Deductive Local Model Checking

On the Verification of CTL Properties of Infinite-State Reactive Systems*

Christoph Sprenger

Ph.D. Thesis

Swiss Federal Institute of Technology
Lausanne, Switzerland

May 2000

Contents

Version Abregée

Cette thèse traite de la vérification formelle de propriétés temporelles des systèmes réactifs à états infinis. Nous proposons un système de preuve basé sur des tableaux et permettant la vérification, par model checking, de propriétés exprimées dans la logique arborescente CTL* construite sur un langage d'assertions $\mathcal{L}$. Le système de preuve est capable de traiter des formules CTL* arbitraires sans avoir besoin de les transformer dans une forme canonique.

Notre méthode repose sur la construction de *structures de preuve* aussi connues sous le nom de *tableaux.* Il existe deux types de structures de preuve (LTL et ELL), chacun correspondant à une sous-logique de CTL*, et deux ensembles de règles locales pour les construire. L'application de quelques unes de ces règles exige la démonstration de la validité de certains assertions de $\mathcal{L}$. Chaque type de structure de preuve est lié à son propre *critère de succès.* Ce dernier assure que les sous-formules exprimant des promesses sont satisfaites d'une manière appropriée: seuls les tableaux satisfaisant ce *critère* sont des démonstrations légales d'une propriété.

Chaque critère de succès est exprimé sous la forme d'une propriété temporelle particulière. Cette dernière doit être satisfaite par le système de transition associé à un tableau donné et au système que l'on vérifie. Un parcours dans ce *système associé*, appelé un *trail* du tableau, combine un parcours du système avec un chemin du tableau. Une règle de preuve supplémentaire est introduite pour chaque critère de succès. Ces règles utilisent un argument de bonne-fondation pour établir le succès d'une structure de preuve. Une *preuve* d'une propriété CTL* est alors une collection finie de structures de preuve LTL et ELL, dont le succès a été démontré par l'application de la règle appropriée. Du fait que les règles de succès sont aussi exclusivement basées sur le raisonnement dans le langage d'assertions $\mathcal{L}$, notre méthode réduit tout raisonnement temporel à la démonstration de la validité d'assertions tirées de $\mathcal{L}$. En conséquence, il n'y a pas besoin de prouver de theorèmes de la logique temporelle elle-même. Nous appelons notre méthode de preuve *model checking local deductive,* car elle généralise en même temps des tech-

niques de model checking pour des systèmes à états finis et des systèmes de preuve pour LTL et CTL qui ont été proposés dans la litérature.

Nous démontrons que notre systéme de preuve est correct et complet relatif à la validité d'assertions tirée de $\mathcal{L}$. Une partie majeure de cette preuve est basée sur la théorie des jeux. Dans une première étape la notion d'un jeux CTL* infini pour deux joueurs est introduite. Selon ce jeu l'objectif du premier joueur (appelé $\exists$) est de montrer qu'une propriété CTL* est satisfaite, alors que l'autre (appelé $\forall$) essaie de montrer le contraire. Nous donnons une charactérisation de la satisfaction d'une propriété CTL* en fonction de l'existence d'une stratégie gagnante pour le joueur $\exists$. Dans un deuxième temps, nous analysons la structure interne des chemins et trails d'une structure de preuve et nous mettons cette structure en relation avec l'idée des jeux dévelopée auparavant. En particulier, à chaque trail d'une structure de preuve LTL (ELL) correspond une stratégie du joueur $\forall$ ($\exists$) pour un jeu LTL (ELL). Nous montrons qu'une preuve d'une propriété LTL ou ELL par notre système existe précisement si le joueur $\exists$ possède une stratégie gagnante pour le jeu correspondant. Pour ce faire, on doit comparer la notion de succès à celle d'admissibilité, qui est une notion alternative de succès proposée dans la literature. Les résultats pour LTL et ELL sont ensuite généralisés à CTL* entier. La dernière étape consiste en la démonstration de la correction et de la complétude des règles de succès.

Différents types d'équité sont ensuite étudiés et nos règles de succès sont étendues pour les prendre en compte. Finalement, l'application du système de preuve est illustrée sur un exemple non banal.

Abstract

The present thesis is about the formal verification of temporal properties of infinite-state reactive systems. We propose a tableau proof system for the model-checking of properties expressed in the full branching time temporal logic CTL* over an assertion language $\mathcal{L}$. The proof system applies to an arbitrary CTL* formula. There is no need to transform formulas into some canonical form.

The basic proof object in our method is a *proof structure* (a.k.a. tableau). There are two types of proof structures (LTL and ELL), each corresponding to a sublogic of CTL*. Accordingly, there are two dual sets of proof rules. Some of these rules require the validity of an assertion from $\mathcal{L}$ to be proven as part of their application. Each type of proof structure has its own *success criterion,* which ensures that eventuality subformulas of the original formula are satisfied as appropriate. Only successful tableaux qualify as legal *proofs* of a property.

Each success criterion is formulated as a temporal property of some specific form. The latter has to be satisfied by a certain transition system associated with a given tableau and the reactive system to be verified. A run of this *associated transition system,* called a *trail* of the proof structure, combines a run of the system with a path in the proof structure. We introduce one additional proof rule for each success criterion. These rules employ a well-foundedness argument to establish that a proof structure of the respective type is successful. A proof of a CTL* property of a given system is then a finite collection of LTL and ELL proof structures the success of which has been established using the respective rules. As the success rules also exclusively rely on reasoning in $\mathcal{L}$ our method reduces all temporal reasoning to proving the validity of formulas from $\mathcal{L}$. Therefore, no theorem proving in the temporal logic itself is required. We call our method *deductive local model checking*, as it generalises both local model checking techniques for finite-state systems as well as proof systems for LTL and CTL that have been described in the literature.

We show that our proof system for model checking is sound and complete relative to validity of formulas from the assertion language $\mathcal{L}$. The major part of the proof relies on a game-theoretic argument. As a first and quite independent step we introduce the notion of a CTL* game, an infinite two-player game, where one player ($\exists$) tries to show that a property holds of the system, while the other player ($\forall$) tries to refute it. We give a characterisation of the satisfaction of a CTL* property in terms of the existence of a winning strategy for Player $\exists$. In a second step, we analyse the internal structure of paths and trails in proof structures and link it up with the game-theoretic ideas developed in the previous step. In particular, to each trail of a LTL (ELL) proof structure corresponds a $\forall$-strategy ($\exists$-strategy) of a LTL (ELL) game. We show that a successful proof structure for a given LTL or ELL formula and system exists precisely if Player $\exists$ has a winning strategy for the corresponding game. In doing so, we compare our notion of success with admissibility, an alternative notion of success proposed in the literature. The results for LTL and ELL are then lifted to full CTL*. As a final step we show that the success rules are sound and relatively complete.

We then study different types of fairness and extend our success rules to account for them. Finally, the application of the proof system is illustrated on a non-trivial example.

Acknowledgements

First of all I would like to thank my thesis supervisor Claude Petitpierre for his confidence and for all that liberty he gave me in discovering and following my research interests.

Many thanks to Krzysztof Worytkiewicz for careful proof-reading of parts of this thesis and for all the interesting discussions we had on the topic. Thanks also to Antonio Restrepo and David Barth for reading and commenting on the third chapter.

Antonio and Céline have invited me to their wedding party in a marvellous place in the middle of the vineyards above Lake Geneva, offering a truly pleasant break at a time when the week-ends started to fill with work. By our regular jogging along the lakeside, Claude Amendola has contributed a great deal to prevent my transformation into a pale laboratory mummy in the last two month of the writing. The bakery Boulaz in Lausanne supplied me with their marvellous vegetable cakes, which made up a good share of my stress diet in the last phases of the writing.

Special thanks to the development team of the LyX text (and formula!) processor for providing an excellent tool that makes writing a scientific text as comfortable as it can possibly be and that saved me from the headaches of continuously extracting the sense of my writing from heaps of LaTeX code.

I am particularly grateful for the continuous support from my parents, my brother and all my friends.

Last but not least, my thanks go to Armin Biere, Dilian Gurov, Yonit Kesten and Martin Odersky for serving as members of my examination committee and for their helpful comments on the submitted version of this thesis.

To my parents, Robert and Silvia

Chapter 1

Introduction

Modern society depends more and more on its own technological achievements. Powerful computer systems constitute the backbone of almost any conceivable technology today, be it in its development or in its implementation. The complexity of these systems is growing ceaselessly. Most of today's computing systems are characterised by an ongoing interaction with their environment. This interaction occurs in various forms such as the transmission of data over a communication network to another machine, interaction with a human user, or the exchange of information with the sensors and actuators of an embedded control system. Such systems are called *reactive,* in contrast to *transformational* systems which compute an output from a given input.

Considering our dependency on these systems, it is clear that they should be correct. The development of correct reactive systems represents a serious challenge for hardware and software engineering. Reactive systems are most often composed of several communicating concurrent processes. The inherent complexity of concurrency and communication makes the discovery of design errors a difficult task. Not only may there be mistakes in the *calculations* such systems perform (as in transformational systems), but there is also the possibility of *synchronisation* failures (such as as deadlocks, starvation, unexpected message reception etc.). The behaviour of reactive systems is best described in terms of their ongoing interaction with the environment rather than a relation between input and output data as for transformational systems. In fact, while termination is most often a required property of transformational programs, it is often undesirable for reactive systems.

Traditional software engineering methods for error detection such as simulation and testing can quickly become insufficient in this context. This does not mean that they should be abandoned. Simulation, for example, is a very valuable and efficient method for error detection in the initial phase of design validation. However, due to the sheer number of possible evolutions of concurrent systems, simulation and testing are unable to reach a satisfying coverage, leaving the more subtle errors buried in the depths of the state space. As the cost for the elimination of errors increase the later they are detected, there is also an immediate economical interest to eliminate errors as early as possible in the development cycle.

Formal Specification and Verification

Researchers in the field of *formal methods* tackle this problem with mathematical methods that allow the rigorous verification of designs. A formal framework for the specification and verification of reactive systems should include at least the following parts:

- a *mathematical model* of reactive systems
- a *requirement specification language,* and
- a *verification method*

Models The large majority of frameworks (and this thesis is no exception) that include a verification method use transition systems as their computational model of reactive systems. This is certainly due to the simplicity of this model. A transition system is essentially a graph, where the nodes represent system states and the edges atomic transitions between these states. Concurrency is modeled by non-deterministic *interleaving* of atomic actions. Many other models of reactive computation have been proposed in the literature (see [SNW96] for an overview), some of which like Petri Nets [Rei85] explicitly represent the concept of concurrency as distinct from non-determinism, but many of them lack verification frameworks.

Specification Two different classes of requirement specification can be distinguished. The first class could be called *relational.* In this approach the desired behaviour of a system design is formulated as another, more abstract system. Here the system description language and requirement specification language coincide. Systems are compared w.r.t. some behavioural preorder or equivalence relation. As there is no general agreement on what aspects of the behaviour of a process should be visible to an outside observer, a plethora of different behavioural relations has been proposed in the literature (see [dN87, vG90, vG93] for overviews). This type of requirement specification is complete in the sense that all constraints on the behaviour of an implementation are represented in the abstract system.

The method of *specification refinement* is an example of this approach to specification. Starting from an initial abstract system a series of more and more concrete specifications is produced by the process of refinement until the desired level of detail of the implementation is obtained. Each specification is related to the previous one by a behavioural preorder relation such as simulation. Frameworks based on set theory (see e.g., [Sta88, AL91]) as well as on temporal logic (see e.g., [Pnu92, KMP93, Lam94]) have been proposed. *Process algebras* are another instance of this class (see e.g., [Mil89, Hoa85, BW90, Mil99]), where algebraic theories of process terms and behavioural relations between them are developed.

The second type of requirement specification follows a *logical* approach: the specification language is a modal or temporal logic that is interpreted over the states or computations of a transition system. The behaviour of a design is specified in terms of a collection of desired properties expressed as formulas of the logic. Depending on the particular logic a system satisfies a property if all its initial states or all its computations do. Many useful properties of reactive systems can be expressed in these logics, including safety properties ("nothing bad happens") and liveness properties ("something good happens"). The approach followed in this thesis falls into this category. The logic we use as our specification language is CTL* [EH86], a full branching-time logic that allows quantification over computations to be freely mixed with linear-time operators. As such it is more expressive than linear-time temporal logic (LTL) or computation tree logic (CTL), but less expressive than the modal μ-calculus. For a survey of temporal and modal logics see [Eme90, Sti92].

Verification In this thesis we concentrate on the verification of logical specifications, also called *model checking*. For finite-state systems model checking is decidable and efficient algorithms exist for various logics (see [CGL93, CGP99] for a survey). An obvious advantage of algorithmic model checking is that it is fully automatic. In case a property fails to hold, the algorithm can also produce a counterexample, which is of invaluable help in understanding the reason for the failure. The main drawback of algorithmic model checking is the so-called *state space explosion problem*: the exponential growth of the state space in the number of component processes. Although many sophisticated heuristics have been developed to alleviate this fundamental problem, memory shortage is still the main limiting factor for the application of algorithmic model checking.

Model checking methods based on deductive reasoning on the other hand are applicable to arbitrary infinite-state systems. Their strength lies in their generality. As in general the model checking problem is undecidable, it cannot be fully automatic and therefore requires a certain expertise from the user, which is the main drawback of this method.

More recently, there has been a trend in combining the good sides of both approaches, for example, by constructing a finite-state abstraction of the system by deductive means which is then model-checked algorithmically [DF95, DGG97, GS97, RSS95, MBSU98].

1.1 Scope of the Thesis

In this thesis we address the problem of model checking CTL* properties of systems with possibly infinite state spaces.

Systems are specified by a syntactical representation of transition systems formulated in an assertion language $\mathcal{L}$. In order to compensate for the modeling of concurrency by non-determinism in the transition system model, our system specifications include *fairness* constraints to model the fair scheduling of system components. Only runs of the system where all these components receive fair treatment are considered as computations.

We present a tableau-based proof system for ground-quantified CTL*, which is obtained from pure propositional CTL* by replacing atomic propositions by assertions of $\mathcal{L}$ over the system variables. These assertions may contain first-order quantifiers, but no first-order quantification is allowed in the scope of temporal operators. The proof system is composed of two dual sets of local rules, one dealing with the universal path quantifiers and the other with the existential path quantifiers of CTL*. These rules are used to construct the basic proof objects of our method, called LTL and ELL proof structures. A CTL* proof structure is in turn essentially a collection of LTL and ELL proof structures.

In order to be accepted as a proof of a property, a proof structure has to satisfy a *success criterion* formulated in terms of the runs of an associated system (called *trails*) derived from the original system and the proof structure. An additional (global) rule for proving success is introduced for each type of proof structure. Our proof system reduces all temporal reasoning to showing the validity of *verification conditions* formulated in $\mathcal{L}$ and arising either as side conditions of the local rules used in the construction of proof structures or as premises of the success rules.

We show that our proof system is sound and complete. *Soundness* means that every provable statement is true and *completeness* means that every true statement is provable in our system. By the expressiveness of our assertion language we cannot expect that every verification condition is provable in some formal system, so completeness is proved *relative* to the validity of assertions (verification conditions). Soundness and completeness is shown to hold for LTL and ELL proof structures and is then lifted to CTL* proof structures. A large part of the soundness and completeness proof is based on a game-theoretic argument. We characterise satisfaction of CTL* formulas in terms of the existence of a winning strategy for one player in an infinite two-player game and then proceed by a fine-grained game-theoretic analysis of paths and trails in proof structures, discovering that trails correspond to strategies in CTL* games.

1.2 Contributions

The main contribution of this thesis is the presentation of a sound and relatively complete proof system for the model checking of CTL* properties of infinite-state reactive systems. The system specifications include a quite general type of fairness constraints. To the best of our knowledge no such proof system for CTL* has been proposed before in the literature. Our proof system generalises existing methods for finite-state model checking as well as several existing deductive proof systems.

Another contribution is our novel approach of using a game-theoretic argument to establish soundness and completeness of the proof system. In fact, our game-theoretic analysis of the fine structure of paths and trails in proof structures has an interest of its own as it offers interesting insights into the inner working of proof structures. Games and strategies are particularly attractive in this context as they provide a very intuitive point of view of otherwise rather abstract structures.

1.3 Chapter Outline

In Chapter 2 we introduce the transition system model, the temporal logic CTL* and some background on model checking. An abstract notion of games and strategies is also defined.

LTL proof structures are the topic of Chapter 3. We introduce a suitable sequent format and a set of local rules for the construction LTL proof structures. A success criterion is then presented that qualifies a proof structure as a legal proof. This criterion is defined in terms of *trails* which are runs of a system derived from a proof structure and the original system. Intuitively, a trail combines a run of the system with a path in the proof structure. We then introduce a global proof rule that allows us to establish success. The application of the proof system for LTL is illustrated by a series of examples.

In the following Chapter 4 we establish the soundness and relative completeness of the proof system for LTL. The proof is largely based on a game-theoretic argument. The first step consists in a characterisation of the satisfaction of a CTL* formula in terms of the existence of a winning strategy for one player in an infinite two-player game, where Player $\exists$ tries to establish the truth of the formula, while his opponent, Player $\forall$, tries to refute it. By observing that to each trail corresponds a strategy of Player $\forall$ we can show that a successful proof structure for a system $\mathcal{S}$ with initial condition Θ and property ϕ exists precisely if Player $\exists$ has a winning strategy for the game $\mathcal{G}_{\mathcal{S}}(\Theta, \phi)$ (and hence $\mathcal{S}, \Theta \models \phi$). The final step of the proof consists in showing that the LTL success rule is sound and relatively complete.

In Chapter 5 we extend our proof system to full CTL*. To this end, we first introduce the duals of LTL proof structures called ELL proof structures to handle existentially path-quantified formulas. The ELL success criterion is dual to the one for LTL. A rule is introduced to prove ELL success. Soundness and relative completeness are then shown along the lines of Chapter 4. Some results transfer directly by duality, while others need to be reviewed.

Chapter 6 addresses the problem of proving success for proof structures for systems with fairness constraints. The LTL as well as the ELL success rules are extended to cope with both weak and strong fairness constraints. The new rules are shown to be sound and complete. By this result we can lift the restriction to saturated systems (without fairness constraints) in the soundness and completeness theorem for CTL* proof structures, which was necessary for the only reason that the previous success rules did not account for fairness

The application of our proof system is illustrated on a non-trivial example in Chapter 7, where we verify some properties of the bakery protocol for mutual exclusion. In particular, we prove that the properties of mutual exclusion, accessibility and unboundedness hold for the bakery protocol. The latter shows that there is possibility of unbounded growth of some system variables, which makes the system infinite-state.

The final chapter concludes the thesis by a review of its goals and their achievement, a comparison with related work as well as an outlook on future research.

Chapter 2

Background

In this chapter we introduce transition systems, our computational model of reactive systems, and the temporal logic CTL*, our requirement specification language. We then survey some existing techniques for model checking, that is, for the verification of temporal properties of reactive systems. Algorithmic as well as deductive approaches are considered. Finally, we define the notions of games and strategies. We start with some basics on words and languages.

2.1 Words and Languages

Let A be an alphabet. We denote by A^* the set of finite words (sequences) and by A^ω the set of infinite words over A. Let $A^\infty = A^* \cup A^\omega$. A subset of A^* (A^ω,A^∞) is called a *language (ω-,∞-language)*. We denote the empty word by ϵ and the *concatenation* of a finite word $u \in A^*$ with a word $v \in A^\infty$ by $u \cdot v$ (or just uv). Define the (finite) *prefix* ordering on A^∞ by $u \leq w$ if $u \in A^*$ and there is a $v \in A^\infty$ such that $uv = w$. In this case v is called the *residuum* and is denoted by w/u.

For a finite word u, let $|u|$ denote its length, that is, the number of letters appearing on u. For an infinite word v, we define $|v| = \omega$. Let $w = a_0 a_1 \cdots a_j \cdots$ be a finite or infinite word. Define for $0 \leq i < |w|$:

- $w(i) = a_i$, the ith letter,
- $w[i] = a_o a_1 \cdots a_{i-1}$, its prefix of length i
- $w^i = a_i a_{i+1} \cdots$ be its ith suffix.

We use the quantifier $\exists^\omega$ and its dual $\forall^\omega$ as shorthands for "there are infinitely many" and "for all but finitely many", respectively. Define the set of letters appearing infinitely many times in w by

$$\mathsf{inf}(w) = \{a \mid \exists^{\omega} i.\ w(i) = a\}.$$

The stuttering removal operator $\natural\colon A^{\omega} \to A^{\omega}$ is inductively defined by

$$\begin{array}{lcl} \natural\epsilon & = & \epsilon \\ \natural(abw) & = & \begin{cases} w & \text{if } a = b \text{ and } w = a^{\omega} \\ \natural(bw) & \text{if } a = b \text{ and } w \neq a^{\omega} \\ a\natural(bw) & \text{otherwise} \end{cases} \end{array}$$

It replaces any *finite* repetition of a letter in an infinite word by a single occurrence of that letter.

Given two alphabets A and B, a map $f : A \to B \cup \{\epsilon\}$ is extended to words in the following way: for $w \in A^{\infty}$ we define $f^{\infty}\colon A^{\infty} \to B^{\infty}$ by

$$f^{\infty}(w) = f(w(0)) \cdot f(w(1)) \cdot \cdots \cdot f(w(j)) \cdot \cdots$$

Note that f^{∞} may map some infinite words to finite ones. We write f^{ω} and f^{*} for the restriction of f^{∞} to the domains A^{ω} and A^{*}, respectively.

2.2 Transition Systems

We use transition systems as our computational model of reactive systems.

2.2.1 Basic Transition Systems

DEFINITION 2.2.1. A *labeled transition system (LTS)* is a structure

$$\mathcal{T} = (S, \{\xrightarrow{\lambda} \mid \lambda \in \Lambda\}),$$

where S is a non-empty set of *states,* Λ is a non-empty set of *transition labels* (or *transitions*, for short) and $\xrightarrow{\lambda} \subseteq S \times S$ is a *transition relation* for each label $\lambda \in \Lambda$. ◇

We write $s \xrightarrow{\lambda} s'$ for $(s, s') \in \xrightarrow{\lambda}$ and say that there is an λ-transition leading from state s to state s'. An λ-transition is said to be *enabled* in a state s if there exists a state s' such that $s \xrightarrow{\lambda} s'$. For a set of transitions $\Lambda' \subseteq \Lambda$, we say that Λ' is enabled in state s if some transition $\lambda \in \Lambda'$ is enabled in s. A transition or set of transitions that is not enabled is called *disabled*. Let $\to$ denote the unlabeled *global* transition relation $\bigcup_{\lambda\in\Lambda} \xrightarrow{\lambda}$.

