the union of the A_i. It suffices to take this sequence instead of the original one. Assume first $x \notin \bigcup A_i$. We discard the last A_i, and proceed by induction. In the second case, we assume $x \in A_k$, we select y_i in A_i, not in A_j for $j \neq i$. The same argument as above shows that there are sets $A'_i \in \mathscr{U}$ (for $i \neq k$), subsets of $A_i \cup A_k$, containing y_i and not x. It suffices to proceed by induction in this family.

```
Definition ppr8_hyp R f n:=
  [/\ fgraph f, domain f = Nint n,
  (forall i, i <c n -> inc (Vg f i) R) &
  (forall i, i<c n ->
       ~ (sub (Vg f i) (unionb (restr f (Nint i)))))].

Definition ppr8_conc R B f g  m:=
  [/\ fgraph g, domain g = Nint m,
  (forall i, i <c m -> inc (Vg g i) R),
  (forall i, i <c m -> sub (Vg g i) (unionb f -s B)) &
  (forall i, i<c m ->
       ~ (sub (Vg g i) (unionb (restr g ((Nint m) -s1 i)))))].
Lemma pure_properties_res8 A R:  (* 277 *)
   mobile_ext R A ->
   (forall x z, inc x R -> inc z R -> sub x z -> z = x) ->
   forall r m, natp r -> natp m -> m <> \0c ->
     forall f B, ppr8_hyp R f (m +c r) -> cardinal B = r ->
     exists g, ppr8_conc R B f g  m.
```

Consider now a family of n elements A_i, not in $\mathscr{U}$, but the intersection of A_i and the union of the A_k ($k < i$) is in $\mathscr{U}$. For any set B with less than n elements, the complement of B is the union of the A_i is not in $\mathscr{U}$. Proof: consider A'_i, a subset of A_i in $\mathscr{R}$, and apply the previous result; there is at least one set in $\mathscr{R}$, which is a subset of the complement, contradicting purity.

```
Lemma pure_properties_res9 A R (U := set_of_pure A R): (* 48 *)
  mobile_ext R A ->
  (forall x z, inc x R -> inc z R -> sub x z -> z = x) ->
  forall n f B,
    natp n ->
    cardinal B <c n ->
    fgraph f ->  domain f = Nint n ->
    (forall i, i <c n -> sub (Vg f i) A) ->
    (forall i, i <c n -> ~ inc (Vg f i) U) ->
    (forall i, i <c n ->
      inc ((Vg f i) \cap (unionb (restr f (Nint i)))) U) ->
    ~ (inc ((unionb f) -s B) U).
```

(a) Let's come back to Bourbaki. We assume that we have a mobile set. We know that the set of pure elements is inductive; it follows that each pure subset is contained in a maximal pure set.

```
Section Exercice4_11.
```

```
Variables A R: Set.
Hypothesis mnr: mobile_ext R A.

Lemma Exercise4_11a: (* 1 *)
  inductive (sub_order (set_of_pure A R)).
Lemma Exercise4_11b x: sub x A -> pure R x ->  (* 10 *)
  exists2 y, sub x y & max_pure A R y.
```

(b) Let M be pure maximal, $x \not\in$ M. There is a subset Y of M $\cup \{x\}$ which is in $\mathfrak{R}$ (by maximality) and $x \in$ Y (since M is pure). Then Y $-\{x\}$ is a subset of M, such that adjoining x yields an element of $\mathfrak{R}$. Uniqueness: assume X and Y are distinct subsets of M such that X $\cup \{x\}$ and Y $\cup \{x\}$ are in $\mathfrak{R}$. By mobility, there is Z, which contradicts purity of M.

```
Lemma Exercise4_11c M x: (* 35 *)
  max_pure A R M -> inc x (A -s M) ->
  exists !z, [/\ inc z (\Po A), sub z M & inc (z +s1 x) R].
```

Denote by $\mathrm{E_M}(x)$ the quantity introduced above. This is minimal in $\mathfrak{R}$. If $y \in \mathrm{E_M}(x)$ then T = M $\cup \{x\} - \{y\}$ is pure maximal. Consider a subset P of T that is in $\mathfrak{R}$. By purity of M, it contains x. Consider mobility of P and $\mathrm{E_M}(x) \cup \{x\}$; these sets are distinct (consider y) and contain x; we get a Z contradicting purity of M. Assume T $\subset$ S, where S is pure. If $y \in$ S, then M $\cup \{y\}$ is pure, contradicting maximality. Assume T $\neq$ S; then there is $v \in$ S $-$ T. and T$'$ = M $\cup \{x\} - \{y\} \cup \{v\}$ is pure (it is a subset of S). Note that $y \neq v$ and $v \not\in$ M. We deduce $\mathrm{E_M}(v) \cup \{v\} \in \mathfrak{R}$. By purity of T$'$ this implies $y \in \mathrm{E_M}(v) \cup \{v\}$. Consider mobility of $\mathrm{E_M}(v) \cup \{v\}$ and $\mathrm{E_M}(x) \cup \{x\}$ and the common element y. This contradicts purity of T$'$.

```
Definition Ex4_11EM M x := select   (fun z => (sub z M /\ inc (z +s1 x) R))
  (\Po A).

Lemma Exercise4_11d M x: max_pure A R M -> inc x (A -s M) -> (* 7 *)
  (sub (Ex4_11EM M x) M /\ inc ((Ex4_11EM M x) +s1 x) R).
Lemma Exercise4_11e M x: max_pure A R M -> inc x (A -s M) -> (* 16 *)
  inc (Ex4_11EM M x +s1 x) (min_incl_r R).
Lemma Exercise4_11f M x y: max_pure A R M -> inc x (A -s M) -> (* 71 *)
  inc y (Ex4_11EM M x) -> max_pure A R ((M +s1 x) -s1 y).
```

(c) Let's show that two maximal pure sets have the same cardinal. Let C = M $\cap \complement$N = M $-$ N. Assume C empty. Then M $\subset$ N, and M = N by maximality. If C is finite, we proceed by induction on C. Assume C = $\mathrm{C_1} \cup \{a\}$. The set $\mathrm{E_N}(a)$ is not a subset of M (for otherwise, the same would hold for $\mathrm{E_N}(a) \cup \{a\}$, which is in $\mathfrak{R}$, contradicting purity of M. Thus, there is $b \in \mathrm{E_N}(a)$ not in M. Let N$'$ = N $\cup \{a\} - \{b\}$. This set is pure maximal and has the same cardinal as N. We conclude by M $-$ N$'$ = $\mathrm{C_1}$..

(d) We have M $-$ N $\subset \bigcup_{x \in \mathrm{N-M}} \mathrm{E_M}(x)$. (why ?)

Assume N $-$ M infinite. Then Card($\mathrm{E_M}(x)$) $\le$ Card(N $-$ M) since each $\mathrm{E_M}$ is finite. Thus $\sum_{x \in \mathrm{N-M}}$ Card($\mathrm{E_M}(x)$) $\le$ Card(N $-$ M), so that Card(M $-$ N) $\le$ Card(N $-$ M). If one of N $-$ M and M $-$ N is finite, we know that M and N are equipotent. Otherwise, we deduce Card(M $-$ N) = Card(N $-$ M) and this implies that M and N are equipotent.

```
Lemma  Exercise4_11g M N: max_pure A R M -> max_pure A R N -> (* 45 *)
  finite_set (M -s  N) -> M \Eq N.
Lemma  Exercise4_11h M N:
```

```
  max_pure A R M -> max_pure A R N ->
  sub (M -s N)  (unionb (L (N -s M) (fun z => (Ex4_11EM M z)))).
Admitted.
Lemma  Exercise4_11i M N: max_pure A R M -> max_pure A R N -> (* 34 *)
  M \Eq N.
```

13.6 Section 5

1. Prove the formula

$$\sum_{k=q+1}^{n-p+q+1} \binom{n-k}{p-q-1}\binom{k-1}{q} = \binom{n}{p},$$

where $p \leq n$ and $q < p$ (generalize the argument of no. 8, Corollary to Proposition 14).

Discussion. This formula has been shown above as (6.57) by induction.

The Bourbaki argument is the following: Let X be any subset with p elements of $[0,n[$, and $k-1$ the value of the $(q+1)$-th element of X (in increasing order). Let X_a be the subset of elements of X that are $< k$ and X_b the subset of elements of X that are $> k$. All we need to do is to count the number of sets X_a and X_b.

Solution. Let $\mathfrak{B}$ be the set of all subsets X of $[0,n[$ with cardinal p. Let $f : \mathfrak{B} \to F$ be a function, $\mathfrak{B}_i$ the set of all $x \in \mathfrak{B}$ such that $f(x) = i$. This is a partition of $\mathfrak{B}$ so that $\binom{n}{p} = \text{Card}(\mathfrak{B}) = \sum \text{Card}(\mathfrak{B}_i)$.

Let $X \in \mathfrak{B}$. For any integer x, let $g(x)$ the cardinal of the intersection of X and $[0,x[$. We have $g(x+1) = g(x) + \chi(x)$, where χ is the characteristic function of X. In particular, if $x \in X$ and $x < y$, we have $g(x) < g(y)$. Thus, there is at most one $x \in X$ such that $g(x) = q$. There is a least integer such that $g(x) > q$ (since $g(n) > q$). It is non-zero, since $f(0) = 0$.

Let $f(X)$ be the unique integer x such that $g(x) = q$. Then $f(X) \in X$ and $X \cap [0, f(X)[$ has q elements. This condition obviously shows $f(X) \geq q$. Counting the number of elements of X not in $[0, f(X)[$ gives $f(X) \leq n-p-q$. Let $I = [q, n-p-q]$. We deduce that $\binom{n}{p}$ is the sum for $i \in I$ of the cardinals of $\mathfrak{B}_i$, the set of elements $X \in \mathfrak{B}$ such that $f(X) = i$.

Fix $i < n$. For any subset X of $[0,i[$ with q elements, and any subset Y of $[0,n[- [i+1,n[$ with $p-q-1$ elements, we consider $g(X,Y) = X \cup Y \cup \{i\}$. This is a subset of $[0,n[$ with q elements. It is in $\mathfrak{B}_i$, and all elements of $\mathfrak{B}_i$ are of this form.

```
Lemma Exercise5_1 p n q : (* 193 *)
  natp n ->  p <=c n -> q <c p ->
 binom n p = csumb (Nintcc  q (n -c p +c q)) (fun k =>
    binom (n-c (csucc k)) (p -c (csucc q)) *c binom k q).
```

---

2. If $n \geq 1$, prove the relation

$$\binom{n}{0} - \binom{n}{1} + \binom{n}{2} - \ldots + (-1)^n \binom{n}{n} = 0$$

(Define a one-to-one correspondence between the set of subsets of $[1,n]$ which have an even number of elements, and the set of subsets of $[1,n]$ which have an odd number of elements. Distinguish between the cases n even and n odd.)

Discussion. This is relation (6.64) above with $k = n+1$, proved in $\mathbf{Z}$. If we want to prove something on $\mathbf{N}$, it has to be $A = B$ where $A = \sum_{k\in I_{ne}} \binom{n}{k}$ and $B = \sum_{k\in I_{no}} \binom{n}{k}$, where I_{ne} and I_{no} are the sets of integers $\le n$ that are respectively even and odd. This follows from (6.55) and (6.56). Let E_k be the set of subsets of E with cardinal k. Its cardinal is $\binom{n}{k}$ when E is of cardinal n. Let E_e and E_o be the unions of the E_k for k even or odd. The relation $A = B$ says that E_e and E_o have the same cardinal. If $n = 0$, the formula says $1 = 0$, hence is false.

We introduce the sets E_o, E_e, I_{ne} and I_{no}; we also introduce E'_e the set of non-empty elements of E_e.

```
Definition even_card_sub I := Zo (\Po I) (fun z => evenp (cardinal z)).
Definition even_card0_sub I := even_card_sub I -s1 emptyset.
Definition odd_card_sub I := Zo (\Po I) (fun z => oddp (cardinal z)).
Definition Nintc_even p := Zo (Nintc p) evenp.
Definition Nintc_odd p := Zo (Nintc p) oddp.
```

Solution. We first show that E_e and E_o are equipotent. If the cardinal of E is odd, then $K \mapsto E - K$ is a bijection. Otherwise, fix $x \in E$. If $x \notin K$, we consider $E - (K \cup \{x\})$ otherwise $E - (K - \{x\})$. This gives the desired bijection.

```
Lemma Exercise5_2 E (* 57 *)
  finite_set E -> nonempty E -> (even_card_sub E) \Eq (odd_card_sub E).
Lemma Exercise5_2_alt n: natp n -> n <> \0c ->  (* 46 *)
  csumb (Nintc_even n) (binom n) = csumb (Nintc_odd n) (binom n).
```

---

3. Prove the relations

$$\binom{n}{0}\binom{n}{p} + \binom{n}{1}\binom{n-1}{p-1} + \binom{n}{2}\binom{n-2}{p-2} + \cdots + \binom{n}{p}\binom{n-p}{0} = 2^p\binom{n}{p},$$

$$\binom{n}{0}\binom{n}{p} - \binom{n}{1}\binom{n-1}{p-1} + \binom{n}{2}\binom{n-2}{p-2} - \cdots + (-1)^p\binom{n}{p}\binom{n-p}{0} = 0.$$

(Consider the subsets of p elements of $[1, n]$ which contain a given subset of k elements ($0 \le k \le p$), and use Exercise 2 for the second formula.)

Comment. We have seen these relations as (6.72) and (6.73). The generic coefficient in these formulas is

$$\binom{n}{k}\binom{n-k}{p-k} = \binom{p}{k}\binom{n}{p} \qquad (k \le p)$$

(this follows from (6.22), the direct proof is straightforward, but a lit long). After factoring out the expression $\binom{n}{p}$, we obtain an expression independent of n.

Solution. These formulas are (6.72) and (6.73).

Let E be a set with n elements, $k \le p$, and X_k the set of all pairs (B, C) of subsets of E, where B has cardinal k, C has cardinal $p - k$, B and C are disjoint (this condition is written here $C \subset E - B$). This set has cardinal $\binom{n}{k}\binom{n-k}{p-k}$. The sets X_k are mutually disjoint, so that the LHS of the first equation is the cardinal of the union of the X_k. Let $A = B \cup C$. The union of the X_k is equipotent to the union of the sets of pairs (A, B), where A has cardinal p. The cardinal of this set is the RHS of the first formula.

Let's notice that the generic term in the sum is $\binom{p}{k}\binom{n}{p}$ (if $p > n$ it is zero, otherwise, it suffices to convert the binomial coefficients into factorials and simplify). The second factor is independent of k, and can be factored out. What remains is the sum of the $\binom{p}{k}$, this is 2^p. By the previous exercise, the alternated sum is zero.

```
Definition Exercise5_3V n p k := (binom n k) *c (binom (n -c k) (p -c k)).

Lemma Exercise5_3a E n p k
  (X := Zo (\Po E \times \Po E)
          (fun z => [/\  cardinal (P z) = k,
          cardinal (Q z) = p -c cardinal (P z) & sub (Q z) (E -s P z)])):
  natp n -> natp p -> k <=c p -> cardinal E = n ->
  cardinal X = Exercise5_3V n p k. (* 38 *)
Lemma Exercise5_3b n p: natp n -> natp p ->
  csumb (Nintc p) (Exercise5_3V n p)  = \2c  ^c p *c binom n p. (* 79 *)
Lemma Exercise5_3c n k p: natp n -> natp p -> k <=c p -> (* 42 *)
   (binom n k) *c (binom (n -c k) (p -c k)) = (binom p k) *c (binom n p).
Lemma Exercise5_3d n p: natp n -> natp p ->
  csumb (Nintc p) (Exercise5_3V n p) = \2c ^c p *c binom n p.  (* 7 *)
Lemma Exercise5_3e n p: natp n -> natp p -> p <> \0c ->
    csumb (Nintc_even p) (Exercise5_3V n p)
  = csumb (Nintc_odd p) (Exercise5_3V n p).  (* 15 *)
```

---

4. Prove Proposition 15 of no. 8 by defining a bijection of the set of mappings u of $[1, h]$ into $[0, n]$ such that

$$\sum_{i=1}^{h} u(x) \leq n$$

onto the set of strictly increasing mappings of $[1, h]$ into $[1, n+h]$.

Discussion. This theorem has already been proved in the main text.

---

5. * (a) Let E be a distributive lattice and let f be a mapping of E into a commutative semigroup M (written additively) such that

$$f(x) + f(y) = f(\sup(x, y)) + f(\inf(x, y))$$

for all x, y in E. Show that for each finite subset I of E, we have

$$f(\sup(\mathrm{I})) + \sum_{2n \leq \mathrm{Card}(\mathrm{I})} \Big(\sum_{\mathrm{H} \subset \mathrm{I}, \mathrm{Card}(\mathrm{H}) = 2n} f(\inf(\mathrm{H})) \Big) = \sum_{2n+1 \leq \mathrm{Card}(\mathrm{I})} \Big(\sum_{\mathrm{H} \subset \mathrm{I}, \mathrm{Card}(\mathrm{H}) = 2n+1} f(\inf(\mathrm{H})) \Big)$$

(By induction on Card(I).) $_*$

(b) In particular, let A be a set, let $(\mathrm{B}_i)_{i \in \mathrm{I}}$ be a finite family of finite subsets of A, and let B be the union of the B_i. For each subset H of I, put $\mathrm{B_H} = \bigcap_{i \in \mathrm{H}} \mathrm{B}_i$. Show that

$$\mathrm{Card(B)} + \sum_{2n \leq \mathrm{Card}(\mathrm{I})} \Big(\sum_{\mathrm{Card}(\mathrm{H}) = 2n} \mathrm{Card}(\mathrm{B_H}) \Big) = \sum_{2n+1 \leq \mathrm{Card}(\mathrm{I})} \Big(\sum_{\mathrm{Card}(\mathrm{H}) = 2n+1} \mathrm{Card}(\mathrm{B_H}) \Big).$$

Discussion. Assume that M is some set and $g : \mathrm{M}\times\mathrm{M} \to \mathrm{M}$ is a function satisfying $g(x, y) = g(y,x)$ and $g(x, g(y,z)) = g(g(x, y),z)$. Given a non-empty sequence x_0, x_1, etc., we define $\mathrm{X}_n = \sum_{i<n} x_i$ by induction as: $\mathrm{X}_1 = x_0$ and $\mathrm{X}_{n+1} = g(x_n,\mathrm{X}_n)$. On can show that $\sum_{i<n} x_i = \sum_{i<n} x_{\sigma(i)}$ whenever σ is a permutation of $[0, n-1]$. If $q(x)$ is some property, and if the set of all x satisfying q is the finite nonempty set $\mathrm{Y} = \{x_0,\ldots,x_{n-1}\}$, if p is some function, $y_i = p(x_i)$ we denote by $\sum_{x\in\mathrm{Y}} p(x)$ or by $\sum_{q(x)} p(x)$ the quantity $\sum_{i<n} y_i$. One can show that, if Y is the disjoint union of Y_i, then (general associativity) $\sum_i \sum_{x\in\mathrm{Y}_i} x = \sum_{x\in\mathrm{Y}} x$.

By associativity, the formula to be shown is also

$$f(\sup(\mathrm{I})) + \sum_{\mathrm{H}\subset\mathrm{I},\mathrm{Card(H)\ even}} f(\inf(\mathrm{H})) = \sum_{\mathrm{H}\subset\mathrm{I},\mathrm{Card(H)\ odd}} f(\inf(\mathrm{H})).$$

Note: as stated above, the formula is wrong; we have to assume H non-empty. Note that, if I is finite, and has at least two elements, then each sum has a finite non-zero number of terms; each set H is non-empty and finite so that inf(H) exists, and the formula makes sense. If I has two elements, say x and y, there is a unique non-empty subset H with even cardinal, namely I. Thus, the LHS is $f(\sup(x, y)) + f(\inf(x, y))$. There are two non-empty subsets with odd cardinal, namely $\{x\}$ and $\{y\}$. Thus, the RHS is $f(x) + f(y)$ and the formula holds. Now, if I has a single element, say x, the LHS is $f(x) + s$, where s is the empty sum, and the RHS is $f(x)$. The formula holds, provided that M has an element e such that $g(x, e) = x$ whenever $x \in \mathrm{M}$. Note also that if I is empty, then sup(I) is in general undefined.

We do not introduce semigroups here. We just assume that $f(x)$ is a cardinal.

The proof is as follows. Let I be a non-empty finite subset of E and $z \in \mathrm{E}$. We apply the assumption to sup(I) and z. This gives $f(\sup(\mathrm{I}))+ f(z) = f(\sup(\mathrm{I}\cup\{z\}))+ f(\inf(\sup(\mathrm{I}),z))$. Let I' be the set of all $\inf(i,z)$ for $i \in \mathrm{I}$. We have $\inf(\sup\mathrm{I},z) = \sup(\mathrm{I}')$. We conclude by induction applied to I and I'; this is trivial if the mapping $\mathrm{I} \to \mathrm{I}'$ is bijective, but this assumption may be wrong. For this reason we use an intermediate function.

We thus show the following: let $g : \mathrm{I} \to \mathrm{E}$ be any function, where I is a finite set. Let $g\langle\mathrm{X}\rangle$ be the set of all $g(x)$ for $x \in \mathrm{X}$. We have

$$f(\sup(g\langle\mathrm{I}\rangle)) + \sum_{\mathrm{H}\subset\mathrm{I},\mathrm{Card(H)\ even}} f(\inf(g\langle\mathrm{H}\rangle)) = \sum_{\mathrm{H}\subset\mathrm{I},\mathrm{Card(H)\ odd}} f(\inf(g\langle\mathrm{H}\rangle)).$$

This holds for any set I, but we prove it only in the case of an interval. Proof of the main result: if I is a finite set with n elements, there is a bijection $g : [0, n[\to \mathrm{I}$. Any subset of k elements of I is uniquely of the form $g\langle\mathrm{H}\rangle$, and H has k elements.

We start with some assumptions and the two properties to prove.

```
Lemma odd_nonempty x:  oddp (cardinal x) -> nonempty x.

Section Exercise5_5.
Variables (E r: Set) (f: Set -> Set).
Hypothesis lr:lattice r.
Hypothesis dl: distributive_lattice1 r.
Hypothesis sr: E = substrate r.
Hypothesis card_f: forall x, inc x E -> cardinalp (f x).
Hypothesis hyp_f: forall x y, inc x E -> inc y E ->
    (f x) +c (f y) = (f (sup r x y)) +c f (inf r x y).
Definition  Exercise5_5_conc I :=
   f (supremum r I) +c
       csumb (even_card0_sub I) (fun z => f (infimum r z))
    = csumb (odd_card_sub I) (fun z => f (infimum r z)).
```

```
Definition  Exercise5_5_conc_aux I g :=
  f (supremum r (fun_image I g)) +c
       csumb (even_card0_sub I) (fun z => f (infimum r (fun_image z g)))
    = csumb  (odd_card_sub I) (fun z => f (infimum r (fun_image z g))).
```

Let $\mathrm{I} = [0, n]$, $z = n+1$. We assume (by induction) the result true for any function g defined on I. We consider a function g defined on $\mathrm{I}' = \mathrm{I} \cup \{z\}$, and $g'(i) = \inf(g(i), g(z))$. We assume the result true for the restrictions of g and g' to I, say $f(\sup(g\langle \mathrm{I}\rangle)) + \Sigma_e = \Sigma_o$ and $f(\sup(g'\langle \mathrm{I}\rangle)) + \Sigma'_e = \Sigma'_o$. Our goal is $f(\sup(g\langle \mathrm{I}'\rangle)) + \Sigma''_e = \Sigma''_o$. We write Σ''_e as the sum of two terms, depending on whether $z \in \mathrm{H}$ or not. If $z \not\in \mathrm{H}$, we recognize Σ_e. The other term is the sum over all H of odd cardinal, of $f(\mathrm{H}')$ where $\mathrm{H}' = \inf(g\langle \mathrm{H} \cup \{z\}\rangle)$. Assume that the elements of H are $x_1, x_2, \dots, x_k$. Let $y_i = g(x_i)$. Then $\mathrm{H}' = \inf(y_1, \dots, y_k, g(z))$. We can replace $g(z)$ by k copies of it, putting a copy after each y_i. It follows $\mathrm{H}' = \inf(g'(x_1), \dots, g'(x_k))$. Thus $\mathrm{H}' = \inf(g'\langle \mathrm{H}\rangle)$ (proof by induction on the number of elements of H). Thus, the second term is Σ'_o. The same argument applies to Σ''_o. There is a particular case where H contains z and is a singleton. If we rewrite Σ''_e, Σ''_o and use the induction assumptions, the goal becomes

$$f(\sup(g\langle \mathrm{I}'\rangle)) + f(\sup(g'\langle \mathrm{I}\rangle)) + \Sigma_e + \Sigma'_e = f(g(z)) + f(\sup(g\langle \mathrm{I}\rangle)) + \Sigma_e + \Sigma'_e.$$

Now, $\sup(g\langle \mathrm{I}'\rangle) = \sup(\sup(g\langle \mathrm{I}\rangle), g(z)$. Let $x = \sup(g\langle \mathrm{I}\rangle)$. Our goal becomes

$$f(\sup(x, g(z)) + f(\sup(g'\langle \mathrm{I}\rangle)) + w = f(x) + f(g(z)) + w.$$

Now, we have $\inf(x, g(z)) = \sup(g'\langle \mathrm{I}\rangle)$ (since E is a distributive lattice), and the proof follows by assumption on f.

```
Lemma Exercise5_5_a1 n g (I:=Nintc n):  (* 244 *)
   natp n -> (forall i, inc i I -> inc (g i) E) ->
   Exercise5_5_conc_aux I g.

Lemma Exercise5_5_a2 I: (* 69 *)
   sub I E -> nonempty I -> finite_set I -> Exercise5_5_conc I.
End  Exercise5_5.
```

(b) The result holds for any finite family of sets B_i. In fact, let A be the union of the family, ordered by inclusion. We know that it is a lattice; the supremum of x and y is $x \cup y$ and the infimum is $x \cap y$. Distributivity of intersection and union shows that this is a distributive lattice. Relation $\mathrm{Card}(x) + \mathrm{Card}(y) = \mathrm{Card}(x \cup y) + \mathrm{Card}(x \cap y)$ is easy (if $z = x - y$, it is disjoint from y and $x \cap y$, and the two unions are $x \cup y$ and x).

Note: define $\mathrm{B}_\emptyset$ to be the union of the B_i. The result says: the sum of the cardinals of the B_H for Card(H) even is equal to the sum of the cardinals of the B_H for Card(H) odd. The theorem comes in two variants, one where we have a family of sets, and one where we have a set of sets. The second variant says

$$\mathrm{Card}(\bigcup \mathrm{I}) + \sum_{\mathrm{H} \subset \mathrm{I}, \mathrm{Card}(\mathrm{H}) \text{ even}} \mathrm{Card}(\bigcap \mathrm{H}) = \sum_{\mathrm{H} \subset \mathrm{I}, \mathrm{Card}(\mathrm{H}) \text{ odd}} \mathrm{Card}(\bigcap \mathrm{H}).$$

```
Lemma setP_lattice_d1 A: distributive_lattice1 (subp_order A). (* 9 *)

Lemma Exercise5_5_b1 x y: (* 4 *)
  cardinal x +c cardinal y = cardinal (x \cup y) +c cardinal (x \cap y).

Lemma Exercise5_5_b3 I (f: fterm) : finite_set I ->  (* 122 *)
```

```
  cardinal (unionf I f) +c
       csumb (even_card0_sub I) (fun z => cardinal (intersectionf z f))
    = csumb (odd_card_sub I) (fun z => cardinal (intersectionf z f)).
Lemma Exercise5_5_b2 I: finite_set I -> (* 6 *)
  cardinal (union I) +c
       csumb (even_card0_sub I) (fun z => cardinal (intersection z))
    = csumb (odd_card_sub I) (fun z => cardinal (intersection  z)).
```

---

6. Prove the formula

$$\binom{n+h}{h} = 1 + \binom{h}{1}\binom{n+h-1}{h} - \binom{h}{2}\binom{n+h-2}{h} + \ldots - (-1)^h \binom{h}{h}\binom{n}{h}.$$

(If F denotes the set of mappings u of $[1,h]$ into $[0,n]$ such that $\sum_{x=1}^{h} u(x) \le n$, consider for each subset H of $[1,h]$ the set of all $u \in \mathrm{F}$ such that $u(x) \ge 1$ for each $x \in \mathrm{H}$, and use Exercise 5.)

Note: Bourbaki writes $\ldots + (-1)^h$, which is obviously wrong.

The generic term on the RHS is $\binom{h}{i}\binom{n+h-i}{h}$, for $1 \le i \le h$, with a minus sign for i even. We move these terms to the LHS, and notice that the LHS is the term for $i = 0$. Thus we prove

$$\sum_{i \text{ even}} \binom{h}{i}\binom{n+h-i}{h} = 1 + \sum_{i \text{ odd}} \binom{h}{i}\binom{n+h-i}{h}.$$

Let I be the interval $[0,h[$, it has h elements. We denote by $\mathrm{S}_{\mathrm{I},n}$ the set of mappings $f : \mathrm{I} \to [0,n]$ with $\sum f(i) \le n$. Its cardinal is $\binom{n+h}{h}$ (the LHS of the original formula). For $i \in \mathrm{I}$, let B_i be those $f \in \mathrm{S}_{\mathrm{I},n}$ such that $f(i) \ne 0$. For $\mathrm{H} \subset \mathrm{I}$, let $\mathrm{B_H}$ be the intersection of the B_i for $i \in \mathrm{H}$. This is the set of all functions f such that $f(i) \ne 0$ for $i \in \mathrm{H}$. Let $\bar f$ be the function that agrees with f on the complement of H, and $\bar f(x) = f(x) - 1$ on H, considered as an element of $\mathrm{S}_{\mathrm{I},n-k}$, where k is the cardinal of H. Thus, the cardinal of $\mathrm{B_H}$ is $\binom{n-k+h}{h}$. Let B be the union of the B_i. It contains all functions f such that $f(i) \ne 0$ for at least one i, thus all elements of $\mathrm{S}_{\mathrm{I},n}$ but the constant function zero. The result trivially follows from the previous exercise.

```
Lemma Exercise5_6  n h (I := (Nintc h)) (* 201 *)
    (f := fun i => (binom h i) *c  (binom (n +c h -c i) h)):
   natp n -> natp h ->
   csumb (Zo I evenp) f = \1c +c csumb (Zo I oddp) f.
```

---

7. (a) Let $\mathrm{S}_{n,p}$ denote the number of mappings of $[1,n]$ *onto* $[1,p]$. Prove that

$$\mathrm{S}_{n,p} = p^n - \binom{p}{1}(p-1)^n + \binom{p}{2}(p-2)^n - + \ldots + (-1)^{p-1}\binom{p}{p-1}.$$

(Note that $p^n = \mathrm{S}_{n,p} + \binom{p}{1}\mathrm{S}_{n,p-1} + \binom{p}{2}\mathrm{S}_{n,p-2} + \ldots + \binom{p}{p-1}$ and use Exercise 3.)

(b) Prove that $\mathrm{S}_{n,p} = p(\mathrm{S}_{n-1,p} + \mathrm{S}_{n-1,p-1})$ (method of no. 8, Proposition 13).

(c) Prove that

$$\mathrm{S}_{n+1,n} = \frac{n}{2}(n+1)! \qquad \text{and} \qquad \mathrm{S}_{n+2,n} = \frac{n(3n+1)}{24}(n+2)!$$

(consider the elements r of $[1,n]$ whose inverse image consists of more than one element).

(d) If $\mathrm{P}_{n,p}$ is the number of partitions into p parts of a set of n elements, show that $\mathrm{S}_{n,p} = p!\mathrm{P}_{n,p}$.

Comment. We shall prove

$$p^n = \sum_{i=0}^{p} \binom{p}{i} \mathrm{S}_{ni}.$$

We have explained above how to invert this formula and we get

$$\mathrm{S}_{n,p} = \sum_{i=0}^{p} (-1)^i \binom{p}{i} (p-i)^n,$$

as well as (b). Note that the sums contain one more term than those of Bourbaki; with these modifications, the formulas hold for $p = 0$ as well as $n = 0$. Equation (6.70) provides (c). The formulation is a bit different, it avoids division.

(a) The cardinal of the set of surjective mappings $\mathrm{E} \to \mathrm{F}$ depends only on the cardinals n of E and m of F, since $\mathscr{F}(\mathrm{E};\mathrm{F})$ is canonically isomorphic to $\mathscr{F}(\mathrm{E}';\mathrm{F}')$ when E is equipotent to E' and F is equipotent to F'. This isomorphism maps surjective functions onto surjective functions. In the definition of $\mathrm{S}_{n,p}$ below, we use $[0,n[$ and $[0,p[$ instead of $[1,n]$ and $[1,p]$. We denote these intervals by I_n and I_p. We denote by $\mathfrak{S}(n,p)$ the set of surjections $\mathrm{I}_n \to \mathrm{I}_p$.

```
Definition surjections  E F :=
  Zo (functions E F)(surjection).

Definition nbsurj n p :=
   cardinal(surjections (Nint n)  (Nint p)).

Lemma nbsurj_pr E F:  (* 42 *)
  finite_set E -> finite_set F ->
  cardinal (surjections  E F) = nbsurj (cardinal E) (cardinal F).
```

Let $f : \mathrm{I}_n \to \mathrm{I}_p$ be a function, $\mathrm{R}(f)$ its range, $s(f)$ the cardinal of $\mathrm{R}(f)$. This is an integer between 0 and p and $\mathrm{R}(f)$ is a subset of I_p with $s(f)$ elements. Given an integer k, and a set R with k elements, we count the number of function f such that $\mathrm{R}(f) = \mathrm{R}$. This is $\mathrm{S}_{n,k}$, since f, considered as a function $\mathrm{I}_n \to \mathrm{R}$, is surjective. Thus, the number of functions whose range has k elements is $\binom{n}{k}\mathrm{S}_{n,k}$. This concludes the proof.

```
Definition surjections  E F :=
  Zo (functions E F)(surjection).

Definition nbsurj n p :=
   cardinal(surjections (Nint n)  (Nint p)).

Lemma nbsurj_pr E F:  (* 42 *)
  finite_set E -> finite_set F ->
  cardinal (surjections  E F) = nbsurj (cardinal E) (cardinal F).
Lemma nbsurj_inv n p: natp n -> natp p ->  (* 71 *)
  p ^c n = csumb (Nintc p) (fun k => (binom p k) *c (nbsurj n k)).
```

(b) Let $f \in \mathfrak{S}(n+1, p+1)$. Let $\phi(f) = f(n)$ be the mapping $\mathfrak{S}(n+1, p+1) \to \mathrm{I}_{p+1}$. We apply the shepherd's principle. We have to show: for any x with $x \le p$, the number of elements f in $\mathfrak{S}(n+1, p+1)$ such that $f(n) = x$ is $\mathrm{S}_{n,p+1} + \mathrm{S}_{n,p}$. We consider two cases: there is $y < n$ such that $f(y) = f(x)$, or there is no such y. In the first case, if $\bar f$ is the restriction of f to I_n, then $\bar f \in \mathfrak{S}(n, p+1)$. In the second case, $\bar f$ is in a set that has the same number of elements as $\mathfrak{S}(n, p)$, the set of surjective functions $\mathrm{I}_n \to \mathrm{I}_{p+1} - x$.

```
Lemma nbsurj_rec n p: natp n -> natp p -> (* 126 *)
  nbsurj (csucc n)(csucc p) =
   (csucc p) *c ( nbsurj n p +c  nbsurj n (csucc p)).
```

(c) These two relations follow trivially from (b) and the fact that $\mathrm{S}_{n,n} = n!$. We explain here how they could be shown directly.

Consider a surjection f of $\mathrm{E} = [1, n+1]$ onto $[1, n]$. There is a unique pair (x, y) with $x < y$ such that $f(x) = f(y)$. Let $z = f(x)$. The restriction $\bar f$ of f to $\mathrm{E} - \{x, y\}$ is a bijection onto $\mathrm{F} - \{z\}$. There are $n(n+1)/2$ possibilities for (x, y), n possibilities for y, and $(n-1)!$ for $\bar f$.

Consider a surjection f of $\mathrm{E} = [1, n+2]$ onto $[1, n]$. There are two cases. There can be a triple (x, y, t) such that $f(x) = f(y) = f(t)$; the same argument as above shows that the number of possibilities is $n(n+2)!/6$. The other possibility is that $f(x) = f(y) \neq f(x') = f(y')$. Let z and z' be the two values of f. The number of possibilities is: $\binom{n+2}{4}$ for (x, y, x', y'), $\binom{n}{2}$ for (z, z'), 6 for f restricted to $\{x, y, x', y'\}$ and $(n-2)!$ for f restricted to the complement. Adding the product of these four numbers to $n(n+2)!/6$ gives the result.

(d) We introduce some material needed in Exercise 9. If $f : \mathrm{A} \to \mathrm{B}$ is function, then $f' : \mathfrak{P}(\mathrm{A}) \to \mathfrak{P}(\mathrm{B})$ defined by $f'(\mathrm{X}) = f\langle \mathrm{A} \rangle$ is a function. We consider here $f'' : \mathfrak{P}(\mathfrak{P}(\mathrm{A})) \to \mathfrak{P}(\mathfrak{P}(\mathrm{B}))$, and more precisely, the restriction of f'' to some subsets. If f is a bijection so is f'', and $f \mapsto f''$ is compatible with composition. In what follows, we consider $x \mapsto f''(x)$ which has a simpler form. Note that if f is a bijection, then f'' maps a partition into a partition.

```
Definition extension_p2 g :=  extension_to_parts (extension_to_parts g).
Definition extension_p3 g := Vfs (extension_to_parts g).

Lemma ext2_pr1 g E z:
  function g ->  source g = E -> inc z (\Po (\Po E)) ->
  extension_p3 g z =  Vf (extension_p2 g) z.  (* 2 *)
Lemma ext2_pr2 g E E' z:
  (bijection_prop g E E') -> inc z (\Po (\Po E)) ->
  extension_p3 g z =  Vf (extension_p2 g) z. (* 1 *)
Lemma ext2_pr3 E E' g z: (* 11 *)
  bijection_prop g E E' -> inc z (\Po (\Po E)) ->
  forall t,
  (inc t (extension_p3 g z) <->
     exists2 u, inc u z & t = (Vfs g u)).
Lemma ext2_pr5 E E' g z:
  bijection_prop g E E' -> inc z (\Po (\Po E)) ->
  inc (extension_p3 g z) (\Po (\Po E')).  (* 3 *)
Lemma ext2_pr6 E E' E'' g g' z:
  bijection_prop g E E' -> bijection_prop g' E' E'' ->
  inc z (\Po (\Po E)) ->
  extension_p3 g' (extension_p3 g z) = extension_p3 (g' \co g) z. (* 15 *)
Lemma ext2_pr7 E E' g z:
  bijection_prop g E E' ->
  inc z (\Po (\Po E)) ->
```

```
  extension_p3 (inverse_fun g) (extension_p3 g z) = z. (* 4 *)
Lemma ext2_pr8 E E' g z:
  bijection_prop g E E' ->
  inc z (partitions E) -> inc (extension_p3 g z) (partitions E'). (* 24 *)
Lemma ext2_pr9 E E' g z z':
  bijection_prop g E E' ->
  inc z (partitions E) -> inc z' (partitions E) ->
  (extension_p3 g z) = (extension_p3 g z') -> z = z'. (* 2 *)
```

Clearly, if f is a bijection, then f'' maps a partition of size p into a partition of the same size. So the cardinal of $\mathscr{P}(\mathrm{E}, p)$, the set of partitions with p elements of E, depends only on the cardinal n of E. We denote it by $\mathrm{P}_{n,p}$.

Let $f : \mathrm{E} \to [1, p]$ be surjective. Consider the set $\Phi(f)$ of all $f^{-1}\langle\{i\}\rangle$. This is a partition with p elements of E; it is associated to the equivalence relation $f(x) = f(y)$. We apply the shepherd's principle. It suffices to show that $\mathrm{card}(\mathrm{F}) = p!$, where F is the inverse image by Φ of some partition x with p elements of E. Let $(x_i)_{1\le i\le p}$ be the elements of x, and $f(t)$ be the i such that $t \in x_i$. We have $f \in \mathrm{F}$. If g is a permutation of $[1, p]$, then $g \circ f \in \mathrm{F}$, and all elements of F are of this form.

```
Definition partitionsx E p :=
  Zo (partitions E) (fun z => cardinal z = p).
Definition nbpart n p :=
  cardinal(partitionsx (Nint n) p).

Lemma nbpart_pr1 E F g p:  (* 23 *)
  bijection_prop g E F ->
  bijection (Lf (extension_p3 g)  (partitionsx E p)(partitionsx F p)).
Lemma nbpart_pr E p: (* 5 *)
  finite_set E -> cardinal (partitionsx E p) = nbpart (cardinal E) p.
Lemma nbsurj_part n p: natp n -> natp p -> (* 170 *)
  nbsurj n p = (factorial p) *c (nbpart n p).
```

Complement. The total number of partitions of a set E is its Bell number. This depends only on the cardinal n of E, and $\mathrm{B}_n = \sum_{i\le n} \mathrm{P}_{ni}$ (the non-trivial point is to show that a partition of E cannot have more than n elements).

```
Definition Bell_number n := cardinal (partitions (Nint n)).

Lemma Bell_pr E :  (* 10 *)
  finite_set E ->
  cardinal (partitions E) =
  csumb (Nintc (cardinal E)) (fun p => cardinal (partitionsx E p)).
Lemma Bell_pr1 n: natp n -> (* 2 *)
  Bell_number n = csumb (Nintc n) (nbpart n).
Lemma Bell_pr2 E: finite_set E -> (* 3 *)
  cardinal (partitions E) = Bell_number (cardinal E).
```

We have

$$\mathrm{B}_{n+1} = \sum_{k=0}^{n} \binom{n}{k} \mathrm{B}_k.$$

Let $\mathrm{E} = \mathrm{I}_{n+1}$. This is a set with $n+1$ elements and $n \in \mathrm{E}$. Consider a partition X, and S the set that contains x. This is any subset of $n-k$ elements of I_n (whatever k), and $\mathrm{X} - \{\mathrm{S}\}$ is a partition of $\mathrm{E}' - \mathrm{S}$.

```
Lemma Bell_rec n : natp n -> (* 124 *)
  Bell_number (csucc n) =
  card_sumb (Nintc n) (fun k => (binom n k) *c (Bell_number k)).
```

---

8. Let p_n be the number of permutations of a set E with n elements such that $u(x) \neq x$ for all $x \in \mathrm{E}$. Show that

$$p_n = n! - \binom{n}{1}(n-1)! + \binom{n}{2}(n-2)! - \cdots + (-1)^n$$

* and hence that $p_n \sim n!/e$ as $n \to \infty$ * (same method as in Exercise 7 (a)).

Note. The stars indicate that a part of the exercise uses material not yet defined, here tilde and e. The formula is $p_n = n! \sum_{i\le n}(-1)^i/i!$, hence $p_n/n! = \sum_{i\le n}(-1)^i/i!$. Define $\exp(x) = \sum_i x^i/i!$ where the sum is over all i; considered as a real number. This is called the exponential function and has some properties. The tilde notation says that the finite sum is an approximation of the infinite sum. Here $p_n/n!$ is an approximation of $\exp(-1)$; Let $e = \exp(1)$; one property of the exponential function is that $\exp(-1) = 1/e$.

Discussion. The method of Exercise 7 (a) consists in proving $\sum_k \binom{n}{k} p_k = n!$, then using the inversion formula.

Solution. We say that a permutation of E that has no fixed point is a derangement. Let f be a derangement of E, and $g : \mathrm{E} \to \mathrm{F}$ a bijection. Then $g \circ f \circ g^{-1}$ is a derangement of F. Thus, the number of derangements of E depends only on the cardinal n of E (we state the result in the case E is finite, but it holds in the general case). We call this number p_n.

To each permutation f we associate the set of its fixed points, $\Phi(f)$. If A has $n-k$ elements, $\mathrm{B} = \mathrm{E} - \mathrm{A}$ has k elements and $\Phi(f) = \mathrm{A}$ if and only if f is a derangement on B, the identity on A. The number of such f is p_k. It follows that $\sum_k \binom{n}{k} p_k = n!$.

```
Definition derangements E :=
  Zo (permutations E) (fun z => forall x, inc x E -> Vf z x <> x).

Definition nbder n :=
  cardinal(derangements (Nint n)).

Lemma nbder_pr E: finite_set E -> (* 57 *)
   cardinal (derangements E) = nbder (cardinal E).
Lemma nbder_0: nbder \0c = \1c. (* 8 *)
Lemma nbder_1: nbder \1c = \0c. (* 6 *)

Lemma nbder_pr1 n: natp n -> (* 118 *)
  factorial n = card_sumb (Nintc n) (fun k =>
   binom n k *c nbder k).
```

Complement. We have: $p_{n+1} = n(p_n + p_{n-1})$. Consider a derangement f of a set E with $n+1$ elements; fix $x \in \mathrm{E}$, let $\mathrm{E}' = \mathrm{E} - \{x\}$. Let $y = f(x)$; there are n possibilities for y, since $f(x) \in \mathrm{E}'$. Assume $f(y) = x$. Now f is a derangement of $\mathrm{E}' - \{y\}$. Assume now $f(z) = x$, where $z \neq y$. Exchange the values of x and z. This function g satisfies $g(x) = x$, but is a derangement of E'.

We compute $p_0 = 1$ and $p_1 = 0$ (the empty function is a derangement of the empty set, and a singleton has no derangements). We have $p_2 = 1$ and $p_3 = 2$ (the only derangement of

$(0,1)$ is $(1,0)$ and the derangements of $(0,1,2)$ are $(1,2,0)$ and $(2,0,1)$. We have $p_4 = 9$ (check all permutations).

We have $p_{n+1} = (n+1)p_n + (-1)^{n+1}$. Proof. By induction, $p_n \in \mathbf{N}$. By induction, $p_n > 0$ whenever $n > 1$. In particular, when n is even, $(n+1)p_n > 0$, so that $(n+1)p_n + (-1)^{n+1}$ is defined. The result follows by induction.

```
Lemma nbder_pr2 n: natp n ->  (* 252 *)
   nbder (csucc (csucc n)) = (csucc n) *c (nbder n +c nbder (csucc n)).
Lemma nbder_pr3 f (g := fun n => (succ n) *c (f n)): (* 48 *)
  (f \0c = \1c) -> f \1c = \0c ->
  (forall n, natp n ->
    f (csucc (csucc n)) = (csucc n) *c (f n +c  f (csucc n))) ->
  (forall n, natp n -> (evenp n) -> f n <> \0c)
  /\
  (forall n, natp n ->
     f (csucc n) = Yo (evenp n) (cpred (g n)) (csucc (g n))).
Lemma nbder_pr4 n  (g := fun n => (csucc n) *c (nbder n)): (* 1 *)
  natp n -> nbder (csucc n) = Yo (evenp n) (cpred (g n)) (csucc (g n)).
```

---

9. (a) Let E be a set with qn elements. Show that the number of partitions of E into n subsets each of q elements is equal to

$$(qn)!/(n!(q!)^n).$$

(b) Suppose that $E = [1, qn]$. Show that the number of partitions of E into n subsets each of q elements, no one of which is an interval, is equal to

$$\frac{(qn)!}{n!(q!)^n} - \frac{(qn-q+1)!}{1!(n-1)!(q!)^{n-1}} + \frac{(qn-2q+2)!}{2!(n-2)!(q!)^{n-2}} - \cdots + (-1)^n$$

(same method as in Exercises 7 and 8).

Comment. We fix q. Proving (a) is easy. Assume now $E = [1, qn]$ and consider a partition p. Let J be the set of all j such that $[j, j+q[$ is in p, and assume that J has cardinal k. Let E_J be the union of the $[j, j+q[$ for $j \in J$ and $E' = E - E_J$. This set has cardinal $q(n-k)$ so is equipotent to $[1, q(n-k)]$ via a function f. We may assume f order preserving, so that f, applied to the elements of p that are not of the form $[j, j+q[$ for $j \in J$, is a partition without intervals. The only condition on J is that the distance between two elements is at least q. Hence the number of such J is a binomial number $c_{n,k}$. If $g(n)$ is the answer to (b), then $f(n) = \sum_k c_{n,k} g(n-k)$. One deduces $g(n) = \sum_k c'_{n,k} f(n-k)$. So, we have (1) compute the c, (2) prove that $f(n)$ is indeed the sum, (3) compute the c' and (4) finish the proof. Note that we may use (4) to get the c' and (3) reduces to show that a given matrix is the inverse on another one. This is a bit more complicated than in the case of Exercises 7 and 8.

Implementation. We introduce $P_{E,q}$ the set of partitions p of E such that, if $x \in p$, then x has cardinal q. Assume first $q = 0$. In this case x is empty, as well as E. So: either E is empty, and P has a single element, the empty partition, or P is empty. From now on, we shall assume $q > 0$ (and in most cases finite). If $p \in P$, then $\operatorname{card}(E) = q \operatorname{card}(p)$; conversely $\operatorname{card}(p) = \operatorname{card}(E)/q$ when division is exact.

```
Definition partition_nq E q:=
```

```
  Zo (partitions E) (fun z => forall x, inc x z ->  cardinal x = q).

Lemma partition_nq_pr1 E q p:
 inc p (partition_nq E q) -> cardinal E = q *c (cardinal p).  (* 6 *)
Lemma partition_nq_pr2 E q n p : cardinalp n -> natp q -> q <> \0c ->
  cardinal E = q *c n->
  inc p (partition_nq E q) -> cardinal p = n. (* 3 *)
Lemma partition_nq_pr3 E: E = emptyset \/ partition_nq E \0c = emptyset.
Lemma partition_nq_pr4 E:
  cardinal (partition_nq E \0c) = Yo (E = emptyset) \1c \0c. (* 14 *)
```

We now introduce $f_q(n)$, the answer to (a). It is not quite obvious that division is exact (the property is false when $q = 0$). However, $f_1(n) = 1$ holds when n is an integer, and this is the cardinal of $\mathrm{P}_{\mathrm{E},1}$ (recall that the greatest partition of E is the set of all singletons).

```
Definition Ex59_num q n:= (factorial (q *c n)).
Definition Ex59_den q n:= (factorial n) *c (factorial q) ^c n.
Definition Ex59_val q n:= (Ex59_num q n) %/c (Ex59_den q n).

Lemma partition_nq_pr5 E:
  (partition_nq E \1c) = singleton (greatest_partition E). (* 15 *)
Lemma partition_nq_pr5b E:  cardinal (partition_nq E \1c) = \1c. (* 1 *)
Lemma partition_nq_pr5c E: finite_set E ->
  cardinal (partition_nq E \1c) = partition_nq_nb \1c (cardinal E). (* 4 *)
```

Let's denote by g'' the extension to $\mathfrak{P}(\mathfrak{P}(\mathrm{E}))$ of a bijection $g : \mathrm{E} \to \mathrm{E}'$. We have shown that the image of a partition is a partition. It's clear that it induces a bijection $\mathrm{P}_{\mathrm{E},q} \to \mathrm{P}_{\mathrm{E}',q}$.

Assume E has cardinal qn. Then $\mathrm{P}_{\mathrm{E},q}$ is nonempty (proof: take $\mathrm{E}' = q \times n$ and the set of all $q \times \{i\}$ for $i \in n$). Moreover, if x and y are in $\mathrm{P}_{\mathrm{E},q}$ then there is a permutation σ of E such that $y = \sigma''(x)$. Proof. First, x and y have the same cardinal n, so that there is a bijection $f : x \to y$. Now, if $t \in x$, then t and $f(t)$ have the same cardinal q, so that there is a bijection $f_t : t \to f(t)$. The common extension of these f_t is some function $\mathrm{E} \to \mathrm{E}$, the desired permutation.

```
Lemma partition_nq_pr6c E E' q g:  bijection_prop g E E' -> (* 6 *)
   lf_axiom (extension_p3 g) (partition_nq E q) (partition_nq E' q).
Lemma partition_nq_pr6d E E' q:  E \Eq E' ->
  (partition_nq E q) \Eq (partition_nq E' q).  (* 10 *)
Lemma partition_nq_pr7 E n q: \0c <c q -> cardinal E = q *c n ->
  nonempty (partition_nq E q).  (* 25 *)
Lemma partition_nq_pr8 E q x y: natp q -> q <> \0c ->
  inc x (partition_nq E q) -> inc y (partition_nq E q) ->
  exists2 f, inc f (permutations E)  &
     Vfs (extension_to_parts f) x = y. (* 62 *)
```

Consider an element p of $\mathrm{P}_{\mathrm{E},q}$ (we know that it exists), and a permutation σ of E. The quantity $\sigma''(p)$, that belongs to $\mathrm{P}_{\mathrm{E},q}$, will be denoted by $f(\sigma)$. Let x be another element of $\mathrm{P}_{\mathrm{E},q}$, and τ a permutation such that $\tau''(p) = x$. Define $\mathrm{D}_x = f^{-1}(\{x\})$. This is the set of permutations σ such that $f(\sigma) = x$, or $\sigma''(p) = x$ or $(\tau^{-1} \circ \sigma)''(p) = p$. This last relation shows that the cardinal of D_x is independent of x.

Now, the shepherd principle says that the cardinal of $\mathrm{P}_{\mathrm{E},q}$ is the quotient of the numbers of permutations, namely $(qn)!$ and the cardinal of D_p. This cardinal is $n!(q!)^n$, since we have a bijection $\Phi : \mathrm{S}_n \times (\mathrm{S}_q)^n \to \mathrm{D}_p$, where S_k the set of permutations of I_k. The proof is a bit elaborated. First, if k is finite, then I_k is finite and has k elements, so that S_k is finite and has cardinal $k!$. Second, p has n elements, so that there is a bijection $\mathrm{I}_n \to p$, denoted $i \mapsto \mathrm{E}_i$. Since each element of p has q elements, it follows that each E_i has q elements; so that there is a bijection $f_i : \mathrm{E}_i \to \mathrm{I}_q$.