ASSUMPTION 2.2.2. For convenience, we will only consider LTS with a *total* global transition relations, where some λ-transition is always enabled in each state. We call such LTS *total*.

In case some transition system $\mathcal{T}$ is not total, there is a simple "trick" to transform it into a total one: add an idle transition λ_i to Λ with $s \xrightarrow{\lambda_i} s$ for any state s with Λ disabled.

Runs

A *run* of an LTS $\mathcal{T} = (S, \{\xrightarrow{\lambda} \mid \lambda \in \Lambda\})$ is an infinite sequence of states $\sigma\colon s_0 s_1 \cdots s_j \cdots$ such that for all $i \in \omega$ there is an $\lambda \in \Lambda$ such that $s_i \xrightarrow{\lambda} s_{i+1}$. We say that a λ-transition is *taken* at s_k on σ if $s_k \xrightarrow{\lambda} s_{k+1}$. For a set $\Lambda' \subseteq \Lambda$ of transitions, we say that Λ' is *taken* at s_k on σ if some $\lambda \in \Lambda'$ is taken at s_k.

For $U \subseteq S$ we define a *U-run* to be a run starting in some state $s \in U$. States appearing on a U-run are called *U-reachable*. For singleton sets we will write s-run instead of $\{s\}$-run. We write $\mathcal{R}_\mathcal{T}(U)$ for the set of U-runs and $\mathcal{R}_\mathcal{T}$ for the set of all runs of $\mathcal{T}$.

2.2.2 Fair Transition Systems

Intuitively, fairness [Fra86, Kwi89, AFK88] is a property of runs expressing that if some component of the system is sufficiently often ready to proceed, then its progress will not be delayed indefinitely. An unfair run is then a run along which the execution of some component is unduly delayed, though is is sufficiently often ready for execution. Of course, one has to specify more precisely what is meant by "sufficiently often" and "component".

Fairness is a way to compensate for the modeling of concurrency by non-determinism (interleaving of concurrent transitions) in the transition system model. When it comes to specifying and proving properties of a transition system, one typically only requires that the fair runs of a system satisfy it and generally ignore the unfair ones. Fair runs will be called *computations*. In practice, fairness is realised by a scheduler.

The most common notions of fairness are unconditional, weak and strong fairness (also called impartiality, justice and compassion [LPS81, MP92]) corresponding to different interpretations of "sufficiently often". Consider a transition system $\mathcal{T} = (S, \{\xrightarrow{\lambda} \mid \lambda \in \Lambda\})$, subset $\Lambda' \subseteq \Lambda$ of its transitions. We call a run σ of $\mathcal{T}$

- *unconditionally fair w.r.t.* $\Lambda' \subseteq \Lambda$ if Λ' is taken infinitely many times on σ,

- *strongly fair w.r.t.* $\Lambda' \subseteq \Lambda$ if whenever Λ' is enabled infinitely often, then Λ' is taken infinitely many times on σ, and
- *weakly fair w.r.t.* $\Lambda' \subseteq \Lambda$ if whenever Λ' is enabled continuously from some point on, then Λ' is taken infinitely many times on σ.

Note the decreasing strength: an unconditionally fair run is also strongly fair and a strongly fair is also weakly fair (w.r.t. some $\Lambda' \subseteq \Lambda$).

For our purpose, we define a *fairness constraint* w.r.t. a set of transitions Λ to be a triple $\mathcal{F} = (P, W, F)$, where $P \subseteq \mathcal{P}(\Lambda)$ is a finite partition of Λ, $W \subseteq P$ is a set of weakly fair sets of transitions and $F \subseteq P$ is a set of strongly fair sets of transitions. The elements of P can be seen as an abstract form of processes. They are the "components" referred to in the informal definition of fairness above. A run σ is called *fair* if it is weakly fair w.r.t. all $\Lambda_w \in W$ and strongly fair w.r.t. all $\Lambda_f \in F$. We informally write $\sigma \models \mathcal{F}$ to mean that σ is fair.

By varying the type of partition P of a fairness requirement $\mathcal{F} = (P, W, F)$ we can change the granularity of the "components" that the fairness requirements apply to. For example:

weak process fairness [LT87]:
$\mathcal{F} = (P, W, \varnothing)$ with $W = P$ for any partition P,

strong process fairness [CS87]:
$\mathcal{F} = (P, \varnothing, F)$ with $F = P$ for any partition P, and

transition fairness [MP92]:
$\mathcal{F} = (P_{id}, W, F)$ with the partition P_{id} induced by the identity relation on Λ and W and F arbitrary subsets of P.

There are other notions of fairness, which do not fit this scheme. We refer the reader to [Kwi89] for a survey and to Francez' book [Fra86] for more detailed information.

DEFINITION 2.2.3. A *fair transition system (FTS)* is a structure

$$\mathcal{T} = (S, \{\xrightarrow{\lambda} \mid \lambda \in \Lambda\}, \mathcal{F})$$

with $(S, \{\xrightarrow{\lambda} \mid \lambda \in \Lambda\})$ a LTS and $\mathcal{F}$ a fairness constraint for Λ. A fair run of $\mathcal{T}$ is called a *computation.* $\diamondsuit$

For a set $U \subseteq S$, a *U-computations* is a computation starting in a state $s \in U$. We write $\mathcal{C}_{\mathcal{T}}(U)$ for the set of U-computations and $\mathcal{C}_{\mathcal{T}}$ for the set of all computations of $\mathcal{T}$.

Initial states

It is sometimes useful to add a non-empty set of *initial states* $I_{\mathcal{T}} \subseteq S_{\mathcal{T}}$ to a (fair) transition system $\mathcal{T}$. Such a transition system is called *initialised.* One is then generally interested in the $I_{\mathcal{T}}$-computations only.

What Type of Transition System now?

Any type of transition system introduced above can be seen as an instance of an initialised fair transition system

$$\mathcal{T} = (S, \{\xrightarrow{\lambda} \mid \lambda \in \Lambda\}, I, \mathcal{F}),$$

henceforth just called *transition system* for brevity. Putting $I = S$ is equivalent to dropping I. Setting $\mathcal{F} = (P, \varnothing, \varnothing)$ for any partition P of Λ has the same effect as dropping $\mathcal{F}$. We will call a transition system with trivial or no fairness constraints *saturated,* as all of its runs are computations.

2.3 Syntactic Representation

For the purpose of specification and deductive verification, a more syntactic representation of transition systems is desirable.

2.3.1 Assertion language

Let $\mathcal{L}$ be an assertion language including at least the predicate calculus over the countable set of variables V and some fixed first-order structure $\mathcal{A}$ containing symbols for all the usual operations over integers and booleans that may occur in a system specification. Formulas of $\mathcal{L}$ are called assertions. For $X \subset V$ we write $\mathcal{L}[X]$ for the set of assertions with all free variables in X.

Further Assumptions on $\mathcal{L}$

For the purpose of showing relative completeness of our proof system a pure first-order language is not sufficiently expressive (see, e.g., [SdRG89]) and we have to extend $\mathcal{L}$ to include least and greatest fixed point operators (μ and ν, respectively). We denote this extension of $\mathcal{L}$ by $\mathcal{L}_\mu$ (see also Park's μ-calculus [Par76] and [Mos74, SdRG89]).

Furthermore, we will assume that the first-order structure $\mathcal{A}$ we are working with supports an elementary (first-order definable in the structure) coding scheme, allowing us to code finite sequences of elements of the domain D of the structure $\mathcal{A}$ as single elements of D. Structures with this property are called *acceptable* in [Mos74, SdRG89].

2.3.2 System Specifications

DEFINITION 2.3.1. A *(transition) system specification* (or *system* for short) is a structure

$$\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$$

where

- $X = \{x_1, \ldots, x_n\} \subseteq V$ is a finite set of typed *program variables.* Often these are subdivided into *control* variables indicating the locations in the program where control currently resides and *data* variables. We often use vector notation $\overline{x}$ for the ordered set $x_1, \ldots, x_n$ of program variables.

- Σ is the *state space*, the set of type-consistent interpretations of the program variables. An element $s \in \Sigma$ is called a *state.*

- $\rho_\lambda(\overline{x}, \overline{x}')$ is the *transition relation* for transition label $\lambda \in \Lambda$, an assertion which may refer to two copies of the program variables, an unprimed copy $\overline{x}$ representing the current state and a primed copy $\overline{x}' = x_1', \ldots, x_n'$ representing a successor state,

- $\Theta(\overline{x})$ is the *initial condition,* a satisfiable assertion describing the set of starting states, and

- $\mathcal{F}$ is a fairness constraint over Λ

A *state assertion (over X)* is an assertion all of whose free variables are program variables. Let p be a state assertion. We write $s \models p$ and say that s *satisfies* p if p is true when interpreting the free variables of p by s. If $s \models p$ we also say that s is a *p-state.* We say that p is *satisfiable* if there is a state s such that $s \models p$. An assertion is called *state valid* (w.r.t. structure $\mathcal{A}$), written $\models p$, if $s \models p$ for all states $s \in \Sigma$[1]. A state assertion p describes the set of states $\|p\|$ satisfying it, that is, $\|p\| = \{s \in \Sigma \mid s \models p\}$. For the sake of brevity, we will henceforth just say assertion for state assertion and validity

[1]Note that state validity is not to be confused with general validity of a first-order formula, which states that a formula is true in all first-order structures over the given signature and all interpretations of its free variables.

for state validity. To avoid confusion, we will explicitly indicate the set of free variables of an assertion in case not all of them are program variables.

A pair of states (s, s') satisfies a transition relation $\rho_\lambda(\overline{x}, \overline{x}')$, denoted by $(s, s') \models \rho_\lambda(\overline{x}, \overline{x}')$, if the assertion $\rho_\lambda(\overline{x}, \overline{x}')$ is true when interpreting each unprimed variable $x \in X$ by $s(x)$ and each primed variable $x' \in X'$ by $s'(x)$. For $\Lambda' \subseteq \Lambda$ we write $\rho_{\Lambda'}(\overline{x}, \overline{x}')$ to abbreviate $\bigvee_{\lambda \in \Lambda'} \rho_\lambda(\overline{x}, \overline{x}')$. Hence, $\rho_\Lambda(\overline{x}, \overline{x}')$ denotes the global transition relation.

Using the transition relation ρ_λ, we are now able to express enabledness of a set of transitions $\Lambda' \subseteq \Lambda$ by the assertion

$$en_{\Lambda'}(\overline{x}) \stackrel{\text{def}}{=} \exists \overline{x}'.\rho_{\Lambda'}(\overline{x}, \overline{x}')$$

and write $en_\lambda(\overline{x})$ instead of $en_{\{\lambda\}}(\overline{x})$.

A system specification $\mathcal{S}$ as above induces the obvious transition system $\mathcal{T}_\mathcal{S} = (\Sigma, \{\xrightarrow{\lambda} \mid \lambda \in \Lambda\}, I, \mathcal{F})$, with transitions $\xrightarrow{\lambda} = \{(s, s') \mid (s, s') \models \rho_\lambda\}$ for each $\lambda \in \Lambda$, initial states $I = \{s \in \Sigma \mid s \models \Theta\}$. Given a system $\mathcal{S}$, we will write $\mathcal{R}_\mathcal{S}$ for $\mathcal{R}_{\mathcal{T}_\mathcal{S}}$ and $\mathcal{C}_\mathcal{S}$ for $\mathcal{C}_{\mathcal{T}_\mathcal{S}}$. For a state assertion Ξ let $\mathcal{R}_\mathcal{S}(\Xi)$ and $\mathcal{C}_\mathcal{S}(\Xi)$ denote $\mathcal{R}_\mathcal{S}(\|\Xi\|)$ and $\mathcal{C}_\mathcal{S}(\|\Xi\|)$, respectively.

2.4 Temporal Logic

In this section, we present the syntax and semantics of our requirement specification language, the temporal logic CTL* [EH86] and its sublogics.

2.4.1 Syntax

Let Prop be a set of atomic propositions and define the set of literals by $\mathsf{Lit} = \mathsf{Prop} \cup \{\neg p \mid p \in \mathsf{Prop}\}$.

DEFINITION 2.4.1. The syntax of the logic CTL* is defined by

$$\varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \varphi\,\mathsf{V}\,\varphi \mid \varphi\,\mathsf{U}\,\varphi \mid \mathsf{A}\,\varphi \mid \mathsf{E}\,\varphi$$

where $p \in \mathsf{Lit}$ is a literal. The connectives Next (X), Release (V) and Until (U) are called temporal operators. The operators A and E are called universal and existential path quantifiers, respectively. We will often call a formula with top-level connective $\nabla \in \{\wedge, \vee, \mathsf{X}, \mathsf{V}, \mathsf{U}, \mathsf{A}, \mathsf{E}\}$ a ∇-formula and use Z as a placeholder for either U or V. So a Z-formula is either a U- or a V-formula. We write $\phi \preccurlyeq \phi'$ to mean that ϕ is a subformula of ϕ'. The relation $\preccurlyeq$ induces a partial order on CTL* formulas. Let $\mathsf{V}(\phi)$ denote the set of V-subformulas of ϕ and similarly for $\mathsf{U}(\phi)$. The (path-)quantifier depth of a CTL* formula is inductively defined by:

$$\begin{aligned}
&\mathsf{qd}(p) = 0 \\
&\mathsf{qd}(\mathsf{X}\,\phi) = \mathsf{qd}(\phi) \\
&\mathsf{qd}(\phi_1 \diamond \phi_2) = \max(\mathsf{qd}(\phi_1), \mathsf{qd}(\phi_2)) \qquad \text{for } \diamond \in \{\wedge, \vee, \mathsf{V}, \mathsf{U}\} \\
&\mathsf{qd}(\mathsf{A}\,\phi) = \mathsf{qd}(\mathsf{E}\,\phi) = 1 + \mathsf{qd}(\phi)
\end{aligned}$$

We say that a formula ϕ is of level $k \geq 0$ if $\mathsf{qd}(\phi) = k$. We now define three sublogics of CTL*:

- the logic LTL consists of the path-quantifier free (depth 0) formulas of CTL*.
- the logic ALL (ELL) consists of the CTL* formulas of the form $\mathsf{A}\,\varphi$ ($\mathsf{E}\,\varphi$), where φ is a LTL formula.
- the logic CTL consists of those CTL* formulas, where every temporal operator is immediately preceded by a path quantifier. In other words, the syntax of CTL is defined by

$$\varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{Q}\,\mathsf{X}\,\varphi \mid \mathsf{Q}(\varphi\,\mathsf{V}\,\varphi) \mid \mathsf{Q}(\varphi\,\mathsf{U}\,\varphi),$$

  where $p \in \mathsf{Lit}$ and $\mathsf{Q} \in \{\mathsf{A}, \mathsf{E}\}$ is a path quantifier. $\diamondsuit$

In our presentation of these logics, negation on formulas other than assertions is a meta-level notion inductively defined by

$$\begin{array}{rcl@{\qquad}rcl}
\neg(\phi_1 \wedge \phi_2) & = & \neg\phi_1 \wedge \neg\phi_2 & \neg(\phi_1 \vee \phi_2) & = & \neg\phi_1 \vee \neg\phi_2 \\
\neg(\psi_1\,\mathsf{V}\,\psi_2) & = & \neg\psi_1\,\mathsf{U}\,\neg\psi_2 & \neg(\psi_1\,\mathsf{U}\,\psi_2) & = & \neg\psi_1\,\mathsf{V}\,\neg\psi_2 \\
\neg\,\mathsf{X}\,\psi & = & \mathsf{X}\,\neg\psi & & & \\
\neg\,\mathsf{A}\,\psi & = & \mathsf{E}\,\neg\psi & \neg\,\mathsf{E}\,\psi & = & \mathsf{A}\,\neg\psi
\end{array}$$

In this definition the duality of operators is exploited thereby avoiding an exponential blow-up in the size of the formula.

The propositions true and false as well as the connectives for implication ($\rightarrow$) and equivalence ($\leftrightarrow$) are defined in the usual way using the boolean connectives. Some other frequently used temporal operators are 'always' (G), 'eventually' (F) and 'unless' (W) which can be defined from the basic ones by

$$\begin{array}{rcl}
\mathsf{G}\,\psi & \stackrel{\text{def}}{=} & \mathsf{false}\,\mathsf{V}\,\psi \\
\mathsf{F}\,\psi & \stackrel{\text{def}}{=} & \mathsf{true}\,\mathsf{U}\,\psi \\
\psi_1\,\mathsf{W}\,\psi_2 & \stackrel{\text{def}}{=} & \psi_2 \vee (\mathsf{X}\,\psi_2\,\mathsf{V}\,\psi_1)
\end{array}$$

In order to save us some parenthesis, we adopt the convention that unary operators have higher precedence than binary operators and binary temporal connectives have precedence over binary boolean connectives.

2.4.2 Semantics

A model $\mathcal{M}$ for a CTL* formula is a pair $(\mathcal{T}, V)$, where $\mathcal{T}$ is a transition system and $V : \mathsf{Lit} \to 2^{S_\mathcal{T}}$ is a valuation map assigning to each atomic proposition p the states $V(p) \subseteq S_\mathcal{T}$ where p holds. We require that $V(\neg p) = S_\mathcal{T} - V(p)$.

DEFINITION 2.4.2. For a CTL* model $M = (\mathcal{T}, V)$, a run $\sigma \in \mathcal{R}_\mathcal{T}$ and CTL* formulas $\varphi, \varphi_1, \varphi_2$, the *satisfaction relation* $\models$ is inductively defined by

$$\begin{array}{lll}
\mathcal{M}, \sigma \models p & \text{iff} & p \in \mathsf{Lit} \text{ and } \sigma(0) \in V(p) \\
\mathcal{M}, \sigma \models \varphi_1 \wedge \varphi_2 & \text{iff} & \mathcal{M}, \sigma \models \varphi_1 \text{ and } \mathcal{M}, \sigma \models \varphi_2 \\
\mathcal{M}, \sigma \models \varphi_1 \vee \varphi_2 & \text{iff} & \mathcal{M}, \sigma \models \varphi_1 \text{ or } \mathcal{M}, \sigma \models \varphi_2 \\
\mathcal{M}, \sigma \models \mathsf{X}\,\varphi & \text{iff} & \mathcal{M}, \sigma^1 \models \varphi \\
\mathcal{M}, \sigma \models \varphi_1 \,\mathsf{V}\, \varphi_2 & \text{iff} & \forall k \in \omega\colon \mathcal{M}, \sigma^k \models \varphi_2 \text{ if } \mathcal{M}, \sigma^i \not\models \varphi_1 \text{ for all } i < k \\
\mathcal{M}, \sigma \models \varphi_1 \,\mathsf{U}\, \varphi_2 & \text{iff} & \exists k \in \omega\colon \mathcal{M}, \sigma^k \models \varphi_2 \text{ and } \mathcal{M}, \sigma^i \models \varphi_1 \text{ for all } i < k \\
\mathcal{M}, \sigma \models \mathsf{A}\,\varphi & \text{iff} & \mathcal{M}, \sigma' \models \varphi \text{ for all } \sigma' \in \mathcal{C}_\mathcal{T}(\sigma(0)) \\
\mathcal{M}, \sigma \models \mathsf{E}\,\varphi & \text{iff} & \mathcal{M}, \sigma' \models \varphi \text{ for some } \sigma' \in \mathcal{C}_\mathcal{T}(\sigma(0))
\end{array}$$

$\diamondsuit$

Our presentation of CTL* follows the non-standard approach of Stirling [Sti89], in which all formulas are interpreted over paths (i.e., computations of a transition system), thus eliminating the distinction between state and path formulas (interpreted over states and computations, respectively) made in the standard presentation [EH86].

We define a posteriori a *state formula* to be a boolean combination of literals and path-quantified formulas. A formula that is not state formula is called a *path formula*[2]. For any state formula ψ and two runs $\sigma, \sigma' \in \mathcal{R}_\mathcal{T}(s)$ we have

$$\mathcal{M}, \sigma \models \psi \quad \text{iff} \quad \mathcal{M}, \sigma' \models \psi$$

Thus, satisfaction of a state formula depends only on the first state of a run and we can write $\mathcal{M}, s \models \psi$ in this case[3]. We extend the notion of satisfaction to arbitrary CTL* formulas and sets of states. Let $U \subseteq S_\mathcal{T}$ and define

$$\mathcal{M}, U \models \varphi \quad \text{iff} \quad \mathcal{M}, \sigma \models \varphi \text{ for all } \sigma \in \mathcal{C}_\mathcal{T}(U).$$

[2]Note that this definition of path formulas does not coincide with the one in [EH86], where all state formulas are also path formulas.

[3]The converse does not hold; consider for example the path formula $\mathsf{X}\,p \vee \mathsf{X}\,\neg p$.

For $s \in S_{\mathcal{T}}$ we write $\mathcal{M}, s \models \varphi$ instead of $\mathcal{M}, \{s\} \models \varphi$. When there is no risk of confusion we will drop the indication of the model $\mathcal{M}$ and write, for instance, just $\sigma \models \varphi$ instead of $\mathcal{M}, \sigma \models \varphi$ and $U \models \varphi$ for $\mathcal{M}, U \models \varphi$.

We say that a model $\mathcal{M} = (\mathcal{T}, V)$ *satisfies* a CTL* formula φ, written $\mathcal{M} \models \varphi$, if $\mathcal{M}, I_{\mathcal{T}} \models \varphi$. A CTL* formula is *valid,* written $\models \varphi$, if $\mathcal{M} \models \varphi$ for all models $\mathcal{M}$.

Classification of Properties

There are two main classifications of (linear-time) temporal logic properties. The most common classification distinguishes *safety* properties ("nothing bad happens") from *liveness* properties ("something good happens", possibly repeatedly) [AS85]. Examples of safety properties are deadlock freedom, partial correctness and any form of invariant. Examples of liveness properties are freedom from starvation, total correctness and fairness. Topologically, a safety property is a closed set, while a liveness property is a dense set in the Cantor topology on Σ^ω for a set of states Σ [AS85, CMP93].

The alternative *safety-progress* classification [CMP93] is tailored to properties describable in LTL. This classification of properties is hierarchical, according to the alternation of G and F operators. The classes are: *safety* ($\mathsf{G}\,p$), *guarantee* ($\mathsf{F}\,p$), *obligation* ($\bigwedge_{i=1}^{n} \mathsf{G}\,p_i \vee \mathsf{F}\,q_i$), response ($\mathsf{G}\,\mathsf{F}\,p$), persistence ($\mathsf{F}\,\mathsf{G}\,p$) and reactivity ($\bigwedge_{i=1}^{n} \mathsf{G}\,\mathsf{F}\,p_i \vee \mathsf{F}\,\mathsf{G}\,q_i$). A property of a class other than safety is called a *progress property.* It is shown in [LPZ85] that every LTL formula with past operators is equivalent to a reactivity formula. Topologically, this classification corresponds to the lower two and a half levels of the Borel hierarchy (see also [TL94]).