Let's define Φ. Its argument κ is a pair formed of a permutation τ of I_n and a functional graph, defined on I_n such that each value σ_i is a permutation of I_q. If $x \in \mathrm{E}_i$, then $f^{-1}_{\tau(i)}(\sigma_i(f_i(x)))$ belongs to $\mathrm{E}_{\tau(i)}$ hence to E. This gives a function $g_i : \mathrm{E}_i \to \mathrm{E}$. Note that the family $(\mathrm{E}_i)_i$ is a partition of E; so that the family $(g_i)_i$ has a common extension $g : \mathrm{E} \to \mathrm{E}$. In particular, if $x \in \mathrm{E}_i$, then $g(x) = g_i(x)$ and $g(x) \in \mathrm{E}_{\tau(i)}$. Now g is injective: if $g(x) = g(y)$, $x \in \mathrm{E}_i$ and $y \in \mathrm{E}_j$, the sets $\mathrm{E}_{\tau(i)}$ and $\mathrm{E}_{\tau(j)}$ have a common element, thus are equal. It follows $\tau(i) = \tau(j)$ hence $i = j$ by injectivity of τ. Now $g_i(x) = g_i(y)$ implies $i = j$ (everything is injective). As E is finite, g is a permutation of E. We pretend $g \in \mathrm{D}_p$. Consider $x \in p$, let's say $x = \mathrm{E}_i$; define $y = \mathrm{E}_{\tau(i)}$, this is an element of p. By construction $g\langle x\rangle \subset y$; but g is injective, and x and y have the same cardinal q. So $g\langle x\rangle = y$. In particular, the set of all $g\langle x\rangle$ is a subset of p. It is easily seen that every element of p has this form, so that the set of all $g\langle x\rangle$ is p. We can now define $\Phi(\kappa) = g$. This gives an injective function. In effect, assume $\Phi(\kappa) = \Phi(\kappa')$. Take $x \in \mathrm{E}_i$. We have $g(x) \in \mathrm{E}_{\tau(i)}$ and $g'(x) \in \mathrm{E}_{\tau'(i)}$. The two sets $\mathrm{E}_{\tau(i)}$ and $\mathrm{E}_{\tau'(i)}$ have a common element, thus are equal, so that $\tau = \tau'$. It follows that $\sigma_i(x) = \sigma'_i(x)$, hence $\sigma_i = \sigma'_i$. Finally, Φ is surjective. Consider a permutation g such that the set of all $g\langle x\rangle$ for x in p is p. If i is an integer, then $\mathrm{E}_i \in p$, and $g\langle x\rangle = \mathrm{E}_j$ for some j. This gives τ. This function is obviously injective, thus is a permutation of I_n. Moreover, the restriction $g : \mathrm{E}_i \to \mathrm{E}_{\tau(i)}$ is a bijection (it is injective, the two sets have the same cardinal). Consider $\sigma_i(t) = f_{\tau(i)}(g(f_i^{-1}(t)))$, where g is the above restriction. This is a bijection $\mathrm{I}_q \to \mathrm{I}_q$, thus a permutation. One deduces $g(x) = f^{-1}_{\tau(i)}(\sigma_i(f_i(x)))$. Qed. [the proof is a bit long: 150 lines to show that Φ is a function, 100 lines to show that it is surjective).

```
Lemma partition_nq_pr9 E q n:  (* 349 *)
  natp n -> natp q -> \0c <c q -> cardinal E = q *c n ->
  cardinal (partition_nq E q) *c (Ex59_den q n) = (Ex59_num q n) .
```

The previous lemma says: $\mathrm{card}(\mathrm{P}_{\mathrm{E},q}) \cdot \mathrm{D} = \mathrm{N}$. We show here that all quantities are integer, D is non-zero, so that $\mathrm{card}(\mathrm{P}_{\mathrm{E},q}) = f_n(q)$. This result is generally false when $q = 0$.

```
Lemma Exercise5_9a q E: finite_set E ->
   natp (cardinal (partition_nq E q)). (* 5 *)
Lemma Exercise5_9b q n: natp n -> natp q ->
  [/\ natp(Ex59_num q n), natp(Ex59_den q n) & natp (Ex59_val q n) ]. (* 5 *)
Lemma Exercise5_9b' q n: natp n -> natp q -> (Ex59_den q c)  <> \0c. (* 1 *)
Lemma Exercise5_9c q n: natp n -> natp q -> \0c <c q ->  (* 6 *)
   (Ex59_den q n) %|c (Ex59_num q n).
Lemma Exercise5_9c' q n: natp n -> natp q -> \0c <c q ->   (* 2*)
  (Ex59_num q n) = (Ex59_val q n) *c (Ex59_den q n).
Lemma Exercise5_9d q n: natp n -> natp q -> \0c <c q ->
   (Ex59_val q n) <> \0c.  (* 3 *)
Lemma Exercise5_9e E q n: natp n -> natp q -> \0c <c q ->  (* 5 *)
  cardinal E = q *c n ->
  cardinal (partition_nq E q) = Ex59_val q n.
Lemma Exercise5_9f n: natp n ->
  (Ex59_val \0c n) = Yo (n <=c \1c) \1c \0c. (* 12 *)
Lemma Exercise5_9f': partition_nq emptyset \0c = C1. (* 6 *)
```

We consider now (b). The generic term is $(qn-kq+k))!/(k!(n-k)!q!^{n-k})$, with a negative sign if k is odd. This is a bit complicated. We write the expression x in the numerator as

$qn - k(q-1)$; this makes sense as $q > 0$. On the other hand, it is $k + q(n-k)$, so that, when $k \le n$, it is $\ge k$. Hence $\binom{x}{k}$ is $x!/(k!(q(n-k))!)$, and the generic term is $\binom{x}{k} f(n-k)$. So, the expression in (b) is

$$\sum_{k\le n} (-1)^k \binom{qn-k(q-1)}{k} f_q(n-k).$$

```
Definition Ex59b_num1 q n k := (q *c n) -c k *c (q -c \1c).
Definition Ex59b_num q n k := factorial (Ex59b_num1 q n k).
Definition Ex59b_den q n k :=
   (factorial k)*c(factorial (n-c k)) *c (factorial q) ^c (n-c k).
Definition Ex59b_val q n k:= (Ex59b_num q n k) %/c (Ex59b_den q n k).

Lemma Exercise5_9g q n k: natp n -> natp q -> natp k ->   (* 11 *)
  [/\ natp(Ex59b_num1 q n k), natp(Ex59b_num q n k),
     natp (Ex59b_den q n k), natp(Ex59b_val q n k) & (Ex59b_den q n k) <> \0c].
Lemma Exercise5_9h q n k: natp n -> natp q -> natp k -> k <=c n -> \0c <c q ->
  (Ex59b_num q n k) =  (Ex59b_den q n k) *c
    ((binom (Ex59b_num1 q n k) k) *c (Ex59_val q (n -c k))).  (* 21 *)
Lemma Exercise5_9h' q n k: (* 5 *)
   natp n -> natp q -> natp k -> k <=c n -> \0c <c q ->
  (Ex59b_val q n k) = (binom (Ex59b_num1 q n k) k) *c (Ex59_val q (n -c k)).
```

Assume that E is totally ordered and its elements are $x_1, x_2 \dots, x_{qn}$ in increasing order. We say that an element of a partition is an “interval” if it contains all x_j, where $i \le j < i+q$ for some i. Let E' be another totally ordered set, with the same cardinal, with elements $y_1, y_2 \dots, y_{qn}$. Then $x_i \mapsto y_i$ is some bijection $E \to E'$. If p is a partition of E then $f''(p)$ is a partition of E', and these two partitions have the same number of intervals. Thus, the cardinal of $P_{E,q,k}$, the subset of elements of $P_{E,q}$ having k intervals, is independent of E: it depends only on its cardinal qn. For this reason, we consider the case $E = [0, qn[$ rather than $E = [1, qn]$. An interval will be any set of the form $[j, j+(q-1)]$.

We have $\mathrm{card}(P_{E,q}) = \sum_{k\le n} \mathrm{card}(P_{E,q,k})$, because a partition has at most n elements, whenever E has nq elements. Note: assume that no subset of E is an interval (for instance, if no element of E is an integer); with the current definition of an interval, we get $\mathrm{card}(P_{E,q}) = \mathrm{card}(P_{E,q,0})$, as all other terms are zero. On the other hand, with the original definition of an interval, it is obvious that there is at least one partition formed solely of intervals.

```
Definition Ex59_int q j := Nintcc j (j +c (q -c \1c)).
Definition Ex59_intervalp q x :=
   exists2 j, natp j & x = Ex59_int q j.
Definition Ex59_nb_int q p :=
   cardinal (Zo p (Ex59_intervalp q)).
Definition Ex59_k_interval E q k :=
  Zo (partition_nq E q) (fun p => Ex59_nb_int q p = k).

Lemma Exercise5_9i E q n:   (* 11 *)
  natp n -> natp q -> cardinal E = q *c n -> \0c <c q ->
  partition_w_fam (Lg (Nintc n) (Ex59_k_interval E q))  (partition_nq E q).
Lemma Exercise5_9i' E q n:  (* 3 *)
  natp n -> natp q -> cardinal E = q *c n -> \0c <c q ->
  cardinal (partition_nq E q) =
    csumb (Nintc n) (fun k => (cardinal (Ex59_k_interval E q k))).
```

We now fix q, n, $E = [0, qn[$ and $P_k = P_{E,q,k}$. If $p \in P_k$, we denote by p_J the subset of p formed of intervals and by p'_J its complement; we denote by $J(p)$ the set of lower bounds of elements of p_J. We prove a bunch a small lemmas. Obviously $J(p)$ and p_J have k elements, and p_J is the set of intervals. It follows that p'_J has cardinal $n-k$.

```
Definition Ex59_int_lb q x :=  select (fun j => x = Ex59_int q j) Nat.
Definition Ex59_splitA q p :=
  fun_image (Zo p (Ex59_intervalp q)) (Ex59_int_lb q).
Definition Ex59_splitA' q p := fun_image (Ex59_splitA q p) (Ex59_int q).
Definition Ex59_splitB q p := p -s (Ex59_splitA' q p).

Lemma Nintcc_exten a b c d:
   a <=c b -> natp b -> Nintcc a b = Nintcc c d ->
   a = c /\ b = d.
Lemma Nintcc_exten_spec q i j: natp q -> natp i ->  (* 4 *)
   Nintcc i (i +c (q -c \1c)) = Nintcc j (j +c (q -c \1c)) ->
   i = j.
Lemma Ex59_intp q i: natp q -> \0c <c q -> natp i -> (* 5 *)
   i +c q = csucc (i +c (q -c \1c)).
Lemma Ex59_interval_prop q j: natp j -> natp q -> \0c <c q ->  (* 7 *)
  forall x, inc x (Nintcc j (j +c (q -c \1c))) <->
       j <=c x /\ x <c j +c q.
Lemma Exercise5_9j1 q x (j := Ex59_int_lb q x):  (* 3 *)
  natp q -> \0c <c q ->  Ex59_intervalp q x ->
  x =  Ex59_int q j /\ natp j.
Lemma Exercise5_9j2 q x (j := Ex59_int_lb q x):  (* 1 *)
  natp q -> \0c <c q ->  Ex59_intervalp q x -> x = Ex59_int q j.
Lemma Exercise5_9j3 q p j:  natp q -> \0c <c q ->  (* 2 *)
  inc j (Ex59_splitA q p) -> (natp j /\ inc (Ex59_int q j)  p).
Lemma Exercise5_9j4 q p: natp q -> \0c <c q ->
   Zo p (Ex59_intervalp q) = Ex59_splitA' q p.   (* 7 *)
Lemma Exercise5_9j5 E n q p k:    (* 9 *)
  natp n -> natp q -> \0c <c q -> cardinal E = q *c n ->
  inc p (Ex59_k_interval E q k) ->
   cardinal (Ex59_splitA q p) = k /\ cardinal (Ex59_splitA' q p) = k.
Lemma Exercise5_9j6 E n q p k:
  natp n -> natp q -> \0c <c q -> cardinal E = q *c n ->
  inc p (Ex59_k_interval E q k) ->
   cardinal (Ex59_splitB q p) = n -c k. (* 8 *)
```

We consider the property (J) of a set J: it is a subset of $\mathbf{N}$ with k elements, such that, if $i \in J$ and j is either qn or another element of J with $i < j$, then $i + q \le j$. The set $J(p)$ satisfies (J).

Consider now any set satisfying (J) and its enumeration e (this is a strictly increasing bijection $I_k \to J$). By induction $qi \le e(i) \le q(n-1)$. Let e' be the function $e'(i) = e(i) - qi$. Then e' belongs to T, the set of increasing functions $I_k \to [0, q(n-k)]$. Conversely, given an element e' of T, and $e(i) = e'(i) + qi$, then the range of e satisfies (J), and e is the enumeration of the range. Note that the cardinal of T has a nice formula.

```
Definition Ex59_Jprop q n k J:=
  [/\ cardinal J = k, sub J Nat,
  (forall i j, inc i J -> inc j J -> i <c j -> i +c q <=c j) &
  (forall j, inc j J -> j +c q <=c q *c n)].

Lemma Exercise5_9k1 n q p k (E:= Nint (q *c n)):  (* 26 *)
  natp n -> natp q -> \0c <c q ->
  inc p (Ex59_k_interval E q k) -> Ex59_Jprop q n k (Ex59_splitA q p).
Lemma Exercise5_9k2 n q k J (e := nth_elt J):
```

```
  natp k -> Ex59_Jprop q n k J ->
  [/\ forall x y, x <c y -> y <c k -> e x <c e y,
      forall x y, x <=c y -> y <c k -> e x <=c e y,
      forall x y, x <c k -> y <c k -> e x = e y -> x = y,
      forall x, x <c k -> inc (e x) J &
      forall y, inc y J -> exists2 i, i <c k & y = e i].  (* 16 *)
Lemma Exercise5_9k3 n q p k (E:= Nint (q *c n))
  (e := nth_elt (Ex59_splitA q p)):  (* 11 *)
  natp n -> natp q -> \0c <c q -> natp k ->
  inc p (Ex59_k_interval E q k) ->
  (forall i, i<c k -> q *c i <=c e i) /\
  (forall j, j <c k -> e j +c q <=c q *c n).
Lemma Exercise5_9k4 n q k J (e:= nth_elt J) (e' := fun i => e i -c q *c i):
  natp q -> natp n -> k <=c n -> Ex59_Jprop q n k J ->
  [/\ forall i, i <c k -> natp (e' i),
   (forall i, i <c k -> e i = e' i +c q *c i),
   (forall i j, i <=c j -> j <c k -> e' i <=c e' j) &
  (forall j, j <c k -> e' j <=c q *c (n -c k)) ].  (* 35 *)
Lemma Exercise5_9k5 n q k J (e:= nth_elt J) (e' := fun i => e i -c q *c i)
  (T := functions_incr (Nint_co k) (Nint_cco \0c (q *c (n -c k)))):
  natp q -> natp n -> k <=c n -> Ex59_Jprop q n k J ->
  inc (Lf e' (Nint k) (Nintcc \0c (q *c (n -c k)))) T /\
  cardinal T = binom ((q *c (n -c k)) +c k) k.  (* 21 *)
Lemma Exercise5_9k6 n q k f
  (J := fun_image (Nint k) (fun i => Vf f i +c q *c i))
  (T := functions_incr (Nint_co k) (Nint_cco \0c (q *c (n -c k)))):
  natp q -> natp n -> k <=c n -> \0c <c q -> inc f T ->
  [/\ Ex59_Jprop q n k J,
    forall i, i <c k -> (nth_elt J i) = Vf f i +c q *c i &
    forall i, i <c k -> (nth_elt J i)  -c q *c i = Vf f i]. (* 61 *)
```

As above, we assume $p \in \mathrm{P}_k$. Given J we may consider the union of the set of intervals whose lower bounds are in J. If $\mathrm{J} = \mathrm{J}(p)$, this is the union of p_J. We denote the complement in E by E_J. If $\mathrm{J} = \mathrm{J}(p)$, this is the union of p'_J (the union of the elements of p that are not intervals). Consider the property (C) of p': it says that p' is it is a partition of E'_J without intervals, formed of $n-k$ sets of cardinal q. The set p'_J satisfies this property.

Assume now that p' satisfies (C). Assume $t \in x$ and $x \in p'$. In particular $t \in \mathrm{E}$ and $t < qn$. Let e be as above, and l the function defined by $e(l(x)-1) < x \le e(l(x))$. This makes sense only if there is $j \in \mathrm{J}$ such that $< x$, in all other cases $l(x)$ is zero. It also makes sense only when there is $j \in \mathrm{J}$ such that $j \le x$. In all other cases, we define $l(x) = k$. The definition is a bit strange, and we show that l satisfies the desired properties. In particular, if $0 < j < k$ and $e(j-1) < x \le e(j)$ then $l(x) = j$.

```
Definition Ex59_compl n q J :=
  Nint (q *c n) -s union (fun_image J (fun j => Nintcc j (j +c (q -c \1c)))).
Definition Ex59_Cprop n q J k p:=
  [/\ inc p (partition_nq (Ex59_compl n q J) q),
   cardinal p = n -c k & forall x, inc x p -> ~ (Ex59_intervalp q x)].
Definition Ex59_pos_in_J J k x :=
   intersection ((Zo (Nint k) (fun i => x <=c nth_elt J i)) +s1 k).

Lemma Exercise5_9l1 n q p (E:= Nint (q *c n)):  (* 14 *)
  natp n -> natp q -> \0c <c q ->
  inc p (partition_nq E q) ->
  Ex59_compl n q (Ex59_splitA q p) = union (Ex59_splitB q p).
```

```
Lemma Exercise5_912 n q p (E:= Nint (q *c n)):  (* 6 *)
  natp n -> natp q -> \0c <c q ->
  inc p (partition_nq E q) ->
  inc (Ex59_splitB q p) (partition_nq (Ex59_compl n q (Ex59_splitA q p)) q).
Lemma Exercise5_913 n q p k (E:= Nint (q *c n)):
  natp n -> natp q -> \0c <c q ->
  inc p (Ex59_k_interval E q k) ->
  Ex59_Cprop n q (Ex59_splitA q p) k (Ex59_splitB q p). (* 7 *)

Lemma Exercise5_914 n q p k J x y:
  natp n -> natp q -> \0c <c q ->
  Ex59_Cprop n q J k p -> inc x p -> inc y x -> y <c q *c n.  (* 4 *)
Lemma Exercise5_915 J x: Ex59_pos_in_J J \0c x = \0c. (* 3 *)
Lemma Exercise5_916 q n k J:
  Ex59_Jprop q n k J -> natp k -> \0c <c k ->
 [/\ forall x,  Ex59_pos_in_J J k x  <=c k,
    forall x,  Ex59_pos_in_J J k x = \0c <-> (forall y, inc y J -> x <=c y),
    forall x, natp x ->
      (Ex59_pos_in_J J k x = k <-> (forall y, inc y J -> y <c x)),
    forall x, (exists2 y, inc y J & x <=c y) ->
      (exists2 y, inc y J & y <c x) ->
     (\0c <c  Ex59_pos_in_J J k x /\  Ex59_pos_in_J J k x <c k) &
    forall x, let i := Ex59_pos_in_J J k x in
     \0c <c i -> i <c k ->
    [/\ inc (nth_elt J i) J,  x <=c (nth_elt J i),
     inc (nth_elt J (cpred i)) J & (nth_elt J (cpred i)) <c x ]]. (* 56 *)
Lemma Exercise5_916bis q n k J j x :
  Ex59_Jprop q n k J -> natp k -> \0c <c k -> natp x ->
  j <c k -> j <> \0c -> (nth_elt J (cpred j)) <c x -> x <=c (nth_elt J j) ->
  Ex59_pos_in_J J k x = j. (* 22 *)
```

We consider now the greatest element of $\mathrm{E_J}$. Define $d(i) = q(n-i-1)$ and let J_i the interval of length q whose lower bound is $d(i)$. Every element of J is at most $d(0)$, and if $d(j) \in \mathrm{J}$, an element of J less than $d(j)$ is at most $d(j+1)$. So, let i be the greatest integer such that $d(j) \in \mathrm{J}$ for $j < i$. Case 1: $qn-1 \in \mathrm{E}'$. This corresponds to $i = 0$. Case 2: $i = k$. This says that $t \in \mathrm{E}'$ implies $t < q(n-k)$ (note that if $k = n$, then E' is empty). Case 3: $0 < i < k$. Note that J is $d(0), d(1), \dots, d(i-1)$, enumerated in decreasing order (there are $k-i$ other smaller elements). One deduces: there is $j < k$, such that, if $x = q(n-k+j)$, then $x = e(j)$, x is non-zero, and $x-1$ is the greatest element of E'.

We now view $\mathrm{E_J}$ as the union of the p'_J. Let x be in this set. Assume $e(l(x)-1) < x \le e(l(x))$ and write $j = l(x)$ (we also have to consider the case where $j = 0$ or $j = k$). We have $e(j-1)+q \le x < e(j)$ (note that elements between $e(j-1)$ and $e(j-1)+q-1$ are in an interval, hence not in $\mathrm{E_J}$. Introducing e' yields $e'(j-1)+qj \le x < e'(j)+qj$, so $e'(j-1) \le x-qj < e'(j)$. Let $\mathrm{V}(x) = x-qj$. This is a strictly increasing function: assume $x < x'$. We have $e(j-1) < x < x' \le e(j')$, so that $e(j-1) < e(j')$ and $j \le j'$. If $j = j'$ it is immediate that $\mathrm{V}(x) < \mathrm{V}(x')$. So otherwise $j < j'$, $j \le j'-1$ and $e'(j) \le e'(j'-1)$. Now $e'(j-1) \le \mathrm{V}(x')$ and $\mathrm{V}(x) < e(j)$. We pretend that it is an order isomorphism onto $\mathrm{I}_{q(n-k}$. It suffices to show that, for the greatest element t we have $\mathrm{V}(t) < q(n-k)$. Consider the three cases studies above. Case 1: t is $qn-1$; the result follows from $l(t) = k$. Case 2: $t < q(n-k)$, the result follows from $\mathrm{V}(t) \le t$. Case 3: $t = x-1$, where $q(n-k+j)$. The other properties of x easily imply $j = l(x-1)$ so that $\mathrm{V}(x-1) = x-1-l(x-1) = q(n-k)-1$.

```
Lemma Exercise5_917 q n k J  (e := nth_elt J) (E’ := (Ex59_compl n q J)):
```

```
  natp n -> natp q -> \0c <c q -> k <=c n ->
  Ex59_Jprop q n k J -> natp k -> \0c <c k ->
  (inc (cpred (q *c n)) E' \/
  ( (forall t, inc t E' -> t <c q *c (n -c k)) \/
    exists j, let x := (q *c ((n -c k) +c j)) in
     [/\ j <c k, x <> \0c, e j = x, inc (cpred x) E' &
     forall t, inc t E' -> t <=c (cpred x)])). (* 176 *)

Lemma Exercise5_918 q n k J p (E':= union p) (e := nth_elt J)  (* 185 *)
  (e':= fun i => e i -c q *c i) (V:=  Ex59_pos_in_J J k)
  (V' := fun x => x -c  q *c (V x))
  (T:= Nint (q *c (n -c k))):
  natp n -> natp q -> \0c <c q -> k <=c n ->
  Ex59_Jprop q n k J -> natp k -> \0c <c k ->
  Ex59_Cprop n q J k p ->
  [/\ forall x, inc x E' -> V x = k -> e'(k -c \1c) +c q *c k <=c x,
    forall x, inc x E' -> \0c <c (V x) -> (V x) <c k ->
     e'( (V x) -c \1c) +c q *c (V x) <=c x /\ x <c  e'(V x) +c q *c (V x),
    forall x, inc x E' ->  q *c (V x) <=c x,
   forall x, inc x E' ->  x =  q *c (V x) +c V' x &
  order_isomorphism (Lf V' E' T) (graph_on cardinal_le E')
     (graph_on cardinal_le T)
   ].
```

Note. Let p' be the image by V of the elements of the partition that are not intervals. This is a partition, but it may have intervals. What properties does it satisfy that the original set did not?

So let's repeat the process. Example. Take $q = 3$, and $n = 5$. Consider (0,8,9), (1,10,14), (2,6,7), (3,4,5), (10,11,12), and the variant (0,10,14), (1,8,9), (2,6,7), (3,4,5), (10,11,12). Here J contains 2 and 10, and if we remove the two intervals, it happens that (2,6,7) becomes an interval. In the first case, the iterated J is 2, 3, 11, and in the second case, it is 0, 1, 2, 3, 11. Conversely, to such a sequence we associate a set E' as follows. We assume that the sequence is $x_1, x_2, \dots, x_s$, and define $\mathrm{E}'_s = \emptyset$. Now E'_{i-1} is the union of E_i and the first q elements, starting with x_i that are not in E, and the result is E'_0. Example: we start with (11,12,13), then add (3,4,5), then (2,6,7); in the second case, we also add (1,8,9) and (0,10,14).

Let T_k be the set of those J with k elements so that E' is a subset of I_{qn}. Let E'' be the complement of E'; it has $q(n-k)$ elements, and there is an iterated V. Let $p \in \mathrm{P}_{\mathrm{E},q}$; this iterated V, applied to the set of elements of p whose least element is not in J, is an element p' of $\mathrm{P}_{\mathrm{S},q,0}$, where $\mathrm{S} = \mathrm{I}_{q(n-k)}$. Now $p \mapsto (\mathrm{J}, p')$ is a bijection $\mathrm{U}_k \to \mathrm{T}_k \times \mathrm{P}_{\mathrm{S},q,0}$., where the sets U_k form a partition of $\mathrm{P}_{\mathrm{E},q}$. Questions: is this statement true? what is T_k ? what is its cardinal?

---

10. Let $q_{n,k}$ be the number of strictly increasing mappings u of $[1,k]$ into $[1,n]$ such that for each even (resp. odd) x, $u(x)$ is even (resp. odd). Show that $q_{n,k} = q_{n-1,k-1} + q_{n-2,k}$ and deduce that

$$q_{n,k} = \binom{[\frac{n+k}{2}]}{k}.$$

Solution. We give here a direct proof. If u is strictly increasing, then $v(x) = u(x) - x$ is increasing (in the main text, we have shown this for functions defined in the interval $[0,p[$, with value in $[0, n+p[$; we extend the function by setting $u(0) = 0$). Note that, if $n < k$, the binomial coefficient is zero, and there is no such function. So assume $k \le n$, and write $n = k + 2p + r$ (where $r = 0$ or $r = 1$). We have $v(x) \le 2p + r$. The assumption is that $v(x)$ is even for all x. Thus $v(x) \le 2p$. Moreover $x \mapsto v(x)/2$ is any increasing function $[1, h] \to [0, p]$. The binomial coefficient is $\binom{k+p}{k}$, thus the conclusion.

```
Lemma even_compare n p: (* 9 *)
  natp p -> evenp n -> n <=c (\2c *c p) +c \1c -> n <=c (\2c *c p).
Lemma cardinal_set_of_increasing_functions5 p n: (* 12 *)
  natp p -> natp n ->
  cardinal(functions_incr (Nint_cco \1c p) (Nint_cco \0c n)) =
  binom (n +c p) p.
Lemma Exercise5_10 n k (* 186 *)
    (o1 := Nint_cco \1c k) (o2 := Nint_cco \1c n)
    (even_odd_fct := fun f =>
       (forall x, inc x (source f) -> evenp x -> evenp (Vf f x))
       /\ (forall x, inc x (source f) -> oddp x -> oddp (Vf f x))):
   natp n  -> natp k ->
   cardinal (Zo (functions_sincr o1 o2) even_odd_fct) =
   binom  ((n +c k) %/c \2c) k.
```

¶ 11. Let E be a set with n elements and let S be a set of signs such that S is the disjoint union of E and a set consisting of a single element f. Suppose that f has weight 2 and that each element of E has weight 0 (Chapter I, Appendix, Exercise 3).

(a) Let M be the set of significant words in $L_0(S)$ which contain each element of E exactly once. Show that if u_n is the number of elements in M, then $u_{n+1} = (4n-2)u_n$, and deduce that

$$u_n = 2.6\dots(4n-6) \qquad (n \ge 2)$$

(This is the number of products of n different terms with respect to a non-associative law of composition).

(b) Let x_i be the ith of the elements of E which appear in a word of M. Show that the number v_n of words of M, for which the sequence x_i is given, is equal to $\binom{2n-2}{n-1}/n$ and satisfies the relation

$$v_{n+1} = v_1 v_n + v_2 v_{n-1} + \cdots + v_{n-1} v_2 + v_n v_1.$$

¶ 12. (a) Let p and q be two integers ≥ 1, let $n = 2p + q$, let E be a set with n elements and let $N = \binom{n}{p} = \binom{n}{p+q}$. Let $(X_i)_{1\le i\le N}$ (resp $(Y_i)_{1\le i\le N}$) be the sequence of all subsets of E which have p (resp $p+q$) elements arranged in a certain order. Show that there exists a bijection ϕ of $[1, n]$ onto itself such that $X_{\phi(i)} \subset Y_i$ for all i. (The method is analogous to that of Exercise 6 of § 4: observe that for each $r \le N$ the number of sets Y_j which contain at least one of X_1, …, X_r is $\ge r$).

(b) Let h, k be two integers ≥ 1, let n be an integer such that $2h + k < n$, let E be a set with n elements and let $(X_i)_{1\le i\le r}$ be a sequence of distinct subsets of E, each having h elements. Show that there exists a sequence $(Y_j)_{1\le j\le r+1}$ of distinct subsets of E, each having $h+k$ elements, such that each Y_j contains at least one X_i and each X_i is contained in at least one Y_j (by induction on n, using (a)).

¶ 13. Let E be set with $2m$ elements, let q be an integer $< m$, and let $\mathscr{F}$ be the set of all subsets $\mathfrak{S}$ of $\mathfrak{P}(\mathrm{E})$ with the following property: if X and Y are two distinct elements of $\mathfrak{S}$ such that $\mathrm{X} \subset \mathrm{Y}$, then $\mathrm{Y} - \mathrm{X}$ has at most $2q$ elements.

(a) Let $\mathfrak{M} = (\mathrm{A}_i)_{1\le i\le p}$ be an element of $\mathscr{F}$ such that $p = \mathrm{Card}(\mathfrak{M})$ is as large as possible. Show that $m - q \le \mathrm{Card}(\mathrm{A}_i) \le m + q$ for $1 \le i \le p$ (Argue by contradiction. Suppose, for example, that there exists indices i such that $\mathrm{Card}(\mathrm{A}_i) < m - q$ and consider those of the A_i for which $\mathrm{Card}(\mathrm{A}_i)$ has the least possible value $m-q-s$ (where $s \ge 1$). Let $\mathrm{A}_1, \dots, \mathrm{A}_r$, say, these sets. Let $\mathfrak{G}$ be the set of subsets of E each of which is the union of some A_i $(1 \le i \le r)$ and a subset of $2q+1$ elements contained in $\mathrm{E}-\mathrm{A}_i$. Show that $\mathfrak{G}$ contains at least $r+1$ elements (cf. Exercise 12), and that if $\mathrm{B}_1, \dots, \mathrm{B}_{r+1}$ are $r+1$ distinct elements of $\mathfrak{G}$, the set whose elements are B_j $(1 \le j \le r+1$ and A_i $(r+1 \le i \le p)$ belongs to $\mathscr{F}$, contrary to the hypothesis.)

(b) Deduce from (a) that the number of elements p of each $\mathfrak{S} \in \mathscr{F}$ satisfies the inequality

$$p \le \sum_{k=0}^{2q} \binom{2m}{m-q+k}.$$

(c) Establish results analogous to those of (a) and (b) when $2m$ or $2q$ is replaced by an uneven number.

¶ 14. Let E be a finite set with n elements, let $(a_j)_{1\le j\le n}$ be the sequence of elements of E arranged in some order, and let $(\mathrm{A}_i)_{1\le i\le m}$ be a sequence of subsets of E.

(a) For each index j, let k_j be the number of indices i such that $a_j \in \mathrm{A}_i$, and let $\mathrm{S}_i = \mathrm{Card}(\mathrm{A}_i)$. Show that

$$\sum_{j=1}^{n} k_j = \sum_{i=1}^{m} s_i.$$

(b) Suppose that for each subset $\{x, y\}$ of two elements of E, there exists exactly one index i such that x and y are contained in A_i. Show that, if $a_j \not\in \mathrm{A}_i$, then $\mathrm{S}_i \le k_j$.

(c) With the hypotheses of (b), show that $m \ge n$ (Let k_n be the least of the numbers k_j. Show that we may suppose that, whenever $i \le k_n$, $j \le k_n$, and $i \ne j$, we have $a_j \not\in \mathrm{A}_i$ and $a_n \not\in \mathrm{A}_j$ for all $j \ge k_n$.)

(d) With the hypotheses of (b), show that $m = n$ if and only if one of the following two alternatives is true: (i) $\mathrm{A}_1 = \{a_1, a_2, \dots, a_{n-1}\}$, $\mathrm{A}_i = \{a_{i-1}, a_n\}$ for $i = 2, \dots, n$; (ii) $n = k(k-1)+1$, each A_i has k elements, and each element of E belongs to exactly k set A_i.

¶ 15. Let E be a finite set, let $\mathfrak{L}$ and $\mathfrak{C}$ be two disjoint non-empty subsets of $\mathfrak{P}(\mathrm{E})$, and let λ, h, k, l be four integers ≥ 1 with the following properties: (i) for each $\mathrm{A} \in \mathfrak{L}$ and each $\mathrm{B} \in \mathfrak{C}$, $\mathrm{Card}(\mathrm{A} \cap \mathrm{B}) \ge \lambda$; (ii) for each $\mathrm{A} \in \mathfrak{L}$, $\mathrm{card}(\mathrm{A}) \ge h$; (iii) for each $\mathrm{B} \in \mathfrak{C}$, $\mathrm{card}(\mathrm{B}) \le k$; (iv) for each $x \in \mathrm{E}$ the number of elements of $\mathfrak{L} \cup \mathfrak{C}$ which contain x is exactly l. Show that $\mathrm{Card}(\mathrm{E}) \le hk/\lambda$. (Let $(a_i)_{1\le i\le n}$ be the sequence of distinct elements of E arranged in some order, and for each i let r_i be the number of elements of $\mathfrak{L}$ to which a_i belongs. Show that, if $\mathrm{Card}(\mathfrak{L}) = s$ and $\mathrm{Card}(\mathfrak{C}) = t$, then we have

$$\sum_{i=1}^{n} r_i \le sh, \qquad \sum_{i=1}^{n} (l - r_i) \le tk, \qquad \sum_{i=1}^{n} r_i(l - r_i) \ge \lambda st.)$$

For Card(E) to be equal to hk/λ it is necessary and sufficient that for each $\mathrm{A} \in \mathfrak{L}$ and each $\mathrm{B} \in \mathfrak{C}$ we have $\mathrm{card}(\mathrm{A}) = h$, $\mathrm{card}(\mathrm{B}) = k$, $\mathrm{Card}(\mathrm{A} \cap \mathrm{B}) = \lambda$, and that there exists an $r \le l$ such that for each $x \in \mathrm{E}$ the number of elements of $\mathfrak{L}$ to which x belongs is equal to r.

16. Let E be a finite set with n elements, let $\mathfrak{D}$ be a non-empty subset of $\mathfrak{P}(\mathrm{E})$, and let λ, k, l be three integers ≥ 1 with the following properties: (i) if A and B are distinct elements of $\mathfrak{D}$, then $\mathrm{Card}(\mathrm{A} \cap \mathrm{B}) = \lambda$; (ii) for each $\mathrm{A} \in \mathfrak{D}$, $\mathrm{card}(\mathrm{A}) \leq k$; (iii) for each $x \in \mathrm{E}$ the number of elements of $\mathfrak{D}$ to which x belongs is equal to l. Show that

$$n(\lambda - 1) \leq k(k-1),$$

and that if $n(\lambda - 1) = k(k-1)$ then $\lambda = k$ and $\mathrm{Card}(\mathfrak{D}) = n$. (Given $a \in \mathrm{E}$, let $\mathfrak{L}$ be the set of all $\mathrm{A} - \{a\}$ where $\mathrm{A} \in \mathfrak{D}$ and $a \in \mathrm{A}$, and let $\mathfrak{C}$ be the set of all $\mathrm{A} \in \mathfrak{D}$ such that $a \notin \mathrm{A}$. Apply the results of Exercise 15 to $\mathfrak{L}$ and $\mathfrak{C}$.)

¶ 17. Let i, h, k be three integers such that $i \geq 1$, $h \geq i$, $k \geq i$. Show that there exists an integer $m_i(h,k)$ with the following properties: for each finite set E with at least $m_i(h,k)$ elements, and each partition $(\mathfrak{X}, \mathfrak{Y})$ of the set $\mathfrak{F}_i(\mathrm{E})$ of subsets of i elements of E, it is impossible that every subset of h elements of E contains a subset $\mathrm{X} \in \mathfrak{X}$ and that every subset of k elements of E contains a subset $\mathrm{Y} \in \mathfrak{Y}$; in other words, if every subset of h elements of E contains some $\mathrm{X} \in \mathfrak{X}$, there exists a subset A of k elements of E such that every subset of i elements of A belongs to $\mathfrak{X}$. (Proof by induction. Show that we may take $m_1(h,k) = h+k-1$, $m_i(i,k) = k$ and $m_i(h,i) = h$ and finally $m_i(h,k) = m_{i-1}(m_i(h-1,k), m_i(h,k-1)) + 1$. If E is a set with $m_i(h,k)$ elements, if $a \in \mathrm{E}$ and $\mathrm{E}' = \mathrm{E} - \{a\}$, show that if the proposition were false, then every subset of $m_i(h-1,k)$ elements of E' would contain a subset X' of $i-1$ elements such that $\mathrm{X}' \cup \{a\} \in \mathfrak{X}$, and that every subset of $m_i(h,k-1)$ elements of E' would contain a subset Y' of $i-1$ elements such that $\mathrm{Y}' \cup \{a\} \in \mathfrak{Y}$.

18. (a) Let E be a finite ordered set with p elements. If m, n are two integers such that $mn < p$, show that E has either a totally ordered subset of m elements or else a free subset (§ 1, Exercise 5) of n elements (use § 4, Exercise 5).

(b) Let h, k be two integers ≥ 1 and let $r(h,k) = (h-1)(k-1)+1$. Let I be a totally ordered set with at least $r(h,k)$ elements. Show that, for each finite sequence $(x_i)_{i \in \mathrm{I}}$ of elements of a totally ordered set E, there exists either a subset H of h elements of I such that the sequence $(x_i)_{i \in \mathrm{H}}$ is increasing, or else a subset K of k elements of I such that the sequence $(x_i)_{i \in \mathrm{H}}$ is decreasing. (Use (a) applied to $\mathrm{I} \times \mathrm{E}$.)

13.7 Section 6.

1. A set E is infinite if and only if for each mapping f of E into E there exists a non-empty set S of E such that $\mathrm{S} \neq \mathrm{E}$ and $f(\mathrm{S}) \subset \mathrm{S}$.

Solution. Assume first that E is finite and has n elements. Let $f(i) = i+1 \bmod n$. Let f^k be the k-th iterate of f. We have $f^k(0) = k$ for $k < n$, and $f^{n-i}(i) = 0$ for $0 < i < n$. This shows that if $f(\mathrm{S}) \subset \mathrm{S}$, and S is non-empty, then $\mathrm{S} = [0, n[$. We deduce that if E is finite, it does not satisfy the property of the exercise.

Assume now E infinite (in particular, it is non-empty and contains some x), and let $f : \mathrm{E} \to \mathrm{E}$ be any function. Let S be the set of all $f^k(x)$, for $k > 0$ (the function $k \mapsto f^k(x)$ is defined by induction). We have obviously $f(\mathrm{S}) \subset \mathrm{S}$. If $x \notin \mathrm{S}$, we have $\mathrm{S} \neq \mathrm{E}$. Otherwise $x = f^{n+1}(x)$, for some n. By induction $k \mapsto f^k(x)$ is periodic, with period n. By Euclidean division, every element of S has the form $f^{i+1}(x)$ for $i < n$. This shows that S is finite. If E is infinite, we deduce $\mathrm{S} \neq \mathrm{E}$.

```
Lemma Exercise_6_1 E: infinite_set E <-> (* 124 *)
  (forall f, function f -> source f = E -> target f = E ->
    exists S, [/\ sub S E, nonempty S, S <> E & sub (Vfs f S) S]).
```

Alternate proof. Let $f : \mathrm{E} \to \mathrm{E}$ be a function. Assume $x \in \mathrm{E}$, and let S be the intersection of all sets invariant by f that contain $f(x)$. Then S is invariant by f, contains $f(x)$, and any invariant set that contains $f(x)$ is a superset of S. We pretend that $x \in \mathrm{S}$ implies that S is finite. We deduce: if E is infinite, f has a non-trivial invariant set. In effect, if E is infinite, there is such an x, thus such an S. We have $\mathrm{S} \neq \mathrm{E}$, for otherwise we would have $x \in \mathrm{S}$, thus $\mathrm{S} = \mathrm{E}$ is finite, absurd.

In the first part of the report, we defined “chains” as lists with at least two elements: $(x_{n+1}, x_n, x_{n-1}, \ldots, x_0)$. We have two constructors: given$a$, b, we can construct the chain (a, b), given a and $(x_{n+1}, x_n, \ldots, x_0)$ we construct $(a, x_{n+1}, \ldots, x_0)$. The head is x_{n+1}, the tail is x_0. We say that the chain is linked by R if $\mathrm{R}(x_{i+1}, x_i)$ holds. In our case, we assume $x_{i+1} = f(x_i)$. Let's say that $(x_{k+1}, x_{k1}, \ldots, x_0)$ is a subchain of $(x_{n+1}, x_n, \ldots, x_0)$ when $k <= n$.

The definition is a bit tricky; we show here that if p is a subchain of q, if q is linked by R, then q is linked by R, and p and q have the same tail. Assume that our chains are linked by f. Then $x_n = f^n(x_0)$. We state this as: if a is the tail of p, then $(f(a), a)$ is a subchain of p, and if q is a subchain of p, then either $p = q$ or $f(h_q) :: q$ is a subchain of p (where h_q is the head of q).

The value of a chain $(x_{k+1}, x_k, \ldots, x_0)$ is the set of all x_i for $i > 0$. An element x is in the value of the chain p, if and only if it is the head of a subchain of p. Fix some x_0. Let S be the set of all heads of chains (chained by f) with tail x_0. This set contains $f(x_0)$ and is invariant by f. Moreover, if A is any set invariant by f that contains $f(x_0)$, it contains all elements of S. Thus, S is the least subset of E invariant by f that contains $f(x_0)$. We pretend that if $x_0 \in \mathrm{S}$, then S is finite. So, assume that p is chain, linked by f, whose head and tail are x_0. Let A be the value of p. Then, A is obviously a subset of S. We have $f(x_0) \in \mathrm{A}$, and A is invariant by f (assume that x is the head of a subchain q of p; if $p = q$, then $x = x_0$ and $f(x) \in \mathrm{A}$; otherwise, adding $f(x)$ in front of q yields a subchain of p, so that $f(x) \in \mathrm{A}$). It follows $\mathrm{A} = \mathrm{S}$, and S is finite.

```
Fixpoint chain_val x :=
  match x with chain_pair u v => singleton u
    | chain_next u v => chain_val v +s1 u
 end.
Fixpoint sub_chain x y :=
  match y with
    chain_pair u v => x = y
   | chain_next u v =>
     x = y \/ sub_chain x v
end.
Lemma sub_chainedP R p q: sub_chain p q -> chained_r R q -> (* 6 *)
     chained_r R p /\ chain_tail p = chain_tail q.
Lemma chained_prop1 g a c: (* 2 *)
  chained_r (fun a b => a = g b) c -> chain_tail c = a ->
      sub_chain (chain_pair (g a) a) c.
Lemma chained_prop2 g p c: (* 4 *)
  chained_r (fun a b => a = g b) c -> sub_chain p c ->
   p = c \/ sub_chain (chain_next (g (chain_head p)) p) c.
Lemma chain_valP x i: inc i (chain_val x) <-> (* 10 *)
  (exists2 p, sub_chain p x & i = chain_head p).
Lemma chain_val_finite x: finite_set (chain_val x). (* 2 *)
```

```
Lemma Exercise_6_1bis E f: (* 44 *)
    infinite_set E -> function_prop f E E ->
    exists S, [/\ sub S E, nonempty S, S <> E & sub (Vfs f S) S].
```

---

2. Show that, if $\mathfrak{a}$, $\mathfrak{b}$, $\mathfrak{c}$ and $\mathfrak{d}$ are four cardinals such that $\mathfrak{a} < \mathfrak{c}$ and $\mathfrak{b} < \mathfrak{d}$ then $\mathfrak{a}+\mathfrak{b} < \mathfrak{c}+\mathfrak{d}$ and $\mathfrak{a}\mathfrak{b} < \mathfrak{c}\mathfrak{d}$. (cf. Exercise 21 (c)).

Solution. Exercise 21.c gives an example where $\mathfrak{a}^{\mathfrak{b}} < \mathfrak{c}^{\mathfrak{d}}$ is false. In this case, we use commutativity, and may assume $\mathfrak{c} \le \mathfrak{d}$. We may also assume $\mathfrak{d}$ infinite, for otherwise all four cardinals are finite. We use the property that $\mathrm{A}+\mathrm{B} = \mathrm{AB} = \mathrm{B}$ if B is an infinite cardinal and $0 < \mathrm{A} \le \mathrm{B}$. We use it first with $\mathrm{B} = \mathfrak{d}$, in order to simplify the goal to $\mathfrak{a}+\mathfrak{b} < \mathfrak{d}$ and $\mathfrak{a}\mathfrak{b} < \mathfrak{d}$. This is obvious if none of $\mathfrak{a}$, $\mathfrak{b}$ is infinite. Otherwise, we apply the previous rule (using commutativity if needed).

```
Lemma Exercise6_2 a b c d:  (* 42 *)
  a <c c -> b <c d -> ((a +c b) <c (c +c d) /\ (a *c b) <c (c *c d)).
```

---

3. If E is an infinite set, the subsets of E which are equipotent to E is equipotent to $\mathfrak{P}(\mathrm{E})$ (use Proposition 3 of no. 4).

Solution. If n is the cardinal of E, we have $n = n+n$ so that there is a bijection $f : \mathrm{E}_1 \cup \mathrm{E}_2 \to \mathrm{E}$, where $\mathrm{E}_1 = \mathrm{E} \times \{\alpha\}$ and $\mathrm{E}_2 = \mathrm{E} \times \{\beta\}$. For each subset X of E, let $\bar{\mathrm{X}} = \mathrm{X} \times \{\alpha\}$, and let $g(\mathrm{X}) = f\langle \bar{\mathrm{X}} \cup \mathrm{E}_2\rangle$. This a subset of E, and its cardinal is at least the cardinal of $f(\mathrm{E}_2)$, which is the cardinal of E, so that $g(\mathrm{X})$ is equipotent to E, thus is in the set Q of subsets of E equipotent to E. The conclusion follows from the injectivity of g and the relation $\mathrm{Q} \subset \mathfrak{P}(\mathrm{E})$.

```
Lemma Exercise6_3 E: infinite_set E -> (* 55 *)
  (\Po E) \Eq (Zo (\Po E) (fun z => z \Eq E)).
```

---

4. If E is an infinite set, the set of all partitions of E is equipotent to $\mathfrak{P}(\mathrm{E})$ (associate a subset of $\mathrm{E} \times \mathrm{E}$ with each partition of E).

Solution. Let ϖ be a partition, and $\tilde{\varpi}$ be the union of all $\mathrm{A} \times \mathrm{A}$ with $\mathrm{A} \in \varpi$. We know that $\tilde{\varpi} \supset \tilde{\varpi}'$ is an order (see chapter one), so that the function $\varpi \mapsto \tilde{\varpi}$ is injective. Let Q be the set of partitions; this shows $\mathrm{Card}(\mathrm{Q}) \le \mathrm{Card}(\mathfrak{P}(\mathrm{E} \times \mathrm{E}))$, hence $\mathrm{Card}(\mathrm{Q}) \le \mathrm{Card}(\mathfrak{P}(\mathrm{E}))$.

Conversely, consider the mapping $f : \mathrm{I} \mapsto \{\mathrm{I}, \mathrm{E}-\mathrm{I}\}$. Note that $f(\mathrm{I}) = f(\mathrm{E}-\mathrm{I})$ so that f is obviously not injective. Since E is infinite there exists $y \in \mathrm{E}$. Let $\mathrm{F} = \mathrm{E} - \{y\}$. Then f is injective on $\mathfrak{P}(\mathrm{F})$. Assume moreover I non-empty. Then $f(\mathrm{I})$ is a partition of E. All that remains to do is to show that $\mathrm{Card}(\mathfrak{P}(\mathrm{E}-\{y\}) - \{\emptyset\}) = \mathrm{Card}(\mathfrak{P}(\mathrm{E}))$. This shows that the set of all mappings of E onto F and the set of all mappings of E into F are equipotent.

```
Lemma card_powerset_rw x y: cardinal x = cardinal y ->
   cardinal (\Po x) = cardinal (\Po y).  (* 1 *)
Lemma infinite_powerset E: (* 3 *)
  infinite_set E -> infinite_set (\Po E).
Lemma Exercise6_4a E: infinite_set E ->
  cardinal (\Po (coarse E)) = cardinal (\Po E).  (* 3 *)
Lemma Exercise6_4 E: infinite_set E -> (* 49 *)
  (partitions E) \Eq (\Po E).
```

---

5. If E is an infinite set, the set of all permutations of E is equipotent to $\mathfrak{P}(\mathrm{E})$. (Use Proposition 3 of No. 4 to show that, for each subset A of E whose complementary does not consist of a single element, there exists a permutation f of E such that A is the set of elements of E which are invariant under f.)

Solution. Let's say that f is a derangement of E if f is a permutation of E and if the set of elements of E which are invariant under f is empty. Let D(E) be the set of derangements of E. We also show that D(E) is equipotent to $\mathfrak{P}(\mathrm{E})$, whenever E is infinite.

Let G be the set of functions defined on a subset X of E. Then $\mathrm{card}(\mathrm{F}^{\mathrm{E}}) \le \mathrm{Card}(\mathrm{G}) \le \mathrm{Card}(\mathfrak{P}(\mathrm{E}\times\mathrm{F}))$. If E is non-empty, F infinite, $\mathrm{Card}(\mathrm{E}) \le \mathrm{Card}(\mathrm{F})$ then $\mathrm{F}\times\mathrm{E}$ is equipotent to F. Thus $\mathrm{Card}(\mathrm{G}) \le \mathrm{Card}(\mathfrak{P}(\mathrm{F}))$. In particular, if E is an infinite set, and P is the set of permutations of E, we have $\mathrm{Card}(\mathrm{P}) \le \mathrm{Card}(\mathfrak{P}(\mathrm{E}))$. It follows $\mathrm{Card}(\mathrm{D}) \le \mathrm{Card}(\mathfrak{P}(\mathrm{E}))$.

```
Lemma product2_infinite3 E F: nonempty E -> (* 7 *)
   (cardinal E) <=c (cardinal F) -> infinite_set F ->
   (F \times E) \Eq F.

Lemma Exercise6_5a E F: (* 2 *)
   (cardinal (functions E F)) <=c  (cardinal (sub_functions E F)).
Lemma Exercise6_5b E F:  (* 7 *)
  (cardinal (sub_functions E F))
   <=c (cardinal (\Po (product E F))).
Lemma Exercise6_5c E: infinite_set E -> (* 18 *)
 (cardinal (permutations E)) <=c  (cardinal (\Po E)).
Lemma Exercise6_5d E: infinite_set E ->  (* 4 *)
  (cardinal (derangements E)) <=c  (cardinal (\Po E)).
```

Assume that $h : \mathrm{E} \to \mathrm{F}$ is a bijection and f is a derangement of F. Then $h^{-1} \circ f \circ h$ is a derangement of E. If follows that D(E) and D(F) are equipotent. We pretend that if E is not a singleton, it has at least one derangement.

Proof. If E is infinite, it is equipotent to $\mathrm{E}\times\{0,1\}$. This set has $(x,i) \mapsto (x, 1-i)$ as derangement. If E is finite, and has at least two elements, it is equipotent to an interval $[0, n+1]$, that has $i \mapsto i+1$ (modulo $n+1$) as derangement. Finally, the identity function is a derangement of the empty set.

```
Lemma Exercice6_5e E F h: (* 16 *)
   bijection h -> source h = E -> target h = F ->
   (forall f, inc f (derangements F) ->
    inc ((inverse_fun h) \co  (f \co h)) (derangements E)).
Lemma Exercice6_5f E F: E =c F -> (* 32 *)
 (derangements E) =c (derangements F).
```

```
Lemma Exercice6_5g E: (* 67 *)
  singletonp E \/ nonempty (derangements E).
```

The set of derangements of E is equipotent to $\mathfrak{P}(\mathrm{E})$. Let $\mathrm{F} = \mathrm{E} \times \{0,1,2\}$. Let H be a subset of E. We define $f_{\mathrm{H}}(x,i) = (x, i_{\mathrm{H}})$ where $x \in \mathrm{E}$, $i \in \{0,1,2\}$ where i_{H} is $i+1$ or $i-1$, modulo 3, depending on whether x is in H or not. Note that $f(f(f(x))) = x$ so that f is a bijection. It obviously has no fix-point. We have $x \in \mathrm{H}$ if and only if $\mathrm{pr}_2 f_{\mathrm{H}}(x,0) = 1$, so that $\mathrm{H} \mapsto f_{\mathrm{H}}$ is an injection from $\mathfrak{P}(\mathrm{E}) \to \mathrm{D}(\mathrm{F})$. The conclusion follows from $\mathrm{Card}(\mathrm{D}(\mathrm{F})) = \mathrm{Card}(\mathrm{D}(\mathrm{E})) \le \mathrm{Card}(\mathfrak{P}(\mathrm{E}))$.

One deduces that the set of permutations of E is equipotent to $\mathfrak{P}(\mathrm{E})$. Alternate proof: Let S be the set of singletons of E, and $\mathrm{H} = \mathfrak{P}(\mathrm{E}) - \mathrm{S}$. Since S is equipotent to E, the Cantor theorem says that H is equipotent to $\mathfrak{P}(\mathrm{E})$. Let $\mathrm{F} \in \mathrm{H}$. Since the complement of H is not a singleton there exists a derangement f of $\mathrm{E} - \mathrm{H}$; extend it to a function $f : \mathrm{E} \to \mathrm{E}$ by $f(x) = x$ for $x \in \mathrm{H}$. Then H is the set of fix-points of f and $\mathrm{H} \mapsto f$ is an injection. The conclusion is as above.

```
Lemma Exercise6_5h E: infinite_set E -> (* 69 *)
   (permutations E) =c (\Po E).
Lemma Exercice6_5i E: infinite_set E -> (* 57 *)
   (derangements E) =c (\Po E).
```

---

6. Let E, F be two infinite sets such that $\mathrm{Card}(\mathrm{E}) \le \mathrm{Card}(\mathrm{F})$. Show that (i) the set of all mappings of E onto F, (ii) the set of all mappings of E into F, and (iii) the set of all mappings of subsets of E into F are all equipotent to $\mathfrak{P}(\mathrm{F})$.

Note. If $\mathrm{Card}(\mathrm{E}) < \mathrm{Card}(\mathrm{F})$ there is no surjective function $\mathrm{E} \to \mathrm{F}$. For this reason, we assume $\mathrm{Card}(\mathrm{F}) \le \mathrm{Card}(\mathrm{E})$. Then E is the disjoint union of two sets that are respectively equipotent to E and F. If F is non-empty, E is equipotent to $\mathrm{E} \times \mathrm{F}$.

If F is empty, there is no function $\mathrm{E} \to \mathrm{F}$, and the result is false. If F has a single element, the set of functions $\mathrm{E} \to \mathrm{F}$ is equipotent to E and the result is false. The result is true if F has at least two elements.

Solution. Assume $f_1 : \mathrm{G} \to \mathrm{E}$ and $f_2 : \mathrm{E} - \mathrm{G} \to \mathrm{F}$ be two surjective functions, where G is some subset of E. If f is any function, we define a function g as follows. If $x \in \mathrm{G}$, we define $g(x) = f(f_1(x))$, otherwise $g(x) = f_2(x)$. Note that g is surjective since f_2 is surjective. The mapping $f \mapsto g$ is injective (since f_1 is surjective). This shows that the sets in (i) and (ii) are equipotent.

Let A be the set of functions $\mathrm{E} \to \mathrm{F}$, and B the set of functions $\mathrm{X} \to \mathrm{F}$, where X is a subset of E, and let C be the power set of E. We have $\mathrm{A} \subset \mathrm{B}$ so that $\mathrm{Card}(\mathrm{A}) \le \mathrm{Card}(\mathrm{B})$. Let a and b be two distinct elements of F; if $\mathrm{X} \subset \mathrm{E}$ we consider the function that maps an element of X to a, other elements of E to b. This yields an injection $\mathrm{C} \to \mathrm{A}$ and shows $\mathrm{Card}(\mathrm{C}) \le \mathrm{Card}(\mathrm{A})$. To each element of B we associate its graph, a subset of $\mathrm{E} \times \mathrm{F}$. This shows $\mathrm{Card}(\mathrm{B}) \le \mathrm{Card}(\mathfrak{P}(\mathrm{E} \times \mathrm{F})) = \mathrm{Card}(\mathrm{C})$. Thus, A, B and C are equipotent.

```
Section Exercise6_6.
Variables E F: Set.
Hypothesis Einf: infinite_set E.
Hypothesis leFE:  (cardinal F) <=c (cardinal E).
```

```
Hypothesis Finf:  exists a b, [/\ inc a F, inc b F & a <> b].

Lemma Exercise6_6a: (* 26 *)
  exists G, sub G E /\ G =c E /\  (E -s G) =c F.
Lemma Exercise6_6b: (* 33 *)
  (functions E F) =c  (surjections E F).
Lemma Exercise6_6c (p:= \Po E) :  (* 27 *)
  (functions E F)  \Eq p /\ (sub_functions E F) \Eq p.
End Exercise6_6.
```

---

7. Let E, F be two infinite sets such that Card(E) < Card(F). Show that the set of all subsets of F which are equipotent to E and the set of all injections of E into F are both equipotent to the set F^E of all mappings of E into F (for each mapping f of E into F, consider the injection $x \mapsto (x, f(x))$ of E into E × F).

Note. The result is true if Card(E) ≤ Card(F), F infinite. If E is empty, the three sets are singletons, and the result is true. Otherwise F is equipotent to E × F.