Equivalences

We introduce two equivalences on CTL* formulas, one based on states and the other based on computations. Let $\mathcal{M} = (\mathcal{T}, V)$ be a model. Then we define:

$\varphi_1 \equiv_{\mathcal{M},U} \varphi_2$	if $\mathcal{M}, \sigma \models \varphi_1$ if and only if $\mathcal{M}, \sigma \models \varphi_2$ for all $\sigma \in \mathcal{C}_{\mathcal{T}}(U)$
$\varphi_1 \equiv_{\mathcal{M}} \varphi_2$	if $\varphi_1 \equiv_{\mathcal{M},I} \varphi_2$ for I the set of initial states of $\mathcal{T}$
$\varphi_1 \equiv \varphi_2$	if $\varphi_1 \equiv_{\mathcal{M}} \varphi_2$ for all models $\mathcal{M}$
$\varphi_1 \approx_{\mathcal{M},U} \varphi_2$	if $\mathcal{M}, s \models \varphi_1$ if and only if $\mathcal{M}, s \models \varphi_2$ for all $s \in U$
$\varphi_1 \approx_{\mathcal{M}} \varphi_2$	if $\varphi_1 \approx_{\mathcal{M},I} \varphi_2$ for I the set of initial states of $\mathcal{T}$
$\varphi_1 \approx \varphi_2$	if $\varphi_1 \approx_{\mathcal{M}} \varphi_2$ for all models $\mathcal{M}$

Some properties of these equivalences are summarised in

PROPOSITION 2.4.3. (CTL* EQUIVALENCES) *We have:*

(i) $\equiv\subset\approx$,

(ii) $\varphi_1 \approx \varphi_2$ *iff* $\mathsf{A}\,\varphi_1 \equiv \mathsf{A}\,\varphi_2$,

(iii) $\varphi \approx \mathsf{A}\,\varphi$ *for all CTL* formulas* φ;

(iv) $\psi \equiv \mathsf{A}\,\psi$ *for CTL* state formulas* ψ, *and*

(v) $\equiv$ *is a congruence on CTL*, while* $\approx$ *is not.*

PROOF. (i) The inclusion follows from the definition. It is also easy to see that $\mathsf{G}\,p \approx \mathsf{A}\,\mathsf{G}\,p$, while $\mathsf{G}\,p \not\equiv \mathsf{A}\,\mathsf{G}\,p$. (ii),(iii),(iv) Easy. (v) A routine induction on contexts shows that $\equiv$ is a congruence. On the other hand, $\approx$ is not a congruence as $\mathsf{G}\,p \approx \mathsf{A}\,\mathsf{G}\,p$, but $\mathsf{A}\,\mathsf{F}\,\mathsf{G}\,p \not\approx \mathsf{A}\,\mathsf{F}\,\mathsf{A}\,\mathsf{G}\,p$. □

It follows from (ii) and (iii) that the two equivalences coincide on state formulas. As another consequence of (iii) the distinction of the logics LTL and ALL appears rather artificial when talking about satisfaction of formulas w.r.t. a set of states or a model. Therefore, we will take the freedom of identifying the two logics in that context.

The reason for introducing the two equivalences is that $\equiv$ is a congruence, whereas the weaker $\approx$ is useful in comparing the expressive power of CTL* with logics interpreted over states.

Expressiveness

Let L_1 and L_2 be two temporal (or modal) logic languages interpreted over states of a transition system. We say that L_1 is no more expressive than L_2 and write $L_1 \leq L_2$, if for all $\varphi_1 \in L_1$ there is some $\varphi_2 \in L_2$ such that $\varphi_1 \approx \varphi_2$. We say that L_1 is strictly less expressive than L_2, written $L_1 < L_2$, if $L_1 \leq L_2$, but not $L_2 \not\leq L_1$.

Denote by μK the modal μ-calculus [Koz83]. Then the following relations hold:

PROPOSITION 2.4.4. (RELATIVE EXPRESSIVENESS) *We have:*

(i) $L < CTL^*$ *for* $L \in \{ELL, LTL, CTL\}$,

(ii) $CTL^* < \mu K$, *and*

(iii) ELL, LTL *and* CTL *are mutually incomparable w.r.t.* $\leq$. □

In lieu of proving this proposition we just remark that an effective (but double exponential) translation from CTL* to μK was given by Dam in [Dam94] (see also [Ref96]).

2.4.3 Ground-quantified CTL*

Let $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ be a system. If we take as the set of propositions of CTL* the set of state assertions over X, we move from a pure propositional setting to what we call *ground-quantified CTL* (over X), written CTL*$[X]$*. This is a restricted form of first-order CTL*, where no path-quantifier or temporal operator may occur in the scope of a first-order quantifier.

The system $\mathcal{S}$ induces a model $\mathcal{M}_{\mathcal{S}} = (\mathcal{T}_{\mathcal{S}}, V_{\mathcal{S}})$, where $\mathcal{T}_{\mathcal{S}}$ is the transition system induced by $\mathcal{S}$ as above and $V_{\mathcal{S}} \colon \mathcal{L}[X] \to 2^{\Sigma}$ is defined by $V_{\mathcal{S}}(p) = \|p\|$. We then write $\mathcal{S}, \Xi \models \varphi$ for $\mathcal{M}_{\mathcal{S}}, \|\Xi\| \models \varphi$ and $\mathcal{S} \models \varphi$ for $\mathcal{S}, \Theta \models \varphi$.

For surveys on temporal and modal logics we refer the reader to [Eme90, Sti92, Sti96b, MP92].

2.4.4 The Model Checking Problem

Given formula φ of a modal or temporal logic L and model $\mathcal{M}$ for L, the *model checking* problem consists in verifying whether or not

$$\mathcal{M} \models \varphi$$

holds. There are algorithmic as well as deductive approaches to model checking[4]. In the last decade, model checking has developed into a vast field of research, so the following short overview of the existing work on algorithmic and deductive approaches must necessarily remain incomplete.

2.5 Algorithmic Model Checking

The method of algorithmic model checking of finite state systems was pioneered in the early eighties independently by Clarke and Emerson [CE81] and by Queille and Sifakis [QS82] for the logic CTL. A tableau-based algorithm for LTL model checking was developed a few years later by Lichtenstein and Pnueli in [LP85]. Emerson and Lei [EL85] published at the same time the first model checker for CTL*. CTL model checking is more efficient than LTL and CTL* model checking. Its time complexity is linear both in the size of the model and the formula, while for LTL as well as CTL* it is also linear in the size of the model but exponential in the size of the formula. This

[4] The term 'model checking' has traditionally been used for algorithmic methods only. In this thesis, we understand model checking in this broader sense, independently of the method used.

advantage of CTL over LTL and CTL* is contrasted by the fact that fairness is not expressible in CTL. This situation is partially remedied in [CES86], where their CTL model checker is extended to handle fairness constraints. On the other hand, the size of formulas in typical requirement specifications is usually small enough to make LTL or CTL* model checking applicable in practice.

2.5.1 The Automata-theoretic Approach

A uniform theoretical framework for the design of model checking algorithms is provided by the theory of automata on infinite objects [Tho90]. Vardi and Wolper [VW86] were the first to reformulate LTL model checking in terms of automata on infinite words (ω-automata). In this approach, the verification of $\mathcal{M} \models \varphi$ is reduced to checking whether the product of the two ω-automata $\mathcal{A}_{\mathcal{M}}$ and $\mathcal{A}_{\neg\varphi}$ accepts the empty language. The ω-automaton $\mathcal{A}_{\mathcal{M}}$ accepts exactly the computations of $\mathcal{M}$, while the $\mathcal{A}_{\neg\varphi}$ accepts exactly the sequences of states that satisfy $\neg\varphi$. Any computation accepted by the product automaton is thus a computation that does not satisfy φ, that is, a counterexample for $\mathcal{M} \models \varphi$ (see also [Kur94]). More recently, branching time model checking has also been formulated in the automata-theoretic framework using alternating tree automata [BVW].

2.5.2 Local Model Checking

The principal drawback of algorithmic model checking is that it suffers from the so-called *state space explosion* problem, which is due to the state space growing exponentially in the number of concurrent processes.

Local (a.k.a. on-the-fly) algorithms have been developed to potentially reduce the number of states that have to be explored in order to establish or refute a property. Whereas *global* algorithms compute all states satisfying a given formula and hence need to explore the whole reachable state space, local algorithms answer the question whether a given set of states (typically the initial states) satisfy the formula. In this way, counterexamples can be found faster and even in case the property holds it is not always necessary to visit the whole state space. A local algorithm for LTL was presented by Gerth et al. in [GPVW95] and an efficient local CTL* model checker is described by Bhat, Cleaveland and Grumberg in [BCG95].

2.5.3 Symbolic Model Checking

In the early nineties, the advent of symbolic model checking [BCM92] has dramatically pushed the limits of automatic verification. This technique, originally developed for Park's μ-calculus [Par76], is based on computing fixed points of functions $\tau : \mathcal{P}(S) \to \mathcal{P}(S)$ on sets of states and the symbolic representation of these sets ordered binary decision diagrams (OBBDs) [Bry86]. OBDDs provide a canonical representation of boolean functions and relations, therefore enabling model checking algorithms to operate on entire sets of states without enumerating the individual states or transitions. Symbolic model checking algorithms for CTL and the modal μ-calculus are immediately obtained by the straightforward embedding of these logics into Park's μ-calculus, but symbolic model checkers have also been developed for LTL [KRP95, CGH97].

Symbolic model checking has been particularly successful in circuit verification, where state spaces often exhibit considerable regularity and thus allow for compact BDD representations.

2.5.4 Reduction Techniques

State spaces of software systems are usually not as regular as those found in circuit designs. Therefore, alternative methods have been developed to combat the state space explosion problem. These reduction techniques may be grouped into three classes:

Quotienting Techniques construct the quotient of the state space w.r.t. some equivalence relation; examples are partial order reduction, symmetry reduction and partition refinement; these methods are characterised by a *strong preservation* of properties, that is, a property holds of the reduced model precisely if it holds on the complete model

Abstraction Techniques construct an abstract model by collecting many concrete states into a single abstract state; the relation between concrete and abstract system can be described by a simulation or a Galois connection between their state spaces; the technique of network invariants for verification of infinite families of systems falls is also a form of abstraction (of a whole family of systems); abstraction techniques usually only guarantee *weak preservation* of properties, that is, only the truth but not the falsity of properties carries over to the concrete system

Compositional Techniques try to infer a property of a system from properties of its components; properties of components are often established by making assumptions about its environment (assumption-guarantee style of specifications)

For bibliographic references and examples of most of these reduction techniques we refer the interested reader to [CGP99].

General surveys of algorithmic model checking can be found in [CGL93, Sti96b, MP95] and in the recently published book [CGP99].

2.6 Deductive Approaches to Model Checking

The model checking problem is decidable only for finite-state systems and special cases of infinite-state systems, so the applicability of algorithmic methods is restricted to these classes of systems. Limited computing resources even make the automatic verification of many finite-state state systems impossible despite the sophisticated reduction techniques available today. These cases as well as general infinite-state systems can only be tackled by deductive methods.

Deductive approaches to model checking use a set of *proof rules* to reduce the global temporal properties to local first-order *verification conditions*. The proof rules may reduce temporal properties to simpler temporal properties, but ultimately all reasoning is reduced to showing the validity of assertions, thus avoiding reasoning in the temporal logic itself. In the rest of this section, we first introduce some of the ingredients of proof rules and then give a brief overview of some existing deductive approaches to model checking.

2.6.1 Verification Conditions

Hoare triples and possibility triples are (besides implications) the most current verification conditions appearing in proof rules. They abstractly describe properties of (sets of) transitions in terms of two assertions, called pre- and post-conditions. These triples are presented below, along with their relation to some standard predicate transformers.

Let $\Lambda \subseteq \Lambda_{\mathcal{S}}$ be a subset of the transitions of some system $\mathcal{S}$. We use the standard *Hoare triple* notation

$$\{p\}\,\Lambda\,\{q\}$$

for the assertion $p(\overline{x}) \wedge \rho_\Lambda(\overline{x}, \overline{x}') \rightarrow q(\overline{x}')$, meaning that a Λ-transition starting in a p-state always leads to a q-state. The assertion p is called the *pre-condition* and q is called the *post-condition* of the Hoare triple.

Two well-known predicate transformers are the *weakest pre-condition* and the *strongest post-condition* of an assertion p relative to Λ, which are defined as follows:

$$\begin{array}{lcl} \mathsf{wpc}_\Lambda(p)(\overline{x}) & \stackrel{\text{def}}{=} & \forall \overline{x}'.\, \rho_\Lambda(\overline{x}, \overline{x}') \rightarrow p(\overline{x}') \\ \mathsf{spc}_\Lambda(p)(\overline{x}) & \stackrel{\text{def}}{=} & \exists \overline{x_0}.\, p(\overline{x_0}) \wedge \rho_\Lambda(\overline{x_0}, \overline{x}) \end{array}$$

The weakest pre-condition $\mathsf{wpc}_\Lambda(p)$ describes the set of states from which all Λ-transitions lead to a state satisfying p. On the other hand, the strongest post-condition $\mathsf{spc}_\Lambda(p)$ denotes the set of states which are reachable from states satisfying p. Their relation to Hoare triples is characterised by the following chain of equivalences

$$\{p\}\,\Lambda\,\{q\} \;\equiv\; \mathsf{spc}_\Lambda(p) \rightarrow q \;\equiv\; p \rightarrow \mathsf{wpc}_\Lambda(q).$$

The dual notion of Hoare triples, called *possibility triples* in [Sip99], and written

$$\{p\}\ \langle\Lambda\rangle\ \{q\}$$

stands for the predicate $p(\overline{x}) \rightarrow \exists \overline{x}'.\rho_\Lambda(\overline{x}, \overline{x}') \wedge q(\overline{x}')$, meaning that from any p-state there is some Λ-transition leading to a q-state. The dual $\widetilde{\mathsf{wpc}}_\Lambda$ of the predicate transformer wpc_Λ is defined by

$$\widetilde{\mathsf{wpc}}_\Lambda(p) \stackrel{\text{def}}{=} \neg\, \mathsf{wpc}_\Lambda(\neg p),$$

yielding

$$\widetilde{\mathsf{wpc}}_\Lambda(p)(\overline{x}) \;\equiv\; \exists \overline{x}'.\, \rho_\Lambda(\overline{x}, \overline{x}') \wedge p(\overline{x}').$$

Hence we can characterise $\{p\}\ \langle\Lambda\rangle\ \{q\}$ as follows:

$$\{p\}\ \langle\Lambda\rangle\ \{q\} \;\equiv\; p \rightarrow \widetilde{\mathsf{wpc}}_\Lambda(q).$$

Notation. Borrowing some notation from modal logic, we sometimes write $[\Lambda]\,q$ and $\langle\Lambda\rangle\,q$ for the assertions $\mathsf{wpc}_\Lambda(q)$ and $\widetilde{\mathsf{wpc}}_\Lambda(q)$, respectively. For a singleton set $\Lambda = \{\lambda\}$, we write $\{p\}\,\lambda\,\{q\}$, $\mathsf{wpc}_\lambda(p), \ldots$ instead of $\{p\}\,\{\lambda\}\,\{q\}$, $\mathsf{wpc}_{\{\lambda\}}(p), \ldots$.

2.6.2 Well-founded Relations and Rankings

Roughly speaking, satisfaction of liveness properties involves reaching some goal (once or repeatedly). Rules for proving this type of property rely on an important auxiliary device for measuring progress towards a goal: well-founded relations and ranking functions.

DEFINITION 2.6.1. (WELL-FOUNDED RELATION) Let W be some set. A binary relation $\prec \subseteq W \times W$ is *well-founded* if there is no infinite descending sequence

$$w_0 \succ w_1 \succ w_2 \succ \cdots$$

of elements $w_0, w_1, w_2, \ldots$ of W. In this case the structure $(W, \succ)$ is called a *well-founded domain*. We write $u \preceq w$ if $u \prec w$ or $u = w$. $\diamondsuit$

DEFINITION 2.6.2. (RANKING FUNCTION) Let Σ be the state space of a system $\mathcal{S}$ and let $(W, \succ)$ be a well-founded domain. A function $\delta : \Sigma \to W$ mapping states to elements of the well-founded domain W is called a *ranking function*. $\diamondsuit$

2.6.3 Manna and Pnueli's System

Manna and Pnueli present in [MP91] a deductive system for proving that a reactive system $\mathcal{S}$ satisfies a LTL property φ. It is composed of a set of rules for three classes of properties. These are *safety properties* (as expressed by formulas of the form $\mathsf{G}\, p$), response properties (as expressed by formulas of the form $\mathsf{G}(p \to \mathsf{F}\, q)$) and reactivity properties (as expressed by a conjunction of formulas of the form $\mathsf{G}\,\mathsf{F}\, p \vee \mathsf{F}\,\mathsf{G}\, q$), where p and q are formulas that may include *past*, but no future temporal operators. As is shown in [LPZ85] any property specifiable in LTL can be rewritten to an equivalent reactivity property (possibly using past operators).

Let φ and p be assertions.

$$\begin{array}{ll} \text{I1.} & \Theta \to \varphi \\ \text{I2.} & \varphi \to p \\ \text{I3.} & \{\varphi\}\, \Lambda\, \{\varphi\} \\ \hline & \mathcal{S} \vdash \mathsf{G}\, p \end{array}$$

Figure 2.1: General invariance rule INV

As an example, we consider rule INV (see also [MP95]) for invariance properties of the form $\mathsf{G}\, p$, where p is an assertion (see Figure 2.1). The rule has three premises. It requires that we find an assertion φ that is implied by the initial condition (premise I1), strengthens the assertion p (premise I2) and is preserved by all system transitions (premise I3). An assertion satisfying I3 is called *inductive*. It is easy to see that this rule is sound: as φ holds in all initial states (I1) and is preserved by all transitions (I3) φ (and hence p by I2) invariantly holds along any computation. Thus, p is an invariant of $\mathcal{S}$.

Rule INV is also complete relative to assertional validity. This is shown by using for φ the strongest possible invariant, namely, an assertion describing exactly the set of Θ-reachable states (the assertion language is required to be expressive enough to formulate such an assertion). Note that even if p holds on all reachable states, it need not be inductive. In particular, it is possible that a transition from some unreachable state leads to a state not satisfying p. For this reason the auxiliary assertion φ is needed in this rule to obtain completeness.

2.6.4 Diagram-Based Verification

Some of the proof rules mentioned above have been recast into the graphical form of *verification diagrams* in [MP94]. A verification diagram $\mathcal{D} = (N, E, \mu)$ for a system $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ is a graph (N, E) with nodes N and edges E, where each node $n \in N$ is labeled by an assertion $\mu(n)$ over X[5]. Depending on the rule some nodes may be terminal nodes with no outgoing edges. We will below identify a node and its labeling.

Verification diagrams describe a set of Hoare triple verification conditions. Suppose a non-terminal node φ has an outgoing edge to each of the successor nodes $\varphi_1, \dots, \varphi_n$. The Hoare triple associated with φ is

$$\{\varphi\}\, \Lambda\, \{\bigvee_{i=1}^{n} \varphi_i\}$$

No Hoare triple is associated with a terminal node. A verification diagram can be seen as an abstract representation of the system to be verified: the nodes are the abstract states that a system traverses during its execution. Verification conditions other than Hoare triples needed in proof rules are formulated externally to the diagrams.

As an example, *invariance diagrams* are verification diagrams with no terminal nodes used to prove properties of the form $\mathsf{G}\,p$, for an assertion p. Suppose $\{\varphi_1, \dots, \varphi_n\}$ is the set of nodes of an invariance diagram for a system $\mathcal{S}$ with initial condition Θ and let $\varphi \stackrel{\text{def}}{=} \bigvee_{i=1}^{n} \varphi_i$. It easy to see that the

[5] We slightly simplify w.r.t. the presentation in [MP94].

validity of all Hoare triples associated with the nodes of the diagram implies that $\mathcal{S} \models \mathsf{G}(\varphi \rightarrow \mathsf{G}\,\varphi)$ ("once φ, always φ") holds. By proving the additional conditions $\Theta \rightarrow \varphi$ and $\varphi \rightarrow p$ (compare with I1 and I2 in rule INV) we can conclude $\mathcal{S} \models \mathsf{G}\,p$.

Generalised Verification Diagrams

The idea of verification diagrams is further developed and generalised in [BMS95, MBSU98] (see also the theses [Uri98, Sip99]). A generalised verification diagram (GVD) $\Psi = (N, N_0, E, \mu, \nu, \mathcal{A})$ for a system $\mathcal{S}$ and temporal property ϕ extends a verification diagram as described above by

- designating a set of initial nodes $N_0 \subseteq N$,
- adding a labeling function ν that labels nodes with (boolean combinations of) assertions appearing in the property formula ϕ, and
- adding an *acceptance condition* $\mathcal{A} \subseteq \mathcal{P}(N)$ in the form of a subset of the strongly connected subgraphs[6] (SCS) of the underlying graph (a Müller-type acceptance condition as known from ω-automata theory, see [Tho90]).

An infinite sequence $\sigma\colon s_0 s_1 \cdots$ of states of $\mathcal{S}$ is *accepted* by the GVD Ψ, if there is a path $\pi\colon n_0 n_1 \cdots$ such that $\inf(\pi) \in \mathcal{A}$ and we have $s_i \models \mu(n_i)$ for all $i \in \omega$. The verification method proposed by GVDs is the following. In order to verify $\mathcal{S} \models \phi$ a GVD Ψ for $\mathcal{S}$ and ϕ is constructed such that

$$\mathcal{L}(\mathcal{S}) \subseteq \mathcal{L}(\Psi) \subseteq \mathcal{L}(\phi)$$

where $\mathcal{L}(\mathcal{S}) \stackrel{\text{def}}{=} \mathcal{C}_{\mathcal{S}}(\Theta)$, $\mathcal{L}(\Psi)$ is the language accepted by the the GVD Ψ and $\mathcal{L}(\phi)$ is the language described by the formula ϕ (that is, the set of state sequences that satisfy ϕ).

The first inclusion states that the diagram faithfully represents all computations of the system. In contrast to the verification diagrams described above, Ψ not only represents the finitary behaviour of $\mathcal{S}$ but also its infinitary (limit) behaviour. In other words, Ψ is a *sound* abstraction of the system $\mathcal{S}$. This inclusion is proved deductively by discharging a set of verification conditions for Ψ. In addition to the Hoare triples associated with Ψ, the *diagram initiation* condition requires that $\Theta \rightarrow \mu(N_0)$, where $\mu(N_0)$ is the disjunction of $\mu(n)$ over all $n \in N_0$ and the *diagram acceptance* condition

[6] a subgraph S of a graph G is *strongly connected* if for every pair of nodes of S there is a path in S connecting them

requires that any non-accepting SCS S is shown to have a fair exit (that is, any run staying in S is unfair) or is well-founded (that is, there is no run and hence no computation staying in S). The latter condition ensures that non-accepting SCSs do not exclude any computation of $\mathcal{S}$ from being accepted by Ψ.