Solution. Let A be the set of functions E → F, B the set of injections E → F and C the set of subsets of F equipotent to E. To $f \in B$ we associate the range of its graph. This set is equipotent to E, this is in C, and all elements of C have this form. We deduce Card(C) ≤ Card(B) (this is true, whatever E and F).

Consider a bijection $g : F \times E \to F$. If $f : E \to F$ is a function, we consider $h : x \mapsto g((f(x), x))$. This is an injection and $f \mapsto h$ is injective. This shows that A and B have the same cardinal.

We consider now a bijection $g : E \times F \to F$. If f is any function E → F, G is graph, then $g\langle G\rangle$ is a subset of F equipotent to E. The mapping $f \mapsto G$ is injective and so is the mapping $f \mapsto g\langle G\rangle$. This shows Card(A) ≤ Card(C).

```
Lemma image_by_fun_injective f u v: (* 8 *)
  injection f -> sub u (source f) -> sub v (source f) ->
  Vfs f u = Vfs f v -> u = v.
Lemma Exercise6_7a E F  (B := injections E F)  (* 25 *)
    (C := Zo (\Po F)(fun x => x =c E)):
     (cardinal C) <=c (cardinal B).
Lemma Exercise6_7b E F (A:= functions E F) (B := injections E F)  (* 68 *)
    (C:= Zo (\Po F)(fun x => x =c b E)):
  infinite_set F -> (cardinal E) <=c (cardinal F) ->
  (cardinal A = cardinal B /\  cardinal A = cardinal C).
```

---

8. Show that the set of well-orderings on an infinite set E (and *a fortiori* the set of orderings on E) is equipotent to $\mathfrak{P}(E)$ (Use Exercise 5).

Solution. Consider the following sets. A is the set of orderings, B the set of well-orderings, C the set of permutations, D the power set of E, and D_2 the power set of E × E. Let a, b, etc., their cardinals. Consider a well-ordering ≤ of E. If $f \in C$, we consider the relation Γ_f defined by $f(x) \le f(y)$; this is a well-ordering on E. By uniqueness of isomorphisms of well-orderings the mapping $f \mapsto \Gamma_f$ is injective, so that $c \le b$. We have obviously $b \le a \le d_2$. If E is infinite we have $d = d_2 = c$, which proves the theorem.

```
Lemma Exercise6_8b E:  (* 59 *)
  (cardinal (permutations E))  <=c
  (cardinal (Zo (\Po (coarse E)) (fun r => worder_on r E))).
Lemma Exercise6_8c E: infinite_set E -> (* 17 *)
  let s1 := Zo (\Po (coarse E)) (fun r => order_on r E) in
    let s2 := Zo (\Po (coarse E)) (fun r = worder_on r E) in
  (s1 =c s2 /\ s2 =c (\Po E)).
```

---

9. Let E be a non-empty well-ordered set in which every element x other than the least element of E has a predecessor (the greatest element of $]\leftarrow, x[$). Show that E is isomorphic to either **N** or an interval $[0, n[$ of **N** (remark that every segment $\neq$ E is finite by using Proposition 6 of No. 5; then use Theorem 3 of § 2, no. 5).

Solution. We first restate Theorem 3 as: if E and E′ are well-ordered sets, either they are isomorphic, or E is isomorphic to a segment S_x of E′ or E′ is isomorphic to a segment S_x of E. We then state a lemma that says that a segment S_x of **N** (with the induced order) is nothing else than the open interval $[0, x[$ (with its usual ordering). From this, it follows that, any well-ordered set is isomorphic to **N** or to an interval of **N**, or else there is a segment S_x of E which is isomorphic to **N**.

Let's prove the exercise. The assumption $E \neq \emptyset$ is not needed (it suffices to take $n = 0$). The previous remark shows that either the conclusion is true, or there exists some x and a isomorphism $f : S_x \to \mathbf{N}$. Note that x cannot be the least element of E, since this would imply $S_x = \emptyset$ hence $\mathbf{N} = \emptyset$. On the other hand, if $y \in S_x$ then then $y < f^{-1}(1 + f(y))$, so that S_x cannot have a greatest element. There is no need to use Proposition 6 of No. 5.

```
Lemma Exercise6_9a n: natp n -> (* 14 *)
   (int_co n = induced_order Nat_order (segment Nat_order n)).
Lemma Exercise6_9  r: worder r -> (* 26 *)
  (forall x, inc x (substrate r) ->
    (least r x \/
    has_greatest (induced_order r (segment r x)))) ->
  r \Is Nat_order
  \/ (exists2 n, natp n & r \Is (Nint_co n)).
```

¶ 10. Let ω or ω_0 denote the ordinal Ord(**N**) (§ 2, Exercise 14). The set of all integers is then a well-ordered set isomorphic to the set of all ordinals $< \omega$. For each integer n we denote again by n (by abuse of language) the ordinal $\mathrm{Ord}([0, n[)$.

(a) Show that for each cardinal $\mathfrak{a}$ the relation "ξ is an ordinal and $\mathrm{Card}(\xi) < \mathfrak{a}$" is collectivizing (use Zermelo's theorem). Let $W(\mathfrak{a})$ denote the set of all ordinals ξ such that $\mathrm{Card}(\xi) < \mathfrak{a}$.

(b) For each ordinal $\alpha > 0$ define a function f_α on the well-ordered set $O'(\alpha)$ of ordinals $\le \alpha$ by transfinite induction as follows: $f_\alpha(0) = \omega_0 = \omega$, and for each ordinal ξ such that

$0 < \xi \le \alpha$, $f_\alpha(\xi)$ is the least upper bound (§ 2, Exercise 14 (d)) of the set of ordinals ζ such that $\mathrm{Card}(\zeta) \le \mathrm{Card}(f_\alpha(\eta))$ for at least one ordinal $\eta < \xi$. Show that if $0 \le \eta < \xi \le \alpha$, then $\mathrm{Card}(f_\alpha(\eta)) < \mathrm{Card}(f_\alpha(\xi))$ and that, if $\xi \le \alpha \le \beta$, then $f_\alpha(\xi) = f_\beta(\xi)$. Put $\omega_\alpha = f_\alpha(\alpha)$; ω_α is said to be the *initial ordinal with index* α. We have $\omega_\alpha \ge \alpha$. Put $\aleph_\alpha = \mathrm{Card}(\omega_\alpha)$; $\aleph_\alpha$ is said to be the *aleph of index* α. In particular, $\aleph_0 = \mathrm{Card}(\mathbf{N})$.

(c) Show that for each infinite cardinal $\mathfrak{a}$, the least upper bound λ of the set of ordinals $\mathrm{W}(\mathfrak{a})$ is an initial ordinal ω_α, and that $\mathfrak{a} = \aleph_\alpha$ (consider the least ordinal μ such that $\omega_\mu \ge \lambda$); in other words, ω_α is the least ordinal ξ such that $\mathrm{Card}(\xi) = \aleph_\alpha$. For each ordinal α the mapping $\xi \mapsto \aleph_\xi$, defined on $\mathrm{O}'(\alpha)$, is an isomorphism of the well-ordered set $\mathrm{O}'(\alpha)$ onto the well-ordered set of cardinals $\le \aleph_\alpha$; in particular $\aleph_{\alpha+1}$ is the least cardinal $> \aleph_\alpha$. Show that, if α has no predecessor, then for every strictly increasing mapping $\xi \mapsto \sigma_\xi$ of an ordinal β into α such that $\alpha = \sup_{\xi<\beta} \sigma_\xi$, we have

$$\sum_{\xi<\beta} \aleph_{\sigma_\xi} = \aleph_\alpha.$$

(d) Deduce from (c) that ω_ξ is a normal ordinal functional symbol (§ 2, Exercise 17).

Note The mapping $\xi \mapsto \aleph_\xi$ is an isomorphism onto the well-ordered set of infinite cardinals $\le \aleph_\alpha$. The word "infinite" can be omitted if $\alpha > 0$, but is required for $\alpha = 0$. More comments below.

Solution.

(a) is `ord_bound_coll`.

(b) the quantity ω_x is denoted by `omega_fct`. The two main properties are `aleph_pr1` and `aleph_pr4`. Lemma `aleph_pr6` says that this is a strict increasing function.

(d) is `aleph_pr11`. Note that we deduce (c) from (d).

(c) The first claim is `aleph_pr7`. We deduce that ω_x is a cardinal, and $\aleph_x = \omega_x$. Lemma `aleph_pr10` asserts that $\aleph_{\alpha+1}$ is the cardinal successor of $\aleph_\alpha$. The last claim (valid when α is non-zero) is `aleph_sum_pr6`.

We show here that $\xi \mapsto \aleph_\xi$ is an order isomorphism $\mathrm{O}'(\alpha) \to \mathrm{T}(\alpha)$, where $\mathrm{T}(\alpha)$ is the set of infinite cardinals $\le \aleph_\alpha$. Note that $\mathrm{O}'(\alpha)$ is the ordinal successor of α. Since ω_ξ is a cardinal, we have $\aleph_\xi = \omega_\xi$. Since $\le_{\mathrm{Card}}$ and $\le_{\mathrm{Ord}}$ are the same, and ω_ξ is strictly increasing, all we need to show is that the function is well-defined and surjective.

```
Lemma aleph_pr9 x: ordinalp x -> (* 35 *)
  let y:= (omega_fct x) in
  let src := (osucc x) in
   let trg := Zo (cardinals_le y) infinite_c  in
  order_isomorphism
    (Lf (fun z => (omega_fct z)) src trg)
    (ole_on src)(ole_on trg).
```

¶ 11. (a) Show that the ordinal ω is the least ordinal > 0 which has no predecessor, that ω is indecomposable (§ 2, Exercise 16), and that for each ordinal $\alpha > 0$, $\alpha\omega$ is the least indecomposable ordinal which is $> \alpha$ (note that $n\omega = \omega$ for each integer n). Deduce that

$$(\alpha + 1)\omega = \alpha\omega \text{ for each } \alpha > 0.$$

(b) Deduce from (a) that an ordinal is indecomposable if and only if it is of the form ω^β (use Exercise 18 (d) of § 2).

Solution.

(a) Lemma `omega0_limit1` says that ω is limit ordinal and `omega0_limit2` says that it is the least limit ordinal. Lemma `indecomp_omega` says that ω is indecomposable. Lemma `indecomposable_prod2` says that $\alpha\cdot\omega$ is the least indecomposable ordinal $> \alpha$. The last claim is `indecomposable_prod3`.

(b) This relies on the existence of the Cantor normal form (see next exercise). Let α be a non-zero ordinal, write it as $\alpha = \omega^\beta + r$, where r is a sum of powers of ω. By uniqueness of the representation, $r < \alpha$. Thus, if α is indecomposable we must have $\alpha = \omega^\beta$. Conversely, ω^β is indecomposable, since $\omega^a + \omega^b = \omega^b$ for $a < b$.

Note that ω^β is the greatest indecomposable ordinal $\le \alpha$. This shows Exercise 16 (d) and (e) of no. 2.

¶ 12. (a) Show that for each ordinal α and each ordinal $\gamma > 1$, there exist two finite sequences of ordinals (λ_i) and (μ_i) $(1 \le i \le k)$ such that

$$\alpha = \gamma^{\lambda_1}\mu_1 + \gamma^{\lambda_2}\mu_2 + \ldots + \gamma^{\lambda_k}\mu_k,$$

where $0 < \mu_i < \gamma$ for each i, and $\lambda_i > \lambda_{i+1}$ for $1 \le i \le k-1$ (use Exercise 18 (d) of § 2 and Exercise 3 of § 4). Moreover the sequences (λ_i), (μ_i) are uniquely determined by these conditions. In particular there exists a unique finite decreasing sequence $(\beta_j)_{1\le j\le m}$ such that

$$\alpha = \omega^{\beta_1} + \omega^{\beta_2} + \cdots + \omega^{\beta_m}.$$

Let $\phi(\alpha)$ denote the greatest ordinal ω^{β_1} in this sequence.

(b) For each integer n let $f(n) \le n!$ be the greatest number of elements in the set of ordinals of the form $\alpha_{\sigma(1)} + \alpha_{\sigma(2)} + \cdots + \alpha_{\sigma(n)}$, where $(\alpha_i)_{1\le i\le n}$ is an arbitrary sequence of n ordinals and σ runs through the set of permutations of the interval $[1, n]$. Show that

$$f(n) = \sup_{1\le k\le n-1} (k.2^{k-1} + 1) f(n-k). \tag{1}$$

(Consider first the case where all the $\phi(\alpha_i)$ are equal and show that the largest possible number of distinct ordinals of the desired form is equal to n, by using Exercise 16 (a) of § 2. Then use induction on the number of ordinals α_i for which $\phi(\alpha_i)$ takes the least possible value among the set of ordinals $\phi(\alpha_j)$ $(1 \le j \le n)$.) Deduce from (1) that for $n \ge 20$ we have $f(n) = 81 f(n-5)$.

(c) Show that the $n!$ ordinals $(\omega+\sigma(1))(\omega+\sigma(2))\ldots(\omega+\sigma(n))$ where σ runs through the set of permutations of the interval $[1, n]$, are all distinct.

Comment. Point (a) is the subject of Section 11.12 (existence and uniqueness of the Cantor Normal Form). Point (b) is the object of section 11.13. If in (c) we replace the factor by $\omega\cdot\sigma(i)+1$, then (c) holds trivially by the uniqueness of the Cantor product form.

Solution. If a is a non-zero integer, then $a\cdot(\omega+1) = \omega + a$. We deduce, in the case of three integers

$$(\omega + a)\cdot(\omega + b)\cdot(\omega + c) = a\cdot(\omega+1)\cdot b\cdot(\omega+1)\cdot c\cdot(\omega+1)$$

Let $a = \sigma(1)$, $b = \sigma(2)$ and $c = \sigma(3)$. Multiply the RHS on the right by 1, on the left by ω^0. We obtain an expression of the form (11.48), where $k = 4$, $\nu_4 = 1$, the other ν_i are one, $\mu_1 = 1$, and otherwise $\mu_i = \sigma(5-i)$. By uniqueness of the Cantor Normal Product Form, the product uniquely defines σ.

Given a function f (that will correspond to σ), we construct the functions that correspond to μ and ν. They are a bit tricky.

```
Definition Ex6_12_e (n: Set) i:=  (Yo (i = n) \0o \1o).
Definition Ex6_12_c (f: fterm) n i :=
     (Yo (i = n) (Yo (n = \0c) \1o (f \1c))
            (Yo (csucc i = n) \1o (f (csucc (csucc i))))).
Definition  Ex6_12_ec (f: fterm) n i := J (Ex6_12_e n i) (Ex6_12_c f n i).
Definition  Ex6_12_ax f n:=
   (forall i, inc i  (Nint1c n) -> posnatp (f i)).
Definition  Ex6_12_v f n:= CNFpv (Ex6_12_ec f n) n.
```

Let $\mathrm{V}(f)$ be the value of the CNFP associated to f. If $0 < f(i) < \omega$ for $i \le n$, then we can apply the uniqueness theorem, and get: if $\mathrm{V}(f) = \mathrm{V}(g)$ then $f = g$ on $[1, n]$. We then rewrite $\mathrm{V}(f)$ as $f(1)$ times a complicated product, by induction $\mathrm{V}(f)$ is the product of all $\omega + f(i))$, for $i \in [1, n]$.

```
Lemma Exercise6_12a n: natp n -> n <> \0c -> (* 2 *)
  (inc \1c (Nint1c n) /\ inc n (Nint1c n)).
Lemma Exercise6_12b f n: Ex6_12_ax f n -> natp n ->  (* 9 *)
  CNFp_ax  (Ex6_12_ec f n) n.
Lemma Exercise6_12c f g n:   (* 14 *)
  natp n -> Ex6_12_ax f n -> Ex6_12_ax g n ->
  Ex6_12_v f n =  Ex6_12_v g n ->
  forall i, inc i (Nint1c n) -> f i = g i.
Lemma Exercise6_12d n f: natp n -> Ex6_12_ax f n -> (* 10 *)
  Ex6_12_v f n = Yo (n = \0c) \1o
    ((f \1c) *o oprodf
     (fun i => (osucc (omega0 *o (Yo (csucc i = n) \1o (f (csucc (csucc i)))))))
     n).
Lemma Exercise6_12e n: (* 3 *)
  posnatp n -> n *o (osucc omega0) = (omega0 +o n).
Lemma Exercise6_12f f n: (* 41 *)
   natp n -> Ex6_12_ax f n ->
   Ex6_12_v f n = oprodf (fun z => (omega0 +o f(csucc z))) n.
```

The result is now easy.

```
Lemma Exercise6_12g n: (* 23 *)
  natp n ->
  factorial n =
  cardinal (fun_image (permutations  (Nint1c n))
         (fun s => oprodf (fun i => (omega0 +o Vf s (csucc i))) n)).
```

¶ 13. (a) Let $w(\xi)$ be an ordinal functional symbol (§2, Exercise 17), defined for $\xi \ge \alpha_0$ and such that the relation $\alpha_0 \le \xi < \xi'$ implies $w(\xi) < w(\xi')$. Show that, if $\xi \ge \alpha_0$, then $w(\xi + \eta) \ge$

$w(\xi) + \eta$ for every ordinal η (argue by contradiction). Deduce that there exists α such that $w(\xi) \geq \xi$ for all $\xi \geq \alpha$ (take α to be the least indecomposable ordinal $\geq \alpha_0$; cf Exercise 11 (a)).

(b) Let $f(\xi,\eta)$ be the ordinal functional symbol defined in § 2, Exercise 17(b). Suppose that the relations $\alpha_0 \leq \xi \leq \xi'$ and $\alpha_0 \leq \eta \leq \eta'$ imply $g(\xi,\eta) \leq g(\xi',\eta')$ so that the relations $\alpha_0 \leq \xi \leq \xi'$ and $1 \leq \eta \leq \eta'$ imply $f(\xi,\eta) \leq f(\xi',\eta')$ (§ 2, Exercise 17(d)). Show that for each ordinal β there exist at most a finite number of ordinals η for which the equation $f(\xi,\eta) = \beta$ has at least one solution (Note that if ξ_1 is the least solution of $f(\xi,\eta_1) = \beta$ and if ξ_2 is the least solution of $f(\xi,\eta_2) = \beta$ then the relation $\eta_1 < \eta_2$ implies $\xi_1 > \xi_2$.)

(c) A *critical ordinal* with respect to f is any infinite ordinal $\gamma > \alpha_0$ such that $f(\xi,\gamma) = \gamma$ for all ξ such that $\alpha_0 \leq \xi < \gamma$. Show that a critical ordinal (with respect to f) has no predecessor. If there exists a set A of ordinals such that $f(\xi,\gamma) = \gamma$ for all $\xi \in$ A, and if γ is the least upper bound of A, show that γ is a critical ordinal.

(d) Let $h(\xi) = f(\xi,\xi)$ (defined for $\xi \geq \alpha_0$); define inductively $\alpha_1 = \alpha_0 + 2$, $\alpha_{n+1} = h(\alpha_n)$ for $n \geq 1$. Show that that the least upper bound of the sequence (α_n) is a critical ordinal with respect to f.

(e) Show that the least upper bound of every set of critical ordinals with respect to f is again a critical ordinal, and that every critical ordinal is indecomposable (note that $f(\xi,\eta+1) \geq w(\xi)+\eta \geq \xi+\eta$ for all $\xi \geq \alpha_0$).

Solution

(a) is `ord_sum_increasing5`, (c) is `critical_limit`, `sup_critical`, (d) is `sup_critical2`, and (e) is `sup_critical3`. There is a misprint in (e): $\omega(\xi)$ instead of $w(\xi)$.

We consider here (b). The assumptions are that f is increasing in both its arguments, and strictly increasing in its second argument. Fix b and consider

$$\exists x, \quad f(x,y) = b. \tag{*}$$

We know that $f(0,y)$ is strictly increasing, so that there exists c such that $f(0,y) \geq y$ if $y \geq c$. If $x \neq 0$, then $f(x,y) \geq y$. It follows that if y is a solution of (*) we have $y \leq \sup(b,c)$. It follows that there exists a set E_b such that $y \in E_b$ if and only if (*) holds. To each $y \in E_b$ we associate x_y, the least x such that $f(x,y) = b$. We denote by F_b the set of all these x_y. Both sets E_b and F_b contain only ordinals, thus are well-ordered. The mapping $y \to x_y$ is strictly decreasing (if $y < z$ and $x_y \leq x_z$ then $f(x_y,y) \leq f(x_z,y) < f(x_z,z)$ absurd). This means that E_b, ordered by the opposite ordering of $\leq_{\text{ord}}$ is order-isomorphic to F_b and well-ordered. Since E_b is well-ordered, it is finite.

```
Lemma ord_induction_p20 u w0 g b (* 52 *)
  (f:= ord_induction_defined w0 g):
  OIax2 u w0 g ->
 ordinalp b -> exists2 E, finite_set E &
 forall y, (inc y E) <-> (exists x, [/\ u <=o x, ordinalp y & f x y = b]).
```

¶ 14. (a) Show that if $\alpha \geq 2$ and if β has no predecessor, then α^β is an indecomposable ordinal (cf. § 2, Exercise 16 (a)); if α is finite and if $\beta = \omega\gamma$, then $\alpha^\beta = \omega^\gamma$; if α is infinite and if π is the greatest indecomposable ordinal $\leq \alpha$, then $\alpha^\beta = \pi^\beta$ (use Exercise 11).

(b) An ordinal δ is critical with respect to the functional symbol $f(\xi,\eta) = \xi\eta$ if and only if, for each α such that $1 < \alpha \leq \delta$, the equation $\delta = \alpha^\xi$ has a solution; the unique solution ξ of this equation is then indecomposable (Use Exercise 13 (e), together with Exercise 18 (d) of § 2). Conversely, for each $\alpha > 1$ and each indecomposable ordinal π, α^π is a critical ordinal with respect to $\xi\eta$ (use Exercise 13 (c)). Deduce that δ is a critical ordinal with respect to $\xi\eta$ if and only if δ is of the form ω^{ω^μ} (cf. Exercise 11 (b)).

(c) For an ordinal ϵ to be critical with respect to the functional symbol $f(\xi,\eta) = \xi^\eta$, i.e. such that $\gamma^\epsilon = \epsilon$ for each γ satisfying $2 \leq \gamma \leq \epsilon$, it is sufficient that $2^\epsilon = \epsilon$. Show that the least critical ordinal ϵ_0 with respect to ξ^η is countable (cf. Exercise 13 (d)).

Note. There is a misprint in the English Edition. One should read: ϵ is critical when $\gamma^\epsilon = \epsilon$ for each γ satisfying $2 \leq \gamma < \epsilon$.

(a) Let α and β be two ordinals. If $\beta = 0$ then $\alpha^\beta = 1$ is indecomposable. Assume that β has no predecessor, $\beta = \omega\cdot\beta'$ for some β'. We compute α^β via formula (17e) of Section 11.12). It says $\alpha^\beta = \omega^\gamma$ for some γ, thus is indecomposable. If α is finite, then $\gamma = \beta'$ (second part of (17e)). Assume α infinite of degree n, then $\gamma = n\cdot\beta$ (first part of (17e)). Let $\pi = \omega^n$ so that $\alpha^\beta = \pi^\beta$. Since $\omega^n \leq \alpha < \omega^{n+1}$, it follows that π is the greatest power of ω (i.e., the greatest indecomposable ordinal) which is $\leq \alpha$.

(b) Let's show that if α^π has the form ω^μ, for some indecomposable μ, whenever α and π are > 1, and π is indecomposable. Note that we have to exclude the case $\pi = 1$. Now $\pi = \omega^n$ for some non-zero n, and we can apply formula (17e). Assume α is infinite, of degree β, then $\alpha^\pi = \omega^{\beta\cdot\pi}$. We know that $\beta\cdot\pi$ is indecomposable. If α is finite, then $\alpha^\pi = \omega^{\pi'}$ where $\pi = \omega\cdot\pi'$, and π' is again indecomposable.

Consider the following properties of y: (P1) says that y is critical for the product, (P2) says that $y = x^z$ has a solution in z whenever $1 < x \leq z$, (P3) says that $y = x^z$ has a solution in z which is indecomposable whenever $1 < x \leq z$, (P4) says that $y = \omega^{\omega^\mu}$. In the main text we have shown that (P1), (P3), and (P4) are equivalent. Obviously (P3) implies (P2).

Let's show that (P2) implies the other relations. Consider a such that $1 \leq a < y$, and n such that $y = a^n$. Assume first n infinite. Then $1 + n = n$ (proof: consider the CNF of n, and the sum of two CNFs). In this case $a\cdot y = a^{1+n} = a^n = y$. Assume n finite. Assume first $n \geq 3$. Let $m = n-1$, so that $a^m \leq y$ and there is p with $y = (a^m)^p$. The cases $p = 0$ and $p = 1$ are excluded, so that $p \geq 2$ and $mp \geq m+2$. Thus $y \geq ay$, this concludes the proof. Cases $n < 2$ are excluded so that $n = 2$ and $y = a^2$. If $b = a + a$, we have $b \leq y$, so that $y = b^m$ for some m. Simple considerations show $m = 2$, so that $y = a^2 = (a+a)^2$.

This is impossible. Assume $a^2 = (a\cdot 2)^2$, and a non-zero. We get $a\cdot a = a\cdot(2\cdot a\cdot 2)$, and we can simplify by a. So $a = 2\cdot(a\cdot 2)$. It follows $a\cdot 2 \leq a$, thus $a = 0$.

(c) Lemma `ord_epsilon_p10` says that if $2^\epsilon = \epsilon$, then ϵ is critical. The least critical ordinal is ω, and if ϵ is not ω, it is critical if and only if it is an ϵ-ordinal. There are many countable critical ordinals, since ϵ_α is countable whenever α is countable.

```
Lemma CNF_deg0_pr X n:  (* 9 *)
  CNF_axn X n ->  P (Vg X n) = \0o -> n = \0c.
Lemma rev_succ_pr x: ordinalp x ->  (* 1 *)
  x <o \omega \/ x = \1o +o x.
Lemma ord_square_inj a: ordinalp a ->  (* 14 *)
   a ^o \2o =  (a *o \2o) ^o \2o -> a = \0o.
Lemma critical_product_P2: (* 102 *)
  let CP := critical_ordinal \1o oprod2 in
  let p1 := fun y => [/\ infinite_o y, is_ordinal y &
```

```
      (forall z, \1o <o z -> z <=o y ->
         exists2 t, ordinalp t &  y = z ^o t)] in
  forall y, CP y <-> p1 y.

Lemma critical_product_pr3 a b: (* 48 *)
 \1o <o a -> \1o <o b ->
 indecomposable b ->
 critical_ordinal \1o oprod2 (a ^o b).
```

¶ 15. Let γ be an ordinal > 1, and for each ordinal α let $L(\alpha)$ denote the set of exponents λ_i in the expression for α given in Exercise 12 (a).

(a) Show that $\lambda_i \le \alpha$ for each $\lambda_i \in L(\alpha)$, and that $\lambda_i = \alpha$ for one of these ordinal only if $\alpha = 0$ or if α is a critical ordinal with respect to ξ^η (Exercise 14 (c)).

(b) Define $L_n(\alpha)$ by induction on n as follows: $L_1(\alpha) = L(\alpha)$ and $L_n(\alpha)$ is the union of the sets $L(\beta)$ as β runs through $L_{n-1}(\alpha)$. Show that there exists an integer n_0 such that $L_{n+1}(\alpha) = L_n(\alpha)$ whenever $n \ge n_0$, and that the elements of $L_n(\alpha)$ are then either 0 or critical ordinals with respect to ξ^η (Argue by contradiction: for each n, consider the set $M_n(\alpha)$ of elements $\beta \in L_n(\alpha)$ such that $\beta \notin L(\beta)$, and assume that $M_n(\alpha)$ is not empty for any n; use (a) to obtain a contradiction.)

Solution.

If x is non-zero, of degree e, then $b^e \le x$, thus $e \le x$. We deduce: if E is the set of exponents of x, then E is finite; each element of E is an ordinal $\le x$.

```
Section Exercise6_15.
Variable (b: Set).
Hypothesis bg2: \2c <=o b.

Definition CNFB_expos x := cnf_exponents (the_CNFB b x).

Lemma CNFB_monomial_inj  x (z :=  (the_CNFB b x)):  (* 8 *)
  ordinalp x ->
  {when inc ^~ (domain z) & , injective (oexp z)}.
Lemma CNFB_range_fgraph x : ordinalp x -> fgraph (range (the_CNFB b x)). (* 5 *)
Lemma CNFB_card_range x (z :=  (the_CNFB b x)): (* 9 *)
  ordinalp x ->  domain z = cardinal (range z).
Lemma CNFB_expos_zero: CNFB_expos \0o = emptyset. (* 2 *)
Lemma CNFB_e_p2 x (E := CNFB_expos x):  (* 9 *)
  ordinalp x -> (finite_set E /\ ordinal_set E).
Lemma CNFB_e_p3 e c n:
  natp n -> CNFb_ax b e c (csucc n) ->
  e n <=o  (CNFbv b e c (csucc n)).
Lemma CNFB_e_p4 x:  (* 9 *)
  ordinalp x -> (forall y, inc y (CNFB_expos x) -> y <=o x).
```

Since the degree of α is $\le \alpha$, so is any element of $L(\alpha)$. Assume that the degree is α. Then the CNF has a single term, and the coefficient is 1, thus $\alpha = b^\alpha$. We have already noticed that this relation is equivalent to α being critical for exponentiation; this means that either α is a an ϵ-ordinal, or $\alpha = \omega$ and b is finite. Let's call this critical.

We show here the following two properties: if x is critical, then $L(x) = \{x\}$, otherwise, $y \in L(x)$ implies $y < x$.

```
Definition b_critical x := b ^o x = x.

Lemma CNFB_e_p5 e c n (x := CNFbv b e c (csucc n)): (* 35 *)
  natp n -> CNFb_ax b e c (csucc n) ->
  (e n = x ->  b_critical x)
   /\ (b_critical x -> (n = \0c /\ (e n = x))).

Lemma CNFB_e_p6 x (y:=CNFB_expos x): ordinalp x ->  (* 22 *)
   ((b_critical x  -> y = singleton x) /\
   (~ (b_critical x) -> forall a, inc a y -> a <o x)).
```

Let's now define by induction $L_n(x)$ to be $L(x)$ if $n = 0$, and the union of the $L_n(y)$ for $y \in L_{n-1}(x)$ otherwise. This is a finite set of ordinals. Let M_n be the set of all non-critical elements of $L_n(x)$. If M_n is empty then $L_k = L_n$ whenever $k \geq n$ and this set contains only critical elements. We pretend that at least one M_n is empty, for otherwise it would contain a greatest element y_n, and the sequence y_n is strictly decreasing.

```
Definition CNFB_expos_rec x:=
 induction_defined (fun z => union (fun_image z CNFB_expos))
    (CNFB_expos x).
Definition CNFB_expos_rec_nc x n :=
  Zo (Vf (CNFB_expos_rec x) n) (fun z => ~ (b_critical z)).

Lemma CNFB_e_p7 x n (y := Vf (CNFB_expos_rec x) n): (* 17 *)
   ordinalp x -> natp n -> (finite_set y /\ ordinal_set y).

Lemma the_cnf_e_p8 x n (f :=  (the_cnf_expos_rec x)): (* 29 *)
  ordinalp x -> natp n ->
  the_cnf_expos_rec_nc x n = emptyset ->
  ( (forall a, inc a (Vf f n) -> b_critical a)
  /\ (forall k, natp k -> n <=c k -> Vf f k = Vf f n)).
Lemma the_cnf_e_p9 x: ordinalp x -> (* 57 *)
  exists2 n, natp n & the_cnf_expos_rec_nc x n = emptyset.
End Exercise6_15.
```

---

16. Every totally ordered set has a well-ordered cofinal subset (§ 2, Exercise 2). The least of the ordinals Ord(M) of the well-ordered cofinal subsets M of E is called the *final character* of E.

(a) An ordinal ξ is said to be *regular* if it is equal to its final character, and *singular* otherwise. Show that every infinite regular ordinal is an initial ordinal ω_α (Exercise 10). Conversely, every initial ordinal ω_α, whose index α is either 0 or has a predecessor, is a regular ordinal. An initial ordinal ω_α whose index α has no predecessor is singular if $0 < \alpha < \omega_\alpha$; in particular, ω_ω is the least infinite singular initial ordinal.

(b) An initial ordinal ω_α is said to be *inaccessible* if it is regular and its index α has no predecessor. Show that if $\alpha = 0$, then $\omega_\alpha = \alpha$; in other words, α is a critical ordinal with respect to the normal functional symbol ω_η (Exercise 10 (d) and 13 (c)). Let κ be the least critical ordinal with respect to this functional symbol. Show that ω_κ is singular, with final character ω (cf. Exercise 13(d)). In other words, there exists no inaccessible ordinal ω_α such that $0 < \alpha \le \kappa$ (At present, it is not known whether or not there exist inaccessible ordinals other than ω).

(c) Show that there exists only one regular ordinal which is cofinal in a given totally ordered set E; this ordinal is equal to the final character of E, and if E is not empty and has no greatest element, it is an initial ordinal. If $\omega_{\bar\alpha}$ is the final character of ω_α, then $\bar\alpha \le \alpha$; and ω_α is regular if and only if $\alpha = \bar\alpha$.

(d) Let ω_α be a regular ordinal and let I be a well-ordered set such that $\mathrm{Ord}(I) < \omega_\alpha$. Show that, for each family $(\xi_\iota)_{\iota\in I}$ of ordinals such that $\xi_\iota < \omega_\alpha$ for all $\iota \in I$, we have $\sum_{\iota\in I} \xi_\iota < \omega_\alpha$.

Note. In (b) $\alpha = 0$ has to be replaced by $\alpha \ne 0$. Moreover “critical” has to be understood as $f(x) = x$.

(c) This statement can be formalized as: if E is an ordered set, there exists a unique regular ordinal x such that there is a cofinal set F in E, so that the ordinal of F (with the ordering of E) is x. This is the final character of E, and is an initial ordinal if E satisfies the stated properties. We shall only prove partly this statement. Note first that, if E satisfies the stated condition, then any cofinal set F has to be infinite; so that if x exists, it is an initial ordinal. Let x be the final character of E; we know that it exists; there is a cofinal set F, and an order isomorphism $x \to F$. Let $y = \mathrm{cf}(x)$; there is a cofinal function $g : y \to x$. Composing g and f yields a cofinal function, so that $x \le y$. Thus $x = y$. This shows that x is regular. This part is not formally proven.

(d) Not yet done Needs some comments.

Solution. By `exercise2_2b`, for every ordered set $(E, \le)$ there exists a well-ordered subset X that has no upper bound; if E is totally ordered, it follows that X is cofinal. Consider now all ord(X) where X is cofinal in E and well-ordered; this is a set of ordinals, thus has a least element, called the *final character* of E.

Assume now that x is an ordinal and let $(E, \le)$ be its ordering. This is a total ordering. We define $c(x)$, the final character of x, as the final character of E. Note that any subset of X is well-ordered by $\le$, so that the final character of x is the least ord(X), where X is cofinal in x.

```
Lemma Exercise6_16a r: total_order r -> exists2 X, (* 6 *)
  cofinal r X & (worder (induced_order r X)).
Lemma cofinality'_pr1 r: total_order r -> (* 7 *)
  (nonempty (cofinality' r) /\ ordinal_set (cofinality' r)).
Lemma intersection_sub1 A B C:
   A = union2 B C -> (forall x, inc x C -> exists y, inc y B /\ sub y x)
   -> intersection A = intersection B.
Lemma cofinal_trans r x y: (* 4 *)
   order r -> cofinal r x -> cofinal (induced_order r x) y ->
   cofinal r y.

Lemma cofinal_image r r' f x:  (* 10 *)
    order_isomorphism f r r' -> cofinal r x ->
    cofinal r' (Vfs f x).
Lemma worder_image r r' f A: (* 51 *)
  order_isomorphism f r r' -> sub A (substrate r) ->
  let oa := (induced_order r A) in
  let ob := (induced_order r' (Vfs f A)) in
  worder oa  -> (worder ob /\ ordinal oa = ordinal ob).
```

17. A cardinal $\aleph_\alpha$ is said to be *regular* (resp. *singular*) if the initial ordinal ω_α is regular (resp. singular). For $\aleph_\alpha$ to be regular if is necessary and sufficient that for every family $(\mathfrak{a}_\iota)_{\iota\in I}$ of cardinals such that $\mathrm{Card}(I) < \aleph_\alpha$ and $\mathfrak{a}_\iota < \aleph_\alpha$ for all $\iota \in I$, we have

$$\sum_{\iota\in I} \mathfrak{a}_\iota < \aleph_\alpha.$$

$\aleph_\omega$ is the least singular cardinal.

This is `infinite_regular_pr`.

¶ 18. (a) For each ordinal α and each cardinal $\mathfrak{m} \neq 0$ we have $\aleph_{\alpha+1}^{\mathfrak{m}} = \aleph_\alpha^{\mathfrak{m}} \cdot \aleph_{\alpha+1}$ (reduce to the case where $\mathfrak{m} < \aleph_{\alpha+1}$ and consider the mappings of the cardinal $\mathfrak{m}$ into the ordinal $\omega_{\alpha+1}$).

(b) Deduce from (a) that, for each ordinal γ such that $\mathrm{Card}(\gamma) \le \mathfrak{m}$ we have $\aleph_{\alpha+\gamma}^{\mathfrak{m}} = \aleph_\alpha^{\mathfrak{m}} \cdot \aleph_{\alpha+\gamma}^{\mathrm{Card}(\gamma)}$ (by transfinite induction on γ).

(c) Deduce from (b) that, for each ordinal α such that $\mathrm{Card}(\alpha) \le \mathfrak{m}$, we have $\aleph_\alpha^{\mathfrak{m}} = 2^{\mathfrak{m}} \cdot \aleph_\alpha^{\mathrm{Card}(\alpha)}$.

Solution. This exercise corresponds to lemmas `infinite_power2`, `infinite_power5` and `infinite_power6`, and formulas (21b), (21d) and (21e). If $\alpha = 0$, formula (c) reduces to $\aleph_\alpha^{\mathfrak{m}} = 2^{\mathfrak{m}}$, so that we have to add the condition: "either α non-zero or $\mathfrak{m}$ infinite".

¶ 19. (a) Let α and β be two ordinals such that α has no predecessor, and let $\xi \to \sigma_\xi$ be a strictly increasing mapping of the ordinal ω_β into the ordinal α such that $\sup_{\xi<\omega_\beta} \sigma_\xi = \alpha$. Show that

$$\aleph_\alpha^{\aleph_\beta} = \prod_{\xi<\omega_\beta} \aleph_{\sigma_\xi}.$$

(With each mapping f of the ordinal ω_β into the ordinal ω_α associate an injective mapping $\bar f$ of ω_β into the set of all ω_{σ_ξ} $(\xi < \omega_\beta)$ such that $f(\zeta) \le \bar f(\zeta)$ for all $\zeta < \omega_\beta$. Calculate the cardinal of the set of mappings f associated with the same $\bar f$ and observe that

$$\mathfrak{m} = \prod_{\xi<\omega_\beta} \aleph_{\sigma_\xi} \ge 2^{\mathrm{Card}(\omega_\beta)}$$

and $\mathfrak{m} \ge \aleph_\alpha$ (cf. § 3, Exercise 3).)

(b) Let $\bar\alpha$ be the ordinal such that $\omega_{\bar\alpha}$ is the final character of ω_α. Show that $\aleph_\alpha^{\aleph_{\bar\alpha}} > \aleph_\alpha$ and that if there exists $\mathfrak{n}$ such that $\aleph_\alpha = \mathfrak{n}^{\aleph_\gamma}$ then $\gamma < \bar\alpha$ (use (a) and Exercise 3 of § 3).

(c) Show that if $\lambda < \bar\alpha$ then

$$\aleph_\alpha^{\aleph_\lambda} = \sum_{\xi<\alpha} \aleph_\xi^{\aleph_\lambda}$$

(argue as in Exercise 18 (a)).

Solution.

(a) This result corresponds to lemma `infinite_increasing_power1`. The proof of Bourbaki is quite different, so that we give here a second proof.

Let B be an infinite cardinal, X and Y two sets with cardinal < B. Then there is some x in B, which is neither in X nor in Y (because the cardinal of $X \cup Y$ is < B). Assume that x

and y are some ordinals $< \mathrm{B}$, f is a function with source y. Let Y be the image of f. Then $\mathrm{Card}(\mathrm{Y}) \leq \mathrm{Card}(y)$. Now x and y have cardinal $< \mathrm{B}$. Thus we deduce: there exists an element z of B, that is not in the range of f and $z \geq x$ (this last condition is not $z < x$, thus $z \not\in x$). Let g be any function $\mathrm{B} \to \mathrm{B}$. Define by transfinite induction a function f, such that $f(x)$ is the least element z such that $z \geq g(x)$ and z is not in the range of the restriction of f to the interval $]\leftarrow, x[$. This last condition says that $f(x)$ is never $f(y)$ for $y < x$, and implies injectivity of f.

```
Lemma infinite_union2 x y z:   (*  23 *)
    infinite_c z -> cardinal x <c z -> cardinal y <c z ->
    nonempty (z -s (x \cup y)).
Lemma cofinality_pr6 a f (b:= omega_fct a):
   ordinalp a ->
   inc f (functions b b) ->
   exists g, inc g (injections b b) /\
   (forall x, inc x b ->  Vf f x <=o Vf g x).   (* 62 *)
```

Let now σ_ξ be an increasing sequence defined on ω_β, whose supremum is α, a limit ordinal. Bourbaki assumes the sequence strictly increasing; but as mentioned above this condition is not needed. Let f be any function $\omega_\beta \to \omega_\alpha$. Fix $t \in \omega_\beta$. Then $f(t) < \omega_\alpha$. This implies that there is some u such that $f(t) \leq \omega_u$ and $u < \alpha$. Now there is ξ such that $u \leq \sigma_\xi$. Let g be the function $t \mapsto u$. This function satisfies $f(t) \leq \omega_{\sigma_{g(t)}}$. By the previous argument, there exists a injective function h such that $g(t) \leq h(t)$, and $f(t) \leq \omega_{\sigma_{h(t)}}$.

```
Lemma cofinality_pr7 X b f (E :=  omega_fct b): (* 52 *)
   ofg_Mle_leo X -> domain X = omega_fct b -> ordinalp b ->
  limit_ordinal  (\osup (range X)) ->
  inc f (functions E (omega_fct (union (range X)))) ->
  exists2 g, inc g (injections E E) &
    (forall x, inc x (source f) ->  Vf f x <=o omega_fct (Vg X (Vf g x))).
```

Let E_h denote the product of all $\omega_{\sigma^+_{h(t)}}$, where x^+ is the successor of x. Recall that $t \leq x$ is equivalent to $t \in x^+$, so that $f \in \mathrm{E}_h$. Let E be the set of functions $\omega_\beta \to \omega_\alpha$. We have shown that E is a subset of the union of all E_h. It follows $\mathrm{Card}(\mathrm{E}) \leq \sum_h \mathrm{Card}(\mathrm{E}_h)$. Let P be the product of all $\aleph_{\sigma_\xi}$. The objective is to show $\mathrm{Card}(\mathrm{E}) = \mathrm{P}$; the relation $\mathrm{Card}(\mathrm{E}) \leq \mathrm{P}$ is obvious; it remains to show $\sum_h \mathrm{Card}(\mathrm{E}_h) \leq \mathrm{P}$. Note that h belongs to a subset of all functions $\omega_\beta \to \omega_\beta$, whose cardinal is $2^{\aleph_\beta} \leq \mathrm{P}$. It suffices to show $\mathrm{Card}(\mathrm{E}_h) \leq \mathrm{P}$.

```
Lemma infinite_increasing_power3 X b: (* 105 *)
  ofg_Mle_leo X -> domain X = omega_fct b -> ordinalp b ->
  limit_ordinal  (\osup (range X)) ->
  cprod (Lg (domain X) (fun z => \aleph (Vg X z))) =
  \aleph (\osup (range X)) ^c \aleph b.
```

(b) The first point can be restated as $\kappa^{\mathrm{cf}(\kappa)} > \kappa$. The second point follows from $\mathrm{cf}(\mathfrak{n}^\kappa) > \kappa$; note that $\mathfrak{n} \geq 2$.

```
Lemma Exercise_6_19b a     (* 20 *)
  (ba := ord_index (cofinality (omega_fct a)))
  (x := omega_fct a) (y := omega_fct ba):
  ordinalp a ->
  (x <c x ^c y /\
   (forall n c, cardinalp n -> ordinalp c -> x = n ^c (omega_fct c) ->
       c <o ba)).
```

(c) Let $y = \aleph_\lambda$. Condition $\lambda < \tilde{\alpha}$ is equivalent to $y < \aleph_{\tilde{\alpha}} = \mathrm{cf}(\omega_\alpha)$. Since α is a limit ordinal, the condition becomes $y < \mathrm{cf}(\alpha)$ and the result is (20g), `infinite_power7f1`.

¶ 20. (a) For a cardinal $\mathfrak{a}$ to be regular (Exercise 17) it is necessary that for every cardinal $\mathfrak{b} \neq 0$ we should have

$$\mathfrak{a}^{\mathfrak{b}} = \mathfrak{a}.\sum_{\mathfrak{m}<\mathfrak{a}} \mathfrak{m}^{\mathfrak{b}}.$$

(Use Exercise 19 and consider separately the cases (i) $\mathfrak{b}$ is finite, (ii) $\aleph_0 \leq \mathfrak{b} < \mathfrak{a}$, (iii) $\mathfrak{b} \geq \mathfrak{a}$; also use Exercise 3 of § 3.) The generalized continuum hypothesis implies that the above condition is also sufficient.

(b) Show that if a cardinal $\mathfrak{a}$ is such that $\mathfrak{a}^{\mathfrak{m}} = \mathfrak{a}$ for every cardinal $\mathfrak{m}$ such that $0 < \mathfrak{m} < \mathfrak{a}$, then $\mathfrak{a}$ is regular (use Exercise 3 of § 3).

(c) Show that the proposition "for every regular cardinal $\mathfrak{a}$ and every cardinal $\mathfrak{m}$ such that $0 < \mathfrak{m} < \mathfrak{a}$, we have $\mathfrak{a} = \mathfrak{a}^{\mathfrak{m}}$" is equivalent to the generalized continuum hypothesis (use (a)).

(a) is `infinite_power7h` and `infinite_power7h_rev`/

(b) Let $\mathrm{P}(\mathfrak{a})$ be the assumption that $\mathfrak{a}^{\mathfrak{m}} = \mathfrak{a}$ for every cardinal $\mathfrak{m}$ such that $0 < \mathfrak{m} < \mathfrak{a}$. It holds if $\mathfrak{a} \leq 2$; let's exclude this case. We get $\mathfrak{a}^2 = \mathfrak{a}$, so that $\mathfrak{a}$ is infinite. If $\mathfrak{a}$ is singular, the assumption says we may apply `cpow_less_ec_pr3`, and get $\mathfrak{a}^{\mathfrak{a}} = \mathfrak{a}$, absurd. Thus $\mathfrak{a}$ is regular.

(c) Conversely, GCH and $\mathfrak{a}$ regular implies P (second clause of 22i). On the other hand, if P holds for any regular $\mathfrak{a}$, then GCH holds, for if α is any ordinal, $\mathfrak{m} = \aleph_\alpha$ and $\mathfrak{a} = \aleph_{\alpha+1}$ then $0 < \mathfrak{m} < \mathfrak{a}$ and $\mathfrak{a}$ is regular. From $\mathfrak{a}^{\mathfrak{m}} = \mathfrak{a}$, we deduce $2^{\mathfrak{m}} \leq \mathfrak{a}$ thus $2^{\mathfrak{m}} = \mathfrak{a}$.

```
Lemma Exercise_6_20b a: (* 18 *)
   \2c <c a ->
   (forall m, \0c <c m -> m <c a -> a ^c m = a) -> regular_cardinal a.
Lemma Exercise_6_20c1 a: (* 3 *)
   GenContHypothesis -> regular_cardinal a ->
   (forall m, \0c <c m -> m <c a -> a ^c m = a).
Lemma Exercise_6_20c2: (* 12 *)
  (forall a, regular_cardinal a ->
    (forall m, \0c <c m -> m <c a -> a ^c m = a)) ->
 GenContHypothesis.
```

¶ 21. An infinite cardinal $\mathfrak{a}$ is said to be *dominant*, if for each pair of cardinals $\mathfrak{m} < \mathfrak{a}$, $\mathfrak{n} < \mathfrak{a}$ we have $\mathfrak{m}^{\mathfrak{n}} < \mathfrak{a}$.

(a) For $\mathfrak{a}$ to be dominant it is sufficient that $2^{\mathfrak{m}} < \mathfrak{a}$ for every cardinal $\mathfrak{m} < \mathfrak{a}$.

(b) Define inductively a sequence $(\mathfrak{a}_n)$ of cardinals as follows: $\mathfrak{a}_0 = \aleph_0$, $\mathfrak{a}_{n+1} = 2^{\mathfrak{a}_n}$. Show that the sum $\mathfrak{b}$ of the sequence $\mathfrak{a}_n$ is a dominant cardinal. $\aleph_0$ and $\mathfrak{b}$ are the two smallest dominant cardinals.

(c) Show that $\mathfrak{b}^{\aleph_0} = \aleph_0^{\mathfrak{b}} = 2^{\mathfrak{b}}$ (Note that $2^{\mathfrak{b}} \leq \mathfrak{b}^{\aleph_0}$). Deduce that $\mathfrak{b}^{\aleph_0} = (2^{\mathfrak{b}})^{\mathfrak{b}}$, although $\mathfrak{b} < 2^{\mathfrak{b}}$ and $\aleph_0 < \mathfrak{b}$.

(a) is `card_dominant_pr1`.

(b) Lemma `next_dominant_pr` says that the supremum of the sequence is dominant, whatever the value of $\mathfrak{a}_0$, and is the least dominant greater than $\mathfrak{a}_0$. Lemma `card_dominant_pr2` says that $\aleph_0$ is dominant so that it is the least dominant. It follows that $\mathfrak{b}$ is the second least dominant. Note that $\sum x_i = \sup x_i$ if the supremum is infinite and the index set not too big;

(c) is `card_dominant_pr4`.

¶ 22. A cardinal $\aleph_\alpha$ is said to be *inaccessible* if the ordinal ω_α is inaccessible (Exercise 16 (b)). We have then $\omega_\alpha = \alpha$ if $\omega_\alpha \neq \omega_0$. A cardinal $\mathfrak{a}$ is said to be *strongly inaccessible* if it is inaccessible and dominant.

(a) The generalized continuum hypothesis implies that every inaccessible cardinal is strongly inaccessible.

(b) For a cardinal $\mathfrak{a} \geq 3$ to be strongly inaccessible it is necessary and sufficient that, for each family $(\mathfrak{a}_\iota)_{\iota\in I}$ of cardinals such that

$$\mathrm{Card}(I) < \mathfrak{a} \text{ and } \mathfrak{a}_\iota < \mathfrak{a}$$

for all $\iota \in I$, we should have $\prod_{\iota\in I} \mathfrak{a}_\iota < \mathfrak{a}$.

(c) For an infinite cardinal $\mathfrak{a}$ to be strongly inaccessible it is necessary and sufficient that it should be dominant (Exercise 21) and that it should satisfy one of the following two conditions: (i) $\mathfrak{a}^\mathfrak{b} = \mathfrak{a}$ for every cardinal $\mathfrak{b}$ such that $0 < \mathfrak{b} < \mathfrak{a}$; (ii) $\mathfrak{a}^\mathfrak{b} = \mathfrak{a}.2^\mathfrak{b}$ for every cardinal $\mathfrak{b} > 0$ (Use Exercises 20 and 21).

Discussion

Let's write WI and SI wherever Bourbaki uses "inaccessible" and "strongly inaccessible". A cardinal $x = \aleph_\alpha$ is WI if α is regular and not a successor. We do not consider $\aleph_0$ as inaccessible. Thus, x is WI if either $x = \aleph_0$ or is weakly inaccessible. In the second case we know that $\omega_\alpha = \alpha$ (lemma `inaccessible_pr1`). Similarly x is SI if either $x = \aleph_0$ or is inaccessible.

Solution.

(a) is `inaccessible_weak_strong` (this lemma excludes the case of $\aleph_0$ but $\aleph_0$ is WI and SI).

(b) is `inaccessible_pr5` and `inaccessible_pr6`.

(c) is `inaccessible_dominant1` and `inaccessible_dominant2`.

¶ 23. Let α be an ordinal > 0. A mapping f of the ordinal α into itself is said to be *divergent* if for each ordinal $\lambda_0 < \alpha$ there exists an ordinal $\mu_0 < \alpha$ such that the relation $\mu_0 \leq \xi < \alpha$ implies $\lambda_0 \leq f(\xi) < \alpha$. (this condition may be written as $\lim_{\xi\to\alpha,\xi<\alpha} f(\xi) = \alpha$)

(a) Let ϕ be a strictly increasing mapping of an ordinal β into α such that

$$\phi(\sup_{\zeta<\gamma} \zeta) = \sup_{\zeta<\gamma} \phi(\zeta) \text{ for all } \gamma < \beta,$$

and such that

$$\sup_{\zeta<\beta} \phi(\zeta) = \alpha.$$

(if we extend ϕ to $O'(\beta)$ by defining $\phi(\beta) = \alpha$, the conditions above signify that ϕ is continuous.) Then there exists a divergent mapping f of α into itself, such that $f(\xi) < \xi$ for all ξ satisfying $0 < \xi < \alpha$, if and only if there exists a divergent mapping of β into itself of the same type.

(b) Deduce from (a) that there exists a divergent mapping of ω_α into itself, such that $f(\xi) < \xi$ for all ξ satisfying $0 < \xi < \alpha$ if and only if the final character of ω_α is ω_0. (If ω_α is a regular ordinal $> \omega_0$, define inductively a strictly increasing sequence (η_n) as follows: $\eta_1 = 1$, and η_{n+1} is the least ordinal ζ such that $f(\xi) > \eta_n$ for all $\xi \geq \zeta$.)

(c) Let $\omega_{\bar\alpha}$ be the final character of ω_α (Exercise 16). Show that, if $\bar\alpha > 0$ and if f is a mapping of ω_α into itself such that $f(\xi) < \zeta$ for all ξ such that $0 < \xi < \omega_\alpha$, then there exists an ordinal λ_0 such that the set of solutions to the equation $f(\xi) = \lambda_0$ has a cardinal $\geq \aleph_{\bar\alpha}$.

Note in (c), ζ should be replaced by ξ.

¶ 24. Let $\mathfrak{F}$ be a set of subsets of a set E such that for every $A \in \mathfrak{F}$ we have $\mathrm{Card}(A) = \mathrm{Card}(\mathfrak{F}) = \mathfrak{a} \geq \aleph_0$. Show that E has a subset P such that $\mathrm{Card}(P) = \mathfrak{a}$ and such that no set of $\mathfrak{F}$ is contained in P (If $\mathfrak{a} = \aleph_\alpha$, define by transfinite induction two injective mappings $\xi \to f(\xi)$, $\xi \to g(\xi)$ of ω_α into E such that the sets $P = f(\omega_\alpha)$ and $Q = g(\omega_\alpha)$ do not intersect and such that each of them meets every subset $A \in \mathfrak{F}$.)

(b) Suppose, moreover, that for each subset $\mathfrak{G}$ of $\mathfrak{F}$ such that $\mathrm{Card}(\mathfrak{G}) < \mathfrak{a}$, the complement in E of the union of the sets $A \in \mathfrak{G}$ has cardinal $\geq \mathfrak{a}$. Show that E then has a subset P such that $\mathrm{Card}(P) = \mathfrak{a}$ and such that, for each $A \in \mathfrak{G}$, $\mathrm{card}(P \cap A) < \mathfrak{a}$ (similar method).

Note: in (b) the last $\mathfrak{G}$ should be replaced by $\mathfrak{F}$. This is a misprint of the French Edition.

Comments. In (a), instead of defining two functions by induction, we define a single one (the mapping $\xi \mapsto (f(\xi), g(\xi))$). Recall, that, if $p(f)$ is some quantity, X a well-ordered set, there exists a unique surjective function f defined on f such that $f(x) = p(f_x)$, where f_x is the restriction of f to the set of all $y \in X$ such that $y < x$. If α is an ordinal, it is well-ordered by $\leq_{\mathrm{ord}}$, and the source of f_x is x. Thus, it is easy to make p depend on x. Any infinite cardinal $\mathfrak{a}$ has the form $\aleph_\alpha$, and ω_α is an ordinal equipotent to $\mathfrak{a}$. We shall define our function by induction on this ordinal. In our framework, any cardinal is an ordinal, so that we consider $\mathfrak{a}$ as an ordinal α. The important point is that, whenever $\beta < \alpha$, the cardinal of β is $< \mathfrak{a}$. In particular, if $\beta < \alpha$, the source and target of the restriction of t to β has cardinal $< \mathfrak{a}$. On the other hand, if f is injective, it is a bisection, so that its source and target are of cardinal $\mathfrak{a}$. Since $\mathfrak{F}$ has cardinal $\mathfrak{a}$, we may assume that this is the set of all F_β for $\beta < \alpha$.

Let's now make some assumptions.

```
Section Exercise6_24.
Variables (E F a: Set).
Hypothesis FE: forall x, inc x F -> sub x E.
Hypothesis cF: cardinal F = a.
Hypothesis ceF: forall x, inc x F -> cardinal x = a.
Hypothesis iF: infinite_c a.
```

As noted above, in the case (a), we consider a function f with values in $E \times E$. If $\beta < \alpha$, we define $p(f_\beta)$ as follows. Let T be the image of f_β, and T' be the set of all first and second projections. This set has cardinal smaller than that of F_β. Thus, there exists two distinct elements u and v, in F_β, not in T'. We define p to be (u, v).

We get: there is a function f, such that if $\gamma < \beta < \alpha$, if $f(\beta) = (u, v)$ and $f(\gamma) = (u', v')$, then u and v and in F_β (thus in E) and all four elements are distinct. In particular $\beta \mapsto u$ is injective. The set of all u is a solution (if $F_\gamma \subset P$, we get $v' \in F_\gamma$ and $v' \in P$, so that v' is u for some β. Each of the following alternatives $\beta < \gamma$, $\beta = \gamma$ and $\beta > \gamma$ leads to a contradiction.

Consider now (b). Here f takes its values in E. Fix β. Let T be the image of f_β, as above. Let U be the union of all F_γ for $\gamma < \beta$. The assumption is that $\mathrm{Card}(E - U) > \mathrm{Card}(T)$. This means that we can choose $f(\beta)$ to be in E, not in T (so that f will be injective) and not in F_γ for $\gamma < \beta$. Let P be the image of f, and consider $P \cap F_\gamma$. An element of this set is $f(\beta)$, for some β. The above condition says $\beta \le \gamma$. The cardinal of this set is thus the cardinal of $\gamma + 1$, thus $< \mathfrak{a}$.

```
Lemma Exercise6_24a: (* 83 *)
   exists P, [/\ sub P E,cardinal (P) = a &
   forall x, inc x F -> ~ (sub x P)].
Lemma Exercise6_24b:  (* 88 *)
   (forall G, sub G F -> cardinal G <c a ->
       a <=c cardinal (E -s union G)) ->
   exists P, [/\ sub P E, cardinal (P) = a &
   forall x, inc x F -> (cardinal (P \cap x)) <c a].
End Exercise6_24.
```

¶ 25. (a) Let $\mathfrak{F}$ be a covering of an infinite set E. The *degree of disjointness* of $\mathfrak{F}$ is the least cardinal $\mathfrak{c}$ such that $\mathfrak{c}$ is *strictly greater* than the cardinals $\mathrm{Card}(X \cap Y)$ for each pair of distinct sets $X, Y \in \mathfrak{F}$. If $\mathrm{Card}(E) = \mathfrak{a}$ and $\mathrm{Card}(\mathfrak{F}) = \mathfrak{b}$, show that $\mathfrak{b} \le \mathfrak{a}^\mathfrak{c}$ (note that a subset of E of cardinal $\mathfrak{c}$ is contained in at most one set of $\mathfrak{F}$).

(b) Let ω_α be an initial ordinal and let F be set such that $2 \le \mathfrak{p} = \mathrm{Card}(F) < \aleph_\alpha$. Let E be the set of mappings of segments of ω_α, other then ω_α itself, into F. Then we have $\mathrm{Card}(E) \le \mathfrak{p}^{\aleph_\alpha}$. For each mapping f of ω_α into F, let K_f be the subset of E consisting of the restrictions of f to the segments of ω_α (other than ω_α itself). Show that the set $\mathfrak{F}$ of subsets K_f is a covering of E such that $\mathrm{Card}(\mathfrak{F}) = \mathfrak{p}^{\aleph_\alpha}$ and that its degree of disjointness is equal to $\aleph_\alpha$.

(c) Let E be an infinite set of cardinal $\mathfrak{a}$ and let $\mathfrak{c}$, $\mathfrak{p}$ be two cardinals > 1 such that $\mathfrak{p} < \mathfrak{c}$, $\mathfrak{p}^\mathfrak{m} < \mathfrak{a}$ for all $\mathfrak{m} < \mathfrak{c}$ and $\mathfrak{a} = \sum_{\mathfrak{m}<\mathfrak{c}} \mathfrak{p}^\mathfrak{m}$. Deduce from (b) that there exists a covering $\mathfrak{F}$ of E consisting of sets of cardinal $\mathfrak{c}$, with degree of disjunction equal to $\mathfrak{c}$ and such that $\mathrm{Card}(\mathfrak{F}) = \mathfrak{p}^\mathfrak{c}$. In particular, if E is countably infinite, there exists a covering $\mathfrak{F}$ of E by infinite sets such that $\mathrm{card}(\mathfrak{F}) = 2^{\aleph_0}$, and such that the intersection of any two sets of $\mathfrak{F}$ is *finite*.

Note. Assume $\mathfrak{c} = 0$. Then there is no pair of disjoint sets in $\mathfrak{F}$, and $\mathfrak{F}$ has at most one element. Assume $\mathfrak{c} = 1$. Then the elements of $\mathfrak{F}$ are mutually disjoint. For each nonempty $F \in \mathfrak{F}$, consider $x_F \in F$. This is an injective mapping, so that $\mathfrak{b} \le \mathfrak{a} + 1$. More generally, let $\mathfrak{F}'$ be the subset of $\mathfrak{F}$ formed of elements with cardinal $\ge \mathfrak{c}$. Then $F \mapsto x_F$ is injective on $\mathfrak{F}'$. Let $\mathfrak{F}_d$ the subset of $\mathfrak{F}$ formed of elements with cardinal d. By exercise 7, its cardinal is $\le \mathfrak{a}^d$, so that, if $d \le \mathfrak{c}$, it is $\le \mathfrak{a}^\mathfrak{c}$. Thus $\mathfrak{b} \le \mathfrak{a} + \mathfrak{c}\mathfrak{a}^\mathfrak{c}$. This simplifies to $\mathfrak{a}^\mathfrak{c}$.

If A is a set of cardinals, the supremum of the successors of A is the least cardinal $> x$, whenever $x \in A$.

```
Definition csup_s A := \csup (fun_image A cnext).

Lemma csup_s1 A (x := csup_s A): cardinal_set A -> (* 5 *)
  (forall y, inc y A -> y <c x).
```

```
Lemma csup_s2 A z: cardinal_set A -> cardinalp z -> (* 3 *)
  (forall y, inc y A -> y <c z) -> csup_s A <=c z.
```

We define here the degree of disjointness. We consider the case where this degree is zero or one.

```
Definition disjointness_degree F :=
  csup_s (fun_image (coarse F -s diagonal F)
       (fun z => cardinal ((P z) \cap (Q z)))).

Lemma dd_pr1 F x y: inc x F -> inc y F -> x <> y -> (* 4 *)
  cardinal (x \cap y) <c  (disjointness_degree F).
Lemma dd_pr2 F z: cardinalp z -> (* 4 *)
  (forall x y, inc x F -> inc y F -> x <> y ->
     cardinal (x \cap y) <c z)
  -> (disjointness_degree F) <=c z.
Lemma dd_pr3 F : (disjointness_degree F = \0c) <-> small_set F.  (* 4 *)
Lemma dd_pr4 F : (disjointness_degree F = \1c) <->
  (~ small_set F /\ disjoint_set F).
```

Ideas for a proof. If $\mathfrak{c}$ is finite, then $\mathfrak{p}$ is finite, absurd. The second condition on $\mathfrak{a}$ says $\mathfrak{a} = \mathfrak{p}^{<\mathfrak{c}}$. The first condition then says that $\mathfrak{c}$ is a limit cardinal. If GCH holds, then $\mathfrak{a} = \mathfrak{c}$. Otherwise, $\mathfrak{a}$ might be greater, but it is at most $2^{\mathfrak{c}}$.

Write E as a disjoint union of E_i, each of cardinal $x_i = \mathfrak{p}^i$ (where i is infinite, less than $\mathfrak{c}$, greater than $\mathfrak{p}$). Take a cover F_i, of cardinal $\mathfrak{p}^{x_i}$ whose dd is x_i, and such that each element of F_i as cardinal x_i.

¶ 26. Let E be an infinite set and let $\mathfrak{F}$ be a set of subsets of E such that for $A \in \mathfrak{F}$ we have

$$\mathrm{card}(A) = \mathrm{card}(\mathfrak{F}) = \mathrm{card}(E) = \mathfrak{a} \geq \aleph_0.$$

Show that there exists a partition $(B_\iota)_{\iota\in I}$ of E such that

$$\mathrm{card}(I) = \mathrm{card}(B_\iota) = \mathfrak{a}$$

for all $\iota \in I$ and such that $A \cap B_\iota \neq \emptyset$ for all $A \in \mathfrak{F}$ and all $\iota \in I$. (With the notation of Exercise 24 (a), consider first a surjective mapping f of ω_α into $\mathfrak{F}$ such that for each $A \in \mathfrak{F}$ the set of all $\xi \in \omega_\alpha$ such that $f(\xi) = A$ has cardinal equal to $\mathfrak{a}$. Then, by transfinite induction, define a bijection g of ω_α onto E such that $g(\xi) \in f(\xi)$ for every $\xi \in \omega_\alpha$.)

¶ 27. Let L be an infinite set and let $(E_\lambda)_{\lambda\in L}$ be a family of sets indexed by L. Suppose that for each integer $n > 0$ the set of $\lambda \in L$ such that $\mathrm{card}(E_\lambda) > n$ is equipotent to L. Show that there exists a subset F of the product $E = \prod_{\lambda\in L} E_\lambda$, such that $\mathrm{Card}(F) = 2^{\mathrm{Card}(L)}$, and such that F has the following property: for each finite sequence $(f_k)_{1\leq k\leq n}$ of distinct elements of F there exists $\lambda \in L$ such that the elements $f_k(\lambda) \in E_\lambda$ $(1 \leq k \leq n)$ are all distinct. (Show first that there exists a partition $(L_j)_{j\in\mathbf{N}}$ of L such that $\mathrm{Card}(L_j) = \mathrm{Card}(L)$ for all j, and such that $\mathrm{Card}(E_\lambda) \geq 2^j$ for each $\lambda \in L_j$. Hence reduce to the case where L is the sum of the countable family of sets X^j $(j \geq 1)$, where X is an infinite set, and $E_\lambda = 2^j$ for each $\lambda \in X^j$. With each mapping $g \in 2^X$ of X into 2, associate the element $f \in E$ such that $f(\lambda) = (g(x_1), \dots, g(x_j))$ whenever $\lambda = (x_k)_{1\leq k\leq j} \in X^j$; show that the set F of elements $f \in E$ so defined has the required property.)

¶ 28. Let E be an infinite set and let $(\mathfrak{X}_i)_{1\le i\le m}$ be a finite partition of the set $\mathfrak{F}_n(\mathrm{E})$ of subsets of E having n elements. Show that there exists an index i and an infinite subset F of E such that every subset of F with n elements belongs to $\mathfrak{X}_i$. (Proof by induction on n. For each $a \in \mathrm{E}$ show that there exists an index $j(a)$ and an infinite subset $\mathrm{M}(a)$ of $\mathrm{E}-\{a\}$ such that, for every subset A of $\mathrm{M}(a)$ with $n-1$ elements, $\{a\}\cup \mathrm{A}$ belongs to $\mathfrak{X}_{j(a)}$. Then define a sequence (a_i) of elements of E as follows: a_1 is an arbitrary element of E, a_2 is an arbitrary element of $\mathrm{M}(a_1)$, a_3 is defined in terms of $\mathrm{M}(a_1)$ and a_2 in the same way as a_2 was defined in terms of E and a_1, and so on. Show that the set F of elements of a suitable subsequence of the sequence (a_i) satisfies the required conditions).

29. (a) In an ordered set E, every finite union of Noetherian subsets (with respect to the induced ordering) is Noetherian.

(b) An ordered set E is Noetherian if and only if for each $a \in \mathrm{E}$, the interval $]a, \rightarrow[$ is Noetherian.

(c) Let E be an ordered set such that the ordered set obtained by endowing E with the opposite ordering is Noetherian. Let u be a letter and let $\mathrm{T}\{u\}$ be a term. Show that there exists a set U and a mapping f of E onto U such that for each $x \in \mathrm{E}$ we have $f(x) = \mathrm{T}\{f^{(x)}\}$, where $f^{(x)}$ denotes the mapping of $]\leftarrow, x[$ onto $]\leftarrow, f(x)[$ which coincides with f on this interval. Furthermore U and f are determined uniquely by this condition.

(d) Let E be a Noetherian ordered set such that every finite subset of E has a least upper bound in E. Show that, if E has a least element, then E is a complete lattice (§ 1, Exercise 11); and that if E has no least element, the set E′ obtained by adjoining a least element to E (§ 1, no. 7, Proposition 3) is a complete lattice.

(a) We first define "Noetherian" for an ordering, then for a subset X. The empty set is Noetherian, as well as the substrate of a Noetherian order. A set X is Noetherian if any nonempty subset of X has a maximal element (for the ordering of E). The union of two Noetherian sets is Noetherian (let Y be a non-empty subset of $\mathrm{A}\cup\mathrm{B}$, $\mathrm{Y_A} = \mathrm{Y}\cap\mathrm{A}$. If $\mathrm{Y_A}$ is empty, then $\mathrm{Y}\subset\mathrm{B}$ and the result is clear; otherwise there is a maximal element α Let $\mathrm{Y_B}$ the set of all elements $>\alpha$. If this set is empty, then α is maximal. In the other case, it is a non-empty subset of B, and has a maximal element, which is maximal for Y. By induction a finite union of Noetherian sets is Noetherian.

```
Definition Noetherian r :=
  (forall X, sub X (substrate r) -> nonempty X ->
    exists a,  maximal (induced_order r X) a).

Definition Noetherian_set x r:=
   sub x (substrate r) /\ Noetherian (induced_order r x).

Lemma Noetherian_set_pr r x: order r -> sub x (substrate r) ->
   (Noetherian_set r x <->
   (forall X, sub X x -> nonempty X ->
    exists2 a, inc a X & (forall x, inc x X -> gle r a x -> x = a))). (* 9 *)
Lemma Exercise6_29a r: order r ->  (* 1 *)
  Noetherian r ->  Noetherian_set r (substrate r).
Lemma Exercise6_29b r:  Noetherian_set r emptyset. (* 6 *)
Lemma Exercise6_29c r a b: order r -> (* 31 *)
  Noetherian_set r a -> Noetherian_set r b -> Noetherian_set r (a \cup b).
Lemma Exercise6_29d r X: order r -> (* 12 *)
  finite_set X ->
```

```
  (forall x, inc x X -> Noetherian_set r x) ->
  Noetherian_set r (union X).
```

(b) Let's first notice that any subset of a Noetherian set is Noetherian, so that the condition is necessary. Let X be a non-empty subset of E, $a \in \mathrm{X}$. If a is maximal, there is nothing to do; otherwise $\mathrm{X} \cap]a, \rightarrow[$ has a maximal element, which is maximal in X.

```
Lemma Exercise6_29e r X: order r -> Noetherian r ->
  sub X (substrate r) ->  Noetherian_set r X.
Lemma Exercise6_29f r: order r -> (* 11 *)
 (Noetherian r <->
   (forall i, inc i (substrate r) -> Noetherian_set r (interval_ou r i))).
```

30. Let E be a lattice such that the set obtained by endowing E with the opposite ordering is Noetherian. Show that every element $a \in \mathrm{E}$ can be written as $\sup(e_1, e_2, \dots, e_n)$ where $e_1, \dots, e_n$ are irreducible (§ 4, Exercise 7; show first that there exists an irreducible element e such that $a = \sup(e, b)$ if a is not irreducible). Generalize Exercise 7 (b) of § 4 to E; also generalize Exercises 8(b) and 9(b) of § 4.

¶ 31. Let A be an infinite set and let E be the set of all infinite subsets of A, ordered by inclusion. Show that E is completely ramified (§ 2, Exercise 8) but not antidirected (§ 1, Exercise 23) and that E has an antidirected cofinal subset F. (Consider first the set $\mathfrak{D}(\mathrm{A})$ of countable infinite subsets of A (which is cofinal in E) and let $\mathrm{Z} = \mathrm{R}_0(\mathfrak{D}(\mathrm{A}))$ (§ 1, Exercise 23). Write Z in the form $(z_\lambda)_{\lambda \in \mathrm{L}}$, where L is a well-ordered set, and take F to be a set of countable subsets X_n^λ, where λ runs through a suitable subset of L, $n \in \mathbf{N}$, $\mathrm{X}_m^\lambda \supset \mathrm{X}_n^\lambda$ whenever $m \leq n$, $\mathrm{X}_n^\lambda - \mathrm{X}_{n+1}^\lambda$ is infinite for all $n \geq 0$ and $\bigcap_{n \in \mathbf{N}} \mathrm{X}_n^\lambda = \emptyset$; the X_n^λ are to be defined by transfinite induction in such a way that the images of the sets X_n^λ under the canonical mapping $r : \mathfrak{D}(\mathrm{A}) \rightarrow \mathrm{Z}$ (§ 1, Exercise 23) are mutually disjoint and form a cofinal subset of Z.)

¶ 32. * Let (M_n), (P_n) be two sequences of mutually disjoint finite sets (not all empty), indexed by the set $\mathbf{Z}$ of rational integers. Let $\alpha_n = \mathrm{Card}(\mathrm{M}_n)$, $\beta_n = \mathrm{card}(\mathrm{P}_n)$. Suppose that there exists an integer $k > 0$ such that for each $n \in \mathbf{Z}$ and each integer $l \geq 1$ we have

$$\alpha_n + \alpha_{n+1} \cdots + \alpha_{n+l} \leq \beta_{n-k} + \beta_{n-k+1} + \cdots + \beta_{n+l+k},$$

$$\beta_n + \beta_{n+1} \cdots + \beta_{n+l} \leq \alpha_{n-k} + \alpha_{n-k+1} + \cdots + \alpha_{n+l+k}.$$

Let M be the union of the family (M_n) and let P be the union of the family (P_n). Show that there exists a bijection ϕ of M onto P such that

$$\phi(\mathrm{M}_n) \subset \bigcup_{i=n-k-1}^{n+k+1} \mathrm{P}_i \text{ and } \overset{-1}{\phi}(\mathrm{P}_n) \subset \bigcup_{i=n-k-1}^{n+k+1} \mathrm{M}_i$$

for each $n \in \mathbf{Z}$ (consider a total ordering on each M_n (resp. P_n) and take M (resp. P) to be the ordinal sum (§ 1, Exercise 3) of the family $(\mathrm{M}_n)_{n \in \mathbf{Z}}$ (resp. $(\mathrm{P}_n)_{n \in \mathbf{Z}}$). If n_0 is an index such that $\mathrm{M}_{n_0} \neq \emptyset$ consider the isomorphisms of M onto P which transform the least element of M_{n_0} into one of the elements of $\bigcup_{j=n_0-k}^{n_0+k} \mathrm{P}_j$ and show that one of these isomorphisms satisfies the required conditions. Let δ be the least of the numbers

$$\beta_{n-k} + \beta_{n-k+1} + \cdots + \beta_{n+l+k} - (\alpha_n + \alpha_{n+1} \cdots + \alpha_{n+l}),$$

$$\alpha_{n-k}+\alpha_{n-k+1}+\cdots+\alpha_{n+l+k}-(\beta_n+\beta_{n+1}\cdots+\beta_{n+l})$$

for all $n\in\mathbf{Z}$ and all $l\geq 1$. If $n\in\mathbf{Z}$ and $l\geq 1$ are such that, for example $\beta_{n-k}+\beta_{n-k+1}+\cdots+\beta_{n+l+k}=\delta+\alpha_n+\alpha_{n+1}\cdots+\alpha_{n+l}$, we may take ϕ to be such that the least element of P_{n-k} is the image under ϕ of the least element of M_n.) $_*$

¶ **33.** Soient $\mathfrak{a}$, $\mathfrak{b}$ deux cardinaux tels que $\mathfrak{a}\geq 2$, $\mathfrak{b}\geq 1$, l'un au moins des deux étant infini. Soient E un ensemble, $\mathfrak{F}$ une partie de $\mathfrak{P}(E)$, telle que $\mathrm{Card}(\mathfrak{F})>\mathfrak{a}^{\mathfrak{b}}$ et $\mathrm{Card}(X)\leq\mathfrak{b}$ pour tout $X\in\mathfrak{F}$. On se propose de montrer qu'il existe une partie $\mathfrak{G}\subset\mathfrak{F}$ telle que $\mathrm{Card}(\mathfrak{G})>\mathfrak{a}^{\mathfrak{b}}$ et que deux quelconques des ensembles appartenant à $\mathfrak{G}$ aient *la même intersection.* On pourra procéder de la façon suivante.

a) Soit $\mathfrak{c}$ le plus petit des cardinaux $>\mathfrak{a}^{\mathfrak{b}}$ et soit Ω le plus petit ordinal de cardinal $\mathfrak{c}$. On considère une application injective $\nu\mapsto X(\nu)$ de Ω dans $\mathfrak{F}$ et on pose $M=\bigcup_{\nu\in\Omega}X(\nu)$; on peut supposer que $\mathrm{Card}(M)=\mathfrak{c}$, et il y a donc une bijection $\nu\mapsto x_\nu$ de Ω sur M, ordonnant M.

b) Pour tout $\nu\in\Omega$ soit ρ_ν l'ordinal type d'ordre du sous-ensemble $X(\nu)$ de M (on a $\mathrm{Card}(\rho_\nu)\geq\mathfrak{b}$) et soit $\mu\mapsto y_\mu^{(\nu)}$ l'unique application bijective croissante de ρ_ν sur $X(\nu)$. On note M_μ l'ensemble des $y_\mu^{(\nu)}$ lorsque ν parcourt Ω. Montrer qu'il existe au moins un ordinal μ tel que $\mathrm{Card}(M_\mu)=\mathfrak{c}$. On désigne par α le *plus petit* de ces ordinaux; la réunion des M_γ pour $\gamma<\alpha$ a un cardinal $\leq\mathfrak{a}^{\mathfrak{b}}<\mathfrak{c}$.

c) Montrer qu'il existe une partie $N_0\subset\Omega$ telle que $\mathrm{Card}(N_0)=\mathfrak{c}$ et que l'application $\nu\mapsto y_\alpha^{(\nu)}$ de N_0 dans M soit injective. Montrer, par récurrence sur β, qu'il existe une partie $N_\beta\subset N_0$ de cardinal $\mathfrak{c}$ telle que l'élément $y_\lambda^{(\nu)}=z_\lambda$ soit indépendant de ν pour $\nu\in N_\beta$ et pour tout $\lambda\leq\beta$. Montrer que l'intersection N des N_β pour $\beta<\alpha$ a pour cardinal $\mathfrak{c}$ (considérer son complémentaire). Soit Q l'ensemble des z_λ pour $\lambda<\alpha$.

d) Pour tout $\nu\in N$, on définit par récurrence un ordinal λ_ν par la condition suivante : c'est le plus petit ordinal dans N tel que $y_\alpha^{(\lambda_\nu)}$ soit un majorant strict, dans M, de la réunion des $X(\lambda_\mu)$ pour $\mu<\nu$, $\mu\in N$. Montrer que pour $\mu<\nu$ dans N on a $X(\lambda_\mu)\cap X(\lambda_\nu)=Q$.

Solution.

This exercise exists only in the French version. We assume that $\mathfrak{a}$ and $\mathfrak{b}$ are two cardinals. Let $\mathfrak{F}$ be a subset subset of $\mathfrak{P}(E)$ for some E (that plays no role here). We assume that, for any element X of $\mathfrak{F}$, we have $\mathrm{Card}(X)\leq\mathfrak{b}$. We also assume that $\mathrm{Card}(\mathfrak{F})>\mathfrak{a}^{\mathfrak{b}}$. The conclusion is that there exists a subset $\mathfrak{G}$ of $\mathfrak{F}$ of cardinal $>\mathfrak{a}^{\mathfrak{b}}$ so that two elements of $\mathfrak{G}$ have *the same intersection.* More precisely, there is Q such that, whenever x and y and distinct in $\mathfrak{G}$, then $x\cap y=Q$.

We assume $\mathfrak{a}^{\mathfrak{b}}$ infinite; more precisely, $\mathfrak{a}\geq 2$, $\mathfrak{b}\geq 1$, and one of these cardinals is infinite.

```
Section Exercise6_33.
Variables a b E F: Set.
Hypothesis ha: \2c <=c a.
Hypothesis hb: \1c <=c b.
Hypothesis iab: infinite_c a \/ infinite_c b.
Hypothesis HF0: sub F (\Po E).
Hypothesis HF1: a ^c b <c cardinal F.
Hypothesis HF2: forall x, inc x F -> cardinal x <=c b.
   exists G q, [/\ sub G F , a ^c b <c cardinal G &
   forall a b, inc a G -> inc b G -> a <> b -> intersection2 a b = q].
```

Let's denote by $\mathfrak{c}$, the cardinal successor of $\mathfrak{a}^{\mathfrak{b}}$. This is sometimes denoted by Ω, when it is useful to consider it an ordinal. Since $\mathfrak{a}^{\mathfrak{b}}$ is infinite, $\mathfrak{c}$, is infinite as well. It happens that $\mathfrak{b}\leq\mathfrak{c}$

(in fact $\mathfrak{b} < \mathfrak{a}^{\mathfrak{b}} < \mathfrak{c}$) and $\mathfrak{a}^{\mathfrak{b}\cdot\mathfrak{b}} = \mathfrak{a}^{\mathfrak{b}}$ (if $\mathfrak{b}$ is infinite, then $\mathfrak{b}\cdot\mathfrak{b} = \mathfrak{b}$, and otherwise both term as equal to $\mathfrak{a}$).

```
Definition E6_33_c := cnext (a ^c b).

Lemma Exercise6_33a: infinite_c (a ^c b). (* 2 *)
Lemma Exercise6_33a': a ^c (b *c b)  =  a ^c b. (* 4 *)
Lemma Exercise6_33b : a ^c b  <c E6_33_c. (* 1 *)
Lemma Exercise6_33c: infinite_c E6_33_c. (* 1 *)
Lemma Exercise6_33a'': b <c a ^c b /\ b <=c E6_33_c.  (* 2 *)
```

We have $\mathfrak{c} \le \mathrm{Card}(\mathfrak{F})$, so that there is an injection $X : \Omega \to \mathfrak{F}$. As $\mathfrak{F}$ is infinite, we may further assume that no $X(\nu)$ is empty. Let M be the union of $X(\nu)$. Since each $X(\nu)$ has cardinal $\le \mathfrak{b}$, this set has cardinal $\le \mathfrak{c}$.

Let's show $\mathrm{Card}(M) = \mathfrak{c}$. Let T be the set of all ranges of mappings $\mathfrak{b} \to M$. If X is any non-empty subset of M with $\le \mathfrak{b}$ elements, there is a subset Y of $\mathfrak{b}$ and a bijection $Y \to X$, we can extend it to a surjection function $\mathfrak{b}$ to X, and a function $\mathfrak{b} \to M$ with range X. Thus $X \in T \cup \{\emptyset\}$. We deduce $\mathfrak{c} \le \mathrm{Card}(T) + 1$. Since T is infinite, we can ignore the $+1$. We have $\mathrm{Card}(T) \le \mathrm{Card}(M)^{\mathfrak{b}}$; so that $\mathfrak{c} \le \mathrm{Card}(M)^{\mathfrak{b}}$. Assume the result false. Then $\mathrm{Card}(M) \le \mathfrak{a}^{\mathfrak{b}}$. Note that $(\mathfrak{a}^{\mathfrak{b}})^{\mathfrak{b}} = \mathfrak{a}^{\mathfrak{b}}$, so that we get $\mathfrak{c} \le \mathfrak{a}^{\mathfrak{b}}$, contradiction.

```
Definition E6_33_X :=
  choose (fun f => injection_prop f E6_33_c (F - s1 emptyset)).
Definition E6_33_M := unionb (Lg (source E6_33_X) (Vf E6_33_X)).
Lemma Exercise6_33d (f :=  E6_33_X): (* 5 *)
    injection_prop f E6_33_c (F - s1 emptyset).
Lemma Exercise6_33d' n: inc n E6_33_c -> inc (Vf E6_33_X n) F.  (* 2 *)
Lemma Exercise6_33d'' n: inc n E6_33_c -> nonempty (Vf E6_33_X n).  (* 2 *)
Lemma Exercise6_33e n: inc n E6_33_c -> sub (Vf E6_33_X n) E6_33_M. (* 2 *)
Lemma Exercise6_33f: cardinal E6_33_M = E6_33_c. (* 73 *)
```

The last result says that there is a bijection $x : \Omega \to M$. We can use it to transport the natural well-ordering of Ω so as to obtain a well-ordering r on M. Since X_ν is a subset of M, it is well-ordered, and has an ordinal, say ρ_ν, whose cardinal is $\le \mathfrak{b}$. Note that the text says $\ge \mathfrak{b}$, which is wrong. As X_ν is non-empty, this ordinal is non-zero.

```
Definition E6_33_x:= equipotent_ex E6_33_c E6_33_M.
Definition E6_33_r :=
  Vfs (ext_to_prod E6_33_x E6_33_x) (ordinal_o  E6_33_c).
Definition E6_33_rho n := ordinal (induced_order E6_33_r (Vf  E6_33_X n)).

Lemma Exercise6_33g : bijection_prop E6_33_x E6_33_c  E6_33_M.  (* 2 *)
Lemma Exercise6_33h (x := E6_33_x) (r:= E6_33_r) (c := E6_33_c): (* 13 *)
  [/\ worder_on r  E6_33_M,
  order_isomorphism x  (ordinal_o E6_33_c) r &
  (forall a b, inc a c -> inc b c ->
     (a <=o b <-> gle r (Vf x a) (Vf x b)))].
Lemma Exercise6_33i n: inc n E6_33_c -> (* 11 *)
  [/\ worder (induced_order E6_33_r (Vf E6_33_X n)),
   ordinalp (E6_33_rho n), E6_33_rho n <> \0o & cardinal(E6_33_rho n) <=c b].
```

Denote by $y^{(\nu)}$ the unique order isomorphism $\rho_\nu \to X(\nu)$, and by M_μ the set of all $y^{(\nu)}_\mu$, for $\nu \in \Omega$. Here $y^{(\nu)}_\mu$ denotes the value of $y^{(\nu)}$ at μ, which has to be in the source of $y^{(\nu)}$, namely

ρ_ν. Note that $\mathfrak{b} < \mathfrak{a}^\mathfrak{b}$, thus $\rho_\nu < \mathfrak{a}^\mathfrak{b}$ and $\rho_\mu \subset \Omega$. We have then $M = \bigcup M_\mu$, where μ ranges over all ordinals $< \mathfrak{a}^\mathfrak{b}$. Since the index set has smaller cardinality than the union, at least one M_μ has cardinality $\mathfrak{c}$.

We consider α, the least ordinal such that M_α has cardinal $\mathfrak{c}$. Thus, if $\mu < \alpha$, we have $\mathrm{card}(M_\mu) \le \mathfrak{a}^\mathfrak{b}$; moreover $\mathrm{Card}(\bigcup M_\mu) \le \mathfrak{a}^\mathfrak{b}$.

```
Definition E6_33_y n := the_ordinal_iso (induced_order E6_33_r (Vf E6_33_X n)).
Definition E6_33_Mi m := Zo E6_33_M
    (fun z => exists n, [/\ inc n E6_33_c, inc m (source (E6_33_y n)) &
      z = Vf (E6_33_y n) m]).
Definition E6_33_al :=
   intersection (Zo (a ^c b) (fun m => (cardinal (E6_33_Mi m) =  E6_33_c))).

Lemma Exercise6_33j n (* 6 *)
    (yn := E6_33_y n) (r := induced_order E6_33_r (Vf E6_33_X n)):
  inc n E6_33_c ->
    [/\ order_isomorphism yn (ordinal_o (E6_33_rho n)) r,
     source yn = E6_33_rho n & target yn =  (Vf E6_33_X n)].
Lemma Exercise6_33k n: inc n E6_33_c -> (* 10 *)
  ((forall t, inc t (E6_33_rho n)  -> t <o a ^c b)
    /\  sub (E6_33_rho n)  E6_33_c).
Lemma Exercise6_33l:  E6_33_M = unionf (a ^c b) E6_33_Mi. (* 10 *)
Lemma Exercise6_33m x (c:= cardinal (E6_33_Mi x)):  (* 6 *)
  inc x (a ^c b) -> c <=c (a ^c b) \/ c = E6_33_c.
Lemma Exercise6_33n: (* 4 *)
   exists2 m, m <o (a ^c b) & (cardinal (E6_33_Mi m)) = E6_33_c.
Lemma Exercise6_33o (al:=  E6_33_al):
  [/\ al <o (a ^c b), (cardinal (E6_33_Mi al) =  E6_33_c) &
  (forall m, m <o al -> cardinal (E6_33_Mi m) <=c (a ^c b))]. (* 8 *)
Lemma Exercise6_33p: cardinal(unionf E6_33_al E6_33_Mi) <=c a ^c b. (* 4 *)
```

To each $t \in M_\alpha$ we associate some ν such that α belongs to the source of $y^{(\nu)}$ and $y^{(\nu)}_\alpha = t$. Let N_0 be the set of all these quantities. Then $\nu \mapsto y^{(\nu)}_\alpha$ is a bijection $N_0 \to M_\alpha$ and N_0 has cardinal $\mathfrak{c}$.

For any $\beta < \alpha$, there exists N_β, a subset of N_0 of cardinal $\mathfrak{c}$ such that, for all $\nu \le \beta$, $y^{(\nu)}_\lambda$ is independent of ν, for $\lambda \le \beta$. Denote this condition by $P_{\le\beta}$. We consider also $P_{<\beta}$ and $P_{=\beta}$. Let's write $\bar{N}$ instead of N_0 in order to avoid any confusion. Let $\tilde{N}_\beta$ be the intersection of all N_λ for $\lambda < \beta$ (this is $\bar{N}$ for $\beta = 0$. Assume that this set has cardinal $\mathfrak{c}$. We choose N_β to be a subset of $\tilde{N}_\beta$, so that $P_{<\beta}$ holds, and it suffices to verify $P_{=\beta}$. We shall denote by N the set $\tilde{N}_\alpha$ (of cardinal $\mathfrak{c}$). Then $y^{(\nu)}_\lambda$ is independent of ν for $\lambda < \alpha$. This quantity will be denoted by z_λ.

Write $f(\nu)$ instead of $y^{(\nu)}_\beta$, X instead of $\tilde{N}_\beta$. Consider the set T of all $\nu \in X$. Its cardinal is $< \mathfrak{c}$.

```
Definition E6_33_N0 := fun_image (E6_33_Mi E6_33_al)
   (fun t => rep (Zo E6_33_c
      (fun n =>  inc E6_33_al (source (E6_33_y n))
              /\ Vf (E6_33_y n)  E6_33_al = t))).
Lemma Exercise6_33q (N0 :=  E6_33_N0) (al :=  E6_33_al)  (* 36 *)
  (Ma := E6_33_Mi E6_33_al):
  [/\ forall t, inc t N0 ->
   [/\ inc t E6_33_c, inc al (source (E6_33_y t))& inc (Vf (E6_33_y t) al) Ma],
    (forall t1 t2, inc t1 N0 -> inc t2 N0 ->
```

```
    (Vf (E6_33_y t1) al)  = (Vf (E6_33_y t2) al)  -> t1  = t2),
  (forall s, inc s Ma -> exists2 t, inc t N0 & s = (Vf (E6_33_y t) al)),
  sub N0 E6_33_c &
cardinal N0 =  E6_33_c].
```

Chapter 14

Compatibility

This chapter explains the differences between the current version and the previous one.

14.1 Pseudo Ordinals

This section explains the previous implementation of pseudo-ordinals, it has been replaced by the start of Chapter 4.

Consider the following properties:

$$\emptyset \in \mathrm{E}. \tag{1}$$

$$\mathrm{E} \text{ is transitive.} \tag{2}$$

$$\text{The relation “}x \in \mathrm{E} \text{ and } y \in \mathrm{E} \text{ and } (x = y \text{ or } x \in y)\text{” is a well-ordering of E.} \tag{3}$$

$$\text{The relation “}x \in \mathrm{E} \text{ and } y \in \mathrm{E} \text{ and } x \subset y\text{” is a well-ordering of E.} \tag{4}$$

$$\text{The relation “}x \in \mathrm{E} \text{ and } y \in \mathrm{E} \text{ and } x \in y\text{” is a strict well-ordering of E.} \tag{5}$$

$$\mathrm{E} \text{ is asymmetric for } \in. \tag{6}$$

$$\forall x \in \mathrm{E}, x = \,]\leftarrow, x[. \tag{7}$$

$$\text{Every transitive subset of E is E or an element of E.} \tag{8}$$

$$\forall x \in \mathrm{E}, \forall y \in \mathrm{E} \implies \text{ exactly one of } x \in y, y \in x, x = y. \tag{9}$$

$$\forall x \in \mathrm{E}, f(x) = f\langle]\leftarrow, x[\rangle. \tag{10}$$

Different variants of ordinals can be found in the literature. The 1956 Edition of Bourbaki considered (1), (2) and (3), the current edition uses (8); the most common definition is (2) and (5) (for instance [14, 2]. The von Neumann definition (see [23]) uses (4) and (7).

Let's start with a few comments. Condition (6) say that E is decent, condition (5) is equivalent to (3)(6), it implies (9). We shall see that the orderings defined by (3), (4), (5) are the same, they define the natural ordering $o(\mathrm{E})$ of relation (OP).

According to von Neumann, an ordinal is a well-ordered set satisfying (7), where $]\leftarrow, x[$ is the set of elements $y \in \mathrm{E}$ satisfying $y < x$. The relation $x = \,]\leftarrow, x[$ means: for any element x of E, the relation $y \in x$ is equivalent to $y \in \mathrm{E}$ and $y < x$. From $y \in \mathrm{E}$ we deduce that E is transitive. On the other hand, if x and y belong to E, $y \in x$ is equivalent to $y < x$, so that $a \le b$ is equivalent to $a = b$ or $a \in b$. If $x \le y$ we have $]\leftarrow, x[\, \subset \,]\leftarrow, y[$, thus $x \subset y$. Conversely, if $x \subset y$, since E is totally ordered, we have $x \le y$ or $y \le x$. In the second case we have $y \subset x$ thus $x = y$ by extensionality. Thus we have shown: $a \le b$ if and only if $a \subset b$. As a consequence, the ordering of E satisfies (3) and (4). It satisfies (5) since $x \not\in x$ is a consequence of $x \not\in]\leftarrow, x[$. A *numeration* of an ordered set E is function f satisfying (10). This means that $f(x)$ is the set of all $f(y)$ for $y < x$. Relation (7) says that the identity function is a numeration. By transfinite induction, every well-ordered set has a numeration; relation (OP) follows from the fact that the image of a numeration is an ordinal.

In the 1956 version, Bourbaki defined an ordinal via (1), (2), and (3). Consider a set W such that $W = \{\emptyset, W\}$. According to the axiom of foundation, no such set exists; according to AFA (anti-foundation axiom) there is a unique set satisfying this condition. These two axioms being independent of the Bourbaki theory, whether or not W exists is undecidable. This set is not decent, but it satisfies (1), (2) and (3), thus was an ordinal according to the 1956 Edition of Bourbaki. It is isomorphic to the set $\{\emptyset, \{\emptyset\}\}$, contradicting uniqueness of (OP). According to (1), the empty set is not an ordinal, contradicting existence. Note that the least element of a non-empty ordinal cannot have elements (by transitivity), thus must be empty.

We start with the definition of [14]. There are four conditions K_1, K_2, K_3 and K_4. Conditions K_1 and K_4 say that E is decent and transitive. Condition K_2 says that $x \in y$ is a transitive relation on E. We show here that K_1 and K_2 say that the relation "$x \in y$ or $x = y$" is an ordering on E. Condition K_3 will imply that $\in$ is a well-ordering.

```
Definition Kordinal a :=
  ( (forall  x y, inc x a -> inc y a -> inc x y -> inc y x -> False)
   & (forall x y z, inc x a -> inc y a -> inc z a ->
       inc x y -> inc y z -> inc x z)
   & (forall z, sub z a -> nonempty z ->
       exists x, (inc x z & forall y, inc y z -> inc x y \/ x = y))
   & (forall x y, inc x a -> inc y x -> inc y a)).
```

Condition K_3 implies that $x \in y$ is a strict total ordering (at least one of $x \in y$, $y \in x$ or $x = y$ is true). It implies that $x \in y$ is equivalent to $x \subsetneq y$, and that $x \subset y$ is equivalent to "$x \in y$ or $x = y$". Thus $\subset$ is a well-ordering.

```
Lemma Kordinal_asymmetric: forall E, Kordinal E -> asymmetric_set E.
Lemma Kordinal_decent: forall E, Kordinal E -> decent_set E.
Lemma Kordinal_irreflexive: forall E , Kordinal E -> inc E E -> False.
Lemma Kordinal_transitive: forall E, Kordinal E -> transitive_set E.
Lemma Kordinal_trichotomy: forall E x y, Kordinal E ->
  inc x E -> inc y E -> (inc x y \/ inc y x \/ x = y).
Lemma Kordinal_inclusion: forall E, Kordinal E ->
  forall x y, inc x E -> inc y E -> (inc x y = strict_sub x y).
Lemma Kordinal_inclusion1: forall E, Kordinal E ->
  forall x y, inc x E -> inc y E -> (sub x y = (inc x y \/ x = y)).
Lemma Kordinal_inclusion2:  forall E,
  Kordinal E -> worder (inclusion_suborder E).
```

We now show that K_3 implies that $\in$ is a well-ordering, so that the definition of [14] is equivalent to (2), (3) and (6), in other terms: E is transitive, asymmetric and is a well-ordering.

```
Lemma trans_sym_order: forall x,
  (forall y,inc y x ->  transitive_set y) -> asymmetric_set x ->
  (order (ordinal_oa x) & substrate (ordinal_oa x) = x). (* 18 *)
Lemma Kordinal_elt1:  forall E,
  Kordinal E -> worder (ordinal_oa E).
Lemma Kordinal_pr:  forall E,
  Kordinal E =
  (transitive_set E & worder (ordinal_oa E) & asymmetric_set E). (* 24 *)
```

If the set E is ordered by $\in$, the segment $]\leftarrow, x[$ is the set of all elements y in E such that $y \in x$. This is the intersection of x and E. If E is transitive, this is x. In particular, if E is a K-ordinal, we have $x =]\leftarrow, x[$. The same is true if we consider the ordering $\subset$.

We deduce that K-ordinals satisfy the von Neumann condition (7). The converse is true by the argument explained above (relation (7) says that E is transitive and that $x \in y$ is equivalent to $x \subsetneq y$. This implies that E is asymmetric and that the ordering induced by $\subset$ is the same as that of $\in$).

```
Lemma Kordinal_segment1: forall E x, decent_set E ->
  inc x E -> segment (ordinal_oa E) x = intersection2 x E.
Lemma Kordinal_segment2: forall E x, decent_set E -> transitive_set E->
  inc x E -> segment (ordinal_oa E) x = x.
Lemma Kordinal_segment3: forall E x, Kordinal E ->
  inc x E -> segment (ordinal_oa E) x = x.
Lemma Kordinal_segment4: forall E x, Kordinal E ->
  inc x E -> segment (inclusion_suborder E) x = x.

Lemma Kordinal_pr2:  forall E,
  Kordinal E =
    (worder (inclusion_suborder E) &
    (forall x, inc x E -> segment (inclusion_suborder E) x = x)).
```

Every element of a K-ordinal is transitive since we can replace $\in$ by $\subsetneq$ which is transitive. Every transitive subset of E is a K-ordinal (this comes directly from the definition; it is also a consequence of the fact that a subset of a well-ordered set is well-ordered). Thus, every element of a K-ordinal is a K-ordinal. We now state: (9) is true whenever x and y are K-ordinals (there is no need to add the restrictions $x \in$ E and $y \in$ E). In fact, the intersection of two ordinals is a segment of each one. Since the sets are well-ordered, a segment is the whose set or of the form $]\leftarrow, x[$, and we know that this is x. Hence we get $x \cap y = x$ or $x \cap y \in x$, and similarly, $x \cap y = y$ or $x \cap y \in y$; the conclusion follows since $x \cap y \notin x \cap y$. It follows that, if E is a K-ordinal, X a transitive strict subset of E, then X is a K-ordinal, and X $\in$ E, so that E is an ordinal in the Bourbaki sense.

```
Lemma Kordinal_sub_trans: forall E x, Kordinal E -> inc x E ->
  transitive_set x.
Lemma Kordinal_sub_ordinal: forall E x, Kordinal E -> sub x E ->
  transitive_set x -> Kordinal x.
Lemma Kordinal_inc_ordinal: forall E x, Kordinal E -> inc x E ->
  Kordinal x.
Lemma Kordinal_trichotomy1 : forall x y,
  Kordinal x -> Kordinal y ->
  (inc x y \/ inc y x \/ x = y). (* 29 *)
```

Bourbaki defines an ordinal as a set E that satisfies (8). The following result, can be restated (thanks to `Kordinal_pr`) as: a Bourbaki ordinal is transitive, asymmetric, and well-ordered by $\in$. This property has been proved in the main text, page 67.

```
Lemma Kordinal_pr3:  forall E,
  Kordinal E = Bordinal E.
```

Cardinals This is how Bourbaki defines a cardinal, and the cardinals zero, one and two.

```
(*
```

```
  Definition cardinal x := choose (fun z => equipotent x z).
  Definition card_zero := cardinal emptyset.
  Definition card_one := cardinal (singleton emptyset).
  Definition card_two := cardinal (two_points).
*)
```

Using von Neumann ordinals for cardinals makes some theorems easier.

Theorem one [4, p. 159] says that the ordering between cardinals is a well-ordering. The idea is the following. Let E be a set of cardinals, and A its union. Consider a well-ordering on A. Let $\phi(x)$ be the smallest segment of A equipotent to x (if $x \in \mathrm{E}$, x is a subset of A hence isomorphic, hence equipotent, to a segment of A; hence ϕ is well-defined). The relation $a \le_{\mathrm{Card}} b$ on E is equivalent to $\phi(a) \subset \phi(b)$ (if $a \le_{\mathrm{Card}} b$ then a is isomorphic to a subset of $\phi(b)$, hence a is isomorphic to a segment u of $\phi(b)$; by definition $\phi(a) \subset u$, hence $\phi(a) \subset \phi(b)$; converse is easy). From this, one deduces that the relation $a \le_{\mathrm{Card}} b$ is an order on E. This is a well-ordering, since the set of segments is well-ordered. Assume $a \le_{\mathrm{Card}} b$ and $b \le_{\mathrm{Card}} a$. If we consider the doubleton $\{a, b\}$, we have $a \le b$ and $b \le a$ for the induced order, hence $a = b$.

If a is equipotent to a subset of b, and conversely then a is equipotent to b. This is equivalent to the Cantor-Bernstein theorem. Since $a \le_{\mathrm{Card}} b$ is a well-ordering it is a total ordering.

```
Lemma cardinal_le5: forall E, (forall x, inc x E -> is_cardinal x) ->
  substrate (graph_on cardinal_le E) = E.
Theorem wordering_cardinal_le: worder_r  cardinal_le.  (* 137 *)
Lemma cardinal_le7: forall a b c,
  equipotent b c -> equipotent_to_subset a b ->
  equipotent_to_subset a c.
Lemma cardinal_antisymmetry2: forall a b,
  equipotent_to_subset a b -> equipotent_to_subset b a ->
  equipotent a b.
Lemma cardinal_le_total_order: forall a b,
  equipotent_to_subset a b \/ equipotent_to_subset b a.
```

For every cardinal $\mathfrak{a}$, the set of objects of the form $\mathrm{Card}(b)$ for $b \in \mathfrak{P}(\mathfrak{a})$ is the set of cardinals $\le \mathfrak{a}$.

Since $\le_{\mathrm{Card}}$ is a well-ordering, thus total, a finite cardinal is always smaller than an infinite cardinal. Fix some infinite cardinal $\mathfrak{b}$ (for instance the cardinal of $\mathbb{N}$), and consider the subset of $\{\mathfrak{a}, \mathfrak{a}$ is cardinal and $\mathfrak{a} \le_{\mathrm{Card}} \mathfrak{b}\}$ formed of finite cardinals. It contains all finite cardinals. It is independent of $\mathfrak{b}$. It will be called the set of natural integers and denoted by `Bnat` or **N**.

```
Definition set_of_cardinals_le a:=
  fun_image(powerset a)(fun x => cardinal x).
Definition Bnat := Zo(set_of_cardinals_le (cardinal nat))
  (fun z => finite_c z).
```

This is the old definition of the predecessor function.

```
Definition predc n := choose (fun m => is_cardinal m & n = succ m).
Lemma predc_pr0: forall n, is_cardinal n -> n <> card_zero ->
  (is_cardinal (predc n) & n = succ (predc n)).
Theorem exists_predc: forall n, inc n Bnat -> n <> card_zero ->
  exists_unique (fun m => inc m Bnat & n = succ m).
```

The von Neumann Proof. Von Neumann calls a function satisfying (10) a *numeration*; and says that a well-ordered set is *numerable* if there is a numeration. We have shown in the main text that `ordinal_iso` is a numeration. In this section we explain the von Neumann construction. For simplicity, we use here a functional graph, instead of a function, this avoid introducing the target.

```
Definition numeration r f:=
  [/\ fgraph f,  domain f = substrate r &
  forall x, inc x (domain f) ->  Vg f x = direct_image f (segment r x)].
Definition numerable r :=
   worder r & exists f, numeration r f.

Lemma worder_numerable_bis  r:
  worder r -> numeration r (graph (ordinal_iso r)).
```

The proofs are rather straightforward (compare with the proof of existence of a function by transfinite induction). The key relation is that, if $n(\mathrm{E})$ is the numeration of E, and F a segment of E then $n(\mathrm{F})$ is the restriction of $n(\mathrm{E})$ to F. If all segments of E can be numerated, we can extent these numerations to a numeration of E; otherwise, there is a least non-numerable segment; all its segments can be numerated, so that the segment can be numerated; absurd.

```
Lemma numeration_unique r f f':
  worder r -> numeration r f -> numeration r f' -> f = f'.
Lemma sub_numeration r f x:
  let r' := induced_order r (segment r x) in
  worder r ->  numeration r f -> inc x (substrate r) ->
  numeration r' (restr f (segment r x)).
Lemma segments_numerables r:
  let r' := fun x => induced_order r (segment r x) in
  worder r -> (forall x, inc x (substrate r)-> numerable (r' x)) ->
  numerable r. (* 57 *)
Lemma worder_numerable r:
  worder r ->  numerable r.
```

Pseudo-ordinals and the type nat. Our implementation of Bourbaki in Coq relies on the fact that a set is a type, and if a is a set, $a \in \mathrm{B}$ means $a = \mathscr{R}b$ for some b of type B (where $\mathscr{R}$ denotes `Ro`). This is an abstract construction. If we define a type A with two constructors B and C, then $\mathscr{R}\mathrm{B} \in \mathrm{A}$ and $\mathscr{R}\mathrm{C} \in \mathrm{A}$. We assume $\mathscr{R}$ injective; since $\mathrm{B} \neq \mathrm{C}$ by construction, the set A has two distinct elements $\mathscr{R}\mathrm{B}$ and $\mathscr{R}\mathrm{C}$. The only property of $\mathscr{R}\mathrm{B}$ is that it is one of the two elements of A. A property of the form $\mathscr{R}\mathrm{B} = \emptyset$ is undecidable.

Let's now define a more complicated type, `nat`, denoted $\mathbb{N}$; it has a constant constructor O, and another constructor S that is a function on $\mathbb{N}$. This means that, whenever x is of type $\mathbb{N}$, then $\mathrm{S}x$ is also of type $\mathbb{N}$. This set satisfies the principle of induction (that says under which condition a property is true for the elements of this set), and we can define a function by induction. Later on, Bourbaki introduces $\mathbf{N}$, the set of finite cardinals. It satisfies the principle of induction (see section 5.4), and functions can be defined by induction (Chapter six, section 6.6), as a variant of definition by transfinite induction of the well-ordered set $\mathbf{N}$. In the next section, we shall show that $\mathbb{N}$ and $\mathbf{N}$ are isomorphic; in this section we compare $\mathbb{N}$ and the collection of finite pseudo-ordinals (this is in fact a set, but it will not be used).

Since $\mathrm{S}n$ is of type $\mathbb{N}$ whenever n is of type $\mathbb{N}$, we can define a function s such that $s(a) \in \mathbf{N}$ whenever $a \in \mathbb{N}$: if $a \in \mathbb{N}$ and $a = \mathscr{R}(b)$ then $s(a) = \mathscr{R}(\mathrm{S}(b))$. As noted above, a property of the form $\mathscr{R}\mathrm{O} = \emptyset$ is undecidable. Although the exact value of $s(\mathscr{R}\mathrm{O})$ is unknown, we can show some properties of s: for instance, s is injective and not surjective (there is a unique way to construct an object of type $\mathbb{N}$; so that $\mathrm{S}x$ is never O and $\mathrm{S}x = \mathrm{S}y$ implies $x = y$). The COQ parser and pretty printer identify O and 0, SO and 1, SSO and 2. In order to avoid confusion, we shall write **0**, **1** and **2** for the cardinals (note that $\mathbf{0} = \emptyset$).

Let's define by induction a function f by $f(\mathbf{0}) = \mathscr{R}\mathrm{O}$ and $f(n+\mathbf{1}) = \mathscr{R}(\mathrm{S}(\mathscr{B}(f(n)))$ where $a = \mathscr{B}(f(n))$ is defined by $\mathscr{R}a = f(n)$. For instance $f(\mathbf{1}) = \mathscr{R}(1)$. The property "$f$ is the identity function" (more precisely $f = \mathscr{R}$) is undecidable (it cannot be proved; we hope that adding it as an axiom does not make the theory contradictory).