Proving the second inclusion involves showing that $\mu(n) \to \nu(n)$ for each node $n \in N$. Then the inclusion can be model checked algorithmically by abstracting the assertions in $\nu(n)$ and ϕ into atomic propositions.

Deductive Model Checking

The method of deductive model checking [SUM99] (see also the theses [Uri98, Sip99]) also uses diagrams, but follows a different approach. It is based on the successive transformation of diagrams, called *falsification diagrams*. Given a system $\mathcal{S}$ the components of a falsification diagram[7] $\mathcal{G} = (N, N_0, E, \mu, \mathcal{A})$ are the same as the corresponding components of a GVD.

The notion of a sequence of states accepted by $\mathcal{G}$ remains the same as with GVDs. A proof of $\mathcal{S} \models \phi$ starts from an initial falsification diagram $\mathcal{G}_0$ such that $\mathcal{L}(\mathcal{G}_0) = \mathcal{L}(\neg\phi)$ and all edges are labeled by Λ, that is, $\mathcal{G}_0$ represents all sequences of states not satisfying ϕ. The diagram $\mathcal{G}_0$ can be constructed algorithmically. Falsification diagram $\mathcal{G}_{i+1}$ is obtained from $\mathcal{G}_i$ by the application of a *transformation rule*. These are designed in such a way that the invariant

$$\mathcal{L}(\mathcal{S}) \cap \mathcal{L}(\neg\phi) \subseteq \mathcal{L}(\mathcal{G}_i)$$

is preserved. Each diagram $\mathcal{G}_i$ accepts all computations of $\mathcal{S}$ violating ϕ. The idea is to continue the transformations until it can be excluded that there is a counterexample, that is, until a falsification diagram $\mathcal{G}_m$ with $\mathcal{L}(\mathcal{G}_m) = \varnothing$ has been constructed. It then follows from the invariant above that $\mathcal{L}(\mathcal{S}) \subseteq \mathcal{L}(\phi)$, that is, $\mathcal{S} \models \phi$. The method is not guaranteed to terminate for infinite-state systems, but is complete relative to assertional validity.

2.7 Trees, Games and Strategies

Trees

An *A-tree* is a non-empty, prefix-closed subset of A^*. Denote by $\mathrm{Tr}^\infty(A)$ the set of all trees and by $\mathrm{Tr}^*(A)$ the set of finite trees over alphabet A. If $T \subseteq T'$ for two A-trees T and T', we say that T is a *tree-prefix* of T'.

[7] again slightly simplifying w.r.t. the presentation in [Sip99]

Let T be an A-tree. Elements of T are called *nodes*. Define $n_T = \{n \cdot a \in T \mid a \in A\}$, the set of *children* of n in T. A node n is a *leaf* if $n_T = \varnothing$. Otherwise, it is called an *interior* node. The *subtree* at node n is defined as $T/n = \{m/n \mid n \leq m \text{ and } m \in T\}$. A *finite path* in T is a leaf of T. An *infinite path* in T is a sequence $\pi \in A^\omega$ such that all of its finite prefixes are nodes of T. Call $\mathsf{Pth}(T)$ the set of all paths in T.

Games

Let $\mathbb{B}$ be a two-element set whose elements denote players. Ω denotes any player, $\overline{\Omega}$ his opponent. A *game* is a structure $G = (A, T, \lambda, \Omega, W)$, where

- A is an alphabet whose elements are called *configurations*,
- T is an A-tree whose nodes are called *positions* and the paths of which are called *plays* (we write $\mathsf{Play}(G)$ for the set of all plays in T),
- $\lambda : T \to \mathbb{B}$ is a function specifying whose turn it is in each position, and
- $W \subseteq \mathsf{Play}(G)$ is the subset of plays won by player Ω, called the *winning condition* for player Ω.

Player Ω *wins* a play $\mu \in \mathsf{Play}(G)$ if $\mu \in W$, otherwise Player $\overline{\Omega}$ wins. The game G is equivalently specified as $(A, T, \lambda, \overline{\Omega}, \overline{W})$, where $\overline{W} = \mathsf{Play}(G) - W$ is the complement of W. Denote by $\mathsf{Pos}(\Omega) = \{p \in T \mid \lambda(p) = \Omega\}$ the set of Ω-positions, that is, positions where it is player Ω's turn to move.

Strategies

A *(non-deterministic) Ω-strategy* for $G = (A, T, \lambda, \Omega, W)$ is a tree-prefix $\tau \subseteq T$ satisfying the condition

$$p \in \tau,\ \lambda(p) = \overline{\Omega},\ p_\tau \neq \varnothing \ \Rightarrow\ p_\tau = p_T,$$

that is, whenever p is an interior node of τ that is a $\overline{\Omega}$-position then τ contains all the children of p in T. We will write $\mathsf{Strat}(\Omega)$ for the set of all Ω-strategies.

For an Ω-strategy τ and an $\overline{\Omega}$-strategy τ', we define the set of plays resulting from playing these two strategies against each other as

$$\langle \tau \mid \tau' \rangle = \mathsf{Pth}(\tau \cap \tau') \cap \mathsf{Play}(G).$$

We now introduce a couple of properties of strategies. Let τ be an Ω-strategy. Then

- τ is *deterministic* if any Ω-position in τ has at most one child, that is,

$$pa, pb \in \tau,\ \lambda(p) = \Omega \ \Rightarrow\ a = b$$

- τ is *history-free (or memory-less)* if its choices depend only on the last move, i.e.,

$$pa, qa \in \tau,\ \lambda(pa) = \lambda(qa) = \Omega \ \Rightarrow\ (pa)_\tau/(pa) = (qa)_\tau/(qa).$$

- τ is *complete*, if Ω-positions have some successor whenever possible and $\overline{\Omega}$-positions have all successors, that is,

$$\begin{array}{lcl} p \in \tau,\ \lambda(p) = \Omega,\ p_T \neq \varnothing & \Rightarrow & p_\tau \neq \varnothing \\ q \in \tau,\ \lambda(q) = \overline{\Omega} & \Rightarrow & q_\tau = q_T \end{array}$$

- τ is *non-losing,* if all plays resulting from playing against opponent strategies are won by Player Ω, i.e., $\langle \tau \mid \tau' \rangle \subseteq W$ for all $\overline{\Omega}$-strategies τ'.

- τ is a *winning strategy*, if it is complete and non-losing.

We say that player Ω *wins* G if there exists a winning Ω-strategy. A game G is *determined* if one of the players wins.

Useful characterisations of some of the properties of strategies are stated in

PROPOSITION 2.7.1. *Let τ be an Ω-strategy. Then*

(i) τ is complete if and only if all its paths are plays of G, that is, $\mathsf{Pth}(\tau) \subseteq \mathsf{Play}(G)$,

(ii) τ is non-losing if and only if all its paths that are plays of G are won by Player Ω, that is, $\mathsf{Pth}(\tau) \cap \mathsf{Play}(G) \subseteq W$, *and*

(iii) τ is winning if and only if all its paths are plays of G won by Player Ω, that is, $\mathsf{Pth}(\tau) \subseteq W$. □

From the second statement of the proposition we learn that in order to determine whether a Ω-strategy τ is non-losing (losing) it is sufficient to play it against the *vacuous strategy* T and check that each (some) resulting play is won by Player Ω ($\overline{\Omega}$).

Note that any player has a simple (though rather cowardly) non-losing strategy: leave the scene of the game before it is too late!

Chapter 3

LTL Proof Structures

Deductive local model checking is a generalisation of finite-state algorithmic local model checking to infinite state systems. It is a tableau-based proof method that allows us to establish arbitrary CTL* properties of (potentially) infinite-state systems. There is no need to transform formulas into some canonical form and all temporal reasoning is reduced to showing the validity of formulas from the assertion language $\mathcal{L}$. In this chapter we introduce the subsystem for LTL, which will be extended to cover full CTL* in Chapter 5.

Our proof system for LTL consists of a set of *proof rules* that are used to construct graphs, called *LTL proof structures* (a.k.a. tableaux), in a goal-directed way starting from a root node. Some of the rules have side conditions requiring that the validity of an assertion from $\mathcal{L}$ is proved. The construction of a branch stops when we reach a terminal node (an axiom or anti-axiom) or when a rule application generates a subgoal that is already present in the graph constructed so far, in which case we can loop back to that node. As some of the cycles introduced in this way are "bad" ones, a *success criterion* identifies the proof structures that are legal *proofs* of the considered property. Unlike with finite-state systems, where the success criterion can be verified algorithmically by analysing the strongly connected subgraphs of a proof structure and where (at least in the absence of fairness) any unsuccessful cycle immediately produces a counter-example computation, the generality of our approach requires a *well-foundedness argument* for proving that the success criterion holds. We will present an additional proof rule (Rule A(S)) implementing this argument. In summary, the method of deductive local model checking establishes in two steps that a system $\mathcal{S}$ satisfies a LTL property $\mathsf{A}\,\phi$:

1. construct a proof structure Π for $\mathcal{S}$ and $\mathsf{A}\,\phi$, and

2. use Rule A(S) to prove that Π is successful.

The second step can be omitted if there are no anti-axioms in Π and ϕ is syntactically recognisable as describing a safety property, that is, if there are no U-subformulas in ϕ. The method is sound and complete relative to validity of assertions from $\mathcal{L}$ as will be demonstrated in the next chapter.

The outline of the present chapter is as follows. Section 3.1 defines proof structures and the proof rules used to construct them. The rules are then discussed in detail and some useful new rules are derived. The success criterion is defined in Section 3.2 as a property of a system derived from the proof structure and the original system, called the associated system. A syntactic characterisation of the success criterion is formulated, serving as the basis for the design of the proof rule $\mathsf{A}(\mathrm{S})$ for success in Section 3.3. Finally, in section 3.4 we illustrate our proof method with some examples.

3.1 Definition of LTL Proof Structures

Given a transition system specification $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$, a *sequent* in our proof system is of the form $p \vdash \mathsf{A}(\Phi)$ where p is a state assertion over X from our assertion language $\mathcal{L}$ (see Section 2.3) and Φ is a finite, non-empty set of ground-quantified, path-quantifier-free CTL* formulas. A sequent $p \vdash \mathsf{A}(\Phi)$ is called *valid* if all p-computations of $\mathcal{S}$ satisfy the disjunction of the formulas in Φ, that is, $p \models \mathsf{A}(\bigvee_{\phi\in\Phi} \phi)$. Unlike in the finite-state case, where a sequent would have a single state on its left-hand side, the assertion p denotes a (possibly infinite) *set* of states of $\mathcal{S}$.

NOTATION. We will write $p \vdash \mathsf{A}(\phi_1, \dots, \phi_n)$ instead of $p \vdash \mathsf{A}(\{\phi_1, \dots, \phi_n\})$ and use $p \vdash \mathsf{A}(\Phi, \phi)$ to mean $p \vdash \mathsf{A}(\Phi \cup \{\phi\})$. With $\Phi = \{\phi_1, \dots, \phi_n\}$, we also use $\mathsf{X}\,\Phi$ and $\mathsf{X}(\phi_1, \dots, \phi_n)$ as a shorthand for the set $\{\mathsf{X}\,\phi_1, \dots, \mathsf{X}\,\phi_n\}$. Furthermore, for a given sequent $\gamma = p \vdash \mathsf{A}(\Phi)$ we let p_γ and Φ_γ denote p and Φ, respectively.

The *proof rules* (see Table 3.1) are stated and used in a goal-oriented upside-down style with the conclusion above the line and the premises below, with the intention that a rule is applied to a (sub-)goal to reduce it to other subgoals (backwards reasoning). We can distinguish four different groups of rules:

1. The *terminal rules (*$\mathsf{A}(ax)$*,* $\mathsf{A}(nx)$*)* are not, strictly speaking, proper rules. They define the sequents that do not have any successors. A sequent to which rule $\mathsf{A}(ax)$ ($\mathsf{A}(nx)$) is applied is called an *axiom (anti-axiom)*. The axioms and anti-axioms are also called *terminal* sequents.

2. The *propositional rules* ($\mathsf{A}(bsf)$,$\mathsf{A}(\vee)$,$\mathsf{A}(\wedge)$) deal with assertions and with the boolean operators. An assertion can be removed from the sequent in case it does not contribute to its truth (Rule $\mathsf{A}(bsf)$[1]). The boolean rules eliminate the respective boolean connective.

3. The *temporal rules* ($\mathsf{A}(\mathsf{U})$,$\mathsf{A}(\mathsf{V})$,$\mathsf{A}(\mathsf{X})$) deal with the temporal connectives. The rules for Z-formulas exploit the fixed point characterisation of these operators and simply unfold the respective formula. The Next rule $\mathsf{A}(\mathsf{X})$ requires that all right-hand side formulas exhibit a Next operator at the top-level and is applied to all of them at once to eliminate all the Next operators. This is the only rule concerned with state transitions. It requires a Hoare triple to be discharged as part of its side condition.

4. The *Split rule* ($\mathsf{A}(sp)$) is used for case analysis and weakening. It affects only the left-hand side of the sequent by dividing it up into several cases.

In order to prove that for a system $\mathcal{S}$, an assertion Ξ and an LTL formula $\mathsf{A}\,\phi$ the statement $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$ holds, a graph whose nodes are sequents, called a *proof structure for system $\mathcal{S}$ and sequent* $\Xi \vdash \mathsf{A}(\phi)$, is built. The construction is goal-directed, starting from the root sequent $\Xi \vdash \mathsf{A}(\phi)$ and proceeding by successively applying proof rules to the remaining subgoals. The construction of a given branch can be terminated whenever we either reach a terminal node (an axiom or anti-axiom) or a rule application creates a subgoal that already exists in the graph constructed so far, in which case we can loop back to that node. This looping back avoids the construction of infinite branches due to successive unfolding of Z-formulas. Here is the formal definition of a proof structure:

DEFINITION 3.1.1. Given a system $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$, an assertion Ξ and a LTL formula $\mathsf{A}\,\phi$, a *LTL proof structure for system $\mathcal{S}$ and sequent* $\Xi \vdash \mathsf{A}(\phi)$ is a rooted graph

$$\Pi = (\Gamma, \Delta \subseteq \Gamma \times \Gamma, \gamma_r \in \Gamma),$$

where Γ is a finite set of sequents, $\gamma_r = \Xi \vdash \mathsf{A}(\phi)$ is the *root sequent* and for each node $\gamma \in \Gamma$ we require that

(A-SAT) $\quad p_\gamma$ is satisfiable,

[1]The abbreviation '*bsf*' means *basic state formula*, which is either an assertion or a path-quantified formula. The rule applies only to assertions for the moment, but will be extended in Chapter 5 to cover path-quantified formulas.

(A-RCH) γ is reachable from γ_r,

(A-RUL) if γ has $n \geq 0$ successors $\{\gamma_1, \ldots, \gamma_n\} = \{\gamma' \mid (\gamma, \gamma') \in \Delta\}$ then

$$R \; \frac{\gamma}{\gamma_1 \quad \cdots \quad \gamma_n} \; C_R$$

is the correct application of some rule R from Table 3.1, that is, the rules' side condition C_R is satisfied, and

(A-SPL) if $(\gamma, \gamma') \in \Delta$ then rule $\mathsf{A}(sp)$ is *not* applied to both γ and γ'.

If the assertion Ξ is identical with the initial condition Θ of $\mathcal{S}$ then we say that Π is a *proof structure for system* $\mathcal{S}$ *and property* $\mathsf{A}\,\phi$. ♢

Note that we write the side conditions of the proof rules in Table 3.1 in the form $p \models r$ for the respective assertion r, which is in fact equivalent to $\models p \to r$. Recall also that the side condition $p \models [\Lambda]\, q$ of rule $\mathsf{A}(\mathsf{X})$ is equivalent to the Hoare triple $\{p\}\,\Lambda\,\{q\}$. This notational style for the side conditions was chosen, because it will more clearly exhibit the duality with the side conditions of the ELL rules to be introduced in Chapter 5.

DEFINITION 3.1.2. Let Π be a LTL proof structure. We define Γ_R to be the set of sequents of Π where rule R is applied. We also write $\Gamma_{term} \stackrel{\text{def}}{=} \Gamma_{\mathsf{A}(ax)} \cup \Gamma_{\mathsf{A}(nx)}$ for the set of terminal sequents.

A *pseudo-proof structure* $\Pi = (\Gamma, \Delta \subseteq \Gamma \times \Gamma, \gamma_r \in \Gamma)$ for a system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}\,\phi$ is defined in the same way as a proof structure for $\mathcal{S}$ and $\Xi \vdash \mathsf{A}(\phi)$, except that

- the set Γ of sequents need not be finite, and
- condition (A-RUL) of Definition 3.1.1 is relaxed to apply to a sequent γ only if it has at least one successor.

A finite pseudo-proof structure is called a *pre-proof structure*. ♢

Note that in a pseudo-proof structure there may be nodes other than terminal sequents with no successors. Thus, a pre-proof structure can be seen as a partially constructed proof structure.

DEFINITION 3.1.3. A *path* $\pi\colon \gamma_0\gamma_1 \cdots \gamma_i \cdots$ in a pseudo-proof structure Π is a maximal sequence of nodes such that $\gamma_0 = \gamma_r$ and $(\gamma_i, \gamma_{i+1}) \in \Delta$ for all i such that $i + 1 < |\pi|$. ♢

$\mathsf{A}(ax)$	$\dfrac{p \vdash \mathsf{A}(\Phi, q)}{.}$	$p \models q$
$\mathsf{A}(nx)$	$\dfrac{p \vdash \mathsf{A}(q)}{.}$	$p \models \neg q$
$\mathsf{A}(bsf)$	$\dfrac{p \vdash \mathsf{A}(\Phi, q)}{p \vdash \mathsf{A}(\Phi)}$	$p \models \neg q$
$\mathsf{A}(\vee)$	$\dfrac{p \vdash \mathsf{A}(\Phi, \phi_1 \vee \phi_2)}{p \vdash \mathsf{A}(\Phi, \phi_1, \phi_2)}$	
$\mathsf{A}(\wedge)$	$\dfrac{p \vdash \mathsf{A}(\Phi, \phi_1 \wedge \phi_2)}{p \vdash \mathsf{A}(\Phi, \phi_1) \quad p \vdash \mathsf{A}(\Phi, \phi_2)}$	
$\mathsf{A}(\mathsf{U})$	$\dfrac{p \vdash \mathsf{A}(\Phi, \phi_1 \,\mathsf{U}\, \phi_2)}{p \vdash \mathsf{A}(\Phi, \phi_2 \vee (\phi_1 \wedge \mathsf{X}(\phi_1 \,\mathsf{U}\, \phi_2)))}$	
$\mathsf{A}(\mathsf{V})$	$\dfrac{p \vdash \mathsf{A}(\Phi, \phi_1 \,\mathsf{V}\, \phi_2)}{p \vdash \mathsf{A}(\Phi, \phi_2 \wedge (\phi_1 \vee \mathsf{X}(\phi_1 \,\mathsf{V}\, \phi_2)))}$	
$\mathsf{A}(\mathsf{X})$	$\dfrac{p \vdash \mathsf{A}(\mathsf{X}\,\Phi)}{q \vdash \mathsf{A}(\Phi)}$	$p \models [\Lambda]\, q$
$\mathsf{A}(sp)$	$\dfrac{p \vdash \mathsf{A}(\Phi)}{q_1 \vdash \mathsf{A}(\Phi) \quad \cdots \quad q_n \vdash \mathsf{A}(\Phi)}$	$p \models \bigvee_{i=1}^{n} q_i$

Table 3.1: LTL proof rules

3.1.1 Discussion of the Rules

The need to have a set of formulas on the right-hand side of a sequent instead of just a single formula is imposed by the semantics of disjunction. Suppose we want to prove $p \models \mathsf{A}(\phi_1 \vee \phi_2)$. A hypothetic rule like

$$\mathsf{A}(\vee)' \ \frac{p \vdash \mathsf{A}(\phi_1 \vee \phi_2)}{q_1 \vdash \mathsf{A}(\phi_1) \quad q_2 \vdash \mathsf{A}(\phi_2)} \ p \models q_1 \vee q_2$$

would be sound, but put a serious threat on the completeness of the proof system. Supposing $p \models \mathsf{A}(\phi_1 \vee \phi_2)$, it is essential for completeness[2] that we can find assertions q_1 and q_2 such that $q_1 \models \mathsf{A}\,\phi_1$, $q_2 \models \mathsf{A}\,\phi_2$ and $p \rightarrow q_1 \vee q_2$ is valid, but this is not possible in general. Consider a state $s \models p$. Then $s \models \mathsf{A}(\phi_1 \vee \phi_2)$, but neither $s \models \mathsf{A}\,\phi_1$ nor $s \models \mathsf{A}\,\phi_2$ need to hold, since there could be s-computations σ_1 and σ_2 with σ_1 satisfying ϕ_1 but not ϕ_2 and σ_2 satisfying ϕ_2 but not ϕ_1. In this case, the required assertions q_1 and q_2 do not exist. This is the reason for making the right-hand side of a sequent a finite set of formulas which is to be interpreted as a disjunction over all set members. This naturally leads to rule $\mathsf{A}(\vee)$ in Table 3.1 which simply discards the disjunction symbol.

The conjunction rule $\mathsf{A}(\wedge)$ just splits the two conjuncts as expected. The assertion rule $\mathsf{A}(bsf)$ eliminates an assertion q in case it does not contribute to the truth of the disjunction of the formulas on the right-hand side, that is, if no state can at the same time satisfy the left-hand side predicate p and assertion q on the right-hand side. Note that, by the definition of a sequent, the set of formulas Φ must be non-empty. In general, an application of $\mathsf{A}(bsf)$ will have to be preceded by an application of the Split rule $\mathsf{A}(sp)$ in order to separate the states satisfying q from those that do not (see also derived rule $\mathsf{A}(bsf)'$ below).

The rules $\mathsf{A}(\mathsf{U})$ and $\mathsf{A}(\mathsf{V})$ for Z-formulas exploit the fixpoint characterisations of these operators. The following equivalences hold:

$$\begin{aligned} \phi_1 \,\mathsf{U}\, \phi_2 &\equiv \phi_2 \vee (\phi_1 \wedge \mathsf{X}(\phi_1 \,\mathsf{U}\, \phi_2)) \\ \phi_1 \,\mathsf{V}\, \phi_2 &\equiv \phi_2 \wedge (\phi_1 \vee \mathsf{X}(\phi_1 \,\mathsf{V}\, \phi_2)) \end{aligned}$$

The application of a Z-rule simply unfolds the respective Z-formula. We denote by $\mathsf{unf}(\phi_1 \,\mathsf{Z}\, \phi_2)$ the respective unfolding (right-hand side formula above).