The type $\mathbb{N}$ is called `nat` in COQ; it has an order relation, noted $\leq$, and two operations $+$ and $*$, that correspond, via the bijection f, to comparison, sum and product of finite cardinals. We shall import all theorems about natural integers from the COQ library by identification of **N** and $\mathbb{N}$. Given that $\emptyset = f(\mathbf{0}) = \mathscr{R}0 = \mathscr{R}\mathrm{O}$, we may assume $\mathscr{R}\mathrm{O} = \emptyset$. The relation $f(\mathbf{1}) = \mathscr{R}(1)$ suggest that $\mathscr{R}(1)$ should be **1**, but this is a set defined via the axiom of choice, as a set with one element; it could be $\{\emptyset\}$, it could also be any other set. We will add the relation $\mathscr{R}(\mathrm{SO}) = \{\emptyset\}$ as axiom. As a consequence $\mathscr{R}(\mathrm{SO})$ is unlikely to be a cardinal, but it will allow us to construct a function card, such that $\mathrm{card}(x) = 1$ whenever x is a singleton, i.e., whenever $\mathrm{Card}(x) = \mathbf{1}$. The two axioms relating $\mathscr{R}$, S and O have been introduced by Carlos Simpson in the following way:

```
(*
   Axiom nat_realization_O : forall x : Set, ~ inc x (Ro 0).
   Axiom nat_realization_S :
          forall (n : nat) (x : Set),
          inc x (Ro (S n)) = (inc x (Ro n) \/ x = Ro n).
   Lemma nat_zero_emptyset : Ro 0 = emptyset.
*)
```

These axioms are useless, hence have been withdrawn. On the other hand, we can define a function that shares exactly the same properties. The first axioms defines a set $\mathscr{R}0$ that contains no element, hence is the empty set. The second axioms defines $\mathscr{R}(\mathrm{S}n)$, that is equal (by extensionality) to $\mathrm{T}(\mathscr{R}n)$. Thus, we define `natR` denoted by $\mathscr{R}_{\mathbb{N}}$, via $\mathscr{R}_{\mathbb{N}}0 = \emptyset$ and $\mathscr{R}_{\mathbb{N}}(\mathrm{S}n) = \mathrm{T}(\mathscr{R}_{\mathbb{N}}n)$.

```
Fixpoint natR (n:nat) :=
  match n with 0 =>  emptyset
             | S p => tack_on (natR p) (natR p)
  end.
```

The conclusion of Exercise 20 is: *In particular the pseudo-ordinals whose order-type are* $0, 1, 2 = 1+1$, *and* $3 = 2+1$ *are respectively*

$$\emptyset, \quad \{\emptyset\}, \quad \{\emptyset,\{\emptyset\}\}, \quad \{\emptyset,\{\emptyset\},\{\emptyset,\{\emptyset\}\}\}.$$

```
Lemma value_R_0: natR 0 = emptyset.
Lemma value_R_1: natR 1 = singleton emptyset.
Lemma value_R_2: natR 2 = doubleton (singleton emptyset) emptyset.
Lemma value_R_3: let tripleton a b c := tack_on (doubleton a b) c in
  natR 3 = tripleton (doubleton (singleton emptyset) emptyset)
  (singleton emptyset) emptyset.
```

If a is a pseudo-ordinal, then so is $\mathrm{T}(a)$. By induction, we deduce that if n is of type `nat`, $\mathscr{R}_{\mathbb{N}}n$ is a finite pseudo-ordinal.

```
Lemma pseudo_ordinal_Rnat: forall i,  pseudo_ordinal (natR i).
Lemma finite_Rnat: forall i, is_finite_set (natR i).
```

Define a relation $\le$ on nat by the properties: $\forall x, x \le x$ and $\forall x y$, $x \le y \implies x \le \mathrm{S}(y)$. This is reflexive and transitive. Showing that it is an order is not completely trivial (see the Coq library). One can show that the relation $x < y$, defined by $\mathrm{S}x \le y$, is equivalent to $x \le y$ and $x \ne y$.

It is clear by induction that if $x \le y$ then $\mathscr{R}_{\mathbb{N}} x \subset \mathscr{R}_{\mathbb{N}} y$. If $x < y$ then $\mathscr{R}_{\mathbb{N}}(\mathrm{S}x) \subset \mathscr{R}_{\mathbb{N}} y$ hence $\mathscr{R}_{\mathbb{N}} x \in \mathscr{R}_{\mathbb{N}} y$. If $\mathscr{R}_{\mathbb{N}} x \subset \mathscr{R}_{\mathbb{N}} y$, then $x \le y$, for otherwise we would have $y < x$, hence $\mathscr{R}_{\mathbb{N}} y \in \mathscr{R}_{\mathbb{N}} x$, hence $\mathscr{R}_{\mathbb{N}} x \in \mathscr{R}_{\mathbb{N}} x$, absurd. In the same fashion, $x < y$ is equivalent to $\mathscr{R}_{\mathbb{N}}(\mathrm{S}x) \subset \mathscr{R}_{\mathbb{N}} y$. Note that, if $\mathscr{R}_{\mathbb{N}} i = \mathscr{R}_{\mathbb{N}} j$, then $\mathscr{R}_{\mathbb{N}} i \subset \mathscr{R}_{\mathbb{N}} j$ and $\mathscr{R}_{\mathbb{N}} j \subset \mathscr{R} i$, this $i \le j$ and $j \le i$; hence $i = j$. This shows injectivity of $\mathscr{R}_{\mathbb{N}}$.

```
Lemma Rnat_le_implies_sub : forall i j,  i <= j -> sub (natR i) (natR j).
Lemma Rnat_lt_implies_inc : forall i j,  i < j -> inc (natR i) (natR j).
Lemma Rnat_lt_implies_strict_sub : forall i j,
  i < j -> strict_sub (natR i) (natR j).
Lemma Rnat_sub_le : forall i j,  sub (natR i) (natR j) = (i <= j).
Lemma Rnot_inc_itself: forall i, ~ (inc (natR i)(natR i)).
Lemma Rnat_inc_lt : forall i j,  inc (natR i) (natR j) = (i < j).
```

Let $f(i)$ be short for $\mathrm{Card}(\mathscr{R}_{\mathbb{N}} i)$. As a consequence, if $i < j$ then $f(i) <_{\mathrm{Card}} f(j)$, and if $f(i) = f(j)$, then $i = j$. From this, we deduce that each finite cardinal is of the form $f(i)$ for a unique i.

```
Lemma cardinal_Rnat_lt: forall i j,
  i< j -> cardinal_lt (cardinal (natR i)) (cardinal (natR j)).
Lemma  cardinal_Rnat_inj: forall i j,
  cardinal (natR i) = cardinal (natR j) -> i = j.
Lemma exists_nat_cardinal: forall a, is_finite_c a ->
  exists_unique(fun i:nat => cardinal (natR i)=a).
```

We can now introduce a function $\mathrm{card}(n)$ making the following diagram commutative

(Finite cardinals)

$$\begin{tikzcd} \mathrm{Set} \arrow[r, "\mathrm{Card}"] \arrow[d, "\mathrm{card}"] & \mathbf{N} \\ \mathbb{N} \arrow[r, "\mathscr{R}_{\mathbb{N}}"'] & \mathrm{Set} \arrow[u, "\mathrm{Card}"'] \end{tikzcd}$$

In this diagram, $\mathbb{N}$, Set and $\mathbf{N}$ are types; $\mathbb{N}$ is the set of natural numbers (as a Coq type), and $\mathbf{N}$ is the collection of cardinals (the objects x such that x is a cardinal do not form a set, neither a type, we call it a *collection*). We use the notation $\mathbf{N}$ to emphasize that if x is a finite set, then its cardinal is a member of the set of finite cardinals. If x is not finite, we define $\mathrm{card}(x)$ to be 0, in this case the diagram does not commute. If however n is finite we have $\mathrm{Card}(\mathscr{R}_{\mathbb{N}}(\mathrm{card}(n))) = \mathrm{Card}(n)$. By uniqueness, we have $\mathrm{card}(\mathscr{R}_{\mathbb{N}}(i)) = i$ for every nat i. If A is a finite set, then $\mathrm{Card}\,\mathrm{A} = \mathrm{Card}\,\mathrm{B}$ implies $\mathrm{card}\,\mathrm{A} = \mathrm{card}\,\mathrm{B}$. The converse is true if both sets are finite.

```
Definition cardinal_nat x := choosenat(fun i => cardinal (natR i) = x).
Lemma cardinal_nat_cardinal: forall x,
```

```
  cardinal_nat (cardinal x) = cardinal_nat x.
Lemma cardinal_nat_pr: forall x, is_finite_set x ->
  cardinal (natR (cardinal_nat x)) = cardinal x.
Lemma cardinal_nat_pr1: forall i, cardinal_nat(natR i) = i.
Lemma cardinal_nat_finite_eq: forall a b, is_finite_set a ->
  cardinal a = cardinal b -> cardinal_nat a = cardinal_nat b.
Lemma cardinal_nat_finite_eq1: forall a b,
  is_finite_set a -> is_finite_set b ->
  cardinal_nat a = cardinal_nat b -> cardinal a = cardinal b.
```

We have $\mathrm{card}\,\mathbf{0} = 0$, $\mathrm{card}\,\mathbf{1} = 1$ and $\mathrm{card}\,\mathbf{2} = 2$. Note that, if s is the successor function, then $\mathbf{3} = s(\mathbf{2})$ and $3 = \mathrm{S}2$, this implies $\mathrm{card}\,\mathbf{3} = 3$.

```
Lemma cardinal_nat_emptyset: cardinal_nat emptyset = 0.
Lemma cardinal_nat_singleton: forall x, cardinal_nat (singleton x) = 1.
Lemma cardinal_nat_doubleton: forall x y,
  x <> y -> cardinal_nat (doubleton x y) = 2.
Lemma cardinal_nat_zero: cardinal_nat card_zero = 0.
Lemma cardinal_nat_one: cardinal_nat card_one = 1.
Lemma cardinal_nat_two: cardinal_nat card_two = 2.
```

Bijection between nat and the integers Denote by $s(n)$ the successor of n. We have $s(0) = 1$, and $a + s(n) = s(a + n)$ (associativity of the sum). We have $a.s(b) = ab + a$ and $a.0 = 0$. By induction we deduced that the sum and product of two integers are integers.

```
Lemma plus_via_succ: forall a n,
  card_plus a (succ n) = succ (card_plus a n).
Lemma Bnat_stable_plus: forall a b, inc a Bnat -> inc b Bnat ->
  inc (card_plus a b) Bnat.
Lemma mult_via_plus: forall a b, is_cardinal a ->
  card_mult a (succ b) = card_plus (card_mult a b) a.
Lemma Bnat_stable_mult: forall a b, inc a Bnat -> inc b Bnat ->
  inc (card_mult a b) Bnat.
```

We define now by induction a function $\mathcal{N}$ that associates a cardinal to each `nat`; by construction $\mathcal{N}(0) = \mathbf{0}$ and $\mathcal{N}(\mathrm{S}n) = s(\mathcal{N}(n))$. This function converts addition and multiplication on `nat` to cardinal sum and cardinal product (induction on $\mathbb{N}$). This function is injective (because `succ` is injective). By induction on $\mathbf{N}$, this function is surjective. More precisely, every finite cardinal has the form $\mathcal{N}(n)$. Assume $a \le b$; then $\mathcal{N}(a) \le_{\mathrm{Card}} \mathcal{N}(b)$; conversely, if this relation holds then $\mathcal{N}(a) + x = \mathcal{N}(b)$ for some x. By surjectivity $x = \mathcal{N}(c)$, by injectivity $a + c = b$ which implies $a \le b$. By injectivity of $\mathcal{N}$ we deduce that $a < b$ and $\mathcal{N}(a) <_{\mathrm{Card}} \mathcal{N}(b)$ are equivalent.

Let g be the inverse of $\mathcal{N}$; such a function exists using the axiom of choice. By uniqueness this function is card, hence $\mathrm{card}(\mathcal{N}(x)) = x$ and $\mathcal{N}(\mathrm{card}(x)) = x$.

```
Fixpoint nat_to_B (n:nat) :=
  match n with 0 => card_zero | S m => succ (nat_to_B m) end.

Lemma nat_B_0: nat_to_B 0 = card_zero.
Lemma nat_B_1: nat_to_B 1 = card_one.
Lemma nat_B_2:  nat_to_B 2 = card_two.
Lemma nat_B_S: forall n, nat_to_B (S n) = succ (nat_to_B n).
```

```
Lemma inc_nat_to_B: forall n, inc (nat_to_B n) Bnat.
Lemma nat_B_plus: forall a b,
  nat_to_B (a+b) = card_plus (nat_to_B a) (nat_to_B b).
Lemma nat_B_mult: forall a b,
  nat_to_B (a*b) = card_mult (nat_to_B a) (nat_to_B b).

Lemma nat_B_inj: forall a b,
  nat_to_B a = nat_to_B b -> a = b.
Lemma nat_to_B_surjective: forall n, inc n Bnat -> exists m,
  nat_to_B m = n.
Lemma nat_B_le: forall a b,
  (a<= b) =  cardinal_le (nat_to_B a) (nat_to_B b).
Lemma nat_B_lt: forall a b,
  (a< b) =  cardinal_lt (nat_to_B a) (nat_to_B b).
Lemma nat_B_lt: forall a b,
  (a< b) =  cardinal_lt (nat_to_B a) (nat_to_B b).
Lemma nat_B_lt0: forall b,
  (0<b) = cardinal_lt card_zero (nat_to_B b).
Lemma nat_to_B_pr: forall n, inc n Bnat ->
  nat_to_B (cardinal_nat n) = n.
Lemma nat_to_B_pr1: forall n,
  cardinal_nat(nat_to_B n) = n.
```

We finish with the definition of the power function on $\mathbb{N}$.

```
Fixpoint pow (n m:nat) {struct m} : nat :=
  match m with
  | 0 => 1
  | S p => (pow n p) * n
  end
where "n ^ m" := (pow n m) : nat_scope.
```

Ordinals [the second lemma has been withdrawn; in the first lemma, the last assumption is replaced by: some family f_i satisfies a given condition, and the conclusion is now: a function f depending on these f_i satisfies some condition.]

```
Lemma transfinite_aux2: forall r p s, worder r -> (* 70 *)
  (forall z, inc z s -> is_segment r z) ->
  (forall z, inc z s -> (exists f : correspondenceC,
              transfinite_def (induced_order r z) p f)) ->
  exists f : correspondenceC, transfinite_def (induced_order r (union s)) p f.

Lemma order_morphism_pr1: forall f r r',
  order r -> order r' -> is_function f -> substrate r = source f ->
  substrate r' = target f ->
  (forall x y, inc x (source f) -> inc y (source f) ->
    gle r x y = gle r' (W x f) (W y f))
  -> order_morphism f r r'.
```

[These are some original definitions and lemmas.]

```
Definition ord_Bnat n := order_type(interval_Bnat_co n)
Lemma Bordinal_ord_Bnat: forall n, inc n Bnat -> Bordinal (ord_Bnat n).
Lemma cardinal_ord_Bnat: forall n, inc n Bnat ->
```

```
  cardinal (substrate (ord_Bnat n)) = n.
Lemma ord_Bnat_injective: forall n m, inc n Bnat -> inc m Bnat ->
  (ord_Bnat n = ord_Bnat m) -> n = m.
Lemma finite_ordinal: forall x, Bordinal x -> is_finite_set (substrate x)
  ->  x = ord_Bnat (cardinal (substrate x)).
Lemma ord_zero_card: ord_Bnat card_zero = ord_zero.
Lemma ord_one_card: ord_Bnat card_one = ord_one.
Lemma ord_two_card_pr1: forall x,
  Bordinal x -> card_two = cardinal (substrate x) ->
  x = ord_Bnat card_two.
Lemma ord_two_card: ord_Bnat card_two = ord_two.
```

The order-type of the order-sum of a family of order-types will be called the *ordinal sum* and denoted by $\sum_{i\in I}\lambda_i$ (abuse of notations here). The order-type of the order-product of a family will be called the *ordinal product.* If arguments are order-types so is the result of the operation.

```
Definition ord_sum r g := ordinal (order_sum r g).
Definition ord_prod r g := ordinal (order_product r g).
Definition ord_sum2 a b := ordinal (order_sum2 a b).
Definition ord_prod2 a b := ordinal (order_prod2 a b).

Lemma ord_sum_type: forall r g,
  order r -> substrate r = domain g -> fgraph g ->
  (forall i, inc i (domain g) -> order (V i g))
  -> is_order_type (ord_sum r g).
Lemma ord_prod_type: forall r g,
  worder r -> substrate r = domain g -> fgraph g ->
  (forall i, inc i (domain g) -> order (V i g))
  -> is_order_type (ord_prod r g).
```

In Exercice 14(c) Bourbaki say that if α is an ordinal, then the relation "ξ is an ordinal and $\xi \le_{\mathrm{Ord}} \alpha$" is collectivizing in ξ. This means that there is a set O'_α whose elements are all sets ξ such that $\xi \le_{\mathrm{Ord}} \alpha$ (note that this implies that ξ is an ordinal). There is also a set O_α whose elements are all sets ξ such that $\xi <_{\mathrm{Ord}} \alpha$. This set is well-ordered by $\le_{\mathrm{Ord}}$ and $\mathrm{Ord}(\mathrm{O}_\alpha) = \alpha$. Notice that the first set is the set of all order-types of segments S of α, and the second set is obtained by considering the strict segments, which are of the form $]\leftarrow, x[$. The mapping $x \mapsto \mathrm{Ord}(]\leftarrow, x[)$ is the desired isomorphism. The important property here is that two distinct segments of a well-ordered sets are never isomorphic.

```
Definition set_of_ordinal_le a:=
   fun_image (set_of_segments a)  (fun z => order_type (induced_order a z)).
Definition set_of_ordinal_lt a:=
  fun_image (substrate a)(fun z => order_type (induced_order a (segment a z))).

Lemma segments_iso2: forall a A B, worder a ->
   inc A  (set_of_segments a) -> inc B  (set_of_segments a)  ->
   (induced_order a A) \Is (induced_order a B) -> A = B.

Lemma set_ord_le_prop: forall a, is_ordinal a ->
  (forall x, inc x (set_of_ordinal_le a) = ordinal_le x a).
Lemma set_ord_lt_prop: forall a, is_ordinal a ->
  (forall x, inc x (set_of_ordinal_lt a) = ordinal_lt x a). (* 26 *)
Lemma set_ord_lt_prop2: forall a, is_ordinal a ->
```

```
  order_isomorphism (BL (fun z => order_type (induced_order a (segment a z)))
       (substrate a) (set_of_ordinal_lt a))
    a (graph_on ordinal_le (set_of_ordinal_lt a)).  (* 62 *)
Lemma set_ord_lt_prop3: forall a, is_ordinal a ->
  order_type (graph_on ordinal_le (set_of_ordinal_lt a)) = a.
```

For every family of ordinals $(\xi_i)_{i\in I}$, there exists a unique ordinal α such that "λ is an ordinal and $\xi_i \le \lambda$ for all $i \in I$" is equivalent to $\alpha \le \lambda$. It is called the least upper bound or supremum (by abuse of language, since $\le$ is an ordering without graph). Let $Q(\lambda)$ be the property that $\xi_i \le \lambda$ for all $i \in I$, and let $P(\lambda)$ be the property "λ is an ordinal and $Q(\lambda)$". Whatever Q, there is at most one ordinal α such that $\alpha \le \lambda$ is equivalent to $P(\lambda)$ (by antisymmetry of $\le$). Assume $P(\lambda)$ true for some λ. Then the supremum is the least such λ. In what follows, we consider a set of ordinals rather that a family; then the supremum of the family is the supremum of the range. It exists, since the ordinal sum of the family is an upper bound.

```
Definition ord_sup_pr E x:=
   is_ordinal x &
   forall y, ordinal_le x y =
    (is_ordinal y & forall i, inc i E -> ordinal_le i y).
Definition ord_sup E := choose (fun x => ord_sup_pr E x).
Definition ord_supf f := ord_sup (range f).
Lemma ord_sup_pr1: forall E y, Bordinal y ->
  (forall i, inc i E -> ordinal_le i y) ->
  exists x, (ord_sup_pr E x & ordinal_le x y).
Lemma ord_sup_pr2: forall E, (forall i, inc i E -> Bordinal i) ->
  ord_sup_pr E (ord_sup E).
```

Consider a family $(\lambda_i)_{i\in I}$ of ordinals. We can well-order the index set and obtain the ordinal sum λ. The next lemma shows that each i, $\lambda_i \le \lambda$. Thus, there exists a strictly increasing mapping $\lambda_i \to x_i$ such that λ_i is isomorphic to a segment $]\leftarrow, x_i[$ of λ. There is a least x_i, hence a least λ_i. Thus $\le_{\mathrm{Ord}}$ is a well-ordering.

Consider a family $(\mathfrak{a}_i)_{i\in I}$ of cardinals. We can well-order the union E. Each $\mathfrak{a}_i$ is equipotent to at least one segment of E. Let x_i be the least endpoint of such a segment. Thus, there exists a strictly increasing mapping $\mathfrak{a}_i \to x_i$ such that $\mathfrak{a}_i$ is equipotent to a segment $]\leftarrow, x_i[$. There is a least x_i, hence a least $\mathfrak{a}_i$. Thus $\le_{\mathrm{Card}}$ is a well-ordering.

In the current version, we simplified these two theorems as follows. First, we note that, if $\mathfrak{a}_i$ is any cardinal, and λ_i is the ordinal of $]\leftarrow, x_i[$, then λ_i depends only on $\mathfrak{a}_i$, but not on the other elements of the family or the ordering of E. Thus we can assume $\mathfrak{a}_i = \lambda_i$. Then $\le_{\mathrm{Card}}$ is a well-ordering as being the restriction of $\le_{\mathrm{Ord}}$ to cardinals. We can simplify the proof further by assuming $\lambda_i =]\leftarrow, x_i[$. Then $\le_{\mathrm{Ord}}$ is a well-ordering as being the same ordering as that of λ (when suitably restricted). Note that, in the case of von Neumann ordinals, an ordinal is a set with a natural ordering, while Bourbaki considers ordered sets.

```
Lemma ordinal_le_sum: forall r g j,
  worder r -> substrate r = domain g -> fgraph g ->
  (forall i, inc i (domain g) -> worder (V i g)) -> inc j (domain g)
  -> ordinal_le (ordinal (V j g)) (ordinal (order_sum r g)). (* 27 *)
```

14.2 Introduction to Chapter 6

Our current work is based on the SSREFLECT library, which redefined a certain number of functions on natural numbers. For instance the type of $a \leq b$ is now `bool` instead of `Prop`. Below is a theorem `zerop` that says that a number is zero or positive. This is restated in *ssrnat.v* by the lemma `posnP` that says that the boolean value of $n = 0$ or $0 < n$ are mutually exclusive (one is true, the other one is false).

We start this chapter with a list of some definitions and theorems, extracted from the COQ standard library implementing $\mathbb{N}$. Consider for instance `zerop`. We show its type, not its value which is irrelevant for the use we shall make of it; this value is a proof that for every n (of type `nat`), one of A or B is true. This is summarized by the notation {A} + {B} (certified disjoint union). The `heavyside` function is not part of the library, it is an example of how this construction can be used. The underscores in the definition represent the two proofs. The COQ parser and pretty-printer interpret this in the same fashion as '`if zerop n then 0 else 1`'. A property is decidable if it can be shown true of false. For us, all properties are decidable since we have an axiom that says so. It is however useful to know that equality and inequality are decidable. We state also some theorems such as if $n \leq m$ is false then $n > m$ is true. In this case, the result is a consequence of the fact that one of the properties is true.

```
(*
  Definition zerop n : {n = 0} + {0 < n}.
  Definition lt_eq_lt_dec n m : {n < m} + {n = m} + {m < n}.
  Definition gt_eq_gt_dec n m : {m > n} + {n = m} + {n > m}.
  Definition le_lt_dec n m : {n <= m} + {m < n}.
  Definition le_le_S_dec n m : {n <= m} + {S m <= n}.
  Definition le_ge_dec n m : {n <= m} + {n >= m}.
  Definition le_gt_dec n m : {n <= m} + {n > m}.
  Definition le_lt_eq_dec n m : n <= m -> {n < m} + {n = m}.

Definition heavyside n := match (zerop n) with left _ => 0 | right _ => 1 end.

  Theorem dec_le : forall n m, decidable (n <= m).
  Theorem dec_lt : forall n m, decidable (n < m).
  Theorem dec_gt : forall n m, decidable (n > m).
  Theorem dec_ge : forall n m, decidable (n >= m).
  Theorem not_eq : forall n m, n <> m -> n < m \/ m < n.
  Theorem not_le : forall n m, ~ n <= m -> n > m.
  Theorem not_gt : forall n m, ~ n > m -> n <= m.
  Theorem not_ge : forall n m, ~ n >= m -> n < m.
  Theorem not_lt : forall n m, ~ n < m -> n >= m.
*)
```

Here we show how and addition and multiplication behave with respect to ordering.

```
(*
  Lemma plus_reg_l : forall n m p, p + n = p + m -> n = m.
  Lemma plus_le_reg_l : forall n m p, p + n <= p + m -> n <= m.
  Lemma plus_lt_reg_l : forall n m p, p + n < p + m -> n < m.

  Lemma plus_le_compat_l : forall n m p, n <= m -> p + n <= p + m.
  Lemma plus_le_compat_r : forall n m p, n <= m -> n + p <= m + p.
  Lemma le_plus_l : forall n m, n <= n + m.
  Lemma le_plus_r : forall n m, m <= n + m.
```

```
  Theorem le_plus_trans : forall n m p, n <= m -> n <= m + p.
  Theorem lt_plus_trans : forall n m p, n < m -> n < m + p.
  Lemma plus_lt_compat_l : forall n m p, n < m -> p + n < p + m.
  Lemma plus_lt_compat_r : forall n m p, n < m -> n + p < m + p.
  Lemma plus_le_compat : forall n m p q, n <= m -> p <= q -> n + p <= m + q.
  Lemma plus_le_lt_compat : forall n m p q, n <= m -> p < q -> n + p < m + q.
  Lemma plus_lt_le_compat : forall n m p q, n < m -> p <= q -> n + p < m + q.
  Lemma plus_lt_compat : forall n m p q, n < m -> p < q -> n + p < m + q.

  Lemma mult_O_le : forall n m, m = 0 \/ n <= m * n.
  Lemma mult_le_compat_l : forall n m p, n <= m -> p * n <= p * m.
  Lemma mult_le_compat_r : forall n m p, n <= m -> n * p <= m * p.
  Lemma mult_le_compat :
    forall n m p (q:nat), n <= m -> p <= q -> n * p <= m * q.
  Lemma mult_S_lt_compat_l : forall n m p, m < p -> S n * m < S n * p.
  Lemma mult_lt_compat_r : forall n m p, n < m -> 0 < p -> n * p < m * p.
  Lemma mult_S_le_reg_l : forall n m p, S n * m <= S n * p -> m <= p.
  Lemma plus_le_reg_l : forall n m p, p + n <= p + m -> n <= m.
  Lemma plus_lt_reg_l : forall n m p, p + n < p + m -> n < m.
  Lemma mult_S_le_reg_l : forall n m p, S n * m <= S n * p -> m <= p.
*)
```

Here we give some properties of subtraction.

```
(*
   Lemma minus_n_n : forall n, 0 = n - n.
   Lemma minus_n_O : forall n, n = n - 0.
   Lemma le_plus_minus : forall n m, n <= m -> m = n + (m - n).
   Lemma plus_minus : forall n m p, n = m + p -> p = n - m.
   Lemma minus_plus : forall n m, n + m - n = m.
   Theorem le_minus : forall n m, n - m <= n.
   Lemma minus_plus_simpl_l_reverse : forall n m p, n - m = p + n - (p + m).
   Lemma minus_Sn_m : forall n m, m <= n -> S (n - m) = S n - m.
   Lemma le_plus_minus : forall n m, n <= m -> m = n + (m - n).
   Lemma le_plus_minus_r : forall n m, n <= m -> n + (m - n) = m.
   Lemma lt_minus : forall n m, m <= n -> 0 < m -> n - m < n.
   Lemma lt_O_minus_lt : forall n m, 0 < n - m -> m < n.
   Theorem not_le_minus_0 : forall n m, ~ m <= n -> n - m = 0.
*)
```

Additional theorems about integers.

```
Theorem lt_to_plus: forall a b:nat, a<b = exists c:nat, 0<c & c+a=b.
Lemma mult_lt_le_compat : forall n m p q,
  0<q -> n < m -> p <= q -> n * p < m * q.
Lemma mult_le_lt_compat : forall n m p q,
  0< m -> n <= m -> p < q -> n * p < m * q.
Lemma zero_lt_oneN: 0 < 1.
Lemma lt_n_succ_leN: forall a b,  a < b -> S a <= b.
Lemma power_x_0N: forall a, a ^ 0 =  1.
Lemma power_0_0N:  0 ^ 0 = 1.
Lemma power_x_1N: forall a,  a ^ 1 = a.

Lemma plus_simplifiable_leftN: forall a b b':nat,
  a + b = a + b' -> b = b'.
Lemma plus_simplifiable_rightN: forall a b b':nat,
```

```
  b + a = b' + a -> b = b'.
Lemma Sn_is_1plus: forall n, S n = 1 + n.
Lemma Sn_is_plus1: forall n, S n = n +1.
Lemma lt_i_n : forall i n, i< n  -> 1 <= n-i.
Lemma double_subN: forall n p, p <= n -> n - (n- p) = p.
Lemma nonzero_suc: forall n, 0<>n -> exists m, n = S m.
Lemma mult_S_lt_reg_l : forall n m p, S n * m < S n * p -> m < p.
Lemma mult_lt_reg_l : forall n m p, 0<>n -> n * m < n * p -> m < p.
Lemma mult_lt_reg_r : forall n m p, 0<>n -> m * n <  p * n -> m < p.
Lemma minus_wrong: forall n m, n<=m -> n-m = 0.
Lemma pred_minus: forall n m, m<n -> n-m =  S(n - S m).

Lemma plus_n_Sm_subSn: forall n m, n + S m - n - 1 = m.
Lemma plus_n_Sm_subSm: forall n m, n + S m - m - 1 = n.

Lemma minus_SnSi: forall i n, i < S n -> S n - i - 1 = n - i.
Lemma double_compl_nat:forall i n, i< n ->
  i = n - (n - i - 1) -1.
Lemma double_compl_ex:forall i n, i< n -> (n - i - 1) < n.
Lemma plus_reg_r : forall n m p, n + p = m + p -> n = m.
```

14.3 The axiom of choice

Let E be any set, $p(x)$ be a property, $\mathrm{F} = \{x \in \mathrm{E}, p(x)\}$ and $z = \bigcup \mathrm{F}$. If there is a unique x in E that satisfies p, then $\mathrm{F} = \{x\}$, and $p(z)$ holds. This quantity z is denoted by `select p E`. Assume that p depends on a parameter y and $p(x)$ implies $x \in f(y)$. Then '`select (p y) (f y)`' is the same as '`choose p y`', and some calls to `choose` have been replaced by this trick.

Here are the old definitions. In the case of `supremum_pr1` we need the assumption that r is an order.

```
Definition the_least_element r := choose (least_element r).
Definition the_greatest_element r := choose (greatest_element r).
Definition supremum r X  := choose (least_upper_bound r X).
Definition infimum r X  := choose (greatest_lower_bound r X).


Lemma supremum_pr1 X r:
  has_supremum r X ->
  least_upper_bound r X (supremum r X).
Lemma infimum_pr1  X r:
  has_infimum r X ->
  greatest_lower_bound r X (infimum r X).
```

We introduced gge and ggt corresponding to ≥ and >, then removed them. Some statements had to be changed, and some lemmas became useless.

```
Definition gge r x y := gle r y x.
Definition ggt r x y := gle r y x &  x <> y.
Lemma opposite_gge r x y:
  gge (opposite_order r) x y <-> gle r x y.
Lemma glt_inva r x y:
  ggt (opposite_order r) x y <-> glt r x y.
```

```
Lemma ggt_inva r x y:
  glt (opposite_order r) x y <-> ggt r x y.
Lemma ggt_invb r x y:  ggt r x y <-> glt r y x.
```

These definitions have been simplified

```
Definition is_sup_fun r f x := least_upper_bound r (image_of_fun f) x.
Definition is_inf_fun r f x := greatest_lower_bound r (image_of_fun f) x.
Definition is_sup_graph r f x := least_upper_bound r (range f) x.
Definition is_inf_graph r f x := greatest_lower_bound r (range f) x.

Lemma infinite_nonempty x: infinite_c x -> nonempty x.
```

14.4 Theorems removed from Chapter 6

We study here the function `pow` on the type `nat`.

```
Lemma power_of_sumN: forall a b c, a ^ (b+c) = (a ^ b) *(a ^ c).
Lemma nat_B_pow: forall n m,
  nat_to_B (n ^ m) = card_pow (nat_to_B n)(nat_to_B m).
Lemma power_of_prodN: forall a b c,
  (a * b)  ^ c =  (a ^ c) *  (b ^ c).
Lemma power_1_xN: forall a,  1 ^ a = 1.
Lemma nat_not_zero_pr: forall a,  a <> 0 -> nat_to_B a <> card_zero.
Lemma power_0_x: forall a, a <> 0 -> 0 ^ a = 0.
Lemma non_zero_apowbN: forall a b, 0 < a -> 0 < a ^ b.
Lemma finite_power_lt1N: forall a a' b, a < a' -> 0 <  b ->  a ^ b < a' ^ b.
Lemma finite_power_lt2N: forall a b b',
  b < b' -> 1 < a ->  a^ b < a ^ b'.
Lemma mult_simplifiable_leftN: forall a b b':nat,
  0 <>a -> a * b = a * b' -> b = b'.
Lemma mult_simplifiable_rightN: forall a b b',
  0 <>a -> b * a = b' * a -> b = b'.
```

If a and b are integers and $a \le b$ there is a unique integer c such that $b = a + c$, it is called the *difference* and denoted by $b - a$. The operation is called *subtraction*. There is a COQ function, denoted by `sub`, defined for all integers, whose value is zero for $a > b$; we extend our function so that they share the same behavior.

```
Definition card_sub a b :=
   Yo (cardinal_le b a)
     (choose (fun c => inc c Bnat & card_plus b c = a)) card_zero.
Lemma card_sub_pr: forall a b, inc a Bnat-> inc b Bnat ->
  cardinal_le b a -> card_plus b (card_sub a b) = a.
Lemma card_sub_rpr: forall a b, inc a Bnat-> inc b Bnat ->
  cardinal_le b a -> card_plus (card_sub a b) b = a.

Lemma minus_n_nC: forall a, inc a Bnat -> card_sub a a = card_zero.
Lemma prec_pr1: forall a, inc a Bnat -> a <> card_zero
  -> predc a = card_sub a card_one.
Lemma sub_le_symmetry: forall  a b, inc a Bnat -> inc b Bnat ->
  cardinal_le b a -> cardinal_le (card_sub a b) a).
```

These two lemmas we initially use to show that every infinite set has a infinite countable subset.

```
Lemma morphism_range: forall f a b,
  order_morphism f a b -> equipotent (substrate a) (range (graph f)).
Lemma morphism_range1: forall f a b,
  order_morphism f a b -> cardinal (substrate a) = cardinal (range (graph f)).
```

Other removed theorems from chapter 6.

```
Lemma nat_B_sub: forall a b,
  nat_to_B (a-b) = card_sub (nat_to_B a) (nat_to_B b).
Lemma card_sub_pr4N: forall a b a' b',
  a<=b -> a' <= b' ->  (b-a) + (b'-a') = (b+b')- (a+a').
Lemma card_sub_associativeN: forall a b c,
  (b +c) <= a -> (a-b) -c = a - (b+c).
Lemma nat_B_pred: forall a, 0 <> a -> nat_to_B (pred a) = predc (nat_to_B a).
Lemma prec_is_cardinal_prec: forall a, inc a Bnat ->
  a <> card_zero -> cardinal_nat (prec a) = pred (cardinal_nat a).
Lemma card_sub_associative1N: forall a b,
  (S b) <= a -> pred (a - b) = a  - S b.
```

Division Given two integers a and $b \neq 0$, there is a unique pair of integers (q, r) satisfying $a = bq + r$ and $r < q$. Thus q and r are defined via the `choose` function like this.

```
Definition card_rem0 a b:=
  choose (fun r => inc r Bnat  & exists q, inc q Bnat & division_prop a b q r).
Definition card_quo0 a b:=
  choose (fun q => inc q Bnat & exists r, inc r Bnat & division_prop a b q r).
```

In order to simplify proofs, we define the quotient and remainder in the case $b = a$ as $q = 0$ and $r = a$. This means that $a = bq + r$ holds in any case.

```
Definition card_rem a b := variant \0c a (card_rem0 a b) b.
Definition card_quo a b := variant \0c \0c (card_quo0 a b) b.
```

Euclidean division is curiously defined in Coq. The following two lemmas say that if $b > 0$, then for every a we have $\{x : \mathbb{N} \mid \exists y : \mathbb{N},\ Z\}$ where Z is the division property $a = bq + r$ and $r < b$, and (x, y) is (q, r) or (r, q). This expression is a type; from it one can extract q and the associated property (namely that there is r such that Z).

```
(*
  Lemma quotient :
    forall n,
      n > 0 ->
      forall m:nat, {q : nat | exists r : nat, m = q * n + r /\ n > r}.
  Lemma modulo :
    forall n,
      n > 0 ->
      forall m:nat, {r : nat | exists q : nat, m = q * n + r /\ n > r}.
*)
```

The SSREFLECT library has a clever definition of quotient and remainder (the first definition defines quotient and remainder, the second defines only the quotient). The two functions `divn` and `modn` are deduced from these recursive function. We give one theorem that shows how these quantities can be used.

```
Definition edivn_rec d := fix loop (m q : nat) {struct m} :=
  if m - d is m'.+1 then loop m' q.+1 else (q, m).
Definition modn_rec d := fix loop (m : nat) :=
  if m - d is m'.+1 then loop m' else m.

Lemma divn_eq : forall m d, m = m %/ d * d + m %% d.
```

In the previous version of this document, we explained another implementation of these operations. It is not used anymore. These lemmas show existence and uniqueness of division.

```
Lemma least_int_prop0: forall p:nat->Prop,
  ~(p 0) -> (exists x, p x) -> (exists x, p (S x) & ~ p x).

Lemma division_prop_nat: forall a b q r, 0 <>b ->
  (a=b*q+r & r<b) = (b*q <= a & a < b* (S q) & r = a - (b*q)).
Lemma Ndivision_unique: forall a b q q' r r', 0 <> b ->
  a = b* q + r -> r < b -> a = b* q' + r' -> r'< b ->
  (q = q' & r = r').
Lemma Ndivision_existence: forall a b, 0 <> b ->
  exists q, exists r, (a = b* q + r & r < b).
Lemma division_result_integer: forall a b q r, inc a Bnat-> inc b Bnat ->
  b <> card_zero -> division_prop a b q r -> is_cardinal q ->
  (inc q Bnat & inc r Bnat).
```

In the definition that follows $b = 0$ is replaced by $b = 1$ so that the quotient and remainder is well-defined for all arguments. Later on, we modified the definition of quotient and remainder (see above). These two variants give different results. However, we say that b divides a only when b is non-zero.

```
Definition Nquo a b :=
  cardinal_nat (card_quo (Ro (nat_to_B a))
    (Yo (b = 0) card_one (Ro (nat_to_B b)))).
Definition Nrem a b :=
  cardinal_nat (card_rem (Ro (nat_to_B a))
    (Yo (b = 0) card_one (Ro (nat_to_B b)))).
Definition Ndivides b a:= 0 <> b & Nrem a b = 0.

Lemma Ndivision_exists: forall a b,  0 <> b ->
  (a = b* (Nquo a b) + (Nrem a b) & (Nrem a b < b)).
Lemma Ndivision_pr: forall a b q r, 0 <> b ->
  a = b* q + r -> r < b -> (q = Nquo a b & r = Nrem a b).

Lemma Ndivision_pr_q: forall a b q r, 0 <> b ->
  a = b* q + r -> r < b -> q = Nquo a b.
Lemma Ndivision_pr_r: forall a b q r, 0 <> b ->
  a = b* q + r -> r < b -> r = Nrem a b.
```

All properties true for `Nquo` and `Nrem` are true for `card_quo` and `card_rem`. For this reason, we shall only prove our theorems for the case of type `nat`.

```
Lemma nat_B_division: forall a b, 0 <> b ->
  (nat_to_B (Nquo a b) =  card_quo (nat_to_B a) (nat_to_B b) &
    nat_to_B (Nrem a b) =  card_rem (nat_to_B a) (nat_to_B b)).
Lemma nat_B_quo: forall a b, 0 <> b ->
```

```
  nat_to_B (Nquo a b) =  card_quo (nat_to_B a) (nat_to_B b).
Lemma nat_B_rem: forall a b, 0 <> b ->
  nat_to_B (Nrem a b) =  card_rem (nat_to_B a) (nat_to_B b).
```

Now some consequences when division is exact. Bourbaki says: every multiple a' of a multiple a of b is a multiple of b. One can restate this as: if b divides a, then b divides ac.

```
Lemma inc_quotient_bnat:
  forall a b, inc a Bnat-> inc b Bnat -> b <> card_zero ->
  inc (card_quo a b) Bnat.
Lemma inc_remainder_bnat:forall a b,
    inc a Bnat-> inc b Bnat -> b <> card_zero ->
    inc (card_rem a b) Bnat.

Lemma Ndivides_pr: forall a b,
  Ndivides b a -> a =  b * (Nquo a b).
Lemma Ndivides_pr1: forall a b,  0 <> b ->  Ndivides b (b *a).
Lemma Ndivides_pr2: forall a b q,  0 <> b ->
  a = b * q -> q = Nquo a b.
Lemma one_divides_all: forall a, Ndivides 1 a.
Lemma Ndivides_pr3: forall a b q,
  Ndivides b a -> q = Nquo a b -> a =  b * q.
Lemma Ndivides_pr4: forall b q, 0 <> b ->
  Nquo (b * q) b = q.
Lemma Ndivision_itself: forall a, 0 <> a ->
  (Ndivides a a & Nquo a a = 1).
Lemma Ndivides_itself: forall a, 0 <> a -> Ndivides a a.
Lemma Nquo: forall a, 0 <> a -> Nquo a a = 1.
Lemma Ndivision_of_zero: forall a, 0 <> a ->
  (Ndivides a 0 & Nquo 0 a = 0).
Lemma Ndivides_trans: forall a b a', Ndivides a a'-> Ndivides b  a
  -> Ndivides b a'.
Lemma Ndivides_trans1: forall a b a', Ndivides a a'-> Ndivides b a
  -> Nquo a' b =  (Nquo a' a) *(Nquo a b).
Lemma Ndivides_trans2: forall a b c,
  Ndivides b a-> Ndivides b (a *c).
Lemma non_zero_mult: forall a b, 0 <> a -> 0 <> b -> 0 <> (a*b).
Lemma Nquo_simplify:  forall a b c, 0 <> b -> 0 <> c ->
  Nquo (a * c) (b * c) = Nquo a b.
```

If b divides a and a', it divides the sum and the difference.

```
Lemma divides_and_sum:  forall a a' b,  Ndivides b a ->  Ndivides b a'
  -> (Ndivides b  (a + a') &
    Nquo  (a + a')  b =  (Nquo a b) + (Nquo a' b)).
Lemma distrib_prod2_subN: forall a b c, c<= b->
  a * (b-c) = (a*b) - (a*c).
Lemma divides_and_difference:  forall a a' b, a' <= a ->
  Ndivides b a ->  Ndivides b a'
  -> (Ndivides b  (a -a') &
    (Nquo a' b) <= (Nquo a b) &
    Nquo  (a - a')  b =  (Nquo a b) -  (Nquo a' b)).
```

The following lemma may have some interest, but is currently unused. Assume $a = bq + r$ with $r < b$, where a and b are integers, b is non-zero. The last relation says that r is an integer.

The quantity bq is also an integer, so that q is finite. If q is a cardinal, we deduce that q is an integer.

```
Lemma division_result_integer: forall a b q r,
  inc a Bnat-> inc b Bnat ->
  b <> card_zero -> division_prop a b q r -> is_cardinal q ->
  (inc q Bnat & inc r Bnat).

Lemma lt_a_power_b_aN: forall a b,  1< b -> a < pow b a.
```

14.5 Finite sequences and lists

If $\mathrm{P}\{i\}$ is equivalent to $i \in \mathrm{I}$, where I is a finite set of integers, then $(x_i)_{i\in\mathrm{I}}$ may be written as $(x_i)_{\mathrm{P}\{i\}}$. In fact, such a notation can be used whatever I. As an example one can see $(t_i)_{a\le i\le b}$. The sum of such a family may be denoted by $\sum\limits_{i=a}^{b} t_i$.

Lists are defined in Coq by

```
Inductive list (A : Type) : Type :=
    nil : list A
 | cons : A -> list A -> list A
```

A list can be either empty (this is `nil`), or of the form '`cons A a b`' where a is of type A and b is a list of type A. The parameter A is often implicit. The expression '`cons A a b`' is denoted by a::b. There are many functions in the standard library that deal with lists. For instance, `seq` can produce the list containing 1, 2, 3. Given a list containing x_1, x_2 and x_3 (of type A) it is possible to create the list containing $(1, x_1)$, $(2, x_2)$, and $(3, x_3)$ (of type $\mathbb{N}\times\mathrm{A}$) then the set of all these values. This is a finite sequence (i.e., a functional graph, with domain $\{1,2,3\}$). In this section, we explain how to convert operations defined by Bourbaki for finite sequences (like sum and product) into operations on Coq lists.

Lists as functions Given a function g, we define here the list L containing $g(0)$, $g(1)$, $g(2)$ up to $g(n-1)$. The list has length n; it is stored in natural order[1]: On the diagram below, the mapping $g \mapsto \mathrm{L}$ is denoted by `fl`. Conversely given a L of length n, we define a function g that returns the k-th element of the list, and 0 if $k \ge n$. It will be denoted by `lf` on the diagram below.

We consider a variant of `lf` where L is a list of sets (the default value is then $\emptyset$) and, later on, a variant of `fl`, where g is a function in the Bourbaki sense

```
Fixpoint fct_to_list_rev (A:Type) (f: nat->A)(n:nat): list A :=
  match n with O => nil
    | S m => (f m) ::(fct_to_list_rev f m) end.

Definition fct_to_list A f n := rev (fct_to_list_rev (A:=A) f n).

Definition list_to_fct (a: list nat) :=
  fun n => nth n a 0.
```

[1] In the previous version, we used the other order: $g(n-1)$ was the head of the list

```
Definition list_to_fctB (a: list Set) :=
  fun n => nth n a emptyset.
```

```
Lemma card_interval_c0_pr: forall n,
  cardinal_nat (interval_co_0a (nat_to_B n)) = n.
Lemma list_extens: forall (A:Type) (l1 l2 : list A) (u:A),
   length l1 = length l2 ->
   (forall i, i< length l1 -> nth i l1 u = nth i l2 u) -> l1 = l2.

Lemma fct_to_list_length :   forall A (f:nat->A) n,
  length (fct_to_list f n) = n.


Lemma list_to_fct_pr0: forall a l1 l2,
  list_to_fct (l2 ++ a :: l1) (length l2) = a.
Lemma list_to_fct_pr0B: forall a l1 l2,
  list_to_fctB (l2 ++ a :: l1)  (length l2) = a.
Lemma list_to_fct_pr: forall (A:Type) (f:nat->A) (u:A) n i,
  i<n -> nth i (fct_to_list f n) u = f i.
Lemma list_to_fct_pr1: forall f n i,
  i<n -> list_to_fct (fct_to_list f n) i = f i.
Lemma list_to_fct_pr1B: forall f n i,
  i<n -> list_to_fctB (fct_to_list f n) i = f i.
Lemma list_to_fct_pr3: forall l2 l1,
  fct_to_list (list_to_fct (l2++l1)) (length l2) = l2.
Lemma list_to_fct_pr4: forall l,
  fct_to_list (list_to_fct l) (length l) = l.
Lemma list_to_fct_pr3B: forall l2 l1,
  fct_to_list (list_to_fctB (l2++l1)) (length l2) = l2.
Lemma list_to_fct_pr4B: forall l,
  fct_to_list (list_to_fctB l) (length l) = l.
```

Note that if g and g' agree on $[0, n-1]$ then $\texttt{fl}(g) = \texttt{fl}(g')$. On the other hand, if L is a list of size n and $\mathrm{L}' = a$::L, if the associated functions are g and g', then $g'(n) = a$, and g and g' agree on $[0, n-1]$.

```
Lemma fct_to_list_unique: forall (A:Type) (f g: nat-> A) n,
  (forall i, i< n -> f i = g i) -> fct_to_list f n = fct_to_list g n.

Lemma app_nth3 : forall A (a:A),
  forall l' d n, n >= 1 -> nth n (a::l') d = nth (n-1) l' d.
```

Given a list L of elements of $\mathbb{N}$, of length n, if $g = \texttt{lf}(\mathrm{L})$, we construct a function $\mathrm{G} : [0, n[\to \mathbf{N}$ via $g(\mathrm{card}(i)) = \mathrm{card}(\mathrm{G}(i))$. The mapping $\mathrm{L} \mapsto \mathrm{G}$ will be denoted by LF on the diagram below. Similarly, given a list L of sets, we construct $\mathrm{G} : [0, n[\to \mathrm{E}$ via $g(\mathrm{card}(i)) = \mathrm{G}(i)$. This is well-defined if all elements of the list belong to the set E, see later.

```
Definition list_to_f (l: list nat):=
  BL (fun n => nat_to_B (list_to_fct l (cardinal_nat n)))
  (interval_co_0a (nat_to_B (length l))) Bnat.
Definition list_to_fB (l: list Set) E:=
  BL (fun n => list_to_fctB l (cardinal_nat n))
  (interval_co_0a  (nat_to_B (length l))) E.

Lemma list_to_f_axioms: forall (l: list nat),
```

```
  transf_axioms (fun n =>(Ro (nat_to_B (list_to_fct l (cardinal_nat n)))))
  (interval_co_0a (nat_to_B (length l))) Bnat.
Lemma list_to_f_function: forall (l: list nat),
  is_function (list_to_f l).
Lemma list_to_f_source: forall (l: list nat),
  source (list_to_f l) = (interval_co_0a (nat_to_B (length l))).
Lemma list_to_f_target: forall (l: list nat),
  target (list_to_f l) = Bnat.
Lemma list_to_f_W: forall (l: list nat) n,
  inc n (interval_co_0a (nat_to_B (length l))) ->
  W n (list_to_f l) = nat_to_B (list_to_fct l (cardinal_nat n)).
Lemma list_to_f_W1: forall (l: list nat) n,
  n < length l ->
  W (nat_to_B n) (list_to_f l) = nat_to_B (list_to_fct l n).
Lemma list_to_f_W2: forall (l: list nat) n,
  n < length l ->
  cardinal_nat(W (nat_to_B n) (list_to_f l)) = list_to_fct l n.
```

(Finite Lists)

$$\begin{tikzcd}
\text{Lists} \arrow[dd, bend right, "lf"'] \arrow[dr, "\mathrm{LF}"] & \\
& \mathscr{F}(\mathrm{I},\mathbf{N}) \arrow[dl, "\mathrm{BN}"] \\
\mathbb{N}\to\mathbb{N} \arrow[uu, bend right, "fl"'] &
\end{tikzcd}$$

Given a function $[0,n[\to \mathbf{N}$, we can construct a function $\mathbb{N}\to\mathbb{N}$, by extending the function with zero, and using the natural isomorphism between $\mathbb{N}$ and $\mathbf{N}$. It will be denoted by `BN` on the diagram. The composition `fl` $\circ$ `BN` is the inverse of LF.

```
Definition back_to_nat f n:=
  cardinal_nat (Yo (inc (nat_to_B n) (source f))
    (W (nat_to_B n) f) card_zero).

Lemma back_to_nat_pr: forall f n, inc  (nat_to_B n) (source f) ->
  back_to_nat f n =  cardinal_nat (W  (nat_to_B n) f).
Lemma back_to_nat_pr1: forall f n k,
  source f = (interval_co_0a (nat_to_B k)) ->
  n < k -> back_to_nat f n =  cardinal_nat (W (nat_to_B n) f).
Lemma back_to_nat_pr2: forall (l: list nat) n,
  n < (length l) -> back_to_nat (list_to_f l) n = list_to_fct l n.

Lemma list_to_f_pr1: forall f n, is_function f -> target f = Bnat ->
  source f = (interval_co_0a (nat_to_B n)) ->
  f = list_to_f (fct_to_list (back_to_nat f) n).
Lemma list_to_f_pr2: forall l,
  fct_to_list (back_to_nat (list_to_f l)) (length l) = l.
```

Given a list L and a predicate P, we define P(L) to be true if every element of the list satisfies P. Given a predicate with two arguments, we say that the list satisfies the predicate whenever $\mathrm{P}(a,b)$ is true if a comes before b. This means that, if f is the function associated to the list, then $i<j$ implies $\mathrm{P}(f(i),f(j))$.

```
(*
```

```
  Fixpoint single_list_prop (A:Type) (L: list A) (q: A->Prop) :=
    match L with nil => True | a :: b => q a /\ single_list_prop b q end.
  Fixpoint double_list_prop (A:Type) (L: list A) (q: A->A->Prop) :=
    match L with nil => True
        | a :: b => single_list_prop b (q a) /\ double_list_prop b q
  end.
*)
```

These definitions changed in Version 2. The two-arguments case was unused, and removed; the single-argument case has been replaced by an inductive object.

```
Inductive list_prop (A:Type)  (q: A->Prop) : list A -> Prop :=
  | list_prop_nil: list_prop  q nil
  | list_prop_cons: forall (a:A)(l:list A),
    q a -> list_prop q l -> list_prop q (a::l).

Lemma list_prop1: forall A (q: A->Prop),  list_prop q nil.
Lemma list_prop2: forall A a b (q: A->Prop),
  q a -> (list_prop q b) = (list_prop q (a::b)).
Lemma list_prop3: forall A a b (q: A->Prop),
  ~ (q a) -> ~(list_prop q (a::b)).
Lemma list_prop_app: forall A a b (q: A->Prop),
  (list_prop q a) -> (list_prop q b)
  -> (list_prop q (a++b)).
Lemma list_prop_refine: forall A L (p q: A->Prop),
  (forall a, p a -> q a) ->  list_prop p L ->  list_prop q L.
Lemma list_prop_nth: forall  A (q: A->Prop) L u n,
  list_prop q L -> n < length L ->
  q (nth n L u).
```

The contraction $\mathrm{C}_{fv}(\mathrm{L})$ of a list L is inductively defined by $\mathrm{C}_{fv}(a::\mathrm{L}) = f(a,\mathrm{C}_{fv}(\mathrm{L}))$, the value of the empty list being v. If $f(a,b) = b\cup\{a\}$, we call this the *range* of the list and denote it by $r(\mathrm{L})$. We write $\mathrm{L}\subset\mathrm{E}$ if $\mathrm{P_E}(\mathrm{L})$ holds, where $\mathrm{P_E}(x)$ is $x\in\mathrm{E}$; this means that every element of the list belongs to E.

```
Fixpoint contraction (A B: Type) (L: list A) (f: A-> B->B) (v: B):B :=
  match L with | nil => v
    | a :: b => f a (contraction b f v) end.

Definition list_range l := contraction l (fun a b => tack_on b a) emptyset.
Definition list_subset L E := list_prop (fun x => inc x E) L.
```

The range of a list is the smallest set E such that $\mathrm{L}\subset\mathrm{E}$. We show $\mathrm{L}\subset r(\mathrm{L})$ by induction. We have $r(\mathrm{L})\subset r(a::\mathrm{L})$. We then use the fact that if P implies Q, then P(L) implies Q(L), where P is $x\in r(\mathrm{L})$ and Q is $x\in r(a::\mathrm{L})$. We can now state: if $\mathrm{L}\subset\mathrm{E}$ is a list of length n, there is an associated function $[0,n[\to\mathrm{E}$.

```
Lemma list_range_pr: forall L, list_subset L (list_range L).
Lemma list_range_pr1: forall L E, list_subset L E -> sub (list_range L) E.

Lemma list_to_fB_axioms: forall l E, list_subset l E ->
  transf_axioms (fun n => (list_to_fctB l (cardinal_nat n)))
  (interval_co_0a  (nat_to_B (length l))) E.
Lemma list_to_fB_function:  forall l E, list_subset l E ->
```

```
  is_function (list_to_fB l E).
Lemma list_to_fB_W: forall l E n, list_subset l E ->
  inc n (interval_co_0a (nat_to_B (length l))) ->
  W n (list_to_fB l E) = list_to_fctB l (cardinal_nat n).
Lemma list_to_fB_W1: forall  l E n, list_subset l E ->
  n < length l ->
  W (nat_to_B n) (list_to_fB l E) = list_to_fctB l n.
Lemma fct_to_rev: forall (A:Type) (f:nat->A) n,
  rev (fct_to_list f n) =  fct_to_list(fun i=> f(n-i-1)) n.
```

More properties of intervals.

```
Lemma partition_tack_on_intco: forall a, inc a Bnat ->
  partition_fam (variantLc (interval_co_0a a)
    (singleton a)) (interval_co_0a (succ a)).
Lemma interval_co_0a_restr: forall a f, inc a Bnat ->
  (L (interval_co Bnat_order card_zero a) f
    = (restr (L (interval_co Bnat_order card_zero (succ a)) f)
      (interval_co_0a a))).
```

Let L_1 and L_2 be two lists, $L = L_1 ++ a::L_2$. Let G_1 and G be the functions associated to L_1 and L. If L_1 is a list of size n, then $G(n) = a$, and G and G_1 agree on $[0, n-1]$. In fact, G_1 is the restriction of G to the interval $[0, n[$, and if L_2 is empty, then G is the function obtained from G_1 by adding the relation $G(n) = a$.

```
Lemma length_app1: forall (A:Type) (a:A) l l',
  length l < length (l ++ a :: l').
Lemma length_app2:  forall (A:Type) (a:A) l ,
  nat_to_B (length (l++a::nil)) = succ (nat_to_B (length l)).

Lemma list_to_f_cons0: forall a l l',
  W (nat_to_B  (length l)) (list_to_f (l++ a:: l')) = nat_to_B a.
Lemma list_to_f_cons1: forall a l l' n, n < length l ->
  W (nat_to_B n) (list_to_f (l ++ a :: l')) =  W (nat_to_B n) (list_to_f l).
Lemma list_to_f_cons2: forall a l l',
  list_to_f l = restriction (list_to_f (l++ a :: l'))
             (interval_co_0a (nat_to_B (length l))).
Lemma list_to_f_cons3: forall a l,
  list_to_f (l++a::nil) = tack_on_f (list_to_f l)
  (nat_to_B (length l))  (nat_to_B a).

Lemma list_subset_cons: forall a l l' E,
  list_subset l E -> inc a E -> list_subset l' E ->
  list_subset (l'++a::l) E.
Lemma list_to_f_consB0: forall a l l' E,
  list_subset l E -> inc a E -> list_subset l' E ->
  W (nat_to_B  (length l)) (list_to_fB (l++ a:: l') E) = a.
Lemma list_to_f_consB1: forall a l l' n E, n< length l ->
  list_subset l E -> inc a E ->  list_subset l' E ->
  W (nat_to_B n) (list_to_fB(l++ a :: l') E) = W (nat_to_B n) (list_to_fB l E).

Lemma list_to_f_consB0: forall a l E, list_subset l E -> inc a E ->
  W (nat_to_B  (length l)) (list_to_fB (a :: l) E) = a.
Lemma list_to_f_consB1: forall a l n E, n< length l ->
  list_subset l E -> inc a E ->
```

```
  W (nat_to_B n) (list_to_fB (a :: l) E) =  W (nat_to_B n) (list_to_fB l E).
Lemma list_to_f_consB2: forall a l l' E,
  list_subset l E -> inc a E ->  list_subset l' E ->
  list_to_fB l E = restriction (list_to_fB (l++ a :: l') E)
             (interval_co_0a (nat_to_B (length l))).
Lemma list_to_f_consB3: forall a l E,
  list_subset l E -> inc a E ->
  list_to_fB (l++a::nil) E
  = tack_on_f  (list_to_fB l E) (nat_to_B (length l))  a.
```

We denote by LFB the variant of `lf` that converts a list $\mathrm{L} \subset \mathrm{E}$ into a function $\mathrm{I} \to \mathrm{E}$. The source of this function is an interval $[0, n[$; we shall call this an `iid` function. We denote by FLB the variant of `fl` which is the inverse of LFB, i.e. $\mathrm{LFB}_\mathrm{E}(\mathrm{FLB}(f)) = f$ and $\mathrm{FLB}(\mathrm{LFB}_\mathrm{E}(\mathrm{L})) = \mathrm{L}$, whenever f is a function whose source is $[0, n[$ and is target is E, and whenever $\mathrm{L} \subset \mathrm{E}$.

```
Definition fct_to_listB1 f n:=
  fct_to_list (fun n => W (nat_to_B n) f) n.
Definition fct_to_listB f := fct_to_listB1 f (cardinal_nat (source f)).
Definition iid_function f :=
  is_function f & exists n, source f = interval_co_0a (nat_to_B n).

Lemma list_to_fB_pr:  forall l E, list_subset l E ->
  iid_function (list_to_fB l E).
Lemma fct_to_list_lengthB :  forall f,  iid_function f ->
  nat_to_B (length (fct_to_listB f)) = cardinal (source f).
Lemma fct_to_listB_pr0: forall f i,
  iid_function f -> i< cardinal_nat (source f) ->
  list_to_fctB (fct_to_listB f) i = W (nat_to_B i) f.

Lemma fct_to_listB_pr1: forall l E, list_subset l E ->
  fct_to_listB(list_to_fB l E) = l.
Lemma fct_to_listB_pr2: forall f, iid_function f ->
   list_subset (fct_to_listB f) (target f).
Lemma fct_to_listB_pr3: forall f, iid_function f ->
  list_to_fB  (fct_to_listB f) (target f) = f.
```

Contracting lists We show here the following. Assume that $f(n)$ is a cardinal for all n. Let $\mathrm{F}(n)$ be the cardinal sum of the family $i \mapsto f(i)$ on $[0, n-1]$. Then $\mathrm{F}(n+1) = f(n) + \mathrm{F}(n)$, and there is a similar relation for the product. The same formula holds if $\mathrm{F}(n+1)$ is the cardinal sum of the graph of the function f and $\mathrm{F}(n)$ is cardinal sum of the graph of the restriction of f to $[0, n-1]$. We apply this to the case where f is `LF(L++a::nil)`; its restriction is `LF(L)`, and $f(n) = a$.

```
Lemma induction_on_sum: forall a f,  inc a Bnat ->
  (forall a, inc a Bnat -> is_cardinal (f a)) ->
  let iter := fun n=> cardinal_sum (L (interval_co_0a n)f)
    in card_plus (iter a) (f a) = (iter (succ a)).
Lemma induction_on_prod: forall a f,  inc a Bnat ->
  (forall a, inc a Bnat -> is_cardinal (f a)) ->
  let iter := fun n=> cardinal_prod (L (interval_co_0a n ) f)
    in card_mult (iter a) (f a) = (iter (succ a)).
Lemma induction_on_sum1: forall f n,
  is_function f -> source f = interval_co_0a (succ n) -> inc n Bnat ->
  (forall a, inc a (source f) -> is_cardinal (W a f)) ->
```

```
  card_plus (cardinal_sum (graph (restriction f (interval_co_0a n))))
  (W n f) = cardinal_sum (graph f).
Lemma induction_on_prod1: forall f n,
  is_function f -> source f = interval_co_0a (succ n) -> inc n Bnat ->
  (forall a, inc a (source f) -> is_cardinal (W a f)) ->
  card_mult (cardinal_prod (graph (restriction f (interval_co_0a n))))
  (W n f) = cardinal_prod (graph f).
```

Denote by $\mathrm{S}(\mathrm{L})$ the cardinal sum of the family $\mathrm{LF}(\mathrm{L})$. The induction principle says $\mathrm{S}(\mathrm{L}{+}{+}a{::}\texttt{nil}) = \mathrm{S}(\mathrm{L}) + a$. By associativity we get $\mathrm{S}(\mathrm{L}' + +\mathrm{L}) = \mathrm{S}(\mathrm{L}') + \mathrm{S}(\mathrm{L})$. We have $\mathrm{S}(\texttt{nil}) = 0$ and $\mathrm{S}(a{::}\texttt{nil}) = a$. If we take $\mathrm{L}' = a{::}\texttt{nil}$, the associativity formula gives $\mathrm{S}(a{::}\mathrm{L}) = a + \mathrm{S}(\mathrm{L})$. **Note**: in version 2, we changed the ordering of the elements of the list. This changes the properties of S; we proved the previous formula by using commutativity (rather than associativity).

```
Lemma induction_on_sum2: forall a l,
  card_plus (cardinal_sum (graph (list_to_f l))) (nat_to_B a)
  = cardinal_sum (graph (list_to_f (l++a::nil))).
Lemma induction_on_prod2: forall a l,
  card_mult (cardinal_prod (graph (list_to_f l))) (nat_to_B a)
  = cardinal_prod (graph (list_to_f (l++a::nil))).

Lemma induction_on_sum0:
  cardinal_sum (graph (list_to_f nil)) = card_zero.
Lemma induction_on_sum5: forall a,
  cardinal_sum (graph (list_to_f (a::nil))) = nat_to_B a.
Lemma induction_on_prod0:
  cardinal_prod (graph (list_to_f nil)) = card_one.
Lemma induction_on_prod5: forall a,
  cardinal_prod (graph (list_to_f (a::nil))) = nat_to_B a.

Lemma induction_on_sum4: forall l l',
  card_plus (cardinal_sum (graph (list_to_f l)))
  (cardinal_sum (graph (list_to_f l')))
  = cardinal_sum (graph (list_to_f (l++l'))).
Lemma induction_on_prod4: forall l l',
  card_mult (cardinal_prod (graph (list_to_f l)))
  (cardinal_prod (graph (list_to_f l')))
  = cardinal_prod (graph (list_to_f (l++l'))).
```

We define here the sum and product of a list of integers as a contraction. We shall denote this by $\Sigma(\mathrm{L})$ and $\Pi(\mathrm{L})$. This operation is related to the previous one by $\mathscr{N}(\Sigma(\mathrm{L})) = \sum \mathrm{LF}(\mathrm{L})$ and $\mathscr{N}(\Pi(\mathrm{L})) = \prod \mathrm{LF}(\mathrm{L})$.

```
Definition list_sum l := contraction (rev l) plus 0.
Definition list_prod l := contraction (rev l) mult 1.

Lemma list_sum_pr: forall l,
  nat_to_B (list_sum l) = cardinal_sum (graph (list_to_f l)).
Lemma list_prod_pr: forall l,
  nat_to_B (list_prod l) = cardinal_prod (graph (list_to_f l)).
```

If we denote by $a{+}{+}b$ the concatenation of two lists, then $\mathrm{C}_{fv}(a{+}{+}b) = f(\mathrm{C}_{fv}(a), \mathrm{C}_{fv}(b))$ provided that the result is true for the empty list, i.e., $f(v, b) = b$ for all b, and if f is associative. As a consequence $\Sigma(a{+}{+}b) = \Sigma(a) + \Sigma(b)$ and $\Pi(a{+}{+}b) = \Pi(a)\cdot\Pi(b)$. This is a general property of contractions of an associative function f

Denote by $\Sigma'_n(f)$ the expression $\Sigma(\mathtt{fl}(f,n))$. This is the sum of the list of the values $f(i)$ for $i<n$. If we denote by f_1+f_2 the function $i\mapsto f_1(i)+f_2(i)$ then we have $\Sigma'_n f+\Sigma'_n g=\Sigma'_n(f+g)$. There are similar formulas for the product. We have two induction formulas; the trivial one is $\Sigma'_{n+1}(f)=f(n)+\Sigma'_n(f)$; the non-trivial one is $\Sigma'_{n+1}(f)=f(0)+\Sigma'_n(f\circ \mathrm{S})$, where $(f\circ \mathrm{S})(i)=f(i+1)$.

```
Lemma contraction_assoc: forall  (A :Type) (L1 L2: list A)
  (f: A-> A->A) (v: A),
  (forall a b c, f a (f b c) = f (f a b) c) ->
  (forall a, f v a = a) ->
  (contraction (L1++L2) f v) = f (contraction L1 f v)(contraction L2 f v).
```

```
Lemma list_sum_single: forall a, list_sum (a::nil) = a.
Lemma list_prod_single: forall a, list_prod (a::nil) = a.
Lemma list_sum_app: forall a b, list_sum (a++b) = (list_sum a)+ (list_sum b).
Lemma list_sum_cons: forall a b, list_sum (a::b) = a + (list_sum b).
Lemma list_sum_consr: forall a b, list_sum (a++(b::nil)) = (list_sum a)+ b.
Lemma list_prod_app: forall a b,
  list_prod (a++b) = (list_prod a)* (list_prod b) .
Lemma list_prod_cons: forall a b, list_prod (a::b) = a*(list_prod b).
Lemma list_prod_consr: forall a b, list_prod (a++(b::nil)) = (list_prod a)* b.
```

```
Definition fct_sum f n:= list_sum (fct_to_list f n).
Definition fct_prod f n:= list_prod(fct_to_list f n).

Lemma fct_sum0: forall f, fct_sum f 0 = 0.
Lemma fct_prod0: forall f, fct_prod f 0 = 1.
Lemma fct_sum_rec: forall f n, fct_sum f (S n) = (fct_sum f n) + (f n).
Lemma fct_prod_rec: forall f n, fct_prod f (S n) = (fct_prod f n) * (f n).
Lemma fct_sum_rec1: forall f n,
  fct_sum f (S n) = (f 0) + (fct_sum (fun i=> f (S i)) n).
Lemma fct_prod_rec1: forall f n,
  fct_prod f (S n) = (f 0) * (fct_prod (fun i=> f (S i)) n).
Lemma fct_sum_plus: forall f g n,
  (fct_sum f n) +  (fct_sum g n) = fct_sum (fun i=> (f i) + (g i)) n.
Lemma fct_prod_mult: forall f g n,
  (fct_prod f n) *  (fct_prod  g n) =fct_prod (fun i=> (f i) * (g i)) n.
```

We show here some trivial results. The sum of a constant function is the product, and the sum is unchanged if we replace the list by its reverse. A bit more complicated: the reverse of the list associated to a function f is the list associated to $i\mapsto f(n-i-1)$.

```
Lemma fct_sum_const: forall n m, fct_sum (fun _ => m) n = n *m.
Lemma fct_prod_const: forall n m, fct_prod (fun _ => m) n = pow m n.
Lemma list_sum_rev: forall l, list_sum l = list_sum (rev l).
Lemma list_prod_rev: forall l, list_prod l = list_prod (rev l).
Lemma fct_sum_rev: forall f n,
  fct_sum f n =  fct_sum (fun i=> f(n-i-1)) n.
Lemma fct_prod_rev: forall f n,
  fct_prod f n =  fct_prod (fun i=> f(n-i-1)) n.
Lemma fct_to_rev: forall (A:Type) (f:nat->A) n,
  rev (fct_to_list f n) =  fct_to_list(fun i=> f(n-i-1)) n.
```

We consider here the inverse of `BN`: if f is a function of type `nat`→`nat`, we construct a function $[0,n[\to \mathbf{N}$. It is the composition of `fl` and `LF`. We shall denote it by `NB`. The first theorem is a statement about `NB`, the two others are statements about the graph of `NB`. The third theorem is deduced from the second by applying $[0,n+1[=[0,n]$.

```
Lemma l_to_fct: forall f n,
  BL (fun p => nat_to_B(f (cardinal_nat p))) (interval_co_0a (nat_to_B n))
  Bnat = list_to_f (fct_to_list f n).
Lemma l_to_fct1: forall f n,
  L (interval_co_0a (nat_to_B n)) (fun p => nat_to_B(f (cardinal_nat p)))
  = graph (list_to_f (fct_to_list f n)).
Lemma l_to_fct2: forall f n,
  L (interval_Bnat card_zero (nat_to_B n))
  (fun p => nat_to_B(f (cardinal_nat p)))
  = graph (list_to_f (fct_to_list f (S n))).
```

Iterated functions Note: all useful results of this section have been moved to section 14.5. The remaining trivial results are given without comment.

```
Definition function_on_nat f :=
  fun m =>nat_to_B (f (cardinal_nat m)).

Lemma inc_function_on_nat_Bnat : forall f n,
  inc (function_on_nat f n) Bnat.
Lemma function_on_nat_pr : forall f n,
  cardinal_nat(function_on_nat f n) = f (cardinal_nat n).
Lemma function_on_nat_pr1 : forall f n,
  function_on_nat f (nat_to_B n) = nat_to_B (f n).
```

Factorial In a previous version, we defined the factorial function as shown below. This definition is equivalent to the one provided by *ssrnat.v*.

```
Fixpoint factorial (n:nat) : nat :=
  match n with
  | 0 => 1
  | S p => (factorial p) *  S p
  end.

Lemma factorial0: factorial 0 = 1.
Lemma factorial1: factorial 1 = 1.
Lemma factorial2: factorial 2 = 2.

Lemma factorial_succ: forall n,  factorial (S n) = (factorial n) * (S n).
Lemma factorial_nonzero: forall n, 0 <> factorial n.

Lemma factorial_prop: forall f, f 0 = 1 ->
  (forall n, f (S n) =  (f n) * (S n)) ->
   forall x, f x = factorial x.
Lemma factorial_prop1: forall n, factorial n = fct_prod S n.
```

We prove here: if $\mathrm{J}\subset\mathrm{I}$, the product $\prod f_i$ restricted to J divides the product $\prod f_i$ restricted to I. We used this result to show that $n!$ divides $m!$. This requires 44 lines of proof, but proving by induction on c that $b!$ divides $(b+c)!$ requires only 3 lines (for the case of `nat`, ten lines in the case of `Bnat`).

```
Lemma divides_restriction_product: forall f x, fgraph f ->
     (forall i, inc i (domain f) -> is_finite_c (V i f)) ->
     (forall i, inc i (domain f) -> (V i f) <> card_zero) ->
     is_finite_set (domain f) -> sub x (domain f) ->
     BNdivides (cardinal_prod (restr f x)) (cardinal_prod f).

Lemma quotient_of_factorials: forall a b, b <= a ->
  Ndivides (factorial b) (factorial a).
Lemma quotient_of_factorials1: forall a b, b <= a ->
  Ndivides (factorial (a - b)) (factorial a).
Lemma tack_on_nat: forall a b, is_finite_set (tack_on a b) ->
  ~ (inc b a) -> cardinal_nat (tack_on a b) = S (cardinal_nat a).
```

The binomial coefficient In the previous version, the binomial function was defined by induced as follows.

```
Fixpoint binom (n p:nat) {struct n} : nat :=
  match n, p with
    | 0, 0 => 1
    | 0, S m => 0
    | S q, 0 => 1
    | S q, S m =>  (binom q (S m)) + (binom q m)
end.
```

Definition by transfinite induction This is the original definition of ordinal exponentiation.

```
Definition  ord_induction_sup (g: fterm2) x y f :=
  \osup (fun_image (ordinal_interval \1o y) (fun z => g (f z) x)).

Definition ord_induction_p  (w0 w1: fterm) g x f :=
  (Yo (source f = \0o) (w0 x)
    (Yo (source f = \1o) (w1 x)
      (ord_induction_sup g x (source f) (Vf f)))).

Definition ord_induction_aux w0 w1 g x a :=
  transfinite_defined (ordinal_o a) (ord_induction_p w0 w1 g x).

Definition ord_induction_defined w0 w1 g :=
  fun x y => Vf (ord_induction_aux w0 w1 g x (succ_o y)) y.

Definition ord_pow' := ord_induction_defined (fun z:Set => \1o) id ord_prod2.
Definition ord_pow a b :=
  Yo (a = \0o)
     (Yo (b = \0o) \1o \0o)
     (Yo (a = \1o) \1o (ord_pow' a b)).
```

Initial Ordinals This is the definition of an initial ordinal, as proposed by Bourbaki in the exercises.

```
Definition ordinals_card_le y :=
   Zo (\2c ^c y) (fun z => (cardinal z) <= c y).
Definition aleph_aux1 f x :=
```

```
  union (fun_image x (fun z => (ordinals_card_le (cardinal (f z))))).
Definition aleph_aux2 b :=
  transfinite_defined (ordinal_o (succ_o b))
  (fun f => Yo (source f = \0o) \omega
      (\osup (aleph_aux1 (Vf f) (source f)))).
Definition omega_fct x := Vf (aleph_aux2 x) x.
```

This is the definition of the inverse of $x \mapsto \aleph_x$ and the cardinal successor.

```
Definition ord_index x :=
  intersection (Zo (succ_o x) (fun z => \aleph z =x)).
Definition succ_c x:= omega_fct (succ_o (ord_index x)).
```

14.6 Removed theorems

The lemmas and definition shown here existed in previous version, but have been withdrawn.

A correspondence $\Gamma = (G, E, E)$, whose graph G is an order on E, is also called an order by Bourbaki (this definition is in fact never used).

```
Definition order_c r :=
  is_correspondence r & source r = target r & source r = substrate (graph r)
  & order (graph r).
Theorem order_cor_pr: forall f,
  is_correspondence f ->
  order_c f =
  (source f = target f & source f = (domain (graph f)) &
    compose_graph (graph f)(graph f) = graph f &
    intersection2 (graph f) (opposite_order (graph f))
    = diagonal (substrate (graph f))).
```

Here is is the original definition of a product order. One can notice that f is uniquely defined by g. This argument has been removed in the new version.

```
Definition product_order_r (f g:Set): EEP :=
  fun x x' =>
   inc x (productb f) & inc x' (productb f) &
   forall i, inc i (domain f) -> gle (V i g) (V i x)(V i x').

Definition product_order f g:=
  graph_on (product_order_r f g)(productb f).


Definition axioms_product_order f g:=
  fgraph f & fgraph g & domain f = domain g &
  (forall i, inc i (domain f) -> order (V i g)) &
  (forall i, inc i (domain f) -> substrate (V i g) = V i f).


Lemma order_product_order: forall f g,
  axioms_order_product f g -> order (product_order f g).
Lemma related_product_order: forall f g x x',
  axioms_product_order f g ->
```

```
  related(product_order f g) x x' =
  (inc x (productb f) & inc x' (productb f) &
  forall i, inc i (domain f) -> related (V i g) (V i x)(V i x')).
Lemma substrate_product_order: forall f g,
  axioms_product_order f g -> substrate(product_order f g) = productb f.
Lemma product_order_def: forall f g,  axioms_product_order f g ->
  image_by_fun (prod_of_products_canon f f)(product_order f g)
   = (productb g).  (* 36 *)
```

These are the original definitions of the lexicographic order.

```
Definition lexicographic_order_r (r f g:Set): EEP :=
  fun x x' =>
    inc x (productb f) & inc x' (productb f) &
    forall j, least_element (induced_order r (Zo (domain f)
      (fun i =>  V i x <> V i x'))) j -> glt (V j g) (V j x)(V j x').

Definition lexicographic_order_axioms r f g:=
  worder r & substrate r = domain f &
  fgraph f & fgraph g & domain f = domain g &
  (forall i, inc i (domain f) -> order (V i g)) &
  (forall i, inc i (domain f) -> substrate (V i g) = V i f).


Definition graph_order_r(x y g:Set): EEP :=
  fun z z' =>
    inc z (set_of_gfunctions x y) & inc z' (set_of_gfunctions x y) &
    forall i, inc i x-> related g (V i z)(V i z').
Definition graph_order x y g :=
  graph_on(graph_order_r x y g) (set_of_gfunctions x y).
Definition function_order x y r :=
  graph_on(fun u v => function_order_r x y r (sof_value x y u)
        (sof_value x y v))
  (set_of_functions x y).
```

Given two sets A and B, two distinct elements α and β, if I is the set that contains α and β, there is a family $(X_\iota)_{\iota\in I}$ such that such that $A = X_\alpha$ and $B = X_\beta$. This family is `Lvariant`. We shall denote it by $X_{\alpha\beta}(A,B)$. We show here uniqueness of the family.

```
Lemma two_terms_bij1: forall a b x y f,
  y <> x -> fgraph f -> domain f = doubleton x y -> V x f = a -> V y f = b ->
  range f  =  doubleton a b -> f = Lvariant x y a b.
```

Here are some trivial lemmas.

```
Lemma source_pfs:forall y x,
  source (partition_fun_of_set y x) = y.
Lemma target_pfs: forall y x,
  target (partition_fun_of_set y x) = powerset x.
Lemma source_graph_of_function: forall x y,
  source (graph_of_function x y) = set_of_sub_functions x y.
Lemma target_graph_of_function: forall x y,
  target (graph_of_function x y) =   (set_of_graphs x y).
Lemma sup_interval_co_0a: forall n, inc n Bnat ->
  supremum Bnat_order (interval_co_0a (succ n)) = n.
```

Other lemmas The first lemma here is obvious. The second is unused.

```
Lemma cardinal_two_is_doubleton: exists x, exists x',
  x <> x' &  \2c = doubleton  x x'.
Lemma cardinal_equipotent1 x y: is_cardinal x -> is_cardinal y ->
  x \Eq y -> x = y.
Lemma card_lt_succ_le1 a b: inc b Bnat ->
  a <=c (succ b) -> a <> (succ b) -> a <=c b.
```

Definition of a function by induction We explain here the initial implementation of section 6.6, more precisely the case when a function f is defined by (IND0), i.e., $f(0) = a$ and $f(n+1) = h(n, f(n))$ for $n \in \mathbf{N}$, or variants of this formulation.

In Version 1 we had the following two definitions (compare with `induction_defined0_set` and `induction_defined1_set`). They are of the form `choose IND0` and `choose IND1'`. We have two theorems saying that these objects satisfy (IND0) and (IND1') respectively, and two others stating existence and uniqueness of (IND0), and existence of (IND1'). Together with these four theorems, we show a variant of `integer_induction_stable` and the Bourbaki variant of (IND0).

```
Definition induction_defined1 E h a:= choosef(fun f=>
  is_function f & source f = Bnat & target f = E & W card_zero f = a &
  forall n,  inc n Bnat -> W (succ n) f = h n (W n f)).
Definition induction_defined2 E h a p:= choosef(fun f=>
  is_function f & source f = Bnat & target f = E & W card_zero f = a &
  forall n,  cardinal_lt n p -> W (succ n) f = h n (W n f)).

Lemma integer_induction_stable: forall E g  a,
  inc a E -> is_function g -> source g = E -> target g = E ->
  sub (target (induction_defined g a)) E.
Lemma induction_with_var: forall E h a,
  is_function h -> source h = product Bnat E -> target h = E -> inc a E ->
  exists_unique (fun f=>  is_function f & source f = Bnat & target f= E &
    W card_zero f = a
    &  forall n, inc n Bnat -> W (succ n) f = W (J n (W n f)) h).


Lemma induction_with_var1: forall E h a,
  (forall n x, inc n Bnat -> inc x E -> inc (h n x) E) -> inc a E ->
  exists_unique (fun f=>  is_function f & source f = Bnat & target f= E &
    W card_zero f = a
    &  forall n, inc n Bnat -> W (succ n) f = h n (W n f)).

Lemma induction_with_var2: forall E h a p,
  (forall n x, inc n Bnat -> inc x E -> cardinal_lt n p -> inc (h n x) E)
  -> inc a E -> inc p Bnat ->
  exists f,  is_function f & source f = Bnat & target f= E &
    W card_zero f = a
    &  forall n, cardinal_lt n p -> W (succ n) f = h n (W n f).
Lemma induction_defined_pr2: forall E h a p,
  (forall n x, inc n Bnat -> inc x E -> cardinal_lt n p -> inc (h n x) E)
  -> inc a E -> inc p Bnat ->
  let f :=  induction_defined2 E h a p in is_function f &
    source f = Bnat &  target f = E & W card_zero f = a &
    forall n,  cardinal_lt n p -> W (succ n) f = h n (W n f).
```

```
Lemma induction_defined_pr1: forall E h a,
  (forall n x, inc n Bnat -> inc x E -> inc (h n x) E)
  -> inc a E ->
  let f :=  induction_defined1 E h a in is_function f &
    source f = Bnat &  target f = E & W card_zero f = a &
    forall n,  inc n Bnat -> W (succ n) f = h n (W n f).
```

The current definition does not use the `choose` function anymore.

```
Definition induction_defined0 h a := choose(fun f=>
  source f = Bnat & surjection f & W card_zero f = a &
  forall n,  inc n Bnat -> W (succ n) f = h n (W n f)).
Definition induction_defined s a:= choose(fun f=>
  source f = Bnat &  surjection f & W \0c f = a &
  forall n, inc n Bnat -> W (succ n) f = s (W n f)).
Definition induction_defined1 h a p := choose(fun f=>
  source f = Bnat & surjection f & W \0c f = a &
  (forall n,  n <c p -> W (succ n) f = h n (W n f)) &
  (forall n, inc n Bnat -> ~ (n <=c p) -> W n f = a)).

Definition induction_defined_set s a E:= choose(fun f=>
  is_function f &source f = Bnat & target f = E & W \0c f = a &
  forall n, inc n Bnat -> W (succ n) f = s (W n f)).

Definition induction_defined0_set h a E:= choose(fun f=>
  is_function f & source f = Bnat & target f = E & W \0c f = a &
  forall n, inc n Bnat -> W (succ n) f = h n (W n f)).

Definition induction_defined1_set h a p E := choose(fun f=>
  is_function f &source f = Bnat & target f = E & W \0c f = a &
  (forall n, n <c p -> W (succ n) f = h n (W n f)) &
  (forall n, inc n Bnat -> ~ (n <=c p) -> W n f = a)).
```

Intervals We give here the original proof that the intersection of two intervals is an interval.

Let's say that an interval is of type B if it is bounded, of type L' if it is left unbounded, of type R' if it is right unbounded, of type U if it is $]\leftarrow,\rightarrow[$. Let's say that an interval is of type L if it is of type L' or U, of type R if it is of type R' or U.

Let's write $L' \cap L' = L'$ as a short-hand for: the intersection of two intervals of type L' is an interval of type L', this is lemma `intersection_i3` and will be explained later. If we consider the reverse ordering, an interval remains an interval, but the lemmas shown here are more precise (they say for instance that the opposite of L' is R').

```
Lemma opposite_interval_cc: forall r a b,
  order r -> interval_cc r a b = interval_cc (opposite_order r) b a.
Lemma opposite_interval_oo: forall r a b,
  order r -> interval_oo r a b = interval_oo (opposite_order r) b a.
Lemma opposite_interval_oc: forall r a b,
  order r -> interval_oc r a b = interval_co (opposite_order r) b a.
Lemma opposite_interval_co: forall r a b,
  order r -> interval_co r a b = interval_oc (opposite_order r) b a.
Lemma opposite_bounded_interval: forall r x, order r ->
  is_bounded_interval r x -> is_bounded_interval (opposite_order r) x.
Lemma opposite_interval_ou: forall r a,
```

```
  order r -> interval_ou r a = interval_uo (opposite_order r) a.
Lemma opposite_interval_cu: forall r a,
  order r -> interval_cu r a = interval_uc (opposite_order r) a.
Lemma opposite_interval_uu: forall r,
  order r -> interval_uu r = interval_uu (opposite_order r).
Lemma opposite_interval_uo: forall r a,
  order r -> interval_uo r a = interval_ou (opposite_order r) a.
Lemma opposite_interval_uc: forall r a,
  order r -> interval_uc r a = interval_cu (opposite_order r) a.
Lemma opposite_unbounded_interval: forall r x, order r ->
  is_unbounded_interval r x -> is_unbounded_interval (opposite_order r) x.
Lemma opposite_interval: forall r x, order r ->
  is_interval r x -> is_interval (opposite_order r) x.
```

There are 9 types of intervals, thus 81 cases to consider. The case of intervals of type U is trivial, so that the number of cases is really 64. The new proof replaces bounded intervals by unbounded intervals, so that there are only 16 cases to consider. Let's start with these ones.

Case $\mathrm{L}' \cap \mathrm{R}' = \mathrm{R}' \cap \mathrm{L}' = \mathrm{B}$. Consider $\mathrm{X} =]\leftarrow, x[\cap]y, \rightarrow[$. If the intersection is non-empty, there is a such that $y \le a \le x$, thus $y \le x$, and $\mathrm{X} =]y, x[$. Otherwise $\mathrm{X} =]x, x[$. Similarly, if we consider intervals that contain the end-point x or y, the intersection is empty, or an interval that contains the end-point x or y.

Case $\mathrm{L}' \cap \mathrm{L}' = \mathrm{L}'$. Consider $\mathrm{X}(b) =]\leftarrow, b[$ and $\mathrm{Y}(b) =]\leftarrow, b]$. Let $d = \inf(b, c)$. We have $\mathrm{X}(d) \subset \mathrm{X}(b) \cap \mathrm{X}(c) \subset \mathrm{Y}(d)$. If d is in the intersection, then the intersection is $\mathrm{Y}(d)$, otherwise it is $\mathrm{X}(d)$. Replacing one of $\mathrm{X}(b)$ or $\mathrm{X}(c)$ by $\mathrm{Y}(b)$ or $\mathrm{Y}(c)$ is similar. Note: the intersection of two closed intervals is empty or closed, and the intersection of two open intervals is open, only when the order is total.

Using the reverse order, it follows $\mathrm{R}' \cap \mathrm{R}' = \mathrm{R}'$, and this covers all unbounded intervals. All remaining cases are similar. We must consider what happens on the left, and what happens on the right. The big part of the proof (300 lines) consists in showing that the intersection of two bounded intervals is a bounded interval. This does not follow directly from our new theorem, but is easy (for instance, if X is a subinterval of $[a, b]$, of the form $]\leftarrow, x[$, it is $[a, x[$).

```
Lemma intersection_interval1: forall r x y,
  lattice r ->is_closed_interval r x -> is_closed_interval r y  ->
    is_bounded_interval r (intersection2 x y).
Lemma intersection_interval2: forall r x y,
  lattice r ->is_open_interval r x -> is_open_interval r y  ->
  is_bounded_interval r (intersection2 x y).     (* 39 *)
Lemma intersection_interval3: forall r a b a' b',
  lattice r -> inc a (substrate r) -> inc a' (substrate r)  ->
  inc b (substrate r)  -> inc b' (substrate r) ->
  is_bounded_interval r
      (intersection2(interval_co r a b)(interval_co r a' b')) (* 19 *)
Lemma intersection_interval4: forall r a b a' b',
  lattice r -> inc a (substrate r) -> inc a' (substrate r)  ->
  inc b (substrate r)  -> inc b' (substrate r) ->
  is_bounded_interval r (intersection2(interval_oc r a b)(interval_oc r a' b')).
Lemma intersection_interval5: forall r a b a' b',
  lattice r -> inc a (substrate r) -> inc a' (substrate r)  ->
  inc b (substrate r)  -> inc b' (substrate r) ->
  is_bounded_interval r
     (intersection2(interval_co r a b)(interval_oc r a' b')). (* 30 *)
Lemma intersection_interval6: forall r x y,
```

```
  lattice r ->is_semi_open_interval r x -> is_semi_open_interval r y  ->
    is_bounded_interval r (intersection2 x y).
Lemma intersection_interval7: forall r a b a' b',
  lattice r -> inc a (substrate r) -> inc a' (substrate r)  ->
  inc b (substrate r)  -> inc b' (substrate r) ->
  is_bounded_interval r
      (intersection2(interval_cc r a b)(interval_oo r a' b')). (* 30 *)
Lemma intersection_interval8: forall r a b a' b',
  lattice r -> inc a (substrate r) -> inc a' (substrate r)  ->
  inc b (substrate r)  -> inc b' (substrate r) ->
  is_bounded_interval r
      (intersection2(interval_cc r a b)(interval_oc r a' b')). (* 18 *)
Lemma intersection_interval9: forall r a b a' b',
  lattice r -> inc a (substrate r) -> inc a' (substrate r)  ->
  inc b (substrate r)  -> inc b' (substrate r) ->
  is_bounded_interval r
      (intersection2(interval_oo r a b)(interval_oc r a' b')). (* 34 *)
Lemma intersection_interval10: forall r a b a' b',
  lattice r -> inc a (substrate r) -> inc a' (substrate r)  ->
  inc b (substrate r)  -> inc b' (substrate r) ->
  is_bounded_interval r (intersection2(interval_oo r a b)(interval_co r a' b')).
Lemma intersection_interval11: forall r a b a' b',
  lattice r -> inc a (substrate r) -> inc a' (substrate r)  ->
  inc b (substrate r)  -> inc b' (substrate r) ->
  is_bounded_interval r (intersection2(interval_cc r a b)(interval_co r a' b')).
Lemma intersection_interval12: forall r x y, lattice r ->
  is_bounded_interval r x -> is_bounded_interval r y ->
  is_bounded_interval r (intersection2 x y).
```

We consider now the case of unbounded intervals.

```
Lemma intersection_interval13: forall r x,
  is_interval r x -> intersection2 x (interval_uu r) = x.
Lemma intersection_interval14: forall r x y, lattice r ->
  is_left_unbounded_interval r x -> is_left_unbounded_interval r y ->
  is_left_unbounded_interval r (intersection2 x y).  (* 18 *)
Lemma intersection_interval15: forall r x y, lattice r ->
  is_right_unbounded_interval r x -> is_right_unbounded_interval r y ->
  is_right_unbounded_interval r (intersection2 x y).
Lemma intersection_interval16: forall r x y, lattice r ->
  is_left_unbounded_interval r x -> is_right_unbounded_interval r y ->
  is_bounded_interval r (intersection2 x y). (* 19 *)
Lemma intersection_interval17: forall r x y, lattice r ->
  is_unbounded_interval r x -> is_unbounded_interval r y ->
  is_interval r (intersection2 x y).
Lemma intersection_interval18: forall r x y, lattice r ->
  is_left_unbounded_interval r x -> is_bounded_interval r y ->
  is_bounded_interval r (intersection2 x y). (* 97 *)
Lemma intersection_interval19: forall r x y, lattice r ->
  is_right_unbounded_interval r x -> is_bounded_interval r y ->
  is_bounded_interval r (intersection2 x y).
Lemma intersection_interval20: forall r x y, lattice r ->
  is_unbounded_interval r x -> is_bounded_interval r y ->
  is_bounded_interval r (intersection2 x y).
```

The result is now obvious.

```
Theorem intersection_interval: forall r x y,
  lattice r -> is_interval r x -> is_interval r y ->
  is_interval r (intersection2 x y).
```

These are the original tactics used in this section.

```
Ltac uf_interval :=
  uf interval_cc;  uf interval_oo;  uf interval_co; uf interval_oc;
  uf interval_uu;  uf interval_uo;  uf interval_ou;
  uf interval_uc;  uf interval_cu.
Ltac zztac:=
  set_extens; Ztac; ee; match goal with H: inc _ (Zo _ _) |- _ => clear H end.
Ltac zztac2:= uf_interval; set_extens ;
  [  match goal with H: inc _  (intersection2 _ _) |- _ =>
       nin (intersection2_both H)  end ;
     match goal with
       H1:(inc _ (Zo _ _)), H2 :(inc _ (Zo _ _)) |- _
      => nin (Z_all H1); nin (Z_all H2); clear H1; clear H2; ee end;Ztac
    |
      Ztac; match goal with H: inc _ (Zo _ _) |- _ => clear H end;
      app intersection2_inc; Ztac; uf glt;ee; try order_tac].
```

14.7 The size of one

In a previous chapter, we calculated the size of 1, i.e. the number of symbols of the assembly that denotes 1 in the formalization of Bourbaki. We can ask the question: what is the size of 1 in Gaia? The answer depends on which version of the system is considered. Currently, we use von Neumann ordinals to represent cardinals so that 1 is $\{\emptyset\}$. Before that, 1 was defined using the axiom of choice, as a Z satisfying a property P, that says that there is a bijection of $\{\emptyset\}$ onto Z.

An equivalent form of P, corresponding to the text of Bourbaki, is given by

```
exists u : Set,
   exists U : Set,
     u = J (J U Y) Z &
     (forall x : Set, inc x U -> inc x (product Y Z) &
     (forall x : Set, inc x Y -> exists y : Set, inc (J x y) U) &
     (forall x y y' : Set, inc (J x y) U -> inc (J x y') U -> y = y') &
     (forall y : Set, inc y Z -> exists x : Set, inc (J x y) U) &
     (forall x x' y : Set, inc (J x y) U -> inc (J x' y) U -> x = x')
```

The normal form is

```
exists u : Set,
  exists U : Set,
    u = J (J U Y) Z &
    (forall x : Set,
     (exists a : U, Ro a = x) ->
     exists  a : record (IM (fun _ : one_point => emptyset))
             (fun _ : Set => Z), Ro a = x) &
    (forall x : Set,
     (exists a : IM (fun _ : one_point => emptyset), Ro a = x) ->
     exists y : Set, exists a : U, Ro a = J x y) &
```

```
(forall x y y' : Set,
 (exists a : U, Ro a = J x y) -> (exists a : U, Ro a = J x y') -> y = y') &
(forall y : Set,(exists a : Z, Ro a = y) ->
 exists x : Set, exists a : U, Ro a = J x y) &
(forall x x' y : Set, (exists a : U, Ro a = J x y) ->
 (exists a : U, Ro a = J x' y) -> x = x')
```

One deduces that 1 has 500 tokens (we count `forall`, `exists`, `Set`, etc, as a single token). Depending on the version, J may be a primitive or not. In one case the normal form of the last line of the previous code becomes:

```
(exists a : U,
  Ro a = IM (fun t : two_points =>
        match t with
        | two_points_a => IM (fun _ : one_point => x')
        | two_points_b =>
           IM (fun t0 : two_points =>
              match t0 with
              | two_points_a => emptyset
              | two_points_b => IM (fun _ : one_point => y)
               end)
        end)) -> x = x')
```

In this case, the normal form of 1 has less than 2000 tokens.

In reality P has the form: there exists a function f which is injective and surjective; here injective means: whenever x and y are in the source of f, then $f(x) = f(y)$ implies $x = y$. In order to define $f(x)$, one needs to define pr_1 whose normal form is

```
chooseT
   (fun x : Set =>
    (exists x0 : Set, exists y0 : Set, y = J x0 y0) ->
    exists y0 : Set, y = J x y0 &
    ((exists x0 : Set, exists y0 : Set, y = J x0 y0) -> False) ->
    x = emptyset) (nonemptyT_intro emptyset)
```

In this new version, 1 becomes small enough in order to study its structure.

It contains 197 `exists`, 184 `Set`, 117 `Ro`, 96 `J`, 90 `graph`, 64 `fun`, 75 primitives like `IM`, 59 `emptyset`, 42 keywords (like `return`), 28 `chooseT`, 28 `False`, 21 `forall`, 724 identifiers (like x, y), 967 punctuation signs (like ->), 255 operators (like =) (parentheses and commas are not counted). Total: 3100 tokens.

14.8 Changes in Version 6

These were removed.

```
Lemma inc_lt1_substrate r x y: glt r x y -> inc x (substrate r).
Lemma inc_lt2_substrate r x y: glt r x y -> inc y (substrate r).
Lemma not_lt_self r x: glt r x x -> False.
Lemma distrib_inter_prod2 a b c:
  product (union a b) c = union2 (product a c) (product b c).
Lemma ordinal0_emptyset: \0o = emptyset.
Lemma inc0_ord x: is_ordinal x -> x <> \0o -> inc \0o x.
```

`cardinal_of_cardinal` has been renamed into `card_card`

The name of these lemmas has changed.

```
opow_increasing0 opow_increasing1 opow_increasing2 opow_increasing2b
opow_increasing3 opow_increasing4  opow_increasing5  opow_increasing6
ord_pow_omega_increasing1 opow_omega_increasing odiff_pr2.
osum1inf epsilon_fam_pr1 epsilon_fam_pr1bis
```

These were modified

```
Lemma csum_M0le a b: is_cardinal a ->is_cardinal b ->
  a <=c (a +c b).
Lemma cprod_M1le  a b: is_cardinal a ->is_cardinal b ->
  b <> \0c -> a <=c (a *c b).
Lemma oprod0r x: is_ordinal x -> x *o \0o = \0o.
Lemma oprod0l x: is_ordinal x -> \0o *o x = \0o.
Lemma ord_rev_pred x: is_ordinal x -> x <> \0o ->
  exists y, is_ordinal y & x = \1o +o y.
```

We removed these theorems

```
Lemma order_has_graph0 r x:
  order_re r x -> is_graph_of (graph_on r x) r.
Lemma order_has_graph r x:
  order_re r x -> exists g, is_graph_of g r.
Lemma order_if_has_graph r g:
  is_graph g ->  is_graph_of g r ->
  order_r r -> order_re r (domain g).
Lemma order_if_has_graph2 r g:
  is_graph g ->  is_graph_of g r ->
  order_r r -> order g.
Lemma order_has_graph2 r x:
  order_re r x -> exists g,
    g = graph_on r x &
    order g & (forall u v, r u v <-> related g u v).
Lemma induced_trans r x y:
  order r -> sub x y-> sub y (substrate r) ->
  induced_order r x = induced_order (induced_order r y) x.
```

We removed the following tactivc

```
Ltac ord_tac :=
  match goal with
    | h: ordinal_le _ ?x |- is_ordinal ?x
      => move: h => [_ [h _]]; exact h
    | h: ordinal_le ?x _ |- is_ordinal ?x
      => move: h => [h _]; exact h
    | h: ordinal_lt _ ?x |- is_ordinal ?x
      => move: h => [[_ [h _]] _]; exact h
    | h: ordinal_lt ?x _ |- is_ordinal ?x
      => move: h => [[h _] _]; exact h
    | h1: ordinal_le ?x ?y, h2:  ?y <=o ?x  |-  ?x = ?y
      => apply: (ord_leA h1 h2)
    | h1: ?x  <=o ?y, h2: ?y <=o ?z  |-  ?x  <=o ?z
      => apply: (ord_leT h1 h2)
```

```
    | h1: ?x <o ?y, h2: ?y <=o ?z  |-  ?x <o ?z
      => apply: (ord_lt_leT h1 h2)
    | h1: ?x <=o ?y, h2: ?y <o ?z  |-  ?x <o ?z
      => apply: (ord_le_ltT h1 h2)
    | h1: ?x <o ?y, h2: ?y <o ?z  |-  ?x <o ?z
      => apply: (ord_lt_ltT h1 h2)
    | h1: ordinal_le ?x ?y, h2: ordinal_lt ?y ?x  |-  _
      => elim (ord_leA1 h1 h2)
    | h1: is_ordinal ?x, h2: inc ?y ?x |- is_ordinal ?y
      => apply: (elt_of_ordinal h1 h2)
    | h1: is_ordinal ?x |- ordinal_le ?x ?x
      => apply: (ord_leR h1)
    | h1:  ordinal_lt ?x ?y |- ordinal_le ?x ?y
      => by move: h1 => []
end.

Ltac ord_tac1 :=
  match goal with
     | h1: ordinalp ?x |- \0o <=o ?x =>  apply: (ozero_least h1)
     | h1: ordinalp ?x, h2: ?x <> \0o |- \0o <o ?x
       =>  apply: (ord_ne0_pos h1 h2)
     | h: ?x <=o \0o |- ?x = \0o  =>  apply: (ole0 h)
     | h: _ <o ?x |- ?x <> \0o => apply: (ord_gt_ne0 h)
     | h: ?x <o \1o |- ?x = \0o  =>  apply: (olt1 h)
     | h: \1o <=o ?x |- \0o <o ?x => by apply/oge1P
     | h: \0o <o ?x |- \1o <=o ?x => by apply/oge1P
end.

Ltac ord_tac0 :=
 match goal with
  |  h1: ordinalp ?a, h2: inc ?b ?a |- ordinalp ?b =>
    apply: (ordinal_hi h1 h2)
  | h1: ordinalp ?x, h2: inc ?y ?x, h3: inc ?z ?y |- inc ?z ?x =>
    apply: (((ordinal_transitive h1) _ h2) _ h3)
end.

Ltac Nat_tac :=
  match goal with
    | H1: finite_c ?b, H2: cardinal_le ?a ?b |- finite_c ?a
      => apply: (le_finite_finite H1 H2)
    | H1: cardinal_le ?a ?b, H2: natp ?b |-  natp ?a
      => apply: (NS_le_int H1 H2)
    | H1: cardinal_lt ?a ?b, H2:natp ?b |- natp ?a
      => move: H1 => [H1 _ ]; apply: (NS_le_int H1 H2)
end.
```

And this one

```
Ltac co_tac := match goal with
  | Ha:cardinal_le ?a ?b, Hb: cardinal_le ?b ?c |- cardinal_le ?a ?c
     =>  apply: (cardinal_leT Ha Hb)
  | Ha:cardinal_lt ?a ?b, Hb: cardinal_le ?b ?c |- cardinal_lt ?a ?c
     =>  apply: (cardinal_lt_leT Ha Hb)
  | Ha:cardinal_le ?a ?b, Hb: cardinal_lt ?b ?c |- cardinal_lt ?a ?c
     =>  apply: (cardinal_le_ltT Ha Hb)
  | Ha:cardinal_lt ?a ?b, Hb: cardinal_lt ?b ?c |- cardinal_lt ?a ?c
```

```
     => induction Ha; co_tac
  | Ha: cardinal_le ?a ?b, Hb: cardinal_lt ?b ?a |- _
    => elim (not_card_le_lt Ha Hb)
  | Ha:cardinal_le ?x ?y, Hb:  cardinal_le ?y ?x |- _
   => solve [ rewrite (cardinal_leA Ha Hb) ; fprops ]
  | Ha: cardinal_le ?a _ |- is_cardinal ?a => induction Ha; assumption
  | Ha: cardinal_le  _ ?a |- is_cardinal ?a
   => destruct Ha as [_ [Ha _ ]]; exact Ha
  | Ha: cardinal_lt ?a _ |- is_cardinal ?a => induction Ha; co_tac
  | Ha: cardinal_lt  _ ?a |- is_cardinal ?a => induction Ha; co_tac
end.
Ltac eq_aux:= match goal with
    H: is_cardinal ?a |- cardinal ?b = ?a => wr (cardinal_le4 H); aw
  | H: is_cardinal ?a |- ?a = cardinal ?b => wr (cardinal_le4 H); aw
  | H: is_cardinal ?a |- cardinal ?a = ?b => wr (cardinal_le4 H); aw
end.
Ltac eq_aux:= match goal with
  H: cardinalp ?a |- cardinal ?b = ?a => rewrite- (card_card H); aw
| H: cardinalp ?a |- ?a = cardinal ?b => rewrite- (card_card H); aw
end.
```

In `inter_oa_or`, we dropped the condition that the set has to be non-empty.

Many names have changed. For instance `set_of_foo` has been renamed into `foos`: Example: functions, injections, permutations, etc. Moreover `is_foo` has been replaced by `foop`. Example: cardinalp, ordinalp, natp, etc.

The set of natural numbers has been renamed Nat (instead of Bnat).

Chapter 15

Theorems, Notations, Definitions

List of all theorems of Bourbaki, with their COQ equivalent.

Theorems of Chapter 1

Proposition 1 (`order_pr`) « A correspondence Γ between E and E is an ordering on E if and only if ...» , [15].

Proposition 2 (`decreasing_composition`) says that $u(v(x)) \geq x$ and $v(u(x)) \geq x$ and decreasing imply $u \circ v \circ u = u$ and $v \circ u \circ v = v$, [27].

Proposition 3 (`adjoin_greatest`) says that we can add a greatest element to an ordered set, [29].

Proposition 4 (`compare_inf_sup1` and `compare_inf_sup2`) characterizes the supremum and infimum of a subset, [35].

Proposition 5 (`sup_increasing` and `inf_decreasing`) says that $\sup_{\mathrm{A}}$ and $\inf_{\mathrm{A}}$ are increasing functions of the set A, [35].

Proposition 6 (`sup_increasing2` and `inf_decreasing2`) says that $\sup f$ and $\inf f$ are increasing functions of the function f, [36].

Proposition 7 (`sup_A` and `inf_A`) asserts associativity of sup, [36].

Proposition 8 (`sup_in_product` and `inf_in_product`) characterizes supremum and infimum in a product, [38].

Proposition 9 (`sup_induced2` and 3 variants) characterizes supremum of a subset of a subset, [38].

Proposition 10 (`right_directed_maximal` and `left_directed_minimal`) says that «in a right directed ordered set E, a maximal element a is the greatest element of E», [39].

Proposition 11 (`total_order_monotone_injective` and `total_order_increasing_morphism`) characterizes increasing functions and morphism on totally ordered sets, [43].

Proposition 12 (`sup_in_total_order` and `inf_in_total_order`) characterizes supremum and infimum in a totally ordered set, [44].

Proposition 13 (`setI_interval`) says that in a lattice, the intersection of two intervals is an interval, [45].

Theorems of Chapter 2

Proposition 1 (`well_ordered_segment`) says that «in a well-ordered set E, every segment of E other than E itself is an interval $]\leftarrow, a[$, where $a \in$ E», [50].

Proposition 2 (`segments_iso_is` and `segments_worder`) studies $x \mapsto]\leftarrow, x[$, [51].

Proposition 3 (`worder_merge`) studies the supremum of compatible well-orderings, [53].

Lemma 1 (`order_merge1` and `order_merge2`) is a helper for Proposition 3.

Lemma 2 (`transfinite_principle1` and `transfinite_principle2`) is a helper for C59.

Criterion C59 (`transfinite_principle`) is the principle of transfinite induction, [54].

Criterion C60 (`transfinite_definition` and `transfinite_definition_stable`) (Definition of a mapping by transfinite induction), [54].

Lemma 3 (`Zermelo_aux`) is a helper for Theorem 1.

Theorem 1 (`Zermelo`) says that «every set E can be well-ordered», [58].

Proposition 4 (`Zorn_aux`) is a generalization of Zorn's lemma [58].

Theorem 2 (`Zorn_lemma`) says «every inductive ordered set has a maximal element», [58].

Corollary 1 (`inductive_max_greater`).

Corollary 2 (`maximal_in_powerset` and `minimal_in_powerset`).

Theorem 3 (`isomorphism_worder`) studies existence and uniqueness of an isomorphism between two well-ordered sets, [59].

Lemma 4 (`increasing_function_segments`), [59].

Corollary 1 (`unique_isomorphism_onto_segment`), [60].

Corollary 2 (`bij_pair_isomorphism_onto_segment`), [60].

Corollary 3 (`isomorphic_subset_segment`) [60].

Theorems of Chapter 3

Proposition 1 `card_eqP` «two sets X and Y are equipotent if and only if their cardinals are equal», [78].

Theorem 1 (`cardinal_le_wor`) says that the ordering between cardinals is a well-ordering, [680].

Corollary 1 is equivalent to `card_le_to_ell`.

Corollary 2 is equivalant to `cardinal_leA`.

Proposition 2 (`cardinal_supremum2`) says that a family of cardinals has a supremum, [87].

Proposition 3 (`surjective_cardinal_le`) says «if there exists a surjection f of X onto Y, then $\mathrm{Card}(Y) \leq \mathrm{Card}(X)$», [88].

Proposition 4 (`cprod_pr` and `csum_pr`) says that the cardinal sum or cardinal product of the family $\mathrm{Card}(E_\iota)$ is the cardinal of the sum or the product of the sets E_ι; [88].

Corollary (`csum_pr1`).

Proposition 5 (`csum_An`, `cprod_An`, `csum_Cn`, `cprod_Cn` and `cprod_sum_Dn`) asserts commutativity, associativity and distributivity of sum and products, [89].

Corollary. Application to the case of 2 or 3 arguments.

Proposition 6 (`csum_zero_unit` and `cprod_one_unit`) says that one can remove 0 in a sum and 1 in a product, [92].

Corollary 1 (`csum0r`, `csum0l`, `cprod1r`, `cprod1l`).

Corollary 2 (`sum_of_ones` and `sum_of_same`).

Proposition 7 (`cprodnz`) says that a cardinal product is non-zero if and only if each factor is non-zero, [93].

Proposition 8 (`succ_injective`) asserts injectivity of the successor function, [93].

Proposition 9 (`cpow_pr`) says that a^b remains unchanged if letters are replaced by equipotent sets, [94].

Proposition 10 `cpow_pr3` says that a^b is a product where all factors are the same [94].

Corollary 1 (`cpow_sum`).

Corollary 2 (`cpow_prod`).

Corollary 3 (`cpow_prod2`).

Proposition 11 (`cpowx0` and variants), states $\mathfrak{a}^0 = 1$, $\mathfrak{a}^1 = \mathfrak{a}$, $1^\mathfrak{a} = 1$, and $0^\mathfrak{a} = 0$, [95].

Proposition 12 (`card_setP`) says $\mathrm{Card}(\mathfrak{P}(X)) = 2^X$, [95].

Proposition 13 (`cardinal_le_setCP`) states that «$\mathfrak{a} \geq \mathfrak{b}$ if and only if there exists a cardinal $\mathfrak{c}$ such that $\mathfrak{a} = \mathfrak{b} + \mathfrak{c}$», [96].

Proposition 14 (`csum_increasing` and `cprod_increasing`) says the sum and product are increasing functions, [96].

Corollary 1 (`csum_Mlele`, `cprod_Mlele`).

Corollary 2 (`cpow_Mlele`).

Theorem 2 (`cantor`) says $X < 2^X$, [97].

Corollary (`cantor_bis`).

Theorems of Chapter 4

Proposition 1 (`finite_succP`) says that «a cardinal a is finite if and only if $a+1$ is finite», [101].

Proposition 2 (`le_finite_finite`, `cpred_pr`) says that (if n is an integer), if $a \leq n$ then a is an integer, if $n > 0$ there is a unique m with $m+1 = n$ and $a < m+1$ is equivalent to $a < n$, [102].

Corollary 1 (`sub_finite_set`).

Corollary 2 (`strict_sub_smaller`).

Corollary 3 (`finite_image`).

Corollary 4 (`bijective_if_same_finite_c_inj`, `bijective_if_same_finite_c_surj`).