Case analysis is implemented in the Split rule $\mathsf{A}(sp)$. The $n \geq 1$ cases need not be disjoint. It is the only rule not modifying the right-hand side of sequents. We note the following special case of the Split rule, called

[2] Completeness proofs most often rely on the backwards preservation of satisfaction.

weakening

$$\mathsf{A}(wk)\ \frac{p \vdash \mathsf{A}(\Phi)}{q \vdash \mathsf{A}(\Phi)}\ p \models q$$

Weakening can also be useful for replacing a left-hand side assertion by an equivalent one.

The Next rule $\mathsf{A}(\mathsf{X})$ is the only one concerned with state transitions. It reflects the fact that proving $p \models \mathsf{A}(\bigvee_{\phi\in\Phi} \mathsf{X}\,\phi)$ can be reduced to proving $q \models \mathsf{A}(\bigvee_{\phi\in\Phi} \phi)$ provided that any system transition starting in a p-state leads to a q-state as required by the side condition. The rule is not backward sound in general. Choosing q to be the strongest post-condition $\mathsf{spc}_\Lambda(p)$ w.r.t. all system transitions leads to the following special case of $\mathsf{A}(\mathsf{X})$ which is forward and backward sound

$$\mathsf{A}(\mathsf{X})_0\ \frac{p \vdash \mathsf{A}(\mathsf{X}\,\Phi)}{\mathsf{spc}_\Lambda(p) \vdash \mathsf{A}(\Phi)}$$

Rule $\mathsf{A}(\mathsf{X})_0$ would be sufficient for completeness and rule $\mathsf{A}(\mathsf{X})$ is derivable from it with the help of weakening (rule $\mathsf{A}(wk)$ above). However, we have decided to include the more general rule $\mathsf{A}(\mathsf{X})$ for convenience.

3.1.2 The Split Condition (A-SPL)

Condition (A-SPL) in Definition 3.1.1 disallows two successive applications of the Split rule and deserves some explanation. The Split rule $\mathsf{A}(sp)$ is quite a powerful rule as it does not only allow us to reason by case analysis, but also to generalise the sequent to which it is applied (the side condition being an implication in contrast to an equivalence, the latter being sufficient for pure case analysis). It turns out that when applied without the restriction imposed by condition (A-SPL), this rule is even too powerful in the sense that the proof system would become unsound, as is illustrated by the following

EXAMPLE 3.1.4. Consider the three node proof structure displayed below.

$$\begin{tikzcd}
& \boxed{\Xi \vdash \mathsf{A}(\phi)} \arrow[dl, bend left] \arrow[dr, bend right] & \\
\boxed{\Xi \wedge p \vdash \mathsf{A}(\phi)} \arrow[ur, bend left] & & \boxed{\Xi \wedge \neg p \vdash \mathsf{A}(\phi)} \arrow[ul, bend right]
\end{tikzcd}$$

Here the Split rule has been applied to the root sequent $\Xi \vdash \mathsf{A}(\phi)$ to obtain two successor sequents $\Xi \wedge p \vdash \mathsf{A}(\phi)$ and $\Xi \wedge \neg p \vdash \mathsf{A}(\phi)$ which are then reconnected to the root sequent by weakening. Such circular propositional reasoning is obviously unsound and leads to an inconsistent system. ♣

Note that condition (A-SPL) also precludes self-loops on nodes where the Split rule is applied. More generally, soundness requires that every infinite path through a proof structure must contain an infinite number of applications of the Next rule $\mathsf{A}(\mathsf{X})$, each application corresponding to a transition of the system. The reason is that the Next rule is the only one making (qualitative) time advance, thus effectively introducing the temporal aspect into the reasoning.

The following Lemma identifies the successive application of the Split rule $\mathsf{A}(sp)$ as the malefactor and shows that condition (A-SPL) is sufficient to avoid circular reasoning "on the spot".

LEMMA 3.1.5. (TEMPORAL CONSISTENCY) *Let* Π *be a pseudo-proof structure for system* $\mathcal{S}$ *and sequent* $\Xi \vdash \mathsf{A}(\phi)$ *and let* π *be an infinite path in* Π. *Then rule* $\mathsf{A}(\mathsf{X})$ *is applied infinitely often on* π.

PROOF. Let $\pi\colon \gamma_0 \cdots \gamma_j \cdots$ be an infinite path in a pseudo-proof structure Π. Suppose for a contradiction that rule $\mathsf{A}(\mathsf{X})$ is applied only finitely many times on π. Define a ranking function r on sequents by $r(p \vdash \mathsf{A}(\Phi)) = \sum_{\phi \in \Phi} r_0(\phi)$, where the function r_0 is inductively defined on formulas by

$$\begin{aligned}
&r_0(q) = r_0(\mathsf{X}\,\phi) = 1\\
&r_0(\phi_1 \wedge \phi_2) = r_0(\phi_1 \vee \phi_2) = r_0(\phi_1) + r_0(\phi_2) + 1\\
&r_0(\phi_1 \mathsf{V}\, \phi_2) = r_0(\phi_1 \mathsf{U}\, \phi_2) = r_0(\phi_1) + r_0(\phi_2) + 4
\end{aligned}$$

An inspection of the proof rules shows that all rules except $\mathsf{A}(sp)$ and $\mathsf{A}(\mathsf{X})$ decrease the ranking r. Rule $\mathsf{A}(sp)$ applied to some γ_i leaves the rank constant from γ_i to γ_{i+1}, but by condition (A-SPL) in Definition 3.1.1 it is always followed by some rule other than $\mathsf{A}(sp)$ applied to γ_{i+1}. By assumption, rule $\mathsf{A}(\mathsf{X})$ is never applied on π from some point k on. It follows that the ranking r decreases infinitely often along π^k, contradicting the well-foundedness of the natural numbers. □

Translated to (pre-)proof structures this lemma says that there is no cycle without at least one application of the Next rule $\mathsf{A}(\mathsf{X})$ appearing on it.

Finally, we note that condition (A-SPL) does not really restrict the sound use of the Split rule, since successive applications not introducing cycles can always be merged into a single application.

3.1.3 Derived Rules

A number of interesting rules can be derived from the basic ones in Table 3.1. Some of these are displayed in Table 3.2. The derived rule for weakening ($\mathsf{A}(wk)$) has already been discussed above.

$\mathsf{A}(wk)$	$\dfrac{p \vdash \mathsf{A}(\Phi)}{q \vdash \mathsf{A}(\Phi)}$	$p \models q$
$\mathsf{A}(bsf)'$	$\dfrac{p \vdash \mathsf{A}(\Phi, q)}{p \wedge \neg q \vdash \mathsf{A}(\Phi)}$	$p \wedge q, p \wedge \neg q$ both satisfiable
$\mathsf{A}(\mathsf{U})'$	$\dfrac{p \vdash \mathsf{A}(\Phi, \phi_1 \,\mathsf{U}\, \phi_2)}{p \vdash \mathsf{A}(\Phi, \phi_1, \phi_2) \quad p \vdash \mathsf{A}(\Phi, \phi_2, \mathsf{X}(\phi_1 \,\mathsf{U}\, \phi_2))}$	
$\mathsf{A}(\mathsf{V})'$	$\dfrac{p \vdash \mathsf{A}(\Phi, \phi_1 \,\mathsf{V}\, \phi_2)}{p \vdash \mathsf{A}(\Phi, \phi_2) \quad p \vdash \mathsf{A}(\Phi, \phi_1, \mathsf{X}(\phi_1 \,\mathsf{V}\, \phi_2))}$	
$\mathsf{A}(\mathsf{F})$	$\dfrac{p \vdash \mathsf{A}(\Phi, \mathsf{F}\,\psi)}{p \vdash \mathsf{A}(\Phi, \psi, \mathsf{X}\,\mathsf{F}\,\psi)}$	
$\mathsf{A}(\mathsf{G})$	$\dfrac{p \vdash \mathsf{A}(\Phi, \mathsf{G}\,\psi)}{p \vdash \mathsf{A}(\Phi, \psi) \quad p \vdash \mathsf{A}(\Phi, \mathsf{X}\,\mathsf{G}\,\psi)}$	
$\mathsf{A}(\mathsf{W})$	$\dfrac{p \vdash \mathsf{A}(\Phi, \phi_1 \,\mathsf{W}\, \phi_2)}{p \vdash \mathsf{A}(\Phi, \phi_1, \phi_2) \quad p \vdash \mathsf{A}(\Phi, \phi_2, \mathsf{X}(\phi_1 \,\mathsf{W}\, \phi_2))}$	
$\mathsf{A}(\mathsf{X})'$	$\dfrac{p \vdash \mathsf{A}(\mathsf{X}\,\Phi)}{q_1 \vdash \mathsf{A}(\Phi) \quad \cdots \quad q_n \vdash \mathsf{A}(\Phi)}$	$p \models [\Lambda] \bigvee_{i=1}^{n} q_i$

Table 3.2: Derived LTL rules

Rule $\mathsf{A}(bsf)'$ is derivable from rule $\mathsf{A}(bsf)$ and the Split rule $\mathsf{A}(sp)$. It allows one to shift the assertion q from the right to the left-hand side of the sequent.

The unfolded formula in the two Z-rules $\mathsf{A}(\mathsf{U})$ and $\mathsf{A}(\mathsf{V})$ can be further decomposed by applying rules $\mathsf{A}(\vee)$ and $\mathsf{A}(\wedge)$, yielding their more economic versions $\mathsf{A}(\mathsf{U})'$ and $\mathsf{A}(\mathsf{V})'$. In rules $\mathsf{A}(\mathsf{F})$ and $\mathsf{A}(\mathsf{G})$ these are further specialised for the derived operators 'eventually' and 'always'. Rule $\mathsf{A}(\mathsf{W})$ is sound, but — strictly speaking — *not* derivable in our system. However, the following rule is derivable:

$$\frac{p \vdash \mathsf{A}(\Phi, \phi_2, (\mathsf{X}\,\phi_2)\,\mathsf{V}\,\phi_1)}{p \vdash \mathsf{A}(\Phi, \phi_1, \phi_2) \quad p \vdash \mathsf{A}(\Phi, \phi_2, \mathsf{X}(\phi_2, (\mathsf{X}\,\phi_2)\,\mathsf{V}\,\phi_1))}$$

By a slight abuse of notation, writing $\phi_1\,\mathsf{W}\,\phi_2$ for $\phi_2, (\mathsf{X}\,\phi_2)\,\mathsf{V}\,\phi_1$ in sequents (which is semantically justified), we get exactly rule $\mathsf{A}(\mathsf{W})$.

Finally, rule $\mathsf{A}(\mathsf{X})'$ is a variant of the next rule derivable from $\mathsf{A}(\mathsf{X})$ and $\mathsf{A}(sp)$. We will use these derived rules freely in our examples.

A Simple Example

EXAMPLE 3.1.6. Consider the very simple system $\mathcal{S}_0$ with a single natural number variable x, initial condition $\Theta_0 \stackrel{\text{def}}{=} x = M$ for some given parameter $M \geq 0$ and with the only possible transition incrementing x by an arbitrary positive number n:

$$\rho_{inc} \stackrel{\text{def}}{=} \exists n > 0.\, x' = x + n$$

The property we want to prove for this system is expressed by the invariance formula $\psi_0 \stackrel{\text{def}}{=} \mathsf{A}\,\mathsf{G}(x \geq M)$. Figure 3.1 shows a proof structure for $\mathcal{S}_0$ and ψ_0.

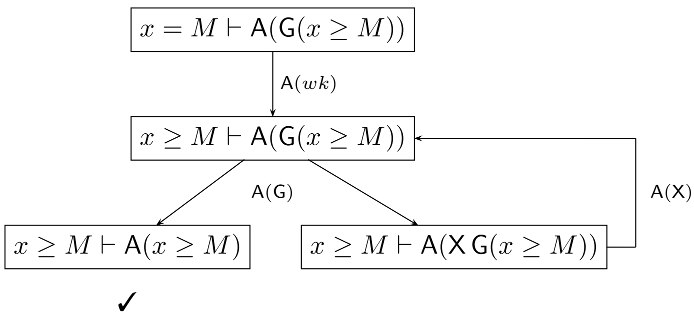

Figure 3.1: Proof structure Π_0 for $\mathcal{S}_0$ and ψ_0

Note the use of weakening in the first step, generalising the statement to be proved. The verification conditions generated by this proof structure are

$x = M \models x \geq M$	from weakening rule $\mathsf{A}(wk)$
$x \geq M \models x \geq M$	from axiom rule $\mathsf{A}(ax)$
$x \geq M \models [inc]\, x \geq M$	from Next rule $\mathsf{A}(\mathsf{X})$

and are all easily discharged. As we will see later in this chapter, this proof structure is indeed a valid *proof* for $\mathcal{S}_0 \models \psi_0$. ♣

3.2 The Success Criterion

Similarly to finite-state model checking, not every proof structure for system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}\,\phi$ can be considered as a valid proof of $\mathcal{S}, \Xi \models \phi$. This is because proof structures generally contain cycles. In this section, we first define a notion of success for paths and then lift it to proof structures. Only successful proof structures will be considered as legal *proofs*.

3.2.1 Successful Paths

Let Π be a proof structure for $\mathcal{S}$ and $\Xi \vdash \mathsf{A}\,\phi$. The significance of successful paths is best stated negatively as:

> a path π of Π is *un*successful precisely if any computation following it provides a counter-example to $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$

Informally, a computation σ follows a path π if σ can be laid along π such that transitions on σ are matched with applications of rule $\mathsf{A}(\mathsf{X})$ and the states on σ satisfy the constraints imposed by the left-hand side assertions of the sequents on π.

For instance, any computation σ of system $\mathcal{S}_0$ of Example 3.1.6 follows the (unique) infinite path in proof structure Π_0 of Figure 3.1, since it starts in the initial state (where $x = M$) and all states of σ satisfy $x \geq M$.

DEFINITION 3.2.1. For a Z-formula ψ define the set U_ψ of *unfolding forms* of ψ by

$$U_\psi \stackrel{\text{def}}{=} \{\theta \mid \psi \preccurlyeq \theta \preccurlyeq \mathsf{unf}(\psi)\}.$$

$\diamondsuit$

An unfolding form of a Z-formula ψ is thus any formula that is a subformula of the unfolding of ψ and contains at the same time ψ as a subformula.

DEFINITION 3.2.2. Let $\Pi = (\Gamma, \Delta, \gamma_r)$ be a proof structure for system $\mathcal{S}$. Define the set of sequents Q_ψ for a Z-formula $\psi = \phi_1 \,\mathsf{Z}\, \phi_2$ by

$$Q_\psi \stackrel{\text{def}}{=} \{\gamma \in \Gamma \mid \Phi_\gamma \cap U_\psi \neq \varnothing \,\wedge\, \phi_2 \notin \Phi_\gamma\},$$

that is, Q_ψ consists of those sequents of Π the right-hand side of which contains some unfolding form of ψ, but not its second subformula ϕ_2. $\diamondsuit$

For example, for $\psi = \phi_1 \,\mathsf{V}\, \phi_2$ we get $U_\psi = \{\psi, \mathsf{X}\,\psi, \phi_1 \vee \mathsf{X}\,\psi, \phi_2 \wedge (\phi_1 \vee \mathsf{X}\,\psi)\}$ and $p \vdash \mathsf{A}(\mathsf{X}\,\psi, \phi_1 \wedge \phi_2) \in Q_\psi$, but $q \vdash \mathsf{A}(\phi_1 \vee \mathsf{X}\,\psi, \phi_2) \notin Q_\psi$ as well as $q \vdash \mathsf{A}(\phi_1 \wedge \mathsf{X}\,\psi, \phi_2 \vee \psi) \notin Q_\psi$.

DEFINITION 3.2.3. Let Π be a proof structure Π for $\mathcal{S}$ and $\Xi \vdash \mathsf{A}(\phi)$. A path π in Π is *successful* if one of the following holds:

(a) π is finite and ends in an axiom, or

(b) π is infinite and there is a V-formula $\psi \in \mathsf{V}(\phi)$ such that $\mathsf{inf}(\pi) \subseteq Q_\psi$.

$\diamondsuit$

Recall from Section 2.4 that $\mathsf{V}(\phi)$ is the set of V-subformulas of ϕ and observe that, since proof structures are finite, $\mathsf{inf}(\pi) \subseteq Q_\psi$ means that from some position in π on all sequents are in Q_ψ.

Let us try to give an intuitive motivation for this definition. Suppose Π is a proof structure for system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}(\phi)$. The first part of the definition is quite obvious: any path ending in an axiom should count as successful, as it provides, by soundness of the proof rules, a definitive contribution to the truth of the root sequent.

The second part states that an infinite path π is successful if, from some point on, some unfolding form of a V-subformula $\phi_1 \mathsf{V} \phi_2$ of the original property ϕ appears in every sequent, but without ϕ_2 ever occurring beyond that point. The absence of ϕ_2 implies that $\phi_1 \mathsf{V} \phi_2$ infinitely often regenerates itself along the path π by unfolding and subsequent elimination of the boolean and next connectives. Let us call *ω-regenerated* (along π) any Z-subformula with this indefinite unfolding property. But why should such a path π count as successful and not others?

Although the definitive answer of this question has to be deferred to the next chapter, we can say at this point that the ω-regeneration of a U-formula $\psi_1 \mathsf{U} \psi_2$ along a path π corresponds to indefinitely delaying the satisfaction of ψ_2, which is in contradiction to the semantics of the Until operator, while ω-regenerating a V-formula $\phi_1 \mathsf{V} \phi_2$ on the other hand corresponds to the possibility that ϕ_2 always holds, which is a possible way to satisfy $\phi_1 \mathsf{V} \phi_2$. As will be shown in the next chapter, some Z-formula is ω-regenerated along any infinite path. Since all propositions are eliminated on an infinite path π on the basis of their falsity (see side condition of rule $\mathsf{A}(bsf)$), there should better be a V-formula that is ω-regenerated along π if π is supposed to be successful. Otherwise, any computation following π provides a counter-example.

3.2.2 A Tentative Success Criterion for Proof Structures

One could now be tempted to define a proof structure to be successful if all its paths are successful. Let us examine this definition in this section. This notion of success is sound. Disregarding fairness issues, it is also complete in the case of finite-state model checking [BCG95]: any unsuccessful path produces a counter-example, namely, the computation that can be extracted from it. However, in the infinite state case, it is insufficient and would make the proof system incomplete as the following two examples illustrate.

Example 3.2.4. Let the system $\mathcal{S}_1$ have a single natural number variable x with initial condition $\Theta_1 \stackrel{\text{def}}{=} \mathsf{true}$ and transition relations ρ_{dec}, ρ_{zero} and ρ_{ten} defined by

$$\begin{array}{lcl} \rho_{dec} & \stackrel{\text{def}}{=} & x > 0 \wedge x' = x - 1 \\ \rho_{zero} & \stackrel{\text{def}}{=} & x = 0 \wedge x' = x \\ \rho_{ten} & \stackrel{\text{def}}{=} & x = 10 \wedge x' = x \end{array}$$

A proof structure Π_1 for this system and property $\phi_1 \stackrel{\text{def}}{=} \mathsf{A\,F}(x = 0)$ is shown in Figure 3.2 below. Clearly, ϕ_1 does not hold for $\mathcal{S}_1$. As required for soundness, proof structure Π_1 is unsuccessful according to our tentative definition of success, since the only infinite path induced by the cycle in Π_1 is unsuccessful.

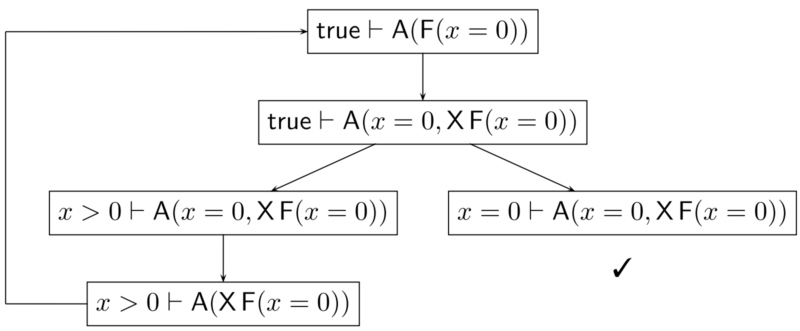

Figure 3.2: Proof structure Π_1 for $\mathcal{S}_1$ ($\mathcal{S}_1'$) and $\phi_1 \stackrel{\text{def}}{=} \mathsf{A\,F}(x = 0)$

Now consider the system $\mathcal{S}_1'$ obtained from $\mathcal{S}_1$ by removing transition *ten*. For this modified system property ϕ_1 holds. Proof structure Π_1 is still an unsuccessful proof structure for system $\mathcal{S}_1'$ and property ϕ_1. The difference between the two cases lies in the way that system computations can follow the infinite path π arising from the cycle in the proof structure. Any computation σ of system $\mathcal{S}_1$ ending in the cycle at $x = 10$ can follow π indefinitely (since all states on σ satisfy $x > 0$). On the other hand, every computation σ' of system $\mathcal{S}_1'$ leaves π towards the axiom when it eventually reaches a state where $x = 0$, as required to fulfill the eventuality.

It is not difficult to see that when working with our tentative definition of success there does not exist any successful proof structure for $\mathcal{S}_1'$ and ϕ_1

at all. Since there is no V-subformula in ϕ_1, such a proof structure must not include an infinite path. Since the property is to be proved for an infinite set of initial states, there is no hope to find a cycle-free, successful proof structure. Nonetheless, for completeness, Π_1 should count as successful. ♣

The second example shows that not even all anti-axioms need to be considered as harmful, as there might be no computation (prefix) following a path leading to it.

EXAMPLE 3.2.5. Let system $\mathcal{S}_2$ be the same as system $\mathcal{S}_1$ of the previous example except that the initial condition is strengthened to $\Theta_2 \stackrel{\text{def}}{=} x > M$ for some parameter $M \geq 0$. Figure 3.3 shows proof structure Π_2 for system $\mathcal{S}_2$ and the simple one-step property $\phi_2 \stackrel{\text{def}}{=} \mathsf{A}\,\mathsf{X}(x \geq M)$.

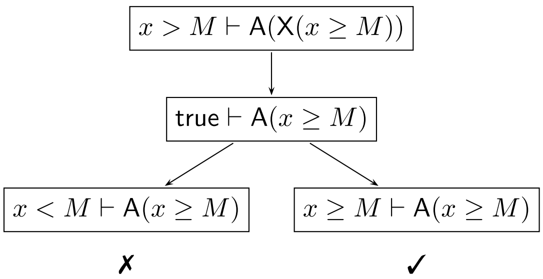

Figure 3.3: Proof structure Π_2 for system $\mathcal{S}_2$ and property $\phi_2 \stackrel{\text{def}}{=} \mathsf{A}\,\mathsf{X}(x \geq M)$

In this proof structure, the Next rule is applied to the root sequent thereby weakening the left-hand side assertion to true. Then Split is applied yielding an anti-axiom and an axiom. Thus, according to our preliminary definition, Π_2 is unsuccessful due to the presence of the anti-axiom. However, the property obviously holds for $\mathcal{S}_2$, so Π_2 should count as successful.[3]

Observe that, since the assertion $x \geq M$ is the strongest post-condition of $x > M$ w.r.t. the transition relation, it is clear that there can be no computation of $\mathcal{S}_2$ following the path to the anti-axiom, that is, reaching a state satisfying $x < M$ after only one transition. ♣

As these examples clearly demonstrate, the infinite-state case requires a refined, weaker notion of success. Of course, the intricacy stems from the fact that the left-hand side assertion of each sequent describes (potentially) infinite set of states. As a consequence, given a path in a proof structure, it is not only possible that infinitely many computations follow it, but also

[3] At least for the strong completeness result we are striving at, which states that *any* proof structure for a system $\mathcal{S}$ and LTL property $\mathsf{A}\,\psi$ is successful, provided $\mathcal{S} \models \mathsf{A}\,\psi$.

that no computation at all follows the path. In the latter case, there is no counter-example computation and the unsuccessful path is thus harmless. Hence, a proof structure should be defined to be successful if

(A-SUC) every path *that is followed by some computation* is successful

This notion of success is now formalised in the next section.

3.2.3 Trails and Success for Proof Structures

In order to formally define the success criterion (A-SUC) for proof structures, we have to make more precise what it means for a run or computation to follow a path in a proof structure. To this end, we will define a system derived from a given original system $\mathcal{S}$ and a proof structure Π for $\mathcal{S}$, called the associated system, a run of which is called a *trail* of Π and combines a path of Π with a run of $\mathcal{S}$ that satisfies the constraints imposed by the assertions along Π. The notion of Π-fairness is then introduced for trails and success of a proof structure defined in terms of Π-fair trails.

The System Associated with a Proof Structure

To this end, we first slightly extend proof structure Π by adding an edge from every axiom to the *pseudo-sequent* $\top \overset{def}{=} \mathsf{true} \vdash \mathsf{true}$ and an edge from every anti-axiom to the *pseudo-sequent* $\bot \overset{def}{=} \mathsf{true} \vdash \mathsf{false}$ plus self-loops on $\top$ and $\bot$.

DEFINITION 3.2.6. Let $\Pi = (\Gamma, \Delta, \gamma_r)$ be a LTL proof structure. Define

$$\begin{array}{rcl}
\Gamma^+ & \overset{\text{def}}{=} & \Gamma \cup \{\top \mid \Gamma_{\mathsf{A}(ax)} \neq \varnothing\} \cup \{\bot \mid \Gamma_{\mathsf{A}(nx)} \neq \varnothing\} \\
\Delta^+ & \overset{\text{def}}{=} & \Delta \cup \Delta_\top \cup \Delta_\bot \\
\Delta_\top & \overset{\text{def}}{=} & \{(\gamma, \top) \in \Gamma^+ \times \Gamma^+ \mid \gamma \in \Gamma_{\mathsf{A}(ax)} \cup \{\top\}\} \\
\Delta_\bot & \overset{\text{def}}{=} & \{(\gamma, \bot) \in \Gamma^+ \times \Gamma^+ \mid \gamma \in \Gamma_{\mathsf{A}(nx)} \cup \{\bot\}\}
\end{array}$$

$\diamondsuit$

A pseudo-sequent and the corresponding edges are added only in case that the corresponding terminal sequent appears in the proof structure. Suppose that we have a fixed enumeration of the (pseudo-)sequents in Γ^+ and let in the following $\lceil\gamma\rceil$ be the number assigned to γ. For a set $\Gamma_0 \subseteq \Gamma^+$ we define $\lceil\Gamma_0\rceil \overset{\text{def}}{=} \{\lceil\gamma\rceil \mid \gamma \in \Gamma_0\}$.

DEFINITION 3.2.7. Let $\Pi = (\Gamma, \Delta, \gamma_r)$ be a proof structure for system $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ and sequent $\Xi \vdash \mathsf{A}\,\phi$. The *system*

$$\mathcal{S}^\Pi = (X^\Pi, \Sigma^\Pi, \{\rho^\Pi_{(\gamma,\lambda,\gamma')} \mid (\gamma, \lambda, \gamma') \in \Lambda^\Pi\}, \Theta^\Pi, \mathcal{F}^\Pi)$$

associated with Π is defined by

$$\begin{array}{lcl}
X^\Pi & \stackrel{\text{def}}{=} & X \cup \{K\} \qquad \text{where } K \in V - X \\
\Sigma^\Pi & \stackrel{\text{def}}{=} & \Sigma \text{ extended by mapping } K \text{ to elements of } \lceil \Gamma^+ \rceil \\
\Theta^\Pi & \stackrel{\text{def}}{=} & (K = \lceil \gamma_r \rceil) \wedge \Xi \\
\\
\Lambda^\Pi & \stackrel{\text{def}}{=} & \Lambda^\Pi_{sys} \cup \Lambda^\Pi_{log} \\
\Lambda^\Pi_{sys} & \stackrel{\text{def}}{=} & \{(\gamma, \lambda, \gamma') \mid (\gamma, \gamma') \in \Delta^+, \gamma \in \Gamma_{sys}, \lambda \in \Lambda\} \\
\Lambda^\Pi_{log} & \stackrel{\text{def}}{=} & \{(\gamma, =, \gamma') \mid (\gamma, \gamma') \in \Delta^+, \gamma \notin \Gamma_{sys}\}
\end{array}$$

where $\Gamma_{sys} \stackrel{\text{def}}{=} \Gamma_{\mathsf{A(X)}} \cup \Gamma_{term} \cup \{\bot, \top\}$, and

$$\begin{array}{lcll}
\rho^\Pi_{(\gamma,\lambda,\gamma')} & \stackrel{\text{def}}{=} & \widehat{p_\gamma}(\overline{x}, K) \wedge \widehat{p_{\gamma'}}(\overline{x}', K') \wedge \rho_\lambda(\overline{x}, \overline{x}') & \text{for } (\gamma, \lambda, \gamma') \in \Lambda^\Pi_{sys} \\
\rho^\Pi_{(\gamma,=,\gamma')} & \stackrel{\text{def}}{=} & \widehat{p_\gamma}(\overline{x}, K) \wedge \widehat{p_{\gamma'}}(\overline{x}', K') \wedge \overline{x}' = \overline{x} & \text{for } (\gamma, =, \gamma') \in \Lambda^\Pi_{log}
\end{array}$$

where $\widehat{p_\gamma} \stackrel{\text{def}}{=} (K = \lceil \gamma \rceil) \wedge p_\gamma$, and finally

$$\begin{array}{lcl}
\mathcal{F}^\Pi & \stackrel{\text{def}}{=} & (P^\Pi, W^\Pi, F^\Pi) \\
P^\Pi & \stackrel{\text{def}}{=} & \{\pi_2^{-1}(\Lambda') \mid \Lambda' \in P\} \cup \{\Lambda^\Pi_{log}\} \\
W^\Pi & \stackrel{\text{def}}{=} & \{\pi_2^{-1}(\Lambda_w) \mid \Lambda_w \in W\} \\
F^\Pi & \stackrel{\text{def}}{=} & \{\pi_2^{-1}(\Lambda_f) \mid \Lambda_f \in F\}
\end{array}$$

where $\pi_2^{-1}(\Lambda') \stackrel{\text{def}}{=} \{(\gamma, \lambda, \gamma') \in \Lambda^\Pi \mid \lambda \in \Lambda'\}$ for $\Lambda' \subseteq \Lambda$. $\diamondsuit$

The state space of $\mathcal{S}^\Pi$ extends the state space of $\mathcal{S}$ with an additional control variable K indicating the position in the proof structure. The label set Λ^Π is divided into a set Λ^Π_{sys} of system-related transitions and a set Λ^Π_{log} of "logical" transitions. Each transition $(\gamma, \lambda, \gamma') \in \Lambda^\Pi_{sys}$ departs from a Γ_{sys}-sequent γ (that is, γ is an $\mathsf{A(X)}$-sequent, a terminal or a pseudo-sequent) and involves the underlying system transition λ. On the other hand, a transition $(\gamma, =, \gamma') \in \Lambda^\Pi_{log}$ departs from a sequent other than a Γ_{sys}-sequent and requires that the values of system variables are preserved. All transitions, in addition to moving control K along an edge in $(\gamma, \gamma') \in \Delta^+$, are constrained by the left-hand side assertions p_γ and $p_{\gamma'}$ appearing in the (pseudo-)sequents γ and γ'. Note that, as these assertions are true for the pseudo-sequents $\top$ and

$\bot$, once control variable K has reached a pseudo-sequent on a run of $\mathcal{S}^\Pi$, its further behaviour is governed essentially by the underlying system transitions only.

The partition P^Π of the fairness constraint $\mathcal{F}^\Pi$ extends the original partition P to account for the modification of the transition labeling set: each set $\Lambda_0 \in P$ is turned into a set $\Lambda_0^\Pi \in P^\Pi$ containing all $(\gamma, \lambda, \gamma') \in \Lambda^\Pi$ such that $\lambda \in \Lambda_0$. The transitions in Λ_{log}^Π have no equivalent in $\mathcal{S}$, so this set appears as an additional element of P^Π. The fairness sets W^Π and F^Π are derived in the same way from W and F. Note that there is no fairness constraint on $\Lambda_{log}^\Pi \in P^\Pi$.

Trails and Π*-Fairness*

For the rest of this section, unless otherwise stated, let $\Pi = (\Gamma, \Delta, \gamma_r)$ be an arbitrary but fixed proof structure for system $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ and sequent $\Xi \vdash \mathsf{A}\,\phi$. Furthermore, let $\mathcal{S}^\Pi$ be the system associated with Π, the components of $\mathcal{S}^\Pi$ being denoted as in Definition 3.2.7.

DEFINITION 3.2.8. A *trail of the proof structure* Π is a Θ^Π-run of its associated transition system $\mathcal{S}^\Pi$. $\diamondsuit$

Loosely speaking, a trail of Π knits together a Ξ-run of the system and a path in the proof structure. We can now define two projections on trails of Π, one to the underlying run in $\mathcal{S}$ and the other to the underlying path in Π. Define the maps $h_\Sigma \colon \Sigma^\Pi \to \Sigma \cup \{\epsilon\}$ and $h_\Gamma \colon \Sigma^\Pi \to \Gamma \cup \{\epsilon\}$ by

$$h_\Sigma(t) \stackrel{\text{def}}{=} \begin{cases} t|_X & \text{if } t(K) \in \lceil \Gamma_{sys} \rceil \\ \epsilon & \text{otherwise} \end{cases}$$

$$h_\Gamma(t) \stackrel{\text{def}}{=} \begin{cases} \gamma & \text{if } t(K) = \lceil \gamma \rceil \text{ and } \gamma \notin \{\bot, \top\} \\ \epsilon & \text{otherwise} \end{cases}$$

These maps are now extended to trails.

DEFINITION 3.2.9. Let ϑ be a trail of Π. Define

- the *run* σ_ϑ *of* $\mathcal{S}$ *induced by* ϑ: $\sigma_\vartheta \stackrel{\text{def}}{=} h_\Sigma^\omega(\vartheta)$, and
- the *path* π_ϑ *of* Π *induced by* ϑ: $\pi_\vartheta \stackrel{\text{def}}{=} h_\Gamma^\omega(\vartheta)$.

Furthermore, we say that a *run* σ *of* $\mathcal{S}$ *follows a path* π *of* Π, if there is a trail ϑ of Π inducing σ and π. $\diamondsuit$

In order to obtain a close match between system computations and fair trails, we need to strengthen the notion of fairness for trails.

DEFINITION 3.2.10. (Π-FAIRNESS) Suppose that $\Lambda_w^\Pi \in W^\Pi$, $\Lambda_f^\Pi \in F^\Pi$ and $\Lambda_w \in W$, $\Lambda_f \in F$ such that $\Lambda_w^\Pi = \pi_2^{-1}(\Lambda_w)$ and $\Lambda_f^\Pi = \pi_2^{-1}(\Lambda_f)$, where $\mathcal{F}^\Pi = (P^\Pi, W^\Pi, F^\Pi)$ and $\mathcal{F} = (P, W, F)$ are the fairness constraints of $\mathcal{S}^\Pi$ and $\mathcal{S}$, respectively. A trail ϑ of Π is called

- *weakly* Π-*fair w.r.t.* Λ_w^Π, if in case Λ_w is enabled on ϑ from some point on, then Λ_w^Π is taken infinitely often on ϑ,
- *strongly* Π-*fair w.r.t.* Λ_f^Π, if in case Λ_f is infinitely often enabled on ϑ, then Λ_f^Π is taken infinitely often on ϑ, and
- Π-*fair* if it is weakly Π-fair w.r.t. each $\Lambda_w^\Pi \in W^\Pi$ and strongly Π-fair w.r.t. each $\Lambda_f^\Pi \in F^\Pi$. $\diamondsuit$

We remark that a Π-fair trail is also fair w.r.t. $\mathcal{F}^\Pi$ in the usual sense. We note some basic properties of trails in

LEMMA 3.2.11. (TRAIL LEMMA) *We have:*

(i) all trails of Π *are infinite, and*

(ii) any run σ *of* $\mathcal{S}$ *follows some path* π *of* Π,

(iii) a trail ϑ *of* Π *is* Π-*fair iff* σ_ϑ *is a computation of* $\mathcal{S}$.

PROOF. (i) and (ii): From the definitions by the totality of the (global) transition relation ρ_Λ of $\mathcal{S}$ (Assumption 2.2.2) and the side condition of the Split rule.

(iii) Let $\Lambda_0^\Pi \in W^\Pi \cup F^\Pi$ and $\Lambda_0 \in W \cup F$ (where $\mathcal{F}^\Pi = (P^\Pi, W^\Pi, F^\Pi)$ and $\mathcal{F} = (P, W, F)$ are the fairness constraints of $\mathcal{S}^\Pi$ and $\mathcal{S}$, respectively) such that $\Lambda_0^\Pi = \pi_2^{-1}(\Lambda_0)$. It is not difficult to see that Λ_0^Π is taken infinitely often on ϑ if and only if Λ_0 is taken infinitely often on σ_ϑ. Furthermore, since Λ_0 is enabled on an extended state $t \in \Sigma^\Pi$ (of $\mathcal{S}^\Pi$) precisely if it is enabled on its projection $t|_X \in \Sigma$ (of $\mathcal{S}$) and σ_ϑ differs from $\vartheta|_X$ only by a repetition of states, we conclude that Λ_0 is enabled on ϑ from some point on (infinitely often) if and only if Λ_0 is enabled on σ_ϑ from some point on (infinitely often). The result then follows immediately. $\square$

The LTL Success Criterion

Now we are ready to formally define our success criterion, by lifting the success condition for paths to trails and then to proof structures.

DEFINITION 3.2.12. (SUCCESSFUL TRAIL) A trail ϑ of a LTL proof structure Π is *successful* if the induced path π_ϑ is successful. $\diamondsuit$

DEFINITION 3.2.13. (LTL SUCCESS CRITERION) A LTL proof structure Π is *successful* if all its Π-*fair* trails are successful. $\diamondsuit$

Note that by the Trail Lemma (ii) and (iii), this definition captures exactly the informal definition given in condition (A-SUC) above. On the other hand, in any unsuccessful proof structure Π for $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}\,\phi$ there is an unsuccessful Π-fair trail ϑ which projects to a Ξ-computation σ_ϑ of $\mathcal{S}$ and to an unsuccessful path π_ϑ. The computation σ_ϑ provides a counter-example, that is, it does not satisfy $\mathsf{A}\,\phi$, a fact that will be proved in the next chapter.

EXAMPLE 3.2.14. Consider proof structure Π_1 of Example 3.2.4 (depicted in Figure 3.2). The (unique) infinite path π is unsuccessful and is followed by exactly the (counter-example) computations σ_m of $\mathcal{S}_1$ of the form

$$\sigma_m\colon \langle x = m\rangle\,\langle x = m-1\rangle \cdots \langle x = 11\rangle\,(\langle x = 10\rangle)^\omega$$

for all $m \geq 10$. Therefore, Π_1 is unsuccessful when viewed as a proof structure for $\mathcal{S}_1$. On the other hand, there is no computation of $\mathcal{S}_1'$ following π. Thus, Π_1 is successful for $\mathcal{S}_1'$. ♣

For the purpose of designing proof rules for success, we need a more syntactic formulation of the success criterion.

DEFINITION 3.2.15. Let $\Psi_\mathsf{A} \stackrel{\text{def}}{=} \mathsf{V}(\phi) \cup \{\top\}$[4] and let Q_ψ be as in Definition 3.2.2. Define the assertions K_ψ for $\psi \in \Psi_\mathsf{A}$ by

$$K_\psi \stackrel{\text{def}}{=} \begin{cases} K \in \lceil Q_\psi \rceil & \text{for } \psi \in \mathsf{V}(\phi) \\ K = \lceil \top \rceil & \text{for } \psi = \top \end{cases}$$

$\diamondsuit$

Note as the sets Q_ψ are finite, the assertions K_ψ are definable in our assertion language. It is now easy to lift the success condition for paths to trails:

PROPOSITION 3.2.16. (LTL SUCCESS, SYNTACTICALLY)

(i) A trail of Π *is* successful *iff it satisfies the* success formula

$$\Omega_\mathsf{A} \stackrel{\text{def}}{=} \bigvee_{\psi \in \Psi_\mathsf{A}} \mathsf{F}\,\mathsf{G}\,K_\psi$$

[4] The index A in Ψ_A and in Ω_A below is to distinguish these sets from their cousins Ψ_E and Ω_E, which will be defined in Chapter 5.

(ii) A proof structure Π *is* successful *iff all its* Π*-fair trails satisfy* Ω_{A}*, that is,*

$$\mathcal{S}^{\Pi} \models \mathsf{A}_{\Pi}\, \Omega_{\mathsf{A}},$$

where A_{Π} *quantifies over all* Π*-fair trails.*

PROOF. (i) Observe that $\vartheta \models \mathsf{F}\,\mathsf{G}\,K_{\top}$ if and only if π_{ϑ} ends in an axiom. Thus, the success formula Ω_{A} holds for a trail ϑ precisely if the induced path π_{ϑ} is successful. (ii) Immediate. □

3.3 A Rule for Proving Success

The success criterion (Definition 3.2.13) lays down the condition for accepting a proof structure for a system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}\,\phi$ as a legal proof of $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$. What is missing for a full-blown proof system is a rule for proving success.

We restrict ourselves, for the time being, to saturated systems (trivial fairness constraint). Success rules including fairness are presented in Chapter 6. Let $\mathcal{S}$ be such a system and suppose Π is a proof structure for $\mathcal{S}$ and some sequent γ_r. By Proposition 3.2.16, success of Π can be established by proving that all trails satisfy Ω_{A}, that is, $\mathcal{S}^{\Pi} \models \mathsf{A}(\bigvee_{\psi\in\Psi_{\mathsf{A}}} \mathsf{F}\,\mathsf{G}\,K_{\psi})$.

3.3.1 Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$

Let us first somewhat generalise the setting and present a rule (see Figure 3.4) allowing us to prove the validity of a formula of the form $\mathsf{A}(\mathsf{F}\,q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i)$ over an arbitrary saturated system $\mathcal{S}$ (but still with a total transition relation). The reason for adding the formula $\mathsf{F}\,q$ will become clear in the next section.

The application of Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ requires that we find an intermediate assertion β_i for each p_i, a well-founded domain $(W, \succ)$ and a ranking function $\delta : \Sigma \to W$ mapping system states to elements of W. Condition P1 states that the initial condition implies q or β, the latter being the disjunction of all the β_i. The Hoare triple in premise P2 requires that from a β_i-state all transitions lead to a q-state, to a β-state with a lower rank or again to a β_i-state with a rank not higher than the source state. By the final premise P3 transitions from a β_i-state where p_i does not hold lead to a q-state or to a β-state with a lower rank.

Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ is derived from Rule F-RESP presented in [MP91] for proving response properties of the form $\mathsf{G}(p \to \mathsf{F}\,q)$ under weak and strong

Let $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta)$ be a *saturated* system and let $p_1, \ldots, p_m$ and q be assertions. In order to apply this rule, find:

(a) a ranking function $\delta \colon \Sigma \to W$ mapping states of $\mathcal{S}$ into elements of a well-founded domain $(W, \succ)$, and

(b) assertions $\{\beta_1, \ldots, \beta_m\}$ (setting $\beta \stackrel{\text{def}}{=} \bigvee_{i=1}^{m} \beta_i$),

and check the validity of conditions P1-P3.

$$\begin{array}{ll} \text{P1.} & \Theta \to q \vee \beta \\ \text{P2.} & \{\beta_i \wedge \delta = w\}\ \Lambda\ \{q \vee (\beta \wedge \delta \prec w) \vee (\beta_i \wedge \delta \preceq w)\} \\ \text{P3.} & \{\beta_i \wedge \delta = w \wedge \neg p_i\}\ \Lambda\ \{q \vee (\beta \wedge \delta \prec w)\} \\ \hline & \mathcal{S} \vdash \mathsf{A}(\mathsf{F}\, q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\, p_i) \end{array}$$

Figure 3.4: Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$

(transition) fairness. Note the equivalences

$$\begin{aligned} \mathsf{F}\, q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\, p_i \quad &\equiv_{\mathcal{S}} \quad \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\, \neg p_i \to \mathsf{F}\, q \\ &\equiv_{\mathcal{S}} \quad \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\, \neg p_i \to \mathsf{G}(\Theta \to \mathsf{F}\, q) \end{aligned} \tag{$e1$}$$

for a system $\mathcal{S}$ with initial condition Θ. The subformula $\bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\, \neg p_i$ on the right-hand side can be interpreted as a (generalised) unconditional fairness constraint. Unconditional fairness being closely related to weak fairness, our rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ is very similar to Rule F-RESP under weak fairness only.