Criterion C61 (`Nat_induction` and variants) (principle of induction), [106]

Proposition 3 (`finite_subset_directed_bounded`, `finite_subset_lattice_inf`, `finite_subset_lattice_sup`, `finite_subset_torder_greatest`, `finite_-subset_torder_least`) gives some properties of a finite subset of an ordered set [108].

Corollary 1 (`finite_set_torder_greatest`, `finite_set_torder_worder`)

Corollary 2 (`finite_set_maximal`).

Theorem 1 (`maximal_inclusion`) says that every nonempty set which is of finite character has a maximal element, [127].

Theorems of Chapter 5

Proposition 1 (`finite_sum_finite` and `finite_product_finite`) says that a finite sum or product of integers is an integer, [117].

Corollary 1 (`finite_union_finite`).

Corollary 2 (`finite_product_finite_set`).

Corollary 3 (`NS_pow`).

Corollary 4 (`finite_powerset`).

Proposition 2 (`card_lt_pr`) says that $a < b$ if and only if there is c such that $0 < c$ and $b = c + a$; [118].

Proposition 3 (`finite_sum_lt` and `finite_product_lt`) says that $\sum a_i < \sum b_i$ and $\prod a_i < \prod b_i$ if $a_i \leq b_i$ for each i and $a_j < b_j$ for some j, [119].

Corollary 1 (`cpow_Mltle`).

Corollary 2 (`cpow_Mlelt`).

Corollary 3 (`csum_simplifiable_left` and variants).

Corollary 4 (`cdiff_pr` and others).

Proposition 4 (`restr_plus_interval_isomorphism`) says that $x \mapsto x + a$ is a bijection (order isomorphism) $[0, b] \to [a, a + b]$, [124].

Proposition 5 (`cardinal_Nintcc`) gives the cardinal of $[a, b]$, [124].

Proposition 6 (`finite_ordered_interval`) asserts that every finite totally ordered set is isomorphic to a unique interval $[1, n]$, [125].

Proposition 7 (`char_fun_setU` and others) states properties of the characteristic function of a set, [126].

Theorem 1 (`cquorem_pr`) asserts existence and uniqueness of Euclidean division, [127].

Proposition 8 is expansion to base b [130].

Proposition 9 (`shepherd_principle`) says that if f is a function from a set with cardinal $\mathfrak{a}$ onto a set with cardinal $\mathfrak{b}$, and if all set $f^{-1}\langle\{x\}\rangle$ have the same cardinal $\mathfrak{c}$, then $\mathfrak{a} = \mathfrak{b}\mathfrak{c}$, [143].

Proposition 10 (`number_of_injections_pr`) gives the number of injections from a finite set into another one, [144].

Corollary (`number_of_permutations`).

Proposition 11 (`number_of_partitions`) gives the number of partitions with p_i elements, [148].

Corollary 1 (`binomial7`).

Corollary 2 (`cardinal_set_of_increasing_functions`).

Proposition 12 (`sum_of_binomial`) $\sum_p \binom{n}{p} = 2^n$ [150].

Proposition 13 is the binomial formula (is a definition in COQ) [150].

Proposition 14 (`cardinal_pairs_lt` and `cardinal_pairs_le`) counts the number of pairs (i, j) such $1 \leq i \leq j \leq n$ or $1 \leq i < j \leq n$, [160].

Corollary (`sum_of_i`).

Proposition 15 counts the number of monomials, [161].

Theorems of Chapter 6

Theorem 1 «The relation 'x is an integer' is collectivizing » (is equivalent to the existence of **N**), [80].

Criterion C62 (restatement on C61, not shown in COQ).

Criterion C63 (`induction_defined_pr`), [194].

Lemma 1 (`infinite_greater_countable`) «Every infinite set, E contains a set equipotent to **N**».

Lemma 2 (`equipotent_N2_N`) «The set **N** × **N** is equipotent to **N**».

Theorem 2 (`equipotent_inf2_inf`) «for every infinite cardinal $\mathfrak{a}$, we have $\mathfrak{a} = \mathfrak{a}^2$» [197]

Corollary 1 (`power_of_infinite`).

Corollary 2 (`finite_family_product`).

Corollary 3 (`notbig_family_sum1`).

Corollary 4 (`sum2_infinite`, `product2_infinite`).

Proposition 1 (`countable_subset`, `countable_product` `countable_union`) states properties of countable sets [200].

Proposition 2 (`countable_finite_or_N`) «Every countable infinite set E is equipotent to **N**», [200].

Proposition 3 (`infinite_partition`) says that every infinite set E has a partition X_i where X_i is equipotent to E and the index set to **N**, [200].

Proposition 4 (`countable_inv_image`) says that if f is a function from E onto F, such that F is infinite and $f^{-1}\langle\{x\}\rangle$ is countable for any $x \in F$, then F is equipotent to E, [200].

Proposition 5 (`infinite_finite_subsets`) says that the set of finite subsets of an infinite set E is equipotent to E, [201].

Proposition 6 (`increasing_stationary`) characterizes stationary sequences, [202].

Corollary 1 (`decreasing_stationary`).

Corollary 2 (`finite_increasing_stationary`).

Proposition 7 (`noetherian_induction`) (Principle of Noetherian induction), [203].

Symbols

$x \wedge y$ is often replaced by "and". The Coq equivalent is `/\`.

$x \vee y$ is often replaced by "or". The Coq equivalent is `\/`.

$\neg x$ is often replaced by "not". The Coq equivalent is `~`.

$(a|b)c$ is a Bourbaki notation, meaning the relation obtained by replacing b by a in c.

$R\{x\}$ is a Bourbaki notation, meaning that R is a relation that may depend on x. If R is a relation that depends on y, it is also $(x|y)R$.

$\tau_x(R)$ is a Bourbaki notation, it is the generic element satisfying $R\{x\}$.

$x \implies y$ is represented in Coq by `x -> y`.

$x \to y$ is a Coq notation meaning the type of functions from type a to type b.

$x = y$ is equality. The Axiom of Extent says that two sets having the seame elements are equal.

x : y is a Coq notation meaning that x is of type y.

$f(x)$ is the value of the function f at point x, parentheses are sometimes omitted.

$f\langle x\rangle$ is the set of all $f(t)$, where t is any element of x.

$\overset{-1}{f}\langle x\rangle$ see `inverse_fun`.

$(\forall x)P$ and `forall x, P` are Bourbaki and Coq notations for: every x satisfies P.

$(\exists x)P$ and `exists x, P` are Bourbaki and Coq notations for: some x satisfies P.

$(\exists! x)P$ and `exists! x, P` are Bourbaki and Coq notations for: a unique x satisfies P.

$x \in y$, (is element of): see `inc`.

$x \subset y$ (is subset of): see `sub`.

$\emptyset$ (empty set): see `emptyset`.

$\{x, \mathrm{R}\}$ (set of x such that R): see `Zo`.

$\{x\}$, $\{x, y\}$: see `singleton` or `doubleton`.

$a - b$, $a \setminus b$, $\complement a$: see `complement`.

(x, y) (ordered pair): see `J`.

$\bigcup \mathrm{X}$, $\bigcup_{\iota\in\mathrm{I}} \mathrm{X}_\iota$, see `union`.

$a \cup b$, $a \cap b$, see `union2`, `intersection2`.

$\mathrm{A} \times \mathrm{B}$, $u \times v$, $\mathrm{R} \times \mathrm{R}'$, see `product`, `ext_to_prod`, `prod_of_relation`.

$f \circ g$, see `fcompose`, `gcompose`, `compose_graph`, `compose`, `composeC`.

Δ_A, see `diagonal`.

$\overset{-1}{\mathrm{G}}$ see `inverse_graph`, `inverse_fun`, `inverse_image`, or `inverseC`.

$x \mapsto y$ or $x \to y$ is the function that maps x to y, for instance $x \mapsto \sin x$ (source and target are implicit).

$\boldsymbol{x} \to \boldsymbol{T}$ ($\boldsymbol{x} \in \boldsymbol{A}$, $\boldsymbol{T} \in \boldsymbol{C}$), is the function with source A, target C that maps x to T.

$(f_x)_{x\in\mathrm{A}}$ is a shorthand for $x \to f(x)$ $(x \in \mathrm{A})$; see above, the piece $\boldsymbol{T} \in \boldsymbol{C}$ is implicit.

$\hat{f}$ may denote `extension_to_parts`.

F^E, set of graphs of functions from E to F, `gfunctions`.

$\mathscr{F}(\mathrm{E};\mathrm{F})$, set of functions from E to F, see see `functions`.

$\Phi(\mathrm{E},\mathrm{F})$, set of functions from a subset of E to F, see `sub_functions`.

f_x, f_y sometimes denotes the mappings $y \mapsto f((x, y))$ or $x \mapsto f((x, y))$, implemented as `first_partial_fun`, `second_partial_fun`.

$\tilde{f}$, implemented as `first_partial_function`, `second_partial_function`, sometimes denotes the mappings $x \mapsto f_x$ or $y \mapsto f_y$.

$f \mapsto \tilde{f}$, implemented as `first_partial_map`, `second_partial_map`, is a bijection from $\mathscr{F}(\mathrm{B} \times \mathrm{C};\mathrm{A})$ into $\mathscr{F}(\mathrm{B};\mathscr{F}(\mathrm{C};\mathrm{A})$ or $\mathscr{F}(\mathrm{C};\mathscr{F}(\mathrm{B};\mathrm{A}))$.

$\prod_{\iota\in\mathrm{I}} \mathrm{X}_\iota$, product of a family of sets, see `productt`.

$(x_\iota)_{\iota\in\mathrm{I}}$ denotes an element of a product indexed by I.

$x \overset{r}{\sim} y$ is sometimes used instead of $r(x, y)$, especially when r is the graph of an equivalence relation.

$g_\mathrm{E}(\sim)$, the graph of $\sim$ on E, see `graph_on`.

$\sim_f$ may denote `eq_rel_associated f`.

$\bar{x}$, may denote the equivalence class of x, see `class`.

$\hat{x}$ may denote a representative of the equivalence class x.

$\mathrm{E}/\sim$, E/R (quotient set of E) see `quotient`.

R/S (quotient of two equivalence relations) see `quotient_of_relations`.

X_f sometimes means $f^{-1}\langle f\langle \mathrm{X}\rangle\rangle$, see `inverse_direct_value`.

R_A see `induced_relation`.

$x \succ_r y$, $y \prec_r x$, notations used when x and y are related by a preorder relation, [12].

$x \le y$, $y \ge x$, $x < y$, $x > y$: notations used when x and y are related by an order relation, see `gle`, `gge`.

$x \le_r y$, $y \ge_r x$, notations used when x and y are related by an order relation.

$x \le_\mathrm{ord} y$ is the orderings of ordinals, see `ordinal_le`.

$x \le_\mathrm{card} y$ is the ordering of cardinals, see `cardinal_le`.

$x \leq_{\mathbf{N}} y$ is the ordering of integers, see Nat_le.

$f \mapsto G_f$ see graph_of_function.

$\varpi \mapsto \tilde{\varpi}$ see graph_of_partition.

$\sup(x,y)$, $\sup_E X$, $\sup X$, $\sup\limits_{x\in A} f(x)$ see supremum, sup, sup_graph.

$\inf(x,y)$, $\inf_E X$, $\inf X$, $\inf\limits_{x\in A} f(x)$, see infimum, inf, inf_graph.

$[a,b]$, $[a,b[$, $]a,b]$, $]a,b[$, $[a,\rightarrow[$, $]a,\rightarrow[$, $]\leftarrow,b]$, $]\leftarrow,b]$, $]\leftarrow,\rightarrow[$, see interval.

$\mathfrak{r} \leq_{\mathrm{Card}} \mathfrak{n}$, order on cardinals, see cardinal_le.

$g^{(x)}$ is the restriction of g to $]\leftarrow,x[$, see restriction_to_segment.

$\mathrm{Card}(x)$, $\mathrm{card}(x)$, is the cardinal of x, see cardinal.

0, 1, 2, 3, 4: generic notation for a cardinal, an element of **Z**, **Q** or **R**.

$\sum\limits_{\iota\in I}\mathfrak{a}_\iota$, $\mathsf{P}_{\iota\in I}\mathfrak{a}_\iota$: cardinal sum or cardinal product of a family of cardinals, see card_sum and card_prod.

REFAIRE $\mathfrak{a}+\mathfrak{b}$, $\mathfrak{a}.\mathfrak{b}$, $\mathfrak{a}\mathfrak{b}$, is the cardinal sum or cardinal product of two cardinals, see card_sum2 card_prod2.

$E_1 + E_2$ denotes also the ordinal sum, see ordinal_sum.

$X_{xy}(a,b)$ is the family $x \mapsto a$ and $y \mapsto b$, see page 82.

$X(a,b)$ is $X_{\alpha,\beta}(a,b)$, for some fixed α and β.

$a_x \cup b_y$ is the disjoint union of $X_{xy}(a,b)$, i.e., $a \times \{x\} \cup b \times \{y\}$.

$\mathfrak{a}^{\mathfrak{b}}$ is the cardinal power of two cardinals, see card_pow.

a^b is the power of two integers, see pow.

x^y, ordinal power, see ord_pow.

$\mathbb{N}$, **N**, set of integers, see Nat.

$a-b$ may be denote the cardinal difference card_diff or ordinal difference ord_diff.

$[a,b]$ is an interval on **N**, [122].

$\sum\limits_{i=a}^{b} t_i$ is $\sum\limits_{i\in[a,b]} t_i$.

ϕ_A is the characteristic function on E see char_fun.

a/b is the quotient of the two cardinals a and b.

$n!$ is the factorial of n.

$\binom{n}{p}$, see binom.

x^+ see succ_o.

x^- is the ordinal predecessor of x.

$\mathrm{ord}(x)$ see ordinal.

$r \preceq_{\mathrm{ord}} r'$ see order_le.

$x \leq_{\mathrm{ord}} y$, $x <_{\mathrm{ord}} y$ see ordinal_le.

$x \ll y$ see ord_negl.

$x^{<y}$ see cpow_less.

$x \sharp y$ see ord_natural_sum.

ω, ω_n is the least infinite ordinal, or the initial ordinal of index n, see omega_fct .

Ω is defined by Cantor as the order type of the second class of numbers, [395].

$\aleph_n$ is the cardinal of ω_n, see omega_fct.

$\aleph_0$, $\aleph_1$, $\aleph_2$, are the first initial cardinals. Note that $\aleph_0 = \omega_0 = \mathbf{N}$, when using von Neumann cardinals.

ϵ_n is the ϵ ordinal of index n, see `epsilon_fam`.

$\mathrm{E}(x)$ a number used in the definition of ϵ_n.

$\mathrm{cf}(x)$ is the cofinality of x.

$\gimel x = x^{\mathrm{cf}(x)}$ is the gimel function, see `gimel_fct`.

$\beth x$ is the beth function, see `beth_fct`.

Φ, Ψ see `Schutte_phi`.

Γ_0 see `Gamma_0`.

$f^{(x)}$, $f_{(x)}$ may denote the restriction of the function f to the set of elements $< x$.

f' may denote the first derivation of f.

$a \oplus b$, $a \sharp b$ may denoet the natural sum of two ordinals see `natural_sum`.

$\mathfrak{S}_n$ iq the set of permutations of the set of integers $< n$.

$\mathrm{X} \Vdash \Phi$ says thay Φ is true in X, see `pformula_valid_on`.

Letters

AF: axiom of foundation.

$\mathscr{B}$ see `Bo`.

CNF: Cantor Normal Form.

$\mathscr{C}_{\mathrm{T}}(p,q)$, $\mathscr{C}(p)$: see `chooseT` and `choose`.

$\mathrm{C}_{xy}a$ stands for `constant_function x y a`, it is the constant function from x to y with value a.

$\mathrm{C}_{\mathrm{R}}x$ may denote the equivalence class of x for R, see `class`.

Cl(X) is the transitive closure of a set X.

$\mathrm{Coll}_x \boldsymbol{R}$ says that R is collectivizing in x.

$\mathscr{E}$, see `Set`.

$\mathrm{E}(x)$ see `epsilon_fct`.

$\mathscr{E}_x(\mathrm{R})$ appears in the English version where $\{x, \mathrm{R}\}$ is used in the French version; see `Zo`.

$\mathscr{F}$ is the set of formuas, see `all_formulas`.

$\mathrm{fv}(x)$: is the set of free variables of the formula x, see `free_vars`.

$\mathrm{G}(f)$ may denote the graph of a function f.

HF: Hereditarily finite.

$\mathrm{J}_{\mathrm{R}}(x)$ is the ordinal of all t such that $t < x$ for the relation R, [318]

$\mathrm{K}_{\mathrm{R}}(\alpha)$, $\mathrm{K}_{\mathrm{B}}(\alpha)$ inverse function of J_{R}, the α-th element of the collection D (ordered by R). [318].

I_{A}, see `identity`.

I_{xy} see `inclusionC`, `canonical_injection`.

$\mathscr{I}(\mathrm{E}, \mathrm{T}, f)$ says that f is defined by transfinite induction, see `transfinite_def`.

$\mathscr{L}_{\mathrm{X}} f$, $\mathscr{L} f$, $\mathscr{L}_{\mathrm{A};\mathrm{B}} f$ (creating functions): see `L`, `acreate`, `Lf`.

LHS is the left hand side of an equality.

$\mathscr{M} f$, $\mathscr{M}_{\mathrm{A};\mathrm{B}} f$ (inverse of $\mathscr{L}$) see `bcreate1` and `bcreate`.

$\mathbb{N}$, $\mathbf{N}$, set of integers, see `Nat`.

$\mathrm{N}_f(x)$ may denote the the least y such that $x \le y$ and $f(y) = y$.

$\mathscr{N}$, is the bijection from $\mathbb{N}$ onto $\mathbf{N}$, see `nat_to_B`.

$o(\mathrm{E})$, $o'(\mathrm{E})$, see `ordinal_o`.

OFS : ordinal functional symbol.

$\mathfrak{P}(x)$, see `powerset`.

$\mathrm{pr}_1 z$, $\mathrm{pr}_2 z$, $\mathrm{pr}_\iota f$, $\mathrm{pr}_J f$ (projections), see P, Q, `pr_i`, `pr_j`.

$\mathscr{R} x$ see `Ro`.

$\mathrm{R}_{ab} f$ (restriction) see restriction2.

RHS is the right hand side of an equality.

$\mathrm{S}(r)$ denotes the substrate of r, see substrate.

$\mathrm{S}(f)$ may denote the source of a function f.

S_x may denote the segment with end-point x (the set of all y such that $y < x$).

$\mathrm{T}(f)$ may denote the target of a function f.

$\mathscr{V}(x, f)$, $\mathscr{V}_f x$ (value of a function): see `Vg`.

V_α is the von Neumann universe of index α; see `universe`.

V is the von Neumann universe, see `universe`.

$\mathscr{V}$ is the set of variables.

Val(ϕ,X) is the value of a formula ϕ in a set X, see `formula_value`.

$\mathscr{W}_f x$ (value of a function): see `Vf`.

$\mathscr{Y}(\mathrm{P}, x, y)$ see `Yo`.

$\mathscr{Z}(x, \mathrm{P})$ see `Zo`.

ZF: the axioms (or theory) of Zermelo Fraenkel.

ZFC: the axioms of Zermelo Fraenkel, including the axiom of choice.

¶ is not defined. We use it as a paragraph separator.

Words

`acreate f`, $\mathscr{L} f$, is the correspondence associated to the Coq function f.

`agrees_on x f f'`, `agreeC x f f'` , is the property that for all $a \in x$, $f(a)$ and $f'(a)$ are defined and equal.

`aleph0`, `aleph1`, `aleph2`, are the first three infinite cardinals, [406], [201].

`aleph_one`, $\aleph_1$, is the next cardinal after $\aleph_0$.

`aleph_succ_comp x`, is the complement of $\aleph_x$ in $\aleph_{x+1}$.

`all_der f b x i`, `all_der_aux f E x i`, `all_der_bound f b`: repeated derivation of f, and auxiliary definitions, [334].

`all_formulas`, the set of formulas [455].

`antisymmetric_r r`: says that the relation r is antisymmetric, [12].

`antisymmetricp r`: says that the graph r is antisymmetric, [12].

`asymmetric_set E` : says that if $x \in \mathrm{E}$ and $y \in \mathrm{E}$, at least one of $x \in y$ and $y \in x$ is false, [66].

`atomic_formulas`, `atomic_formula` : the set of atomic formulas, [455].

`axioms_product_order f g`: is the condition under which `product_order f g` is an order, [23].

`base_two_reverse n`, the number whose expansion in base two is the reverse of that of n, [140].

`bcreate f A B`, $\mathscr{M}_{\mathrm{A};\mathrm{B}} f$, is a kind of inverse of $\mathscr{L}$.

`bcreate1 f`, $\mathscr{M} f$, is a kind of inverse of $\mathscr{L}$.

`Bell` Bell numbers.

`Bezout_pos a b u v`: a specialization of the Bezout relation, [226].

`Bezout_real a b u v`: the relation $au + bv = 1$, [224].

`beth_fct x`, $\beth x$, is the beth function, [438].

`bijective f`, `bijectiveC f`, means that f is a bijection.

`binom n p`, $\binom{n}{p}$, is the binomial coefficient, [150].

`Bo`, $\mathscr{B}$, is an inverse of $\mathscr{R}$.

`Bolzano_hyp f x y` says that f is continuous on the closed interval $[a, b]$, [287].

`bounded_above r X`, `bounded_below r X`, `bounded_both r X`, mean that X is bounded for r (from above, below or both), [31].

`bounded_interval r x`, says x is an interval of the form $[a, b]$, $]a, b[$, $]a, b]$, or $[a, b[$.

`BQ`, `BQm`, `BQms`, `BQp`, `BQps`, the set **Q** and its subsets, [231].

`BQ_four`, \4q, is 4 in **Q**, [233].

`BQ_half`, \2hq, is 1/2 in **Q**, [233].

`BQ_int01`, the open interval $]0, 1[$ in **Q**, [247].

`BQ_int01_ordering` the ordering of $]0, 1[$ in **Q**, [247].

`BQ_le x y`, `x <=q y`, the order relation on **Q**, [235]

`BQ_lt x y`, `x <q y`, the strict order relation on **Q**, [235]

`BQ_mone`, \1mq, is -1 in **Q**, [233].

`BQ_of_nat n` is the rational number associated to $x \in \mathbf{N}$, [232].

`BQ_of_pair a b` the rational number associated to a/b, with a and b in **Z**, [232].

`BQ_of_R x` is the rational number y, which is equal to the real number x, [267].

`BQ_of_Z x` is the rational number associated to $x \in \mathbf{Z}$, [232].

`BQ_of_Zinv x` is the rational number $1/x$ when $x \in \mathbf{Z}$, [232].

`BQ_one`, \1q, is 1 in **Q**, [233].

`BQ_ordering` is the ordering of **Q**, [235].

`BQ_seq x`, says that x is functional graph $\mathbf{N} \to \mathbf{Q}$, [284].

`BQ_two`, \2q, is 2 in **Q**, [233].

`BQ_zero`, \0q, is 0 in **Q**, [233].

`BQabs x`, the absolute value on **Q**, [234].

`BQdiff x y`, `x -q y`, the difference on **Q**, [237].

`BQdiv x y`, `x /q y`, the division on **Q**, [241].

`BQdouble x` is $2x$ on **Q**, [240].

`BQfloor x`, the floor function on **Q**, [245].

`BQhalf x` is $x/2$ on **Q**, [246].

`BQinv x`, the inverse on **Q**, [240].

`BQmiddle x y` is $(x + y)/2$ on **Q**, [246]

`BQopp x` is the opposite of x on **Q**, [234].

`BQpair C`, `BQpairL C`, `BQpairR C`, `BQpair_aux`, etc: pair of adjacent sequences, and related properties. [281]

`BQprod x y`, `x *q y`, the product on **Q**, [238].

`BQps_ordering`, the ordering of $]0, \infty[$ in **Q**, [247].

`BQQsign x`, the sign of $x \in \mathbf{Q}$, as an element of **Q**, [242].

`BQsign x`, the sign of $x \in \mathbf{Q}$, as an element of **Z**, [242].

`BQsquare x` is x^2 on **Q**, [240].

`BQsum x y`, `x +q y`, addition on **Q**, [236].

`BR`, `BRm`, `BRms`, `BRp`, `BRp`, `BRps`, the set of real numbers and its subsets,[267].

`BR_between x a b` means $a \le x \le b$ or $b \le x \le a$ on **R**, [287].

`BR_four`, `\4r`, is the constant 4 in **R**, [267].

`BR_half`, `\2hr`, is the constant 1/2 in **R**, [267].

`BR_le x y`, `x <= y`, the comparison on **R**, [269].

`BR_lt x y`, `x <r y`, the strict comparison on **R**, [269].

`BR_mone`, `\1mr`, the constant −1 in **R**, [267].

`BR_near x e y` says $|x - y| \le e$, [286].

`BR_of_Q x`, the canonical injection $\mathbf{Q} \to \mathbf{R}$, [266].

`BR_one`, `\1r`, is the constant 1 in **R**, [267].

`BR_order`, the order of **R**, [269].

`BR_seq x` says that x is a functional graph $\mathbf{N} \to \mathbf{R}$, [284].

`BR_seq_of_Q s`: conversion of a functional graph $\mathbf{N} \to \mathbf{Q}$ into a functional graph $\mathbf{N} \to \mathbf{R}$, [285].

`BR_three`, `\3r`, is the constant 3 in **R**, [267].

`BR_two`, `\2r`, is the constant 2 in **R**, [267].

`BR_zero`, `\0r`, the constant 0 in **R**, [267].

`BRabs x`: the absolute value on **R**, [279].

`BRdiff x y` :subtraction on **R**, [272].

`BRdiv x y`, division on **R**, [278].

`BRdouble x` is $2x$ on **R**, [280].

`BRhalf x` is $x/2$ on **R**, [280].

`BRinv x`, the inverse of x on **R**, [277].

`BRmiddle x y` is $(x + y)/2$ on **R**, [280].

`BRopp x`, the opposite of x on **R**, [271].

`BRprod x`, the product of x and y on **R**, [274].

`BRsqrt x` is the positive square root of a real number, [277].

`BRsqrt2` the square root of 2 on **R**, [267].

`BRsquare x` is x^2 on **R** [277].

`BRsum x y` is addition on **R**, [271].

`BZ, BZms, BZp, BZps, BZm, BZs` are the sets **Z** and the subsets of numbers respectively $z < 0$, $z \ge 0$,$z > 0$, $z \le 0$ and $z \ne 0$, [206].

`BZ_four`, `\4z`, is 4 in **Z**, [206].

`BZ_ideal x`, says that x is an ideal of **Z**, [222].

`BZ_ideal2 a b`, `BZ_ideal1 a`, define the ideal generated by a and b, [222].

`BZ_le x y`, `x <=z y`, $x \le_{\mathbf{Z}} y$, is the order relation on **Z**, [209].

`BZ_lt x y`, `x <z y`, $x <_{\mathbf{Z}} y$, is the strict order relation on **Z**, [209].

`BZ_mone`, `\1mz`, is −1 in **Z**, [206].

`BZ_of_nat x` is the canonical injection $\mathbf{N} \to \mathbf{Z}$, [206].

`BZ_of_Z x` is bijection between **Z** and the COQ integers.

`BZ_one`, `\1z`, is 1 in **Z**, [206].

`BZ_ordering` is the ordering of **Z**, [209].

`BZ_three`,`\3z`, is 3 in **Z**, [206].

`BZ_two`, `\2z`, is 2 in **Z**, [206].

`BZ_zero`,`\0z`, is 0 in **Z**, [206].

`BZabs x`, is the absolute value of $x \in \mathbf{Z}$, [208].

`BZBezout a b` says that there is a Bezout relation between a and b, [224].

`BZcoprime a b` says that a and b are coprime in **Z**, [224].

`BZdiff x y`, `x -z y`, is $x - y$ in **Z**, [213].

`BZdivides x y`, `x %|z y`, divisibility on **Z**, [221]

`BZdivision_prop a b q r`, says $a = bq + r$, and $0 \le r < |b|$, [220].

`BZdvdordering`, the ordering induced by the divisibility relation on $\mathbf{Z}^+$, [225].

`BZgcd a b`, the gcd on **Z**, [223].

`BZgcd_prop a b p`, `BZgcdp_prop a b p`, characteristic property of the gcd on **Z** or $\mathbf{Z}^+$, [225]

`BZlcm a b`, the lcm on **Z**, is $ab/gcd(a,b)$, [223]

`BZm_of_nat x` is the function $x \mapsto -x$, $\mathbf{N} \to \mathbf{Z}$, [206].

`BZopp x`, is the opposite of $x \in \mathbf{Z}$, [208].

`BZpred x` is $x - 1$ in **Z**, [213].

`BZprod x y`, `x *z y`, is the product of two elements of **Z**, [214].

`BZquo x y`, `x %/z y`, quotient on **Z**, [220].

`BZrem x y`, `x %%z y`, remainder on **Z**, [220].

`BZsign x` is the sign function in **Z**, [214].

`BZsucc x` is $x + 1$ in **Z**, [213].

`BZsum x y`, `x +z y`, is the addition on **Z**, [210].

`C0`, `C1`, `C2`, are respectively 0, 1, 2, sets with zero, one and two elements.

`canon_proj r`, is the mapping $x \mapsto \bar{x}$ from E onto E/R, where E/R is the quotient set of r.

`canonical_doubleton_order x y` is the well-ordering on the canonical doubleton, [48].

`canonical_du2 x y` is the canonical disjoint union of two sets, [294].

`canonical_injection x y`, I_{xy}, is the inclusion map on $x \subset y$.

`Cantor_eta`: is the order type of the ordering of **Q**, [474].

`cantor_mon b e c i`: is the value of one monomial in a CNF, namely $b^{e(i)} \cdot c(i)$, [340].

`Cantor_Omega` is $\aleph_1 - \aleph_0$ the set of infinite countable ordinals.

`cantor_pmon p i`: a factor of the cantor product form [357].

`card_dominant x`, says that x is a dominant cardinal, [433].

`card_five`, the cardinal five, successor of 4, [136].

`card_four`, `\4c`, the cardinal four, successor of 3, [101].

`card_max x y`, is the cardinal of the maximum of x, y and ω, [315].

`card_nine`, `\9c`, is 3×3, [136].

`card_nz_fam f`, says that no $f(i)$ is zero.

`card_one`, `\1c`, the cardinal one, [78].

`card_ten`, `\10c`, the cardinal ten, is $5 + 5$, [136].

`card_three`, `\3c`, the cardinal three, successor of 2, [101]

`card_two`, `\2c`, the cardinal two, [78].

`card_zero`, `\0c`, the cardinal zero, [78].

`cardinal x`, Card(x), is some set equipotent to x, [77].

`cardinal_fam x`, says that x is a family of cardinals, [77].

`cardinal_le x y`, `x <=c y`, $x \leq_{\text{card}} y$, says that x and y are two cardinals such that x is equipotent to a subset of y, [83].

`cardinal_lt x y`, `x <c y`, is $x \leq_{\text{card}} y$ and $x \neq y$, [83].

`cardinal_pos_fam g`, says that for all x we have $g(x) > 0$.

`cardinal_prop x y`, explaisn that y is teh cardinal of x, [77].

`cardinal_set x`, says that x is a set whose elements are cardinals, [77].

`cardinalp x`, says that x is of the form card(z), [77].

`cardinals_le a`, `cardinals_lt a`: is the set of cardinals $\leq a$, or $< a$, [83].

`cardinalU U x`, cardinal number, [451].

`cardinalV x`, says that x is a von Neumann cardinal.

`CauchyQ x`, `CauchyR x`, says that x is a Cauchy sequence with terms in **Q** or **R**,[284].

`cdiff a b`, `a -c b`, cardinal difference, dfpdid9.

`cdivides b a`, `x%| c y`, says that $a = bq$ for some q, [127].

`cdivision_prop a b q r`: the relation $a = bq + r$, $0 < r$ for cardinals, [127].

`cdouble n`: is $2n$ as a cardinal, [138].

`chalf n`: is $n/2$ as a cardinal, [138].

`change_target_fun f t`, is the same function as f, with target t.

`char_fun A B`, is the characteristic function of A, as a mapping from B into $\{0,1\}$, [95].

`choiceA`, axiom of choice, [449].

`choose p`, $\mathscr{C}(p)$, is some x such that $p(x)$ is true, the empty set if no x satisfies p.

`chooseT p q`, $\mathscr{C}_{\mathrm{T}}(p,q)$, is our basic axiom of choice.

`class r x`, is the class of x for the equivalence relation r.

`classp r x`, says that x is an equivalence class for r.

`clog2 n`, the base two logaritm of n, [139]

`closed_formula`: a formula without free variables [458].

`closed_interval r x`, says x has the form $[a,b]$.

`cmax x y`, is the greatest of the two cardinals x and y, [84].

`cmin x y`, is the least of the two cardinals x and y, [84].

`cnext c`, is the next cardinal after c, [97].

`CNF_npec px py n m`, auxiliary for cantor product form, [360].

`CNFbv b e c n`, `CNFb_ax b e c n`, CNF of type b, [340].

`CNFbvo e c n`, `CNFb_axo e c n`, CNF of type b, for base ω, [343].

`CNFB_ax b X`, `CNFBv b X`, `CNFB_bound x`, CNF of type B, [344].

`CNF_psi_ax p n`; `CNF_psi_ax2 p n`, `CNF_from_psi p z`, `CNF_psiv p n`, CNF for the Shütte ψ function, [404].

`CNF_simple_ax a n b x`, `the_CNF_simpl x`, [403]

`CNFp_value1 e c`, `CNFp_value2 e c`, `CNFp_ax p n`, `CNFp_ax1 p n`, `CNFpv p n`, `CNFpv1 p n`, `CNFp_ax4 p n x`, functions for the cantor product form, [357],[357], [360].

`CNFq_ax b e c n`, CNF of type q, [340]

`CNFrv f n`, `CNFr_ax f n`, CNF of type r, [336].

`cnf_s f k`, `cnf_m f1 f2 k`, `cnf_c f n x`, `cnf_nr x k`, `cnf_ns x k`, `cnf_nm x y`, `cnf_nr x k`, `cnf_ns x k`, `cnf_nm x y`, `cnf_nc y c`, `cnf_ncms x y k`, `cnf_nck y k`, Various operations on cnfs, [340], [350], [351]. .

`cnf_all_small x y`, says that all exponents of x ate smallet than all exponents of y; [350].

`cnf_and_val x y`, says that x is the cnf of y; [345].

`cnf_bound x`, a bound for the cnf of x, [345].

`cnf_degree x`, the degree of a cnf x, [345].

`cnf_exponents x`, the set of exponents of the cnf x, [347].

`cnf_lc x`, the leading coefficient of a cnf x, [345].

`cnf_le x y`, `cnf_lt x y`, `x <=f y`, `x <f y`; comparison of two cnfs, [348].

`cnf_nat_sum x y`, `x +#f y`, natural sum of two cnfs[383]

`cnf_nat_sum_mons`, aux for natural sum [383]

`cnf_prod_gen x y`, the generic formula for cnf multiplications [355].

`cnf_rangep E`, says that E is the range of a cnf, [347].

`cnf_rem x`, the remainder of a cnf x, [345].

`cnf_shift_expo x d`, adds d to each exponent of x, [354].

`cnf_size x`, one less than the length of the cnf x, [343].

`cnf_sort E`, converts the set E into a cnf, [347].

`cnf_subset E c`, `cnf_subset1 x`, aux for natural sum, [385]

`cnf_sum x y`, `x +f y`, the sum of two cnfs, [349].

`cnf_sum_compat x y`, says that addition of cnfs is compatible with addition of ordinals, [349].

`cnf_sum_monp x y c p`, `cnf_sum_mons x y d`, auxiliary definitions for the cnf sum , [349].

`cnf_val x`, the ordinal value of a cnf x, [345].

`cnfp x`, says that x is a cnf, [345].

`cnfp_nz x`, says that x is a non-zero cnf, [345].

`coarse x`: is $x \times x$.

`coarser x`: is the order on the set of partitions of x associated to `coarser_cs`, [14].

`coarser_cs I J`, `coarser_cg f g`, two definitions that say for all $j \in J$ there is $i \in I$ such that $j \subset i$ or for all j there is i such that $g_j \subset f_i$.

`coarser_preorder` is the order induced by $\subset$ on preorders, [21].

`cofinal r A` says that if x in the substrate of r there is an $y \in A$ such that $x \leq_r y$, [30].

`cofinal_function f x y`, `cofinal_function_ex x y`: says that f is a cofinal function $y \to x$, or that there is such a function, [413].

`cofinal_ordinal x y`, says that every element of x is bounded above by an element of y and vice-versa for the ordinal ordering, [307].

`cofinality x`, defines the cofinality of an ordinal, [413].

`cofinality_c x` defines the cofinality of a cardinal, [409].

`cofinality_aux r`, `cofinality' r`, `cofinality_alt`, are variant of cofinality, [412].

`coinitial r A`: says that if x in the substrate of r there is an $y \in A$ such that $y \leq_r x$, [30].

`collectivising_srel r`, says that the segment $]\leftarrow,x[$ for the order relation r (that may be without graph) is always a set, [318].

`common_extension_order_axiom g`: are the conditions on g for which an order can be put on the union, [52].

`common_extension_order g h`: says that h is an order on the union of the substrate of the g_i that coincides with the restriction, [52].

`common_ordering_set r r'`, aux for Zermelo's theorem.

`common_worder_axiom g`: are the conditions on g for which a well-ordering can be put on the union of the family, [52].

`compatible_with_equiv_p p r`, means that $p(x)$ and $x \overset{r}{\sim} y$ implies $p(y)$.

`compatible_with_equiv f r`, means that $x \overset{r}{\sim} y$ is equivalent to $f(x) = f(y)$.

`compatible_with_equivs f r r'`, means that $x \overset{r}{\sim} y$ is equivalent to $f(x) \overset{r'}{\sim} f(y)$.

`complement a b`, $a - b$, $a \setminus b$, $\complement b$, is the set of element of a not in b.

`composableC f g`, `composable f g`, is the condition on correspondences (resp. functions) f and g for $f \circ g$ to be a correspondence (resp. function).

`compose_graph f g`, $f \circ g$, composition of two graphs.

`compose f g`, `composeC f g`, $f \circ g$, is the composition of two functions.

`composite n` is true when n has a non-trivial divisor, [142].

`comprehensionA_pr x p c`, `comprehensionA`, axiom of comprehension, [447].

`consecutive x y`, means $x < y$ and there is no z such that $x < z < y$, [218].

`constant_graph s x`, is the graph of the constant function with domain s and value x.

`ContHypothesis` is the continuum hypothesis $\aleph_1 = 2^{\aleph_0}$.

`cont_ofn f x`, says that $f(a) = \sup_{t<a} f(t)$ whenever a is a limit ordinal $< x$, [311].

`continuous f`, `continuous2 f` says that f is continuous everywhere (resp. everywhere in each of its two arguments), [286].

`continuous_at f x` says that f is continuous at x, [286].

`continuous_left f x`, `continuous_right x` says that f is continuous at the left or right of x, [287].

`contraction_rec a`, `contraction_rep a`, is $\sum a_i$ where the a_i's are the digits of a in bases b, [135].

`coprime a b` says that the gcd on $\mathbf{N}$ of a and b is trivial, [142].

`correspondence f`: says that f is a triple $(\mathrm{G},\mathrm{A},\mathrm{B})$, with $\mathrm{G} \subset \mathrm{A} \times \mathrm{B}$.

`correspondences A B`: means the set of correspondences $\mathrm{A} \to \mathrm{B}$.

`countable_ordinal x`, says that x is countable and an ordinal,[306].

`countable_infinite x`, says that x is equipotent to a subset of $\mathbf{N}$, [200].

`countable_set x`, says that x is equipotent to $\mathbf{N}$, [200].

`covering f x`, `covering_f I f x`, `covering_s f x`, three variants of a family of sets (defined by f and I) whose union contains x.

`cpow a b`, `a ^c b`, $\mathfrak{a}^{\mathfrak{b}}$, is the cardinal is the set of functions from b into a, [94].

`cpow_less x y`, $x^{<y}$, `a ^<c b`, is the supremum of all x^z, with $z < y$, [428].

`cpow_less_ecb x y`, says that there is $a < x$ such that forall b, $a \le b < c$ implies $a^a = 2^{\mathrm{B}}$, [429].

`cpow_less_ec_prop x y a`, says $a < y$ and if $a \le b < y$ then $x^a = x^b$, [429].

`cpred x`, is cardinal predecessor of x; it is the cardinal y such that $y+1=x$, is x if x is infinite or zero, [102].

`cprod x`, `cprodb a f`, $\prod_{\iota\in I}\mathfrak{a}_\iota$, is the cardinal of the product of the family of sets, [88].

`cprod2 a b`, `a *c b`, $\mathfrak{a}.\mathfrak{b}$ or $\mathfrak{a}\mathfrak{b}$, is the cardinal product of a family of two elements, [90].

`cprod_of_small f x`, says that f is a functional graph, whose domain and values are x, [435].

`cquo a b`, `a %/c b`, is the quotient in the division of a by b [127].

`crem a b`, `a %%c b`, is the remainder in the division of a by b [127].

`critical_ordinal u f y`, says that $f(x,y)=y$ whenever $x<y$, [328].

`csucc x`, is the cardinal successor of x, [80].

`csum x`, $\sum_{\iota\in I}\mathfrak{a}_\iota$, is the cardinal of the disjoint of the family of sets, [88].

`csum2 a b`, `a +b c`, $\mathfrak{a}+\mathfrak{b}$, is the cardinal sum of a family of two elements, [90].

`csum_of_small0 x f`, `csum_of_small1 x f`, says that f is a family of cardinals, that are $\le x$ or $< x$, [409].

`csum_to_increasing_fun y n p`, is the function $[0,p]\to[0,n]$ that maps i to $\sum_{j\le i} y_j$, [163].

`decent_set x`, says that no element y of x satisfies $y\in y$, [66].

`decreasing_fun f r r'`, is a function such that $x\le_r y$ implies $f(x)\ge_{r'} f(y)$, [24].

`decreasing_sequence f r`, says that f is a strictly decreasing function with source $\mathbf{N}$ and target r, [202].

`decreasing_strict_sequence f r`, says that f is a decreasing function with source $\mathbf{N}$ and target r, [202].

`derange n` number of derangements, [165].

`diagonal A`, Δ_A, is the set of all (x,x) such that $x\in A$.

`diagonal_application A`, is the diagonal mapping $x\mapsto(x,x)$ of A into Δ_A.

`diagonal_graphp I E`, is the set of graphs of constant functions from I to E.

`disjoint x y`, means $x\cap y=\emptyset$.

`disjoint_union f`, `disjoint_union_fam f` are two variants of the disjoint union of the family of sets f.

`distributive_lattice1 r` (and 5 other definitions) says that r is a distributive lattice, [42].

`domain f`, is the set of x for which there is an y with $(x,y)\in f$, it is $\mathrm{pr}_1\langle f\rangle$.

`doubleI_E a b`, `doubleI_F a b`, set for double ordinal induction, [386].

`doubleI_restr f a b`, `doubleI_restr2 f a b`, `doubleI_fix1 T f`, `doubleI_fix2 T f`, `doubleI_def A B T f`, `doubleI_tdef2 T`, `doubleI_rg f a b` auxiliary definitions for double ordinal induction, [387], [389]. [390].

`doubleI_transdef A B T`, `doubleI_tdef T a b`, definition by double transfinite induction, [388][388]

`doubleton x y`, $\{x,y\}$, is a set with elements x and y.

`doubletonU a b`, axiom of the pair, [447].

`doubleton_fam f x y`, means that f is a functional graph defined on a doubleton whose range is $\{x,y\}$,[82].

`double_list_prop A L Q`, says that all elements x and y of the list L of type A satisfy the predicate $Q(x,y)$, whenever x comes before y, [697].

`empty_function`, `empty_functionC`, is the identity on $\emptyset$.

`empty_function_tg F`, is the function defined on the empty set with target F, [29].

`emptyset`, $\emptyset$, is a set without elements.

`emptysetA_pr e`, `emptysetU`, axiom of the empty set, [447].

`EP_fun`, `EPperm_???`, `EPpermi_???`, definitions local to the section 9.3.

`epsilon0`, ϵ_0 the first epsilon number, [404]

`epsilonp x`, `epsilon_fct`, `epsilon_fam`: the epsilon-numbers and the property of being an epsilon-number, [393].

`eq_rel_associated f`: is the graph of the equivalence relation $f(x) = f(y)$.

`eqmod B x y`: x and y have the same remainder in the division by B, [135].

`equipotent x y`, means that there is a bijection from x into y.

`equipotent_ex x y`, denotes a bijection from x into y (it exists if x and y are equipotent), [81].

`equipotent_to_subset x y`: means that x is equipotent to a subset of x, [83].

`equipotentU U X Y`, equipotency [451].

`equivalence r`, says that the graph r is an equivalence.

`equivalence_associated f`, is the equivalence relation $f(x) = f(y)$.

`equivalence_associated_o r`, is the equivalence relation "$x \prec_r y$ and $y \prec_r x$" between x and y (when r is a preorder), [16].

`equivalence_corr r`, says that the correspondence r is associated to an equivalence.

`equivalence_r r`, `equivalence_re r x`, says that the relation r is an equivalence relation (in x).

`eta_like r`, `eta_like0 r`, says that r is an order relation satisfying the same properties as the ordering of **Q**, [247].

`euler`, Euler numbers, [169].

`evenp x`, says that x is an even integer, [137].

`exists_unique p`, means that there exists a unique x such that $p(x)$.

`expansion f b k`, says that f is a functional graph, defined for $i < k$ and such that $f_i < b$, [131].

`expansion_ax f n`, says that f is a functional graph, defined for $i < n$ and such that f_i is an ordinal, [301].

`expansion_ext f k` says that f is a functional graph, defined for $i < k$ and such that $f_i \le 2$, [259].

`expansion_ext_of_a f a` says that f is a function graph, $f_i \le 2$, the last f_i is nonzro, $a = \sum f_i 2^i$, [259]

`expansion_normal_of f b k a`, is expansion-of (see below) plus the condition that the last term of the sum is non-zero, [134].

`expansion_of f b k a`, says that f is a functional graph, defined for $i < k$ and such that $f_i < b$, and $\sum f_i b^i = a$, [134].

`expansion_ten f k`, says that f is a functional graph on the interval $[0, k[$, with values in the interval $[0, 9]$, [136].

`expansion_value f b`, assume that f is a functional graph, defined for $i < k$ and such that $f_i < b$; the value is $\sum f_i b^i$, [131].

`expansions_ext_of n` is the set of f satisfying `expansion_ext_of_a f n`, [259].

`exp_boundary f k`, says $k = 0$ or $f(k-1) \neq 0$, [134].

`ext_map_prod I X Y g`, is the function $(x_\iota)_{\iota\in I} \mapsto (g_\iota(x_\iota))_{\iota\in I}$ from $\prod_I X_\iota$ into $\prod_I Y_\iota$.

`ext_to_prod u v`, is the function $(x, y) \mapsto (u(x), v(y))$, sometimes denoted $u \times v$.

`extends g f`, `extendsC g f`, says $g(x) = f(x)$ whenever $f(x)$ is defined.

`extends_in E F`, is the relation `extends` in $\Phi(E,F)$, [13].

`extensional_set x`, says that x is an extensional set, [446].

`extensionalityA`, axiom of extent [447].

`extension_order E F`, is the order associated to `extends_in E F`, [13].

`extension_to_parts f`, denotes the function $x \mapsto f\langle x\rangle$, from $\mathfrak{P}(A)$ into $\mathfrak{P}(B)$.

`factorial n`, $n!$, is the factorial function, [143].

`fam_of_opp g`: is the family of opposite orders of the elements of the family g, [22].

`fam_of_substrates g`: is the family of substrates of the elements of the family g, [22].

`fcompose f g`, $f \circ g$, composition of two graphs, without assumption.

`fcomposable f g`, says that graphs g and $f \circ g$ have the same domain.

`fct_to_list A f n`: is the list of type A containing $f(i)$ for $i < n$, [695]

`fct_sum f n`, `fct_prod f n`, is the sum or product of the values $f(k)$ for $k < n$, computed via `fct_to_list`, [701].

`fg_Mle_lec X`, `fg_Mle_leo X`, `fg_Mlt_lto X`, `fg_Mlt_ltc X`, says that if $i \le j$ or $i < j$, then $X_i \le X_j$ or $X_i < X_j$ (the second comparison is between ordinals or cardinals), [411].

`fgraph f`, says that f is a functional graph.

`fgraph_to_fun f`, the funtion associated to the functional graph f, [54].

`fgraphs E F`, is the set of functional graphs from E to F, [21].

`fgraphs_equipotent x y`, says that each x_i is equipotent to y_i.

`Fib n`, F_n, the n-th Fibonacci number, [141].

`Fib2_rec`, recursive definition of (F_n, F_{n+1}), [141].

`Fib_fusc` auxiliary function that gives the position where the fusc function attains a maximum as a Fibonacci value, [258].

`fincr_prop f r r'`, says that $x \le_r y$ implies $f(x) \le_{r'} f(y)$, [19].

`finer_equivalence s r`, comparison of equivalences, $x \overset{s}{\sim} y$ implies $x \overset{r}{\sim} y$.

`finite_c x`, means that x is a finite cardinal, [80].

`finite_character x`, says that x is of finite character, [110].

`finite_depth x`: says that x is a set of finite depth, [445].

`finite_int_fam f`, says that f is a functional graph, with a finite domain and whose range is a subset of the set of integers, [117].

`finite_o x`, means that x is a finite ordinal, [76].

`finite_set x`, means that x is a finite set, [76].

`first_derivation f`, is the enumeration of the fixed-points of f, [321].

`first_non_int p n`: index of the first non-integer in the list p; with index $< n$, [365].

`first_proj g`, is the function $x \mapsto \mathrm{pr}_1 x$ ($x \in g$).

`first_proj_equiv x y`, `first_proj_equivalence x y`, is the equivalence associated to `first_proj` on the set $x \times y$.

`fiso_prop f r r'`, says that $x \le_r y$ is equivalent to $f(x) \le_{r'} f(y)$, [19].

`fixpoints f`, is the set of all x such that $f(x) = x$, [314].

`formula_len x`, is the length of the formula, [456].

`formula_val`, `formula_values`: value of a formula, the set of possible values, [458].

`formulap x`, says x is a formula, [455].

`formulas_rec`, auxiliary definition for the set of formulas, [455].

`foundation_prop`, `foundation_axiom`, the Axiom of Foundation, [444].

`foundationA`: axiom of foundation, [449].

`free_vars`: the set of free variables of a formula, [458].

`fsimpl_sum p i` is the sum $\sum_{p\le j\le p+i} 1/(s_j s_{j+1})$, where s is the fusc function, [259]

`fsincr_prop f r r'`: says that $x <_r y$ implies $f(x) <_{r'} f(y)$, [19].

`fsiso_prop f r r'`: says that $x <_r y$ is equivalent to $f(x) <_{r'} f(y)$, [19].

`fterm`, `fterm2`, `fterm3`, type of a function that takes 1, 2 or 3 sets as arguments and returns a set.

`fun_image x f`, $f\langle x\rangle$, is the value of f on the set x.

`fun_on_quotient r f`, `function_on_quotient r f b`, `function_on_quotients`, `fun_on_quotients r r' f`, the function obtained from f on passing to the quotient of r (or r and r').

`fun_set_to_prod E X` is the canonical bijection between $(\prod X_\iota)^E$ and $\prod X_\iota^E$.

`function f`, says that f is a function in the sense of Bourbaki.

`function_order E F G`, `function_order_r E F G`, is the order defined on $\mathscr{F}(E;F)$ by $\forall x, f(x) \prec_G g(x)$, [23].

`function_prop f s t`, `function_prop_sub f s t`. This is the property that f is a function from s into t, or into a subset of t.

`functional_graph f`, says that f is a functional graph.

`functions E F`, denoted $\mathscr{F}(E;F)$, is the set of functions $E \to F$.

`functions_incr E F`, `functions_sincr E F`, is the set of (strictly) increasing functions from E to F, [158].

`functions_incr_int p n`, is the set of increasing function $[0,p] \to [0,n]$, [162].

`functions_sum_le E n`, `functions_sum_eq E n`, is the set of functions $f : E \to [0,n]$ such that the sum $\sum f(i)$ is $\le n$ or $= n$, [161].

`funimageU x p f`, image of a set by a function, [447].

`funsetU U X Y Z`, set of functions, [451].

`fusc`, the fusc function, satisfying `fusc_prop`, [254]

`fusc_next F n`, aux function for defining fusc, [254].

`fusc_prop f`: the fusc property, $f_{2n} = f_n$, $f_{2n+1} = f_n + f_{n+1}$, [254]

`fusc_quo n` is s_n/s_{n+1}, where s is the fusc function, [255].

`fusc_quo_inv`, the function F such that $F(x_{2n}) = x_n$ and $F(x_{2n+1}) = x_n$, [254].

`Fusci n a b`: iterative version of the fusc function, [257].

`Fuscj n m`: variant of the fusc function, [263].

`fuscz n` is the fusc function, considered in **Z**, [255].

`Gamma_0`, Γ_0: the least strongly critical ordinal, the Feferman-Schütte ordinal, [400].

`gcompose f g`, $f \circ g$, composition of two graphs, assumes that range g is a subset of domain f.

`GenContHypothesis` is the generalised continuum hypothesis; for all ordinal n, $\aleph_{n+1} = 2^{\aleph_n}$.

gfunctions E F, denoted F^E, is the set of graphs of functions from E to F.

gimel_fct x: is the gimel function, [422].

gle r x y, $x \le y$, says that x and y are related by r, [12].

glt r x y, $x < y$, says $x \le y$ and $x \neq y$, [12].

graph f: is the graph of a function.

graph_of_function X Y: is the function $f \mapsto G_f$ defined on $\Phi(E,F)$, where G_f is the graph of f, [21].

graph_of_partition x, is partition_relset, considered as a function with source partitions x and target $\mathfrak{P}(x \times x)$, see [21].

graph_on r X: is the graph of the relation r restricted to X.

graph_order E F G, graph_order_r E F G: is the order defined on F^E by $\forall x, f(x) \prec_G g(x)$, [23].

graphs_sum_le E n, graphs_sum_eq E n, are the sets of graphs of function $f : E \to [0,n]$ such that the sum $\sum f(i)$ is $\le n$ or $= n$, [161].

graphs_sum_le_int p n, is the set of graphs of function $f : [0,p] \to [0,n]$ such that $\sum f(i) \le n$, [162].

greatest r a, is the property that a is the greatest (unique maximal) element of the substrate of the order r, [27].

greatest_induced r X x, says that x is the greatest of r on X, [31]

greatest_lower_bound r X a, is the property that a is the greatest element of the set of lower bounds of X, [31].

has_greatest r, says that r has a greatest element, [27].

has_infimum r X, says that X has a infimum, [32].

has_inf_graph r f, says that the image of the graph f has an infimum, [35].

has_least r says that r has a least element, [27].

has_supremum r X, says that X has an supremum, [32].

has_sup_graph r f, says that the image of the graph f has a supremum, [35].

hereditarily_finite x: says that x is hereditarily finite, [445].

iclosed_set E, iclosed_collection C says that the set E, ot the collection C is closed (for the toplogy of the ordinals), [316].

iclosed_proper C, says that C is a closed collection, but not a set, [316].

identity A, I_A, is the identity function on the set A.

identity_g A, I_A, is is the graph of the identity function on the set A.

IM stands for the image of a function. Its axioms implement the Axiom Scheme of Replacement.

image_by_fun f A, $f\langle A\rangle$, is $\{t, \exists x \in A, t = f(x)\}$ (renamed to Vfs).

image_by_graph f x, $f\langle A\rangle$, is $\{t, \exists x \in A, (x,t) \in f\}$.

image_of_fun f, is the image of f (renamed to Imf).

Imf f, is the image of f.

in_same_coset f, is the relation "there exists i such that $x \in f(i)$ and $y \in f(i)$" between x and y.

inaccessible x, says that x is an inaccessible cardinal, [435].

inaccessible_w x, says that x is a weakly inaccessible cardinal, [416],

inaccessibleU U x, inaccessible cardinal, [451].

inc x y, $x \in y$, means that x is an element of y.

`inclusionC x y`, I_{xy}, it is the inclusion map on $x \subset y$ as a COQ function.

`increasing_fun f r r'`, is a function such that $x \le_r y$ implies $f(x) \le_{r'} f(y)$, [24].

`increasing_pre f r r'`, says that f is an increasing function for preorders r and r', [516].

`increasing_sequence f r`, says that f is an increasing function with source **N** and target r, [202].

`indecomposable z`, says that z is an indecomposable ordinal, [322].

`induced_order R A`, R_A, is the order induced by R on A [12].

`induced_relation R A`, R_A, is the equivalence induced by R on A.

`induction_defined s a`, is the function f defined on **N** by $f(0) = a$ and $f(n+1) = s(f(n))$, [129].

`induction_defined0 h a`, is the function f defined by $f(0) = a$ and $f(n+1) = h(n, f(n))$, [129].

`induction_defined1 h a p`, is the function f defined for $n < p$ by $f(0) = a$ and $f(n+1) = h(n, f(n))$, [194].

`induction_defined_set s a E`, is the function f with target E defined on **N** by $f(0) = a$ and $f(n+1) = s(f(n))$, [194].

`induction_defined_set0 h a E`, is the function f with target E defined by $f(0) = a$ and $f(n+1) = h(n, f(n))$, [194].

`induction_defined_set1 h a p E`, is the function f with target E defined for $n < p$ by $f(0) = a$ and $f(n+1) = h(n, f(n))$, [194].

`induction_term s a`, is the term f defined by $f(0) = a$ and $f(n+1) = s(f(n))$, [130].

`inductive r`, means that r is an order whose substrate is inductive, [58].

`inf r x y`, $\inf(x, y)$, is the greatest lower bound of pair $\{x, y\}$ (if it exists), [32].

`inf_fun r f`, $\inf_{x\in A} f(x)$, is the greatest lower bound of the image of the function f (if it exists), [35].

`inf_funp r f x`, says that x is the infimum of the image of the function f for the order r, [34].

`inf_graph r f`, $\inf_{x\in A} f(x)$, is the greatest lower bound of the image of the graph f (if it exists), [35].

`inf_graphp r f x`, says that x is the infimum of the image of the graph f for the order r, [34].

`infimum r X`, $\inf_E X$, is the greatest lower bound of X (if it exists), [32].

`infinite_c x`, means that x is a not a finite set, [80].

`infinite_o x`, means that x is equipotent to x^+, [76].

`infinite_set x`, means that x is a not a finite set, [80].

`infiniteA`, axiom of the infinite set, [449].

`injections E F`, is the set of injective functions from E into F, [144].

`injective f`, `injectiveC f`, means that f is an injection.

`intersection X`, $\bigcap X$, is the intersection of a set of sets.

`intersection2 X Y`, `X \cap Y`, $X \cap Y$, is the intersection of two sets.

`intersectiont I f`, `intersectionf x f`, `intersectiont g`, $\bigcap_{\iota\in I} X_\iota$ is the set of elements a such that for all $\iota \in I$ we have $a \in X_\iota$.

`intersectionU2 a b`, intersection of two sets, [449].

intersection_covering, intersection of coverings.

interval r x, says that x is an interval.

interval_oo r a b, interval_oc r a b, interval_ou r a, interval_co r a b, interval_cc r a b, interval_cu r a, interval_uo r b, interval_uc r b interval_uu r: Intervals, [44].

intp x, means $x \in \mathbf{Z}$, [206].

inv_graph_canon G, is the bijection $(x, y) \mapsto (y, x)$ from G to G^{-1}.

inv_image_by_fun r x, $\overset{-1}{f}\langle x\rangle$, renamed to Vfi.

inv_image_relation f r, is the inverse image of the relation r under the function f.

inv_psi_omega x: the (a, b) such that $\psi(a, b) = \omega^x$, [402].

inverse_direct_value f X, X_f, is $f^{-1}\langle f\langle X\rangle\rangle$.

inverse_epsilon x, the y such that $\epsilon_y = x$, [397].

inverse_graph G, $\overset{-1}{G}$, inverse graph of the graph G.

inverse_fun f, inverseC a b f H, $\overset{-1}{f}$, inverse of the function f.

inverse_image x f, $\overset{-1}{f}\langle x\rangle$, is the inverse value of f on the set x.

irrationalp x says that x is irrational (as a real Dedekind number), [266].

is_left_inverse r f, means that r is a retraction or left-inverse of f, and $r \circ f$ is the identity.

is_natural_sum f: says that f is a natural sum for ordinals, [389].

is_right_inverse s f, means that s is a section or right-inverse of f, and $f \circ s$ is the identity.

J x y, or (x, y), is an ordered pair, formed of two items x and y.

L X f, fcreate X f, $\mathscr{L}_X f$ is the graph formed of all $(x, f(x))$ with $x \in X$.

largest_partition x, is the set of all singletons of x.

lattice r, is a relation for which $\sup(x, y)$ and $\inf(x, y)$ exist, [40].

least r a, is the property that a is the least (unique minimal) element of the substrate of the order r, [27].

least_fixedpoint_ge f x y, says that y the least fixed-point of f that is $\geq x$, [314].

least_induced r X x, says that x is the least of r on X, [31]

least_non_trivial_dominant, is the least non-trivial (i.e. $> \aleph_0$) dominant cardinal.

least_ordinal p x, is the least ordinal that satisfies p, provided that $p(x)$ holds, [68].

least_upper_bound r X a, is the property that a is the least element of the set of upper bounds of X, [31].

left_directed r, means that each doubleton is bounded below, [39].

left_unbounded_interval r x, says that x has the form $]\leftarrow, b[$ or $]\leftarrow, b]$.

left_inverseC, left inverse of a Coq function.

length_smaller_formulas n: set of formulas smaller than n, [457].

Lf f A B, $\mathscr{L}_{A;B} f$, is function from A to B whose graph is $\mathscr{L}_A f$.

Lfs f s, variant of Lf, where the target is the image of the source s.

`lf_axiom f A B`, says that for all $x \in \mathrm{A}$ we have $f(x) \in \mathrm{B}$, case where $\mathscr{L}_{\mathrm{A};\mathrm{B}} f$ is a function.

`Lg X f`, $\mathscr{L}_{\mathrm{X}} f$, is the graph formed of all $(x, f(x))$ with $x \in \mathrm{X}$.

`limit_ordinal x`, is a non-zero ordinal that is not a successor, [75].

`limitQ x v`, `limitR x v`, says that the sequence x converges to the numbers x (real and rational case), [284].

`list_range L`, is the smallest set containing all elements of the list, [698].

`list_subset L E`, says that all elements of the list L belong to the set E, [698]

`list_sum L`, `list_prod L`, is the sum or product of the element of the list L [701].

`list_to_fct L`, `list_to_f L`, `list_to_fctB L`, `list_to_fB L E`, converts the list L into a mapping $\mathbb{N} \to \mathbb{N}$, or a function with source $[0, n[$ and target $\mathbf{N}$ or E.[695], [696].

`lower_bound r X x`, says that for all $y \in \mathrm{X}$, we have $x \leq_r y$, [30].

`LS_nextP X P`, `LS_nextP_aux X P` used by `LS_res`, [463].

`LS_rec X P`, `LS_res X P` the set appearing in the conclusion of the Löwenheim-Skolem theorem, [463].

`lu_interval r x` see interval.

`Lvariant a b x y`, `variant a x y`, `Lvariantc x y`, these are functions whose range is the doubleton $\{x, y\}$.

`many_der f E`, iterated derivation of a function f on a set E, [333].

`maximal r a`, says that $x \leq_r a$ implies $x = a$, [27].

`maximal_critical a x`, a property of Schütte ϕ, [401].

`merge_int n m`, is a bijection $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$, [197].

`minimal r a`, says that $x \geq_r a$ implies $x = a$, [27].

`monotone_fun f r r'`, says that f is an increasing or decreasing function for r and r', [24].

`monotone_order_fam g`: says that g is a family of orders on E_i such that if $\mathrm{E}_\mathrm{I} \subset \mathrm{E}_j$, then g_j extends g_i, [52].

`multiple_interpolation f x`, the value at x (rational and ≥ 0) of the function g, which is linear on each interval $[n, n+1]$ and takes the value $f(n)$ at n, [252]

`multiple_interpolation2 f x`, the value at x (rational, positive) of the function g, which is linear on each interval $[1/(n+1), 1/n]$ and take the value $f(n)$ at $1/(n+1)$, [253]

`mutually_disjoint f`, says that for all distinct i and j, $f(i)$ and $f(j)$ are disjoint.

`mutually_cofinal x y`, says x and y are sets of ordinals such each element of one set is bounded by an element of the other set, [307].

`nat`, $\mathbb{N}$, is the type of natural integers in COQ.

`Nat`, $\mathbf{N}$, is the set of natural integers of Bourbaki, [101].

`nat_factor_listp nx`, says that $(p(i_{i<n}$ is a decreasing list, of prime integers, [362].

`Nat_le x y`, `Nat_lt x y` (or `x <=N y`, `x <=N y`), are the relations $x \leq y$ or $x < y$ on $\mathbf{N}$, [105].

`Nat_order`, is the ordering on $\mathbf{N}$, [105].

`nat_prime x`, says that x is a prime integer, [362].

`nat_to_B`, $\mathscr{N}$ is the canonical bijection from `nat` to the set of finite cardinals, [684].

`natp x`, says that $x \in \mathbf{N}$.

`Nats`, $\mathbf{N}^*$, is the set of non-zero natural integers, [206].

`natR n`, maps a natural number onto a pseudo-ordinal, [682].

`natural_sum x y`, `x +#o y`, $x\natural y$, is the natural sum of two ordinals [384].

`Nbexp n` the number of ways to write n as a sum of powers of two, each power is used at most twice, [259].

`Ncoprime a b` says that the gcd on $\mathbf{N}$ of a and b is trivial, [227].

`next_dominant x`, is the next dominant cardinal after x, [433].

`next_formulas F`, auxiliary definition for the set of formulas [455].

`Ngcd n m` the gcd on $\mathbf{N}$, [227].

`Ngcd_prop a b p` the characteristic property of the gcd on $\mathbf{N}$, [227].

`Nint a`, `Nintcc a b`, `Nintc a`, `Nint1c a`, are the intervals $[0,a[$, $[a,b]$, $[0,a]$ and $[1,a]$ on $\mathbf{N}$, [122].

`Nint_cco a b`, is the ordering of the interval $[a,b]$, [122].

`Nint_co a`, is the ordering of the interval $[0,a[$, [124].

`non_coll p`, says that there is no set E such that $p(x)$ is equivalent to $x \in \mathrm{E}$.

`normal_function f x y`, says that f is a normal function $x \to y$, [311].

`normal_ofu_aux f u`, `normal_of_aux f` , says that for every limit ordinal a, $f(a) = \lim_{t<a} f(t)$k (where $t < a$ (in the first definition $u < a$ and $t < a$), [311].

`normal_ofs f`, `normal_ofs1 f u`, `normal_ofs2 f u`, says that f is a normal functional symbol defined for $\geq u$, [311].

`nth_more K S`, `nth_elts K`, `nth_elt K`, `nth_set_fct`, enumeration of a set of integers, [146].

`number_of_injections b a` is $a!/(a-b)!$, [144].

`ocoef x i`, the i-th coefficient of the cnf x, [343].

`ocomparable r x y` says that x and y are comparable for the order r, [41].

`oddp x`, says that x is an odd integer, [137].

`odegree x`, the degree of the cnf of x, [345].

`oleading_coef x`, the leading coefficient of the cnf of x, [355].

`ovaluation x`, the leadt exponent of the cnf of x, [346]

`odiff a b`, `x -o y`, is the difference between two ordinals, [308].

`odiv_pr0 a b q r`, says that $a = bq + r$, $r < b$, [309].

`odiv_pr1 a b c q r`, says that $a = bq + r$, $r < b$ and $q < c$, [309].

`oexp x i`, the i-th exponent of the cnf x, [343]

`ofact_list_succ p n` says that if $i < n$, $p(i)$ is a prime successor; [365].

`ofg_Mlt_lto X`, `ofg_Mle_leo X`, says that X is a (striclty) increasing family of ordinals.

`ofs f`, `ofsu f u`, says that $f(x)$ is an ordinal, whenever x is an ordinal (resp, x is an ordinal $x \geq u$), [310].

`OIax???`, various assumptions for ordinal induction, [325].

`olog x`: the y such that $x = \omega^y$, [402].

`ole_on x`: the restriction of $\leq_{\mathrm{ord}}$ on x, [72].

`omax x y`, is the greatest of the two ordinals x and y, [72].

`omega0`, `omega1`, `omega2`, are the first three initial ordinals, [81], [406].

`omega_fct x`, `\aleph x`, is the initial ordinal (cardinal) of index x, [406].

`omin x y`, is the least of the two ordinals x and y, [72].

`omonomp m`, says that m is a monomial of a cnf, [345].

On, denotes the class of all ordinals.

`oopow a`, is ω^a, [331].

`open_interval r x`, says that x is an interval $]a,b[$.

`opow a b`, `a ^o bb`, is the ordinal power, [329].

`opp_order r`, is inverse graph of r, [12].

`opposite_relation r`, is the relation $r(y,x)$ between x and y, [12].

`opred x`, is the ordinal Predecessor of x, [74].

`oprod r g`, is the ordinal of the product of a family g whose index set is ordered by r, [296].

`oprod2 x y`, `x *o y`, is the ordinal of the product two ordinals, [296].

`oprod_comm`, `oprod2_comm_P4` , `oprod2_comm_P1`, are helper definition for the sudy of $x \cdot y = y \cdot x$.

`oprod_expansion f n`, defines an ordinal product indexed by an interval $[0,n[$, [301].

`oprdf f n`, is the ordinal product of the $f(i)$ for $i < n$, [301].

`oquo a b`, is the quotient of the ordinal division of a by b, [309].

`oquorem a b`, is the quotient and remainder of the ordinal division of a by b, [309].

`or_complete r` says that r is an order relation for which every element has a successor and a predecessor, [219].

`or_connected r` says that only trivial subsets of the substrate of r are stable by successor and predecessor, [219].

`or_cut r B` says that B is a Dedekind cut for the order r, [265].

`or_cuts r` is the set of cuts for the order r, [265],

`or_cut_order r` is the order of the set of cuts for the order r, [265].

`or_likeZ r` says that the order r is complete and connected, [219].

`or_pred x` the predecessor of x for the order r, [218].

`or_stable r E` says that E is stable for the successor and predecessor defined by the order relation r, [219].

`or_succ r x` the successor of x for the order r, [218].

`ord_below f n`, says that $f(i)$ is an ordinal for $i < n$, [301].

`ord_cofinal a b`, says that a is a cofinal subset of b for the ordinal ordering, [307].

`ord_ext_div_pr a b x y z`, says that $b = a^x \cdot y + z$ with $z < a^x$, $0 < y < a$, [331].

`ord_factor_list p n`, says that $(p_i)_{i<n}$ is a sorted list of prime ordinals, ordinals,ordpreimele

`ord_index x`, is y such that $x = \aleph_y$, [407].

`ord_induction???`, various definitions for ordina induction definition, [324].

`ord_negl x y`, `x <<o y`, $x \ll y$, says that $x + y = y$, [337].

`ord_one`, `\1o` is the ordinal one, [78].

`ord_pair_le a b` is the canonical ordering of pairs of ordinals, [112].

`ord_prime x`, `ord_sprime x` says that x is a prime ordinal, [362].

`ord_prime_le x y`, a compatison for prime ordinals,ordpreimele

`ord_ptypeF a`, `ord_ptypeT a`, `ord_ptypeL a`, a classification of prime ordinals, [364].

`ord_sup_pr E x` says that x is the least ordinal such that $y \le x$, whenever $y \in E$, [74].

`ord_prime_le x y`, a compatison for prime ordinals,ordpreimele

`ord_two`, \2o, is the ordinal two, [78].

`ord_zero`, \0o, is the ordinal zero [78], [74].

`order r`, says that the graph r is an order, [12].

`order_associated r`, is the order associated to a preorder by passing on the quotient of "$x \prec y$ and $y \prec x$", [16].

`order_axioms r s`, is the condition on which r is an order on s, [18].

`order_extends r r'` says that r is the ordering induced by r' on the substrate of r, [52].

`order_fam f`, says that f is a family of ordered sets, [22].

`order_function x y r`, is the order defined by `order_function_r x y r f g`, [23].

`order_function_r x y r f g`, says that f and g are functions $x \to y$ such that, for all t, $f(t) \le g(t)$, [23].

`order_graph`, `order_graph_r`, like `order_function` but for graphs of functions instead of functions [23].

`order_isomorphic r r'`,`x \Is y`, says that there is an order-isomorphism f from $\mathrm{S}(r) \to \mathrm{S}(r')$, [19].

`order_isomorphism f r r'`, says that f is a strictly increasing bijection $\mathrm{S}(r) \to \mathrm{S}(r')$, [19].

`order_le r r'`,`r <=O r'`, says $r \le r'$ when r and r' are orderings, [70].

`order_morphism f r r'`, says that f is a strictly increasing function $\mathrm{S}(r) \to \mathrm{S}(r')$, [19].

`order_on r E`, says that the graph r is an order on E.

`order_prod r g`, is the lexicographic order of the family g whose domain is the well-ordered set r, [294].

`order_prod_ax r g`, assumptions for the lexicographic product, [294].

`order_prod_r r g x y`, is the order relation between x and y associated to `order_prod r g`,[294].

`order_product g`, `order_product_r g`, is the product order of the family g, [22].

`order_product2 f g`, is the product of two orders, [23].

`order_prod2 r r'`, $\mathrm{E}_1 \cdot \mathrm{E}_2$, is ordinal product of two sets. [63].

`order_r r`, says that the relation r is an order, [12].

`order_re r x`, says that the relation r is an order on x, [12].

`order_sum_r r g x x'`, is the order relation between x and y associated to `order_sum r g`, [62].

`order_sum r g`, $\sum_{\iota\in\mathrm{I}} \mathrm{X}_\iota$, is the ordinal sum of the family of orders $g(i)$, the index set being ordered by r, [62].

`order_sum2 r r'`, $\mathrm{E}_1 + \mathrm{E}_2$, is ordinal sum of two sets. [63].

`order_type x` the order type of x, [466].

`order_type_fam g` says that g is a family of order types, [467].

`order_type_le x y`,`x <=t y`, comparison for order types x, [466].

`order_with_greatest r a`, is the order obtained from r by adjoining a greatest element a, [30].

`ordinal x`, is some ordinal y such that $o(y)$ is order-isomorphic to x, [70].

`ordinal_alt`, alternate definition of an ordinal, assuming the Axiom of Foundation, [444].

`ordinal_fam f`, says that f is a family of ordinals, [296].

`ordinal_interval a b`, is the set of ordinals x such that $a \leq x < b$, [310].

`ordinal_iso r`, `ordinal_isog r`, is the isomorphism (resp. its graph) between a well-order r and its ordinal, [56].

`ordinal_le r r'`, `x <=o y`, denoted $x \leq_{\text{ord}} y$, is the ordering on ordinals [72].

`ordinal_leD1 r r'`, `ordinal_ltD1 r r'`, variant of ordinal comparison, [71].

`ordinal_lt r r'`, `x <o y`, denoted $x <_{\text{ord}} y$, is the strict ordering on ordinals [72].

`ordinal_o E`, `ordinal_oa E`, (denoted $o(\mathrm{E})$ and $o'(\mathrm{E})$) are the relations $x \subset y$ or "$x \in y$ or $x = y$ defined on E; they coincide if E is an ordinal, [66].

`ordinal_pair x`, says that x is a pair of ordinals, [112].

`ordinal_prop p`, says that x is an ordinal, whenever $p(x)$ holds, [316].

`ordinal_ub E x`, says that x is an ordinal upper bound of E, [74].

`ordinalp x`, says that x is an ordinal, [67].

`ordinalr r x`, is the ordinal of $]\leftarrow, x[$ for the well-order relation r (that may be without graph), [318].

`ordinals r a`, is the x such that `ordinalr r x` is a, [318].

`ordinals_card_le y`, is the set of ordinals whose cardinal is `ordinalU U X`, ordinal number [451].

`ordinals_card_le y`, is the set of ordinals whose cardinal is $\leq y$.

`ordinalsE E B`, is the function $\mathrm{E} \to \mathrm{E}$ that maps a to `ordinals r a`, where r is the restriction of ordinal comparison to B, [320].

`ordinalsf p`, is `ordinals r`, where r is the restriction of ordinal comparison to ordinals satisfying property p, [320].

`orem a b`, is the remainder of the ordinal division of a by b, [309].

`orprod_ax r g`, are the conditions for the lexicographic order product to exist, [294].

`orsum_ax r g`, are the conditions for the order sum to exist, [62].

`osucc x`, is the ordinal successor of x, [66].

`osuccp x`, says that x is an ordinal successor, [75].

`osum r g`, is the ordinal of the ordinal sum of a family g whose index set is ordered by r, [296].

`osum2 a b`, `a +o b`, is the sum of two ordinals, [296].

`osum_expansion f n`, defines an ordinal sum indexed by an interval $[0, n[$, [301].

`osumf f n`, is the ordinal sum of the $f(i)$ for $i < n$, [301].

`osups X`, strict upper bound of the set of ordinals X, [389].

`otrans_def a T f`, `otrans_defined a`, definition by trandfinite induction on an ordinal a, [387]

`OT_opposite x` is the opposite of the order type x, [469].

`OT_prod r g` is the product of family of order types g over r , [467].

`OT_prod2 a b`, `a *t b`, is the product of two order types, [467].

`OT_sum r g` is the sum of family of order types g over r, [467].

`OT_sum2 a b`, `a +t b`, is the product of two order types, [467].

`P z`, $\mathrm{pr}_1 z$ denotes x if z is the pair (x, y).

`pairp x`, says that x is an ordered pair.

`pairpA_pr a b p`, `pairA`, axiom of the pair, [447].

`part_fun Y X`, is the canonical injection from Y into $\mathfrak{P}(X)$, (if Y is a partition of X then $Y \in \mathfrak{P}(\mathfrak{P}(X))$) [14].

`partial_fun f x m`, a restriction.

`partition y x`, `partition_s y x`, `partition_fam f x`, thee variants that say that y or f is a partition of x.

`partition_relation f x`, is the equivalence relation associated to the partition graph(f) of x.

`partition_relset y x`, is the equivalence associated to the partition y of x, [15].

`partition_with_complement X A`, is the partition of X formed of A and its complementary set.

`partition_with_pi_elements p E f`, says that the sets $f(i)$ are of cardinal p_i, mutually disjoint and form a covering of E, [148].

`partitions` is the set of all partitions of X, [14].

`partitions_pi p E`, is the set of all partitions X_i of E, where each X_i has p_i elements, [148].

`pclosed x`, a formula with parameters without free variables,[461].

`permutations E`, is the set of bijections $E \to E$, [144].

`pformula x`, `pformula_in X x`, `pformulas_in X`, `pcformulas_in X`, a formula with parameters, [461].

`pformula_in_val X P`, the formulas in P valid on X, [462].

`pformula_valid_on X x`, a formula valid on X, [462].

`pformula_vals X x`, value of a formula, [462].

`pfree_vars x`, the free variables of a formula with parameters, [461]

`pmonomp m`, says that m is a monomial of a cnf with positive exponent , [357]

`position_in_cnf x d`, the position of an exponent d with regards to a cnf x, [352]

`posnatp x`, says that x is a non-zero integers, [340].

`pow x y`, x^y, is the power function on the type `nat`, [685].

`powerset x`, $\mathfrak{P}(x)$, is the set of subsets of x.

`powersetA_pr x o`, `powersetA`, `powersetU x`, axiom of the powerset, [447].

`powerU U x y`, cardinal power [451].

$\mathrm{pr}_1 z$, $\mathrm{pr}_2 z$ stand for `pr1 z` and `pr2 z`. These are also denoted by P and Q. If z is the pair (x, y), these functions return x and y respectively.

`pr_i f i`, $\mathrm{pr}_i f$, denotes a component of an element of a product.

`pr_j f J`, $\mathrm{pr}_J f$, is the function $(x_\iota)_{\iota\in I} \mapsto (x_\iota)_{\iota\in J}$.

predecessor of x: is the greatest y such that $y < x$; In the case of ordinals is the union, on the case of cardinals is `cpred`.

`preorder r`, is a reflexive and transitive graph, [16].

`preorder_on r E`, says that r is a preorder on E, [16].

`preorder_r`, is a reflexive and transitive relation, [16].

`preorders E`, is the set of preorders on E, [21].

`prod_assoc_map`, is the function whose bijectivity is the “theorem of associativity of products”.

`prod_of_function u v`, is the function $x \mapsto (u(x), v(x))$.

`prod_of_products_canon F F'`, is the bijection between $\prod F_\iota \times \prod F'_\iota$ and $\prod (F_\iota \times F'_\iota)$.

`prod_of_relation R R'`, $R \times R'$, is the product of two equivalences.

`prod_of_substrates g`, is the cartesian product of the family of substrates of the elements of the family g, [22].

`product A B`, $A \times B$, is the set of all pairs (a, b) with $a \in A$ and $b \in B$.

`productb g`, `productf I f`, $\prod_{\iota \in I} X_\iota$ is the product of a family of sets.

`product1 x a`, is the product of the family defined on the singleton $\{a\}$ with value x.

`product1_canon x a`, is the canonical application from x into `product1 x a`.

`product2 x y`, is the product of the family defined on the doubleton $\{a, b\}$ with values x and y.

`product2_canon x y`, is the canonical application from $x \times y$ into `product2 x y`.

`product_compose`, auxiliary function used for change of variables in a product.

`product_order f g`, `product_order_r f g`, is the order on the product $\prod X_i$ induced by Γ_i (where f defines the family X_i and g defined the family Γ_i), [23].

`Q z`, $\mathrm{pr}_2 z$ denotes y if z is the pair (x, y).

`quasi_bij f I J`, auxiliary definition used to prove commutativity of cardinal addition and multiplication.

`qsum f n` is $\sum_{i<n} f(i)$ in $\mathbf{Q}$, [246].

`quotient R`, E/R, is the set of equivalence classes of R.

`quotient_of_relations r s`, R/S, is the quotient of two equivalences.

`quotient_order_r r s`, is the preorder relation induced in the quotient E/S by the preorder $\prec_R$, where E is the common substrate of R and S, [516].

`range f`, is the set of y for which there is an x with $(x, y) \in f$, it is $\mathrm{pr}_2\langle f \rangle$.

`rat_below f n` says $f(i) \in \mathbf{Q}$ for $i < n$, [246].

`rat_iterator x` is the function $f(x)$ such that $f(x_n) = x_{n+1}$ where $x_n = s_n/s_{n+1}$, s_n is the fusc function, [264].

`rationalp x` says that x is a rational (as a Dedekind real number), [266].

`ratp x` says that $x \in \mathbf{Q}$, [231].

`real_dedekind` says that x is a real Dedekind number, [266].

`realp x` says that $x \in \mathbf{R}$, [267].

`rec_finite x`: says that x is recursively finite, [445].

`regular_cardinal x`, says that x is a regular cardinal, [409].

`regular_ordinal x`, says that x is a regular ordinal, [414].

`reflexive_r r x`, says that the relation r is reflexive in x.

`reflexive_rr r`, says that the relation r is reflexive, [12].

`reflexivep r`, says that the graph r is reflexive.

`rel_strong_card A B`, says that A is B-strong, [434].

`related r x y`, is a short-hand for $(x, y) \in r$.

`relation_on_quotient p r`, is the relation induced by $p(x)$ on passing to the quotient (with respect to x) with respect to R.

`rep x`, is an element y such that $y \in x$, whenever x is not empty.

`replacementA_pr x p f r`, `replacementA` axiom of replacement, [447]

`representative_system s f x`, means that, for all i, $s \cap X_i$ is a singleton, where X_i is a partition of x associated to the function f.

`representative_system_function g f x`, means that g is an injection whose image is a system of representatives (see definition above).

`rept_union x`, repeated union; [445].

`restr x G`, is the restriction to x of the graph G.

`rest_plus_interval a b`, `rest_minus_interval a b` are the functions $x \mapsto x + b$ and $x \mapsto x - b$ as bijections between $[0, a]$ and $[b, a+b]$, [124].

`restricted_eq E`, is the relation "$x \in E$ and $y \in E$ and $x = y$".

`restriction_function f x`, is like `restr`, but f and the restrictions are functions.

`restriction_product f j`, is the product of the restrictions of $\prod f$ to J.

`restriction_to_image f`, is the restriction of the function f to its range, [83].

`restriction_to_segment r x g`, $g^{(x)}$, is the restriction of g to the segment S_x defined by the order r, [54]

`restriction_to_segment_axiom r x g`, is the property for `restriction_to_segment` to be well-behaved, [54].

`restriction2_axioms f x y`, is the condition: f is a function whose source contains x, whose target contains y, moreover $a \in x$ implies $f(a) \in y$.

`restriction2 f x y`, `restriction2C f x y`, restriction of f as a function $x \to y$.

`restrictionC f H`, is the restriction to x of the function $f : a \to b$, where H proves $x \subset a$ implicitly.

retraction: see `is_left_inverse`.

`right_directed r`, `right_directed_prop r`, means that each doubleton is bounded above, [39].

`right_inverseC`, right inverse of a Coq function.

`right_unbounded_interval r x`, says that x has the form $[a, \to[$ or $]a, \to[$.

`ru_interval r x`, see interval.

`rmin x y` the smallest of x and y as a real number, [281].

`Ro x` or $\mathscr{R}x$ converts its argument x of type u to a set, which is an element of u.

`same_below e e' n` says $e(i) = e'(i)$ for $i < n$, [246].

`saturated r x`, means: for every $y \in x$, the class of x for the relation r is a subset of x.

`saturation_of r x`, is the saturation of x for r.

`Schutte_Cr a b`, aux for Schütte Φ; [399].

`second_proj g`, is the function $x \mapsto \mathrm{pr}_2 x$ ($x \in g$).

section: see `is_right_inverse`.

`section_canon_proj R`, is the function from E/R into E induced by `rep`.

`segment r x`, S_x, is the interval $]\leftarrow, x[$ for the order r, [50].

`segmentc r x`, is the interval $]\leftarrow, x]$, [50].

`segmentp r s`, says that s is the interval $]\leftarrow, x[$ or the whole substrate of a well ordered relation r, [50].

`segmentr r x`, is the segment $]\leftarrow, x[$ for the order relation r (that may be without graph), [318].

`segments r`, `segmentss r`, is the set of all segments of an ordered set (with possible exclusion of the whole set), [51].

`segments_iso r`, the isomorphism that maps x to the segment defined by x; [51].

`segment_nat K S`, enumeration of a set of integers, [146] `semi_open_interval r x`, says that x is $[a, b[$ or $]b, a]$.

`Set` is the type of sets.

`set_of_finite_subsets x`, is the set of finite subset sof x, [595].

`setofU x p`, axiom of comprehension, [447].

`sgraph f`, says that f is a set of pairs.

`similar_seq x y` says that the sequence $x_n - y_n$ converges to zero, [285].

`simple_interpolation n n u v`, the function $z \mapsto az+b$ whose value at n or $n+1$ is u and v respectively, [252].

`simple_interpolation2 i u v`, the function $z \mapsto az+b$ whose value at $1/(i+1)$ or $1/i$ is u and v respectively, [253].

`sincr_ofn f x`, says $f(a) < f(b)$ whenever $a < b$, all quantities being ordinals, a and b are in x, [310].

`sincr_ofs f`, says $f(x) < f(y)$ whenever $x < y$, all quantities being ordinals, [308].

`sincr_ofsu f u`, says $f(x) < f(y)$ whenever $u \leq x < y$, all quantities being ordinals, [310].

`single_list_prop A L Q`, says that all elements of the list L of type A satisfy the predicate Q, [697].

`singleton x`, $\{x\}$, is a set with one element.

`singletonp x`, means that x is a singleton.

`singular_cardinal x`, says that x is a singular cardinal.

`singular_ordinal x`, says that x is a singular ordinal, [414]

`small_formulas n`: formulas smaller than n, [457].

`small_set x`, means that x hast at most one element.

`smallest_partition x`, is the singleton $\{x\}$.

`source f`, contains (resp. is equal to) the domain of the graph of a correspondence f (resp. function f).

`Sphi x y`, $\Phi(x, y)$, the Schûtte function;, [398].

`Spsi x y`, $\Psi(x, y)$, `Spsip p`, the Schûtte function;, [402].

`stationary_sequence f`, says that the restriction of f to some interval $[n, \rightarrow[$ is constant, [202].

`stirling1`, `stirling2`, stirling numbers of the first and second class, [167].

`stratified_set`, `stratified_setr`, `stratified_fct`, proof by stratified induction, [196].

`strict_decreasing_fun f r r'`, is a function such that $x <_r y$ implies $f(x) >_{r'} f(y)$, [24].

`strict_increasing_fun f r r'`, is a function such that $x <_r y$ implies $f(x) <_{r'} f(y)$, [24].

`strict_monotone_fun f r r'`, is a strictly increasing or strictly decreasing function, [24].

`strong_critical`, [400].

`ssub x y`, $x \subsetneq y$, means $x \subset y$ and $x \neq y$.

`sub x y`, $x \subset y$, means that x is a subset of y.

`sub_fgraphs A B` is the set of all functional graphs f, whose domain is a subset of A, and whose range is a subset of B.

`sub_functions E F`, denoted $\Phi(E;F)$, is the set of triples (G,A,F) associated to functions from $A \subset E$ into F.

`sub_order A`, is the order induced by $\subset$ on A, [13].

`subp_order A`, is the order induced by $\subset$ on $\mathfrak{P}(A)$, [13].

`substrate r`: this is E, if r is an order on E.

`subsets_with_p_elements p E`, is the set of subsets of E having p elements, [155].

`subU U x y`, set inclusion [451].

successor of x: is the least y such that $x < y$; In the case of ordinals is called `osucc`, in the case of cardinals is `csucc`.

`sum_diag_pascal2 n` is the sum of the binomial coefficients $\binom{i}{j}$, taken module 2, where $i + j = n$, [262].

`sum_of_substrates g`, is the disjoint union of the substrates of the family g, [62].

`sup r x y`, $\sup(x, y)$, is the least upper bound of the pair $\{x, y\}$ (if it exists), [32].

`sup_fun r f`, $\sup_{x\in A} f(x)$, is the least upper bound of the image of the function f (if it exists), [35].

`sup_funp r f x`, says that x is the supremum of the image of the function f for the order r, [34].

`sup_graph r f`, $\sup_{x\in A} f(x)$, is the least upper bound of the image of the graph f (if it exists), [35].

`sup_graphp r f x`, says that x is the supremum of the image of the graph f for the order r, [34].

`supremum r X`, $\sup_E X$, is the least upper bound of X (if it exists), [32].

`surjective f`, `surjectiveC f`, means that f is a surjection.

`symmetric_r r`, says that the relation r is symmetric.

`symmetricp r`, says that the graph r is symmetric.

`symmetrize r`, is the symmetric relation "$r(x, y)$ and $r(y, x)$", [16].

`Szeta`, is the Schütte zeta function, [401].

`target f`, contains the range of the graph of a correspondence or function f.

`the_contraction a` is the sum of the terms of the expansion of a, [134].

`the_expansion a` is the unique normalized base b expansion of a, [134].

`the_CNFB b x`, the CNF of type B of x, [344].

`the_cnf x`, the cnf of x, [345].

`the_cnf_lc x`, leading coefficient of the the cnf of x, [345].

`the_cnf_rem x`, remainder of the cnf of x, [345].

`the_greatest r`, denotes the greatest element of the ordering r, [70].

`the_least r`, denotes the least element of the ordering r, [70].

`the_least_fixedpoint_ge f x`, is the least fixed-point of f that is $\geq x$, [314].

`the_nondominant_least A B`, a property of strong cardinals, [434].

`the_ordinal_iso r`, is the order isomorphism between $o(\mathrm{ord}(r))$ and r for a well-order r, [70].

`total_order r`, means that r is a total order, [41].

`trans_dec_set x`, says x is transitive and decent, [66].

`transdef_ord_prop p f x`, says $f(x) = p(F_x)$ where F_x is the functional graph $z \mapsto f(z)$ on x, [195].

`transdef_ord p x`, is $f(x)$, where f is the unique function satisfying $f(t) = p(\mathrm{F}_t)$, see above, [195].

`transfinite_closure X`, transitive closure of a set X, [443].

`transfinite_def r p f`, $\mathscr{I}(\mathrm{E}, p, f)$, says that f is defined by transfinite induction on the set E, well-ordered by r, via the property p, [54].

`transfinite_defined r p`, is the function defined by the property p by transfinite induction on the well-ordered set r, [56].

`transfiniteg_def r p f`, variant of `transfinite_def`,, [54].

`transfiniteg_defined r p`, is the graphdefined by transfinite induction, [55].

`transitivep r`, says that the graph r is transitive.

`transitive_r r`, says that the relation r is transitive.

`transitive_set x`, says that if $a \in b$ and $b \in x$ then $a \in x$, [66].

`transitive_setU U X`, transitive set [451].

`triple a b c`, is the ordered pair $(a, (b, c))$.

`tripleton a b c`, is the set $\{a, b, c\}$.

`Ufacts U x`, properties of a von Neumann universe [451].

`unbounded_interval r x`, says that x is an interval, but not a bounded one.

`union X`, $\bigcup \mathrm{X}$, is the union of a set of sets.

`unionA_pr x u`, `unionA`, `unionU x`, `unionU2 a b`, axiom of the union, [447], [449].

`uniont I f`, `unionf x f`, `uniont g`, $\bigcup_{\iota \in \mathrm{I}} \mathrm{X}_\iota$ is the set elements a with $a \in \mathrm{X}_\iota$ for some $\iota \in \mathrm{I}$.

`union2 a b`, `a \cup b`, $a \cup b$, is the union of two sets.

`universe`, `universe_i`: The von Neumann universe, [441].

`universe_permutation_fun` a permutation of the universe, [454].

`upper_bound r X x`, says that for all $y \in \mathrm{X}$, we have $y \le_r x$, [30].

`upper_bound1 r x` is used in an example.

`urank`, `urank0`, `urankA`, `urank_prop`, `urankA_prop`, the rank of an element of the von Neuman universe, [442].

`Val_of_not X v V`, `Val_of_or X v1 v2 v1 V2`, `Val_of_ex X v V`, `Val_of_eq X a b`, `Val_of_inc X a b`, value of a formula, [458].

`variables`, `variablep x`: variables in the set of formulas, [454].

`variant`, see `Lvariant`.

`well_founded_set x`: says that x is a well-founded set, [444].

well-ordered set, well-ordering relation, well-ordering: see `worder`.

`Vf f x`, $\mathscr{W}_f x$, is the value at the point x of the function f.

`Vfi f x`, $f^{-1}\langle x \rangle$, the set of all t such that $f(t) \in x$.

`Vfi1 f x`, $f^{-1}\langle \{x\} \rangle$, the set ot all t such that $f(t) = x$.

`Vfs f A`, $f\langle \mathrm{A} \rangle$, is $\{t, \exists x \in \mathrm{A}, t = f(x)\}$, when f is a function.

`Vg f x`, $\mathscr{V}(x, f)$ or $\mathscr{V}_f x$, is the value at the point x of the graph f.

`Vr x`, is Vg for the range of x, [347].

`worder r`, says that r is a well-ordering, [47].

`worder_rc r`, says that r is a well-ordering, such that $]\leftarrow, x[$ is always a set, [318].

`worder_on r E`, says that r is a well-ordering on E, [47].

`worder_fam f`: says that f is a family of well-ordered sets, [52].

`worder_of E`, the well-ordering of E given by Zermelo, [49].

`worder_r r`: says that r is a relation, that induces a well-ordering on each set where it is reflexive, [47].

`worder_rc r`: like `worder_r r`, moreover for every x, the collection of elements $< x$ is a set, [318].

`Yo P x y`, $\mathscr{Y}(\mathrm{P}, x, y)$, is a function that evaluates to x is P is true, and to y if P is false.

`z_infinite X` says that X contains zero and is stable by successor, [111]

`Zermelo_axioms E p s r`, auxiliary definition for Zermelo's theorem (version 1) [57].

`Zermelo_chain E F`, auxiliary definition for Zermelo's theorem (version 2), [49].

`Zermelo_like r` says that r is a well-ordering such that $c(\mathrm{S}_x) = x$, where S_x is the set of elements $\geq x$, and $c(\mathrm{X})$ is the element chosen by the axiom of choice for the nonempty set E, [49].

`ZF_axioms1`, axioms of Zermelo Fraenkel, [447]

`Zo x R`, $\mathscr{Z}(x, \mathrm{R})$, $\mathscr{E}_x(\mathrm{R})$ or $\{x, \mathrm{R}\}$: it is the set of all x that satisfy R.

Index

absolute value, 208, 234
addition, 88
associated, 16
associativity, 36, 299

bijective, 19
binomial coefficient, 150

cardinal, 65
choice, 39, 58, 318, 511
cofinal, 30, 409, 413
coinitial, 30
commutative, 408
comparable, 41
compatible, 16
contraction, 698

decreasing, 24
degree, 336
difference, 120
disjoint, 82
doubleton, 82

empty, 33, 35, 74
equivalence, 16
extension, 13
extensional, 446
extensionality, 20

factorial, 143
finite, 101
first derivation, 320

generated, 222
graph, 12
greatest, 27
greatest common divisor, 223
greatest lower bound, 31

ideal, 222
inaccessible, 416
increasing, 24
induced, 20
induction, 54

inf, 32
infimum, 32
infinite, 191
integer, 101
intersection, 29, 33, 59, 68
interval, 44, 106
isomorphism, 19, 21, 23, 43, 51

lattice, 40
least, 27
least common multiple, 223
least upper bound, 31
lexicographic product, 61
limit, 75
lower bound, 30

max, 27
maximal, 27
min, 27
minimal, 27
monotone, 24
morphism, 19
multiplication, 88

opposite, 12, 234
ordinal, 65

predecessor, 75
principal, 223
product, 22, 81, 88

quotient, 127

range, 698
reflexive, 12
regular, 409, 414
remainder, 127

segment, 50
singleton, 27
singular, 414
substrate, 12
subtraction, 120
successor, 66, 101

sum, 88
sup, 32
supremum, 32, 87
symmetric, 12

total, 27, 41
transitive, 12

union, 29, 33, 59, 74, 82, 102
upper bound, 30

well-ordering, 47

Bibliography

[1] D. S. Asche. Minimal dependent sets. *Journal of the Australian Mathematical Society*, pages 259–262, 1966.

[2] Jon Barwise and Lawrence Moss. *Vicious Circles*. Number 60. CLSI Publications, 1996.

[3] Yves Bertot and Pierre Castéran. *Interactive Theorem Proving and Program Development*. Springer, 2004.

[4] N. Bourbaki. *Elements of Mathematics, Theory of Sets*. Springer, 1968.

[5] N. Bourbaki. *Éléments de mathématiques, Théorie des ensembles*. Diffusion CCLS, 1970.

[6] N. Bourbaki. *Elements of Mathematics, Algebra I*. Springer, 1989.

[7] Georg Cantor. *Contributions to the Founding of the Theory of Transfinite Numbers*. Dover Publications Inc, 1897. Trans. P. Jourdain, 1955.

[8] Philip Carruth. Arithmetic of ordinals with applications to the theory of ordered abelian groups. *Bull. Amer. Math. Soc*, 48:262–271, 1942.

[9] Edsgert Dijkstra. EWD578: More about the function "fusc" (a sequel to EWD570). http://www.cs.utexas.edu/users/EWD/ewd05xx/EWD578.PDF, 1976. In "Selected Writings on Computing: A Personal Perspective, Springer-Verlag, 1982. ISBN 0–387–90652–5".

[10] Paul Erdös. Some remarks on set theory. *Proceedings of the American Mathematical Society*, 1(2):127–141, 1950.

[11] José Grimm. Implementation of Bourbaki's Elements of Mathematics in Coq: Part One, Theory of Sets. Research Report RR-6999, INRIA, 2009. http://hal.inria.fr/inria-00408143/en/.

[12] Gerhard Hessenberg. Grundbegriffe der Mengenlehre. Zweiter Bericht über das Unendliche in der Mathematik. *Abhandlungen der Fries'schen Schule*, pages 478–706, 1906.

[13] Douglas Hofstadter. *Gödel, Escher, Bach: An Eternal Golden Braid*. Basic Books, 1979.

[14] Jean-Louis Krivine. *Théorie axiomatique des ensembles*. Presses Universitaires de France, 1972.

[15] Jean-Louis Krivine. *Théorie des ensembles*. Cassini, 1998.

[16] Alan Mathias. A term of length 4 523 659 424 929. *Synthese*, 133:75–86, 2002. DOI: 10.1023/A:1020827725055.

[17] A. P. Roberstson and J. D. Weston. A general basis theorem. *Proc. Edinburgh Math. Soc.*, 2(11):139–141, 1958-1959.

[18] Kurt Schütte. *Proof theory*. Springer, 1977.

[19] Moritz Stern. Ueber eine zahlentheorische Funktion. *Journal für die reine und angewandte Mathematik*, 55:193–220, 1858.

[20] Alfred Tarski. Quelques théorèmes sur les alephs. *Fundamenta Mathematicae*, 7:1–14, 1925.

[21] The Coq Development Team. The Coq reference manual. http://coq.inria.fr.

[22] Oswald Veblen. Continous increasing functions of finite and transfinite ordinals. *Transactions of ther AMS*, 9(3), 1908. DOI=10.2307/1988605.

[23] Jean von Heijenoort, editor. *From Frege to Gödel: a source book in mathematical logic, 1879-1931*. Harvard University Press, 1967.

[24] Andrzej Wakulicz. Sur la somme d'un nombre fini de nombres ordinaux. *Fundamenta Mathematicae*, 36(1):254–266, 1949.

[25] Andrzej Wakulicz. Correction au travail "sur les sommes d'un nombre fini de nombres ordinaux" de a. wakulicz (fund. math. 36, p. 259). *Fundamenta Mathematicae*, 38(1):239, 1951. http://matwbn.icm.edu.pl/ksiazki/fm/fm38/fm38123.pdf.

Contents

1 Introduction **3**
1.1 Objectives . . . 3
1.2 Content of this document . . . 3
1.3 Terminology . . . 4
1.4 Notations . . . 5
1.5 Tactics . . . 9

2 Order relations. Ordered sets **11**
2.1 Definition of an order relation . . . 11
2.2 Preorder relations . . . 16
2.3 Notation and terminology . . . 17
2.4 Ordered subsets. Product of ordered sets . . . 20
2.5 Increasing mappings . . . 24
2.6 Maximal and minimal elements . . . 27
2.7 Greatest element and least element . . . 27
2.8 Upper and lower bounds . . . 30
2.9 Least upper bound and greatest lower bound . . . 31
2.10 Directed sets . . . 39
2.11 Lattices . . . 40
2.12 Totally ordered sets . . . 41
2.13 Intervals . . . 44

3 Well-ordered sets **47**
3.1 Segments of a well-ordered set . . . 47
3.2 The principle of transfinite induction . . . 53
3.3 Zermelo's theorem . . . 57
3.4 Inductive sets . . . 58
3.5 Isomorphisms of well-ordered sets . . . 59
3.6 Lexicographic products . . . 61

4 Equipotent Sets. Cardinals **65**
4.1 The cardinal of a set . . . 83
4.2 Order relation between cardinals . . . 83
4.3 Operations on cardinals . . . 88
4.4 Properties of the cardinals 0 and 1 . . . 92
4.5 Exponentiation of cardinals . . . 94
4.6 Order relation and operations on cardinals . . . 95

5 Natural integers. Finite sets **99**
5.1 Definition of integers . . . 101
5.2 Inequalities between integers . . . 101
5.3 The set of natural integers . . . 105
5.4 The principle of induction . . . 106
5.5 Finite subsets of ordered sets . . . 108
5.6 Properties of finite character . . . 110

6 Properties of integers **115**
6.1 Operations on integers and finite sets . . . 115
6.2 Strict inequalities between integers . . . 118
6.3 Intervals in sets of integers . . . 122
6.4 Finite sequences . . . 126
6.5 Characteristic functions on sets . . . 126
6.6 Euclidean Division . . . 127
6.7 Expansion to base b . . . 130
6.8 Combinatorial analysis . . . 143
6.9 More combinatorial analysis . . . 163
6.9.1 Number of derangements . . . 165
6.9.2 Number of increasing mappings . . . 165
6.9.3 Stirling numbers . . . 167
6.9.4 Euler numbers . . . 168
6.9.5 Sum of powers. . . . 172
6.9.6 Other formulas . . . 174
6.9.7 Dyck paths . . . 177
6.9.8 Derangements . . . 185
6.9.9 Formulas on Z . . . 186
6.9.10 Formulas using polynomials . . . 188
6.9.11 Partitions . . . 189

7 Infinite sets **191**
7.1 The set of natural integers 191
7.2 Definition of mappings by induction 192
7.3 Properties of infinite cardinals 196
7.4 Countable sets 200
7.5 Stationary sequences 202

8 Rational integers **205**
8.1 A definition of the set of rational integers 205
8.2 Opposite and absolute value 208
8.3 Ordering the integers 209
8.4 The sum of two integers 210
8.4.1 Subtraction 213
8.4.2 The sign function 214
8.5 Multiplication 214
8.6 The principle of induction 216
8.7 Properties of order 217
8.8 Euclidean Division 220
8.8.1 Divisibility 221
8.8.2 Ideals and Gcd 222
8.8.3 Gcd of natural numbers 227
8.9 Equivalence of definitions 228

9 Rational numbers **231**
9.1 Definition 231
9.1.1 Basic properties 232
9.1.2 Opposite 234
9.1.3 Absolute value 234
9.1.4 Order 235
9.2 Field operations 236
9.2.1 Addition 236
9.2.2 Subtraction 237
9.2.3 Multiplication 238
9.2.4 Inverse 240
9.2.5 Division 241
9.2.6 Sign 242
9.2.7 Compatibility of order and operations 243
9.2.8 Floor 245

9.2.9 Sums . . . 246
9.3 The ordering of Q . . . 247
9.4 Stern Brocot sequences . . . 254

10 Real numbers **265**
10.1 Definition and basic properties . . . 265
10.1.1 Dedekind cuts . . . 265
10.1.2 Rational cuts . . . 266
10.1.3 Real numbers . . . 267
10.1.4 Order . . . 269
10.2 Field Structure . . . 270
10.2.1 Opposite . . . 270
10.2.2 Addition . . . 271
10.2.3 Difference . . . 272
10.2.4 Multiplication . . . 274
10.2.5 Inverse and division . . . 277
10.2.6 Absolute value . . . 279
10.2.7 Half and middle . . . 280
10.3 Cauchy sequences and continuity . . . 281
10.3.1 Adjacent sequences . . . 281
10.3.2 The cardinal of R . . . 283
10.3.3 Cauchy sequences . . . 284
10.3.4 Continuity . . . 286
10.3.5 The power function . . . 290
10.3.6 Fibonacci . . . 290

11 Ordinal numbers **293**
11.1 Notations and vocabulary . . . 293
11.2 Basic Properties . . . 295
11.3 Operations on ordinals . . . 296
11.4 Operations and ordering . . . 302
11.5 Ordinal subtraction and division . . . 308
11.6 Normal ordinal functional symbols . . . 309
11.7 Enumerating a collection of ordinals . . . 317
11.8 Indecomposable ordinals . . . 322
11.9 Definition by transfinite induction . . . 324
11.10 Ordinal power . . . 329
11.11 Repeated derivations . . . 332

11.12 Cantor Normal Form 336
11.12.1 The simple normal form 336
11.12.2 Indecomposable ordinals 337
11.12.3 The general normal form 339
11.12.4 Properties of the Cantor Normal Form 347
11.12.5 Cantor normal form and operations 353
11.12.6 The product form 357
11.12.7 Factorisation into prime numbers 361
11.13 An exercise of Bourbaki 366
11.13.1 Study of the equation 367
11.13.2 Solution of the equation 369
11.13.3 Bourbaki integers 370
11.13.4 Solution of the problem. 372
11.14 Natural sum and products of ordinals 382
11.14.1 Natural sum 383
11.14.2 Double Induction Principle 386
11.14.3 More about natural sums 389
11.15 Numbers equal to their degree 393
11.15.1 Range of epsilon 394
11.15.2 Critical points of product 395
11.16 The functions phi and psi 396
11.16.1 First derivation of exponentiation 396
11.16.2 Many derivations 397
11.16.3 The function phi 398
11.16.4 The function psi 401
11.16.5 The Cantor Normal Form 403
11.17 Initial ordinals 406
11.18 Cardinal Cofinality 408
11.19 Ordinal cofinality 412
11.20 Infinite products 417
11.21 Sums of powers 426
11.22 Inaccessible cardinals 432
11.23 Consequences of GCH 436
11.24 The beth function 438
11.25 Consequences of SCH 440
11.26 Von Neumann universe 441
11.27 The set of formulas 454

11.28 Order types 465
11.28.1 The eta ordering of Cantor 470
11.29 The eta ordering of Cantor 470
11.29.1 A partial implementation of Q 470
11.29.2 Definition of eta 473
11.30 The cardinals, according to Zermelo, 1908 474

12 The size of one **481**
12.1 Comments 482
12.2 Computations 484

13 Exercises **501**
13.1 Additional theorems 501
Zermelo's theorem. 501
Monotonicity examples. 507
An example of well-ordering. 507
Examples of inductive sets. 508
An exercise of the first part. 509
Order relations. 510
The Cantor Bernstein Theorem. 511
The transfinite principle. 512
Well-ordering according to Cantor. 512
The Anti-Foundation Axiom. 513
Ordinal subtraction and division. 514
13.2 Section 1 515
1. 515
2. 515
3. 519
4. 524
5. 527
6. 528
7. 532
8. 534
9. 534
10. 535
11. 536
12. 539
13. 540

14. 541
15. 541
¶ 16. 544
¶ 17. 545
¶ 18. 548
19. 550
¶ 20. 552
21. 554
¶ 22. 557
¶ 23. 562
24. 564
13.3 Section 2 567
1. 567
2. 568
3. 569
¶ 4. 570
5. 571
¶ 6. 571
¶ 7. 573
¶ 8. 576
9. 578
10. 578
¶ 11. 578
¶ 12. 579
13. 581
¶ 14. 583
15. 584
¶ 16. 585
¶ 17. 585
¶ 18. 586
19. 587
¶ 20. 591
13.4 Section 3 592
¶ 1. 592
2. 592
¶ 3. 593
4. 593

5. 594
6. 595
13.5 Section 4 595
1. 595
2. 596
3. 596
4. 596
¶ 5. 597
¶ 6. 607
7. 611
¶ 8. 613
¶ 9. 616
¶ 10. 619
¶ 11. 621
13.6 Section 5 628
1. 628
2. 628
3. 629
4. 630
5. 630
6. 633
7. 633
8. 637
9. 638
10. 645
¶ 11. 646
¶ 12. 646
¶ 13. 647
¶ 14. 647
¶ 15. 647
16. 648
¶ 17. 648
18. 648
13.7 Section 6. 648
1. 648
2. 650
3. 650

4. 650
5. 651
6. 652
7. 653
8. 653
9. 654
¶ 10. 654
¶ 11. 655
¶ 12. 656
¶ 13. 657
¶ 14. 658
¶ 15. 660
16. 661
17. 663
¶ 18. 663
¶ 19. 663
¶ 20. 665
¶ 21. 665
¶ 22. 666
¶ 23. 666
¶ 24. 667
¶ 25. 668
¶ 26. 669
¶ 27. 669
¶ 28. 670
29. 670
30. 671
¶ 31. 671
¶ 32. 671
¶ 33. 672

14 Compatibility **677**

14.1 Pseudo Ordinals 677
Cardinals 679
The von Neumann Proof. 681
Pseudo-ordinals and the type nat. 681
Bijection between nat and the integers 684

Ordinals 685
14.2 Introduction to Chapter 6 688
14.3 The axiom of choice 690
14.4 Theorems removed from Chapter 6 691
Division 692
14.5 Finite sequences and lists 695
Lists as functions 695
Contracting lists 700
Iterated functions 703
Factorial 703
The binomial coefficient 704
Definition by transfinite induction 704
Initial Ordinals 704
14.6 Removed theorems 705
Other lemmas 707
Definition of a function by induction 707
Intervals 708
14.7 The size of one 711
14.8 Changes in Version 6 712

15 Theorems, Notations, Definitions **717**

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399