Let us now explain why this rule is sound. As a consequence of P1 and P2, on any run of $\mathcal{S}$ the assertion β is invariant and the ranking never increases *unless* q becomes true. The idea behind this rule is then that by the decrease in rank each time $\beta_i \wedge \neg p_i$-state is met, a run σ is forced by well-foundedness to either reach q (hence $\sigma \models \mathsf{F}\, q$) or stabilise eventually in $\beta_j \wedge p_j$-states (hence $\sigma \models \mathsf{F}\,\mathsf{G}\, p_j$). In words corresponding to the second formula in ($e1$), each encounter of a $\beta_i \wedge \neg p_i$-state brings us closer to a q-state. With this in mind, it is not difficult to see that

Proposition 3.3.1. (Soundness of Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$) *Let $\mathcal{S}$ be a saturated system and let $p_1, \ldots, p_m$ (with $m \geq 1$) and q be assertions. Then $\mathcal{S} \vdash \mathsf{A}(\mathsf{F}\, q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\, p_i)$ implies $\mathcal{S} \models \mathsf{A}(\mathsf{F}\, q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\, p_i)$.* □

Note that the application of the rule requires that the set of assertions $\{p_1, \ldots, p_n\}$ is non-empty. However, in order to apply the rule to prove $\mathsf{A}\,\mathsf{F}\,q$ (with an empty set of assertions p_i) we can simply use the single dummy assertion $p_1 \stackrel{\text{def}}{=} \mathsf{false}$. Condition P3 then requires that the ranking decreases on every transition until q is reached.

3.3.2 *Rule* $\mathsf{A}(\boldsymbol{S})$

In order to obtain a proof rule for LTL success, all we have to do is to instantiate Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ with the associated system $\mathcal{S}^\Pi$ for $\mathcal{S}$, assertions $\{K_\psi \mid \psi \in \Psi_\mathsf{A}\}$ for $\{p_1, \ldots, p_n\}$ and to set $q \stackrel{\text{def}}{=} \mathsf{false}$, thus yielding a rule for proving $\mathcal{S}^\Pi \models \mathsf{A}(\bigvee_{\psi\in\Psi_\mathsf{A}} \mathsf{F}\,\mathsf{G}\,K_\psi)$. This is correct, since all trails are infinite by Lemma 3.2.11. However, as we know that control K always stabilises at $\top$ once it got there, there is no need to prove this every time. So a more practical instantiation is based on the equivalence:

$$(\Omega_\mathsf{A} =) \quad \bigvee_{\psi\in\Psi_\mathsf{A}} \mathsf{F}\,\mathsf{G}\,K_\psi \equiv_{\mathcal{S}^\Pi} \mathsf{F}\,K_\top \vee \bigvee_{\psi\in\mathsf{V}(\phi)} \mathsf{F}\,\mathsf{G}\,K_\psi \tag{$e2$}$$

The right-hand side formula more closely reflects the semantic definition of success[5] (Definition 3.2.13).

In order to account for the case where there are no V-subformulas (that is, $\mathsf{V}(\phi) = \varnothing$), we define

$$\widehat{\Omega}_\mathsf{A} \stackrel{\text{def}}{=} \mathsf{F}\,K_\top \vee \bigvee_{\psi\in\widehat{\Psi}_\mathsf{A}} \mathsf{F}\,\mathsf{G}\,K_\psi$$

where $\widehat{\Psi}_\mathsf{A} \stackrel{\text{def}}{=} \mathsf{V}(\phi) \cup \{\bullet\}$. Assertion $K_\bullet$ could be set to false as suggested at the end of the previous section. Again for practical reasons, we can do better than that:

$$\begin{array}{lcl} K_\bullet & \stackrel{\text{def}}{=} & \neg K_{sys} \\ K_{sys} & \stackrel{\text{def}}{=} & K \in \lceil \Gamma_{sys} \rceil \end{array}$$

where K_{sys} is the syntactical counterpart of Γ_{sys}. Observe that since in each trail the control variable K either traverses infinitely many $\mathsf{A}(\mathsf{X})$-sequents (by Lemma 3.1.5), or eventually stabilises at $\bot$ or $\top$, we have $\mathsf{F}\,\mathsf{G}\,K_\bullet \equiv_{\mathcal{S}^\Pi} \mathsf{F}\,\mathsf{G}\,\mathsf{false} \equiv_{\mathcal{S}^\Pi} \mathsf{false}$ and therefore $\widehat{\Omega}_\mathsf{A} \equiv_{\mathcal{S}^\Pi} \Omega_\mathsf{A}$. For later reference we state

[5] At this point the reader may wonder why we did not use the right-hand side formula in the syntactic characterisation of LTL success (Proposition 3.2.16) right from the beginning. This is because the formula $\Omega_\mathsf{A} = \bigvee_{\psi\in\Psi_\mathsf{A}} \mathsf{F}\,\mathsf{G}\,K_\psi$ more cleanly exhibits the duality between LTL success and ELL success (defined in Section 5.2).

Let Π be a LTL proof structure for a *saturated* system $\mathcal{S}$. Let $\left\{K_\psi \mid \psi \in \widehat{\Psi}_{\mathsf{A}}\right\}$ be as discussed in the text. Find:

(a) a ranking function $\delta : \Sigma^\Pi \to W$ mapping states of $\mathcal{S}^\Pi$ into elements of a well-founded domain $(W, \succ)$, and

(b) assertions $\left\{\beta_\psi \mid \psi \in \widehat{\Psi}_{\mathsf{A}}\right\}$ (setting $\beta \stackrel{\text{def}}{=} \bigvee_{\psi \in \widehat{\Psi}_{\mathsf{A}}} \beta_\psi$),

such that conditions A1-A3 are valid.

$$\begin{array}{ll}
\text{A1.} & \Theta^\Pi \to K_\top \vee \beta \\
\text{A2.} & \{\beta_\psi \wedge \delta = w\}\ \Lambda^\Pi\ \{K_\top \vee (\beta \wedge \delta \prec w) \vee (\beta_\psi \wedge \delta \preceq w)\} \\
\text{A3.} & \{\beta_\psi \wedge \delta = w \wedge \neg K_\psi\}\ \Lambda^\Pi\ \{K_\top \vee (\beta \wedge \delta \prec w)\} \\
\hline
 & \mathcal{S}^\Pi \vdash \mathsf{A}\,\widehat{\Omega}_{\mathsf{A}}
\end{array}$$

Figure 3.5: Rule A(S) for proving success

Proposition 3.3.2. *A proof structure* Π *is successful iff* $\mathcal{S}^\Pi \models \mathsf{A}_\Pi\,\widehat{\Omega}_{\mathsf{A}}$. □

Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ can thus be instantiated with $\mathcal{S}^\Pi$ for $\mathcal{S}$, $K_\top$ for q and $\{K_\psi \mid \psi \in \widehat{\Psi}_{\mathsf{A}}\}$ for $\{p_1, \ldots, p_n\}$ yielding Rule A(S) of Figure 3.5.

To see the practical advantage of defining $K_\bullet$ as above rather than setting it to false, consider Condition A3 of Rule A(S) for $\psi = \bullet$. It requires that the ranking decreases along every transition from a $\beta_\bullet$-state where $K_\bullet$ does not hold. With $K_\bullet$ set to false this would mean *every* transition from a $\beta_\bullet$-state, while with $K_\bullet$ as defined above, a decrease in rank is only required at K_{sys}-states (where control K is at a Γ_{sys}-sequent). The chosen definition is thus more permissive. Moreover, the transitions from K_{sys}-states correspond to underlying system transitions, which we feel to be more intuitive and to make the rule easier to apply. Note that if $\mathsf{V}(\phi)$ is non-empty, $K_\bullet$ is not needed and may be "switched off" by setting $\beta_\bullet \stackrel{\text{def}}{=} \mathsf{false}$.

Proofs and Soundness of Rule $\mathsf{A}(\boldsymbol{S})$

Definition 3.3.3. Let Π be proof structure for $\mathcal{S}$ and $\Xi \vdash \mathsf{A}\,\phi$. We say that Rule A(S) is *applicable* to Π if $\mathcal{S}^\Pi \vdash \mathsf{A}\,\widehat{\Omega}_{\mathsf{A}}$, that is, we can find a well-founded domain $(W, \succ)$, a ranking function $\delta\colon \Sigma^\Pi \to W$ and assertions $\{\beta_\psi \mid \psi \in \widehat{\Psi}_{\mathsf{A}}\}$ such that conditions A1-A3 are valid. ♢

DEFINITION 3.3.4. (LTL PROOFS) Let Π be proof structure for $\mathcal{S}$ and $\Xi \vdash \mathsf{A}\,\phi$.

- Π is a *proof of* $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$, written $\Pi\colon \mathcal{S}, \Xi \Vdash \mathsf{A}\,\phi$, if it is successful, and
- Π is a *S-proof of* $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$, written $\Pi\colon \mathcal{S}, \Xi \vdash \mathsf{A}\,\phi$, if Rule A(S) is applicable to Π.

We say that $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$ is *provable* (*S-provable*), written $\mathcal{S}, \Xi \Vdash \mathsf{A}\,\phi$ ($\mathcal{S}, \Xi \vdash \mathsf{A}\,\phi$), if there is a proof structure Π for $\mathcal{S}$ and $\Xi \vdash \mathsf{A}\,\phi$ such that $\Pi\colon \mathcal{S}, \Xi \Vdash \mathsf{A}\,\phi$ ($\Pi\colon \mathcal{S}, \Xi \vdash \mathsf{A}\,\phi$). $\diamondsuit$

NOTATION. If Π proves $\mathcal{S}, \Theta \models \mathsf{A}\,\phi$ with Θ being the initial condition of $\mathcal{S}$, we write $\Pi\colon \mathcal{S} \vdash \mathsf{A}\,\phi$ and $\mathcal{S} \vdash \mathsf{A}\,\phi$ instead of $\Pi\colon \mathcal{S}, \Theta \vdash \mathsf{A}\,\phi$ and $\mathcal{S}, \Theta \vdash \mathsf{A}\,\phi$, respectively, and similarly for S-proofs and S-provability.

From Propositions 3.3.1 and 3.3.2 we immediately get

PROPOSITION 3.3.5. (SOUNDNESS OF RULE A(S) FOR PROVING SUCCESS) *Let $\mathcal{S}$ be a saturated system. If $\Pi\colon \mathcal{S}, \Xi \vdash \mathsf{A}\,\phi$ then $\Pi\colon \mathcal{S}, \Xi \Vdash \mathsf{A}\,\phi$.* $\square$

Note that the fact that Π is a proof of $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$ does of course not *a priori* imply that $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$ holds. That this is indeed the case follows from the soundness of our proof system, which will be proved along with relative completeness in the next chapter.

Safety Formulas

The case where there is no V-subformula in ϕ has already been discussed extensively. To conclude this section, we also consider the other extreme where there are no U-subformulas in ϕ. In this case ϕ describes a safety property, since there are no (sub-)formulas that are promised to become true in the future (as is the case with ϕ_2 in $\phi_1 \,\mathsf{U}\, \phi_2$). We would then expect that no well-foundedness argument is necessary to establish success of a proof structure for a system $\mathcal{S}$ and a sequent $\Xi \vdash \mathsf{A}(\phi)$, since all infinite paths are successful. Indeed, the following proposition, the proof of which is deferred to the next chapter (see Proposition 4.2.21), exempts us from applying Rule A(S) altogether, at least in case all terminals are axioms. Note that the proposition holds for any system $\mathcal{S}$, with or without fairness constraints, since only liveness properties but not safety properties depend on fairness.

PROPOSITION 3.3.6. *Let Π be a proof structure for system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}\,\phi$, where ϕ does not contain any U-subformulas. Suppose that all terminals in Π are axioms. Then $\Pi\colon \mathcal{S} \Vdash \mathsf{A}\,\phi$.*

As will also be shown in the next chapter, provided that the property to be shown is true, there is always a proof structure all of whose terminals are axioms, so this restriction in the proposition is only a mild one.

3.4 Some Examples

In this section, we illustrate the application of our proof method with three examples. In particular, we will prove a guarantee, a safety and a persistence property.

3.4.1 A Guarantee Property

In order to complete Example 3.2.4, it remains to show that proof structure Π_1 is successful for system $\mathcal{S}_1'$ (and property ϕ_1). Recall that system $\mathcal{S}_1'$ simply decrements a natural number variable x or loops at $x = 0$. Proof structure Π_1 is reproduced in Figure 3.6 in a decorated form for reference.

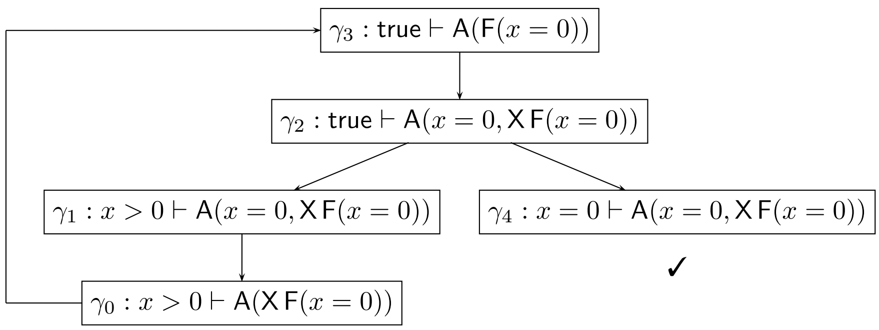

Figure 3.6: Proof structure Π_1 for system $\mathcal{S}_1'$ and property $\phi_1 \stackrel{\text{def}}{=} \mathsf{A}\,\mathsf{F}(x = 0)$.

In the following we suppose that the coding $\lceil\cdot\rceil$ is defined by $\lceil\gamma_i\rceil = i$ for $0 \leq i \leq 4$, $\lceil\top\rceil = 5$. In order to apply Rule A(S) we choose the auxiliary assertion $\beta_\bullet$ and the ranking δ as follows:

$$\beta_\bullet \stackrel{\text{def}}{=} \mathsf{true} \qquad\qquad \delta(x, K) \stackrel{\text{def}}{=} x$$

Note that $\beta \equiv \beta_\bullet$, since $\beta_\bullet$ is the only auxiliary assertion, and that $\neg K_\bullet \equiv K \in \{0, 4, 5\}$. Condition A1 is trivially satisfied. For condition A2, we can restrict ourselves to the cases where $K_\bullet$ holds, the others being covered by A3. Condition A2 then boils down to showing

$$\{x = n \wedge (1 \leq K \leq 3)\}\, \Lambda^\Pi \,\{K_\top \vee x \leq n\}$$

For $1 \leq K \leq 3$, the relevant transitions are $(\gamma_1, =, \gamma_0)$, $(\gamma_2, =, \gamma_1)$, $(\gamma_2, =, \gamma_4)$ and $(\gamma_3, =, \gamma_2)$ and all of them maintain the ranking constant. Finally, for condition A3 we have to check

$$\{x = n \wedge K \in \{0, 4, 5\}\} \, \Lambda^{\Pi} \, \{K_{\top} \vee x < n\}$$

It is now easy to see that the only enabled transition for $K = 0$ is $(\gamma_0, dec, \gamma_3)$ and decreases the rank, while the relevant transitions from $K \in \{4, 5\}$, namely $(\gamma_4, zero, \top)$ and $(\top, zero, \top)$, lead to $K_{\top}$.

Hence Π_1 is a proof of $\mathcal{S}_1' \models \mathsf{A}\,\mathsf{F}(x = 0)$ by soundness of Rule A(S) (Proposition 3.3.5).

3.4.2 A Safety Property

Consider the system $\mathcal{S}_3$ with a boolean-valued variable b, a natural number variable x, initial condition $\Theta_3 \stackrel{\text{def}}{=} (b = \mathsf{tt}) \wedge (x = 0)$ and transition relations:

$$\begin{array}{lcl} \rho_{inc} & \stackrel{\text{def}}{=} & (x' = x - 1) \wedge (b' = b) \\ \rho_{up} & \stackrel{\text{def}}{=} & (b = \mathsf{ff}) \wedge (b' = \mathsf{tt}) \wedge (x' = x) \end{array}$$

The labeled transition system for this specification is depicted in Figure 3.7.

$$\begin{tikzcd}
(0, \mathsf{tt}) \arrow[r, "inc"] & (1, \mathsf{tt}) \arrow[r, "inc"] & (2, \mathsf{tt}) \arrow[r, "inc"] & (3, \mathsf{tt}) \arrow[r, dash] & \cdots \\
(0, \mathsf{ff}) \arrow[r, "inc"] \arrow[u, "up"] & (1, \mathsf{ff}) \arrow[r, "inc"] \arrow[u, "up"] & (2, \mathsf{ff}) \arrow[r, "inc"] \arrow[u, "up"] & (3, \mathsf{ff}) \arrow[r, dash] \arrow[u, "up"] & \cdots
\end{tikzcd}$$

Figure 3.7: LTS for system $\mathcal{S}_3$

The property we want to verify for this system is $\phi_3 \stackrel{\text{def}}{=} \mathsf{A}((b = \mathsf{ff})\,\mathsf{W}(b = \mathsf{tt}))$. Figure 3.8 shows proof structure Π_3 for system $\mathcal{S}_3$ and property ϕ_3. Recall that $\psi_0 \,\mathsf{W}\, \psi_1$ is defined as $\psi_2 \vee (\mathsf{X}\,\psi_2)\,\mathsf{V}\,\psi_1$, so there are no U-subformulas in ϕ_3. By Proposition 3.3.6, this proof structure is successful, since all its terminal sequents are axioms. Therefore, proof structure Π_3 is a proof of $\mathcal{S}_3 \models \phi_3$.

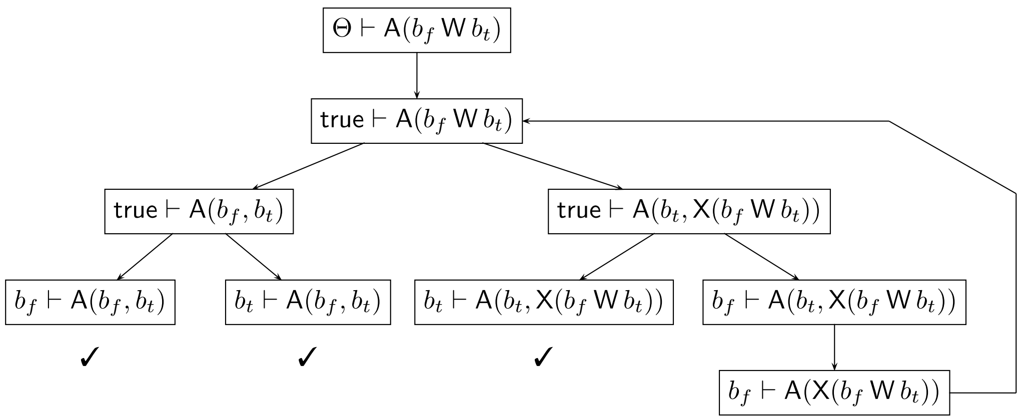

Figure 3.8: Proof structure Π_3 for system $\mathcal{S}_3$ and property $\phi_3 \stackrel{\text{def}}{=} \mathsf{A}((b = \mathsf{ff})\,\mathsf{W}(b = \mathsf{tt}))$, writing b_t for $b = \mathsf{tt}$ and b_f for $b = \mathsf{ff}$.

3.4.3 A Persistence Property

System $\mathcal{S}_4$ has a single natural number variable x with initial condition $\Theta_4 \stackrel{\text{def}}{=} \mathsf{true}$ and the transition relations ρ_{dec} and ρ_{ev} defined by

$$\begin{aligned} \rho_{dec} &\stackrel{\text{def}}{=} x > 0 \wedge x' = x - 1 \\ \rho_{ev} &\stackrel{\text{def}}{=} ev(x) \wedge x' = x \end{aligned}$$

As a property to be proved for this system consider the persistence formula $\phi_4 \stackrel{\text{def}}{=} \mathsf{A}\,\mathsf{F}\,\mathsf{G}\, ev(x)$ expressing that x eventually stabilises on an even value. The labeled transitions system for $\mathcal{S}_4$ appears in Figure 3.9 and a proof structure for $\mathcal{S}_4$ and ϕ_4 is displayed in Figure 3.10.

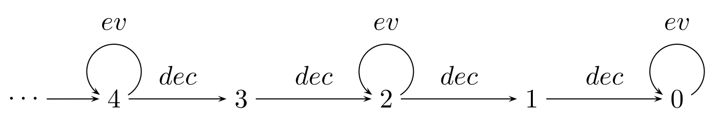

Figure 3.9: LTS for system $\mathcal{S}_4$

Successful paths in this proof structure are exactly those that are either finite, ending in the axiom γ_5, or infinite, ending in $(\gamma_1\gamma_6\gamma_7)^\omega$.

In order to prove that Π_4 is successful using Rule $\mathsf{A}(\mathrm{S})$, we quite naturally

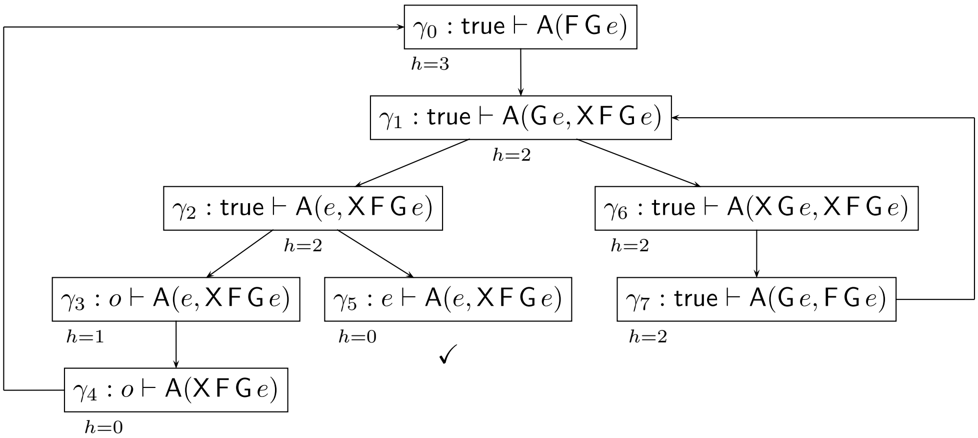

Figure 3.10: Proof structure Π_4 for $\mathcal{S}_4$ and $\phi_4 \stackrel{\text{def}}{=} \mathsf{A\,F\,G}\,ev(x)$ with $ev(x)$ abbreviated to e and $od(x)$ to o

choose the auxiliary assertions $\beta_\bullet$ and $\beta_{\mathsf{G}\,e}$ as follows:

$$\begin{array}{lcl} \beta_\bullet & \stackrel{\text{def}}{=} & od(x) \\ \beta_{\mathsf{G}\,e} & \stackrel{\text{def}}{=} & ev(x) \end{array}$$

This yields $\beta \equiv \mathsf{true}$, so condition A1 of Rule A(S) is trivially satisfied. We propose the following ranking function:

$$\delta(x, K) \stackrel{\text{def}}{=} (x, h(K))$$

with the lexicographic ordering, where the value of $h(K)$ is as indicated below each node in Figure 3.10 and is defined for $K = \top$ by $h(\top) = 0$. We suppose that the coding $\lceil \cdot \rceil$ is defined by $\lceil \gamma_i \rceil = i$ for $0 \leq i \leq 7$ and $\lceil \top \rceil = 8$. This yields

$$\begin{array}{lcl} K_\bullet & \equiv & K \in \{0, 1, 2, 3, 7\} \\ K_{\mathsf{G}\,e} & \equiv & K \in \{1, 6, 7\} \end{array}$$

It remains to prove that A2 and A3 are satisfied. For condition A2 we again assume that K_ψ holds, the other cases being covered by A3. The case $\psi = \bullet$ then boils down to

$$\begin{array}{l} \{od(x) \wedge (x, h) = (m, n) \wedge K \in \{0, 1, 2, 3, 7\}\} \\ \Lambda^\Pi \\ \{K_\top \vee (x, h) < (m, n) \vee (od(x) \wedge (x, h) \leq (m, n))\} \end{array}$$

All relevant transitions are of the form $(\gamma, =, \gamma')$, thus preserving the value of x. Also, the value of h does not increase along these transitions, so neither does the overall ranking.

For $\psi = \mathsf{G}\, e$, the verification condition reads

$$\begin{array}{l}\{ev(x) \wedge (x,h) = (m,n) \wedge K \in \{1,6,7\}\} \\ \Lambda^{\Pi} \\ \{K_{\top} \vee (x,h) < (m,n) \vee (ev(x) \wedge (x,h) \leq (m,n))\}\end{array}$$

For $K \in \{1,7\}$, the relevant transitions are $(\gamma_1, =, \gamma_2)$, $(\gamma_1, =, \gamma_6)$ and $(\gamma_7, =, \gamma_1)$ which clearly preserve the value of x and h. For $K = 6$, transition (γ_6, ev, γ_7) on one hand preserves the ranking (hence $ev(x)$) and transition $(\gamma_6, dec, \gamma_7)$ on the other hand decreases the ranking.

Condition A3 for $\psi = \bullet$ is equivalent to

$$\begin{array}{l}\{od(x) \wedge (x,h) = (m,n) \wedge K \in \{4,5,6,8\}\} \\ \Lambda^{\Pi} \\ \{K_{\top} \vee (x,h) < (m,n)\}\end{array}$$

For $K = 4$ or $K = 6$, the relevant transitions $(\gamma_4, dec, \gamma_0)$ and $(\gamma_6, dec, \gamma_7)$ decrease x, hence the ranking. For $K = 5$ and $K = 8\,(\equiv K_{\top})$, we clearly have $\{K = 5\}\,\Lambda^{\Pi}\,\{K_{\top}\}$ and $\{K_{\top}\}\,\Lambda^{\Pi}\,\{K_{\top}\}$, respectively.

The remaining case is A3 for $\psi = \mathsf{G}\, e$, which boils down to

$$\begin{array}{l}\{ev(x) \wedge (x,h) = (m,n) \wedge K \in \{0,2,3,4,5,8\}\} \\ \Lambda^{\Pi} \\ \{K_{\top} \vee (x,h) < (m,n)\}\end{array}$$

We distinguish three cases. For $K \in \{0,2\}$ the all non-trivial transitions are of the form $(\gamma, =, \gamma')$, thus preserving x while decreasing the value of h. Hence, the rank decreases along these transitions. For $K \in \{3,4\}$ the condition holds trivially, since any transition departing from these positions requires that $od(x)$ holds. Finally, any transition departing from $K \in \{5,8\}$ reaches $K_{\top}$. We conclude that Π_4 is a proof of $\mathcal{S}_4 \models \mathsf{A}\,\mathsf{F}\,\mathsf{G}\, ev(x)$.

Chapter 4

Soundness and Completeness via Games

This chapter is devoted to proving the following

THEOREM 4.0.1. (SOUNDNESS AND RELATIVE COMPLETENESS) *Let $\mathcal{S}$ be a saturated system, Ξ an assertion and ϕ a LTL formula. We have*

$$\mathcal{S}, \Xi \models \phi \text{ if and only if } \mathcal{S}, \Xi \vdash \phi.$$

Due to the expressiveness of the assertion language $\mathcal{L}$ we cannot expect that all verification conditions are provable in some formal system. Therefore, completeness is shown *relative* to the validity of the verification conditions, thereby decoupling the reasoning in our proof system from the reasoning in the assertion language $\mathcal{L}$.

$$\begin{array}{c} \mathcal{S}, \Xi \models \phi \\ \Updownarrow 1. \\ \text{Player } \exists \text{ wins } \mathcal{G}_{\mathcal{S}}(\Xi, \phi) \\ \Updownarrow 2. \\ \mathcal{S}, \Xi \Vdash \phi \\ \Updownarrow 3. \\ \mathcal{S}, \Xi \vdash \phi \end{array}$$

Figure 4.1: Road-map to soundness and relative completeness

A major part of this theorem will be proved by a game-theoretic argument. The proof proceeds in three stages as depicted in Figure 4.1. As an outline of the present chapter, we will briefly discuss each of the three equivalences.

1. CTL* Games (Section 4.1) As a preparatory step we will define, for a given model $\mathcal{M} = (\mathcal{T}, V)$, run $\sigma \in \mathcal{R}_{\mathcal{T}}$ and CTL* formula ψ, the CTL* game $\mathcal{G}_{\mathcal{M}}(\sigma, \psi)$, an infinite two-player game, where – intuitively speaking – one player (called $\exists$) tries to show that the property holds ($\mathcal{M}, \sigma \models \psi$) and the other player (called $\forall$) tries to refute it ($\mathcal{M}, \sigma \not\models \psi$). We then show that the truth of $\mathcal{M}, \sigma \models \psi$ can be characterised by the existence of a winning strategy for Player $\exists$ in the game $\mathcal{G}_{\mathcal{M}}(\sigma, \psi)$. Furthermore, this game is *determined*, that is, one of the players has a winning strategy. This characterisation is not directly related to proof structures and has an interest of its own.

2. Trails and Strategies (Sections 4.2 and 4.3) In this second step we investigate the internal structure of paths in proof structures and show that to any trail ϑ of a LTL proof structure Π for $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}\,\phi$ corresponds a $\forall$-strategy τ_ϑ for the LTL game $\mathcal{G}_{\mathcal{S}}(\sigma_\vartheta, \phi)$. Clearly, if there is a trail ϑ such that the $\forall$-strategy τ_ϑ is winning then σ_ϑ provides a counter-example to the truth of $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$.

We introduce an alternative notion of success proposed in the literature called *admissibility* [Dam94] and matching more closely the winning conditions of our games. We show that a trail ϑ inducing a computation σ_ϑ is admissible precisely if τ_ϑ is a loosing strategy for $\mathcal{G}_{\mathcal{S}}(\sigma_\vartheta, \phi)$. This can be interpreted as a failed attempt to produce a counterexample. On the other hand, we demonstrate that if Player $\forall$ has a winning strategy τ for some game $\mathcal{G}_{\mathcal{S}}(\sigma, \phi)$ with σ a Ξ-run of $\mathcal{S}$, then τ is represented in Π in the sense that there exists a trail ϑ such that $\tau_\vartheta = \tau$ and $\sigma_\vartheta = \sigma$. As a consequence, if all trails ϑ projecting to σ are admissible, we can conclude that Player $\exists$ has a winning strategy for the game $\mathcal{G}_{\mathcal{S}}(\sigma, \phi)$ and hence $\mathcal{S}, \sigma \models \phi$. Summing up, we can say that Player $\exists$ wins $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{A}\,\phi)$ precisely if Π is admissible.

We then compare admissibility with success. Although the two notions do not precisely match on the level of individual paths or trails, they do coincide on the level of proof structures, that is, Π is admissible exactly if it is successful. As a final ingredient for the establishment of the second equivalence in the figure above, we show in Section 4.3 that, provided Player $\exists$ wins the game $\mathcal{G}_{\mathcal{S}}(\Xi, \phi)$, a proof structure does indeed exist for system $\mathcal{S}$ and sequent $\Xi \vdash \phi$.

3. Soundness and Completeness of Rule $\mathsf{A}(\boldsymbol{S})$ ***(Section 4.4)*** The third equivalence follows from the fact that Rule A(S) is sound and relatively complete for proving success of LTL proof structures.

We do not claim that our proof is the shortest possible one. However, our game-theoretic analysis of proof structures provides interesting insights into their fine structure and has as such an interest of its own. We therefore think that the little "detour" via games is well worth its price.

4.1 CTL* Games

In this section we will give a characterisation of the CTL* satisfaction relation in terms of winning strategies in an infinite two-player game. This characterisation is similar in spirit to the one for the modal μ-calculus [Sti95, Sti96a, Sti97]. An alternative notion of CTL* games is proposed in the very recent, as yet unpublished work by Lange and Stirling [LS00].

4.1.1 Game Definition

Given a CTL* model $\mathcal{M} = (\mathcal{T}, V)$ and a run $\sigma \in \mathcal{R}_{\mathcal{T}}$, the CTL* game $\mathcal{G}_{\mathcal{M}}(\sigma, \phi)$ is defined as follows. There are two players, called $\exists$ and $\forall$. Intuitively, Player $\exists$ is trying to establish $\mathcal{M}, \sigma \models \phi$ while his opponent, Player $\forall$, is trying to refute $\mathcal{M}, \sigma \models \phi$ (that is, establish $\mathcal{M}, \sigma \not\models \phi$). Game configurations are pairs consisting of a run of $\mathcal{T}$ and an CTL* formula. The initial configuration is $(\sigma_0, \phi_0) = (\sigma, \phi)$. The rules of the game are described in Table 4.1.

ψ	action	new configuration
p	end of play	-
$\psi_1 \vee \psi_2$	Player $\exists$ chooses one of the ψ_i	(ς, ψ_i)
$\psi_1 \wedge \psi_2$	Player $\forall$ chooses one of the ψ_i	(ς, ψ_i)
$\psi_1 \, \mathsf{Z} \, \psi_2$	Player $\exists$ unfolds $\psi_1 \, \mathsf{Z} \, \psi_2$	$(\varsigma, \mathsf{unf}(\psi_1 \, \mathsf{Z} \, \psi_2))$
$\mathsf{X} \, \psi$	Player $\forall$ advances ς by one state	(ς^1, ψ)
$\mathsf{E} \, \psi$	Player $\exists$ chooses some $\widehat{\varsigma} \in \mathcal{C}_{\mathcal{M}}(\varsigma(0))$	$(\widehat{\varsigma}, \psi)$
$\mathsf{A} \, \psi$	Player $\forall$ chooses some $\widehat{\varsigma} \in \mathcal{C}_{\mathcal{M}}(\varsigma(0))$	$(\widehat{\varsigma}, \psi)$

Table 4.1: Game moves in configuration (ς, ψ)

The possible moves in a configuration (ς, ψ) depend on the top-level connective of the formula ψ. A game play ends if ψ is an atomic proposition. For boolean formulas it is Player $\exists$ that chooses one of the disjuncts if ψ is a disjunction and Player $\forall$ chooses one of the conjunct in case ψ is a conjunction. For the temporal connectives, Player $\exists$ unfolds Z-formulas and Player $\forall$ eliminates Next connectives and advances the run by one state. Note that for these connectives, it is quite irrelevant which player actually moves, since there is no choice to be made. In case of a top-level path quantifier in ψ there is a choice — namely, of a computation $\widehat{\varsigma}$ starting in the same state as ς — and it is made (not surprisingly) by Player $\exists$ in case of an existential path quantifier and by his opponent $\forall$ in case of a universal path quantifier.

Table 4.1 defines a relation $\triangleright$ on configurations. The game tree $T_{\mathcal{G}_{\mathcal{M}}(\sigma,\phi)}$ induced by this relation contains the root ϵ together with all positions $p = c_0 c_1 \cdots c_m$ with $m \geq 0$ such that

- $c_0 = (\sigma, \phi)$, and
- $c_i \triangleright c_{i+1}$ for all $0 \leq i < m$.

A play either ends at an atomic proposition or proceeds ad infinitum by repeated unfolding of some Z-formula. The winning conditions are summarised in Table 4.2.

play type	Player $\exists$ wins	Player $\forall$ wins
μ finite, ending in (ς, p)	$\varsigma(0) \in V(p)$	$\varsigma(0) \notin V(p)$
μ infinite	$\exists^\omega i.\ \pi_2(\mu(i)) \in \mathsf{V}(\phi)$	$\exists^\omega i.\ \pi_2(\mu(i)) \in \mathsf{U}(\phi)$

Table 4.2: Winning conditions for game $\mathcal{G}_{\mathcal{M}}(\sigma, \phi)$

We say that a configuration (ς, ψ) is *true* if $\varsigma \models \psi$, and *false* otherwise. For finite plays the winner is determined according to the truth or falsity of the final configuration. For infinite plays μ, Player $\exists$ wins if there is some V-subformula of ϕ appearing in infinitely many configurations on μ and Player $\forall$ wins if some U-subformula of ϕ appears infinitely often along μ.

The game $\mathcal{G}_{\mathcal{M}}(U, \phi)$ for a non-empty set $U \subseteq S_{\mathcal{T}}$ is initiated by Player $\forall$ choosing a U-computation σ of $\mathcal{T}$, yielding the initial configuration (σ, ϕ). Then the game proceeds as $\mathcal{G}_{\mathcal{M}}(\sigma, \phi)$, with the same winning conditions. Thus,

$$T_{\mathcal{G}_{\mathcal{M}}(U,\phi)} = \bigcup_{\sigma \in \mathcal{C}_{\mathcal{T}}(U)} T_{\mathcal{G}_{\mathcal{M}}(\sigma,\phi)}.$$

Given a system $\mathcal{S}$ with system variables X and a ground-quantified CTL* formula ϕ over X, the games $\mathcal{G}_{\mathcal{S}}(\sigma, \phi)$ for $\sigma \in \mathcal{R}_{\mathcal{S}}$ and $\mathcal{G}_{\mathcal{S}}(\Xi, \phi)$ for a satisfiable assertion Ξ are defined in the obvious way. As usual, when there is no confusion possible we will drop indices $\mathcal{M}$ or $\mathcal{S}$. Finally, we speak of an LTL, ELL, CTL game when the formula ϕ is in the respective sublogic of CTL*.

4.1.2 Characterisation of CTL* satisfaction

For this section, consider a fixed, but arbitrary CTL* model $\mathcal{M} = (\mathcal{T}, V)$, run $\sigma \in \mathcal{R}_\mathcal{T}$ and CTL* formula ϕ. The following lemma states that the winning conditions stated for each player in Table 4.2 are indeed complementary, that is, any play is won by some player (there are no draws).

Lemma 4.1.1. (No Draws) *Any play of $\mathcal{G}(\sigma, \phi)$ won by some player. In particular, any infinite play μ ends in the following pattern for some run $\varsigma \in \mathcal{R}_\mathcal{T}$ and Z-formula $\psi = \phi_1 \,\mathsf{Z}\, \phi_2$:*

$$\mu\colon \; \cdots (\varsigma, \psi)(\varsigma, \mathsf{unf}(\psi))(\varsigma, \phi_1 \flat \mathsf{X}\,\psi)(\varsigma, \mathsf{X}\,\psi)(\varsigma^1, \psi)(\varsigma^1, \mathsf{unf}(\psi)) \cdots$$

where $\flat \in \{\wedge, \vee\}$ according to Z.

Proof. The statement is trivial for finite plays. For infinite plays, let us call Z-move a move from a configuration with a Z-formula. It is clear that any infinite play μ must exhibit an infinite number of Z-moves, since any other type of move decreases the size of the formula. As there are only a finite number of possible formulas occurring in configurations – namely, subformulas of ϕ and subformulas of unfoldings of Z-subformulas of ϕ – there must be some Z-formula, say $\psi = \phi_1 \,\mathsf{Z}\, \phi_2$, which is unfolded infinitely often in μ. But the only way to do so is to follow the sequence in the statement of the Lemma from the first point on where ψ occurs in a configuration on μ. Any other sequence would prevent the regeneration of ψ in a later configuration. $\square$

Note that game moves are designed to preserve the respective goal of each player. More precisely, if it is Player $\exists$'s ($\forall$'s) turn to move and the current configuration is true (false), then he has the choice of making a move to a true (false) next configuration. This observation provides the basis for

Proposition 4.1.2.

1. *if $\sigma \models \phi$ then Player $\exists$ has a (deterministic) history-free winning strategy for $\mathcal{G}(\sigma, \phi)$, and*
2. *if $\sigma \not\models \phi$ then Player $\forall$ has a (deterministic) history-free winning strategy for $\mathcal{G}(\sigma, \phi)$.*

Proof. We prove the first case, the second one follows by a symmetrical argument. Suppose $\sigma \models \phi$. Player $\exists$ determines his moves according to a fixed *choice function ε,* a partial function that is defined at least on *true* configurations of the forms $(\varsigma, \phi_1 \vee \phi_2)$, $(\varsigma, \phi_1 \,\mathsf{Z}\, \phi_2)$ and $(\varsigma, \mathsf{E}\,\psi)$. On the former two types of configurations ε is defined by

$$\varepsilon(\varsigma, \psi_1 \vee \psi_2) \stackrel{\text{def}}{=} \begin{cases} (\varsigma, \psi_1) & \text{if } \varsigma \models \psi_1 \text{ and } (\varsigma \not\models \psi_2 \text{ or } |\psi_1| \leq |\psi_2|) \\ (\varsigma, \psi_2) & \text{otherwise} \end{cases}$$

$$\varepsilon(\varsigma, \psi_1 \,\mathsf{Z}\, \psi_2) \stackrel{\text{def}}{=} (\varsigma, \mathsf{unf}(\psi_1 \,\mathsf{Z}\, \psi_2))$$

Furthermore, on true configurations of the form $(\varsigma, \mathsf{E}\,\psi)$ we require that

$$\varepsilon(\varsigma, \mathsf{E}\,\psi) = (\eta, \psi) \qquad \text{such that } \eta \in \mathcal{C}_{\mathcal{T}}(\varsigma(0)) \text{ and } \eta \models \psi \qquad (*)$$

Clearly, such a function ε exists by the semantics of the existential path quantifier.

We show by induction on the length of a play that ε induces a strategy for Player $\exists$ that allows him to preserve the truth of configurations regardless of the moves of his opponent, that is, we have $\varsigma \models \psi$ for any configuration (ς, ψ) occurring along a play. The initial configuration of the game is $(\sigma_0, \phi_0) = (\sigma, \phi)$ and is true by assumption. Suppose the game play has proceeded for k moves to position

$$(\sigma_0, \phi_0)(\sigma_1, \phi_1) \cdots (\sigma_k, \phi_k)$$

such that $\sigma_k \models \phi_k$. Then, according to the structure of ϕ_k, we have:

- $\phi_k = p$ for an atomic proposition p: Player $\exists$ wins
- $\phi_k = \psi_1 \vee \psi_2$: By induction hypothesis we have $\sigma_k \models \psi_1 \vee \psi_2$. Thus, the choice function ε gives us configuration (σ_k, ψ_i), where ψ_i is the smaller of ψ_1 and ψ_2 such that $\sigma_k \models \psi_i$ holds (ψ_1 in case of a tie-break). Player $\exists$ sets $(\sigma_{k+1}, \phi_{k+1}) = (\sigma_k, \psi_i)$.
- $\phi_k = \psi_1 \wedge \psi_2$: Since $\sigma_k \models \psi_1 \wedge \psi_2$ by induction hypothesis, whichever of (σ_k, ψ_1) or (σ_k, ψ_2) Player $\forall$ chooses as $(\sigma_{k+1}, \phi_{k+1})$, we always have $\sigma_{k+1} \models \phi_{k+1}$.
- temporal operators: there is no real choice and truth is easily seen to be preserved across moves.
- $\phi_k = \mathsf{E}\,\psi$: since $(\sigma_k, \mathsf{E}\,\psi)$ is true by induction hypothesis, Player $\exists$ can set $(\sigma_{k+1}, \phi_{k+1}) = \varepsilon(\sigma_k, \mathsf{E}\,\psi)$ which is a true configuration and a legal move by constraint $(*)$ above.

- $\phi_k = \mathsf{A}\,\psi$: By induction hypothesis $\sigma_k \models \mathsf{A}\,\psi$, so whatever computation ς with $\varsigma(0) = \sigma_k(0)$ Player $\forall$ may choose as σ_{k+1}, we always have $\sigma_{k+1} \models \psi$.

By the above construction, the choice function ε induces a (unique) deterministic, complete and history-free $\exists$-strategy τ. We have already seen that Player $\exists$ wins any finite play. It remains to be shown that, following the strategy τ, Player $\exists$ also wins any infinite play

$$\mu\colon (\sigma_0, \phi_0)(\sigma_1, \phi_1) \cdots (\sigma_k, \phi_k) \cdots$$

Suppose for a contradiction that his opponent, Player $\forall$, wins such an infinite play μ, with $\psi = \theta_1 \,\mathsf{U}\, \theta_2$ appearing infinitely often on μ. By Lemma 4.1.1 μ ends in the pattern

$$\cdots (\varsigma, \psi)(\varsigma, \theta_2 \vee (\theta_1 \wedge \mathsf{X}\,\psi))(\varsigma, \theta_1 \wedge \mathsf{X}\,\psi)(\varsigma, \mathsf{X}\,\psi)(\varsigma^1, \psi)(\varsigma^1, \mathsf{unf}(\psi)) \cdots$$

for some computation ς. Since all configurations on μ are true by our invariant, it follows that $\varsigma^i \models \theta_1 \,\mathsf{U}\, \theta_2$ for all $i \geq 0$. By the semantics of U we also have $\varsigma^j \models \theta_2$ for infinitely many $j \geq 0$. Thus for some $m \geq 0$ there is a configuration $c = (\varsigma^m, \theta_2 \vee (\theta_1 \wedge \mathsf{X}\,\psi))$ on μ such that $\varsigma^m \models \theta_2$. But this means that μ is not played according to τ, since strategy τ moves from c to configuration (ς^m, θ_2) (because $\varsigma^m \models \theta_2$ and $|\theta_2| < |\theta_1 \wedge \mathsf{X}\,\psi|$) and not to $(\varsigma^m, \theta_1 \wedge \mathsf{X}\,\psi)$ as is the case on μ. Contradiction. Hence, Player $\forall$ cannot win the play μ. According to Lemma 4.1.1, Player $\exists$ wins μ . $\square$

THEOREM 4.1.3. *Let $\mathcal{M} = (\mathcal{T}, V)$ be a CTL* model, σ a run of $\mathcal{T}$ and ϕ a CTL* formula. Then*

(i) The game $\mathcal{G}_{\mathcal{M}}(\sigma, \phi)$ is determined, and

(ii) Player $\exists$ wins $\mathcal{G}_{\mathcal{M}}(\sigma, \phi)$ if and only if $\mathcal{M}, \sigma \models \phi$.

PROOF. Directly from Proposition 4.1.2. $\square$

COROLLARY 4.1.4. *Let $\mathcal{M} = (\mathcal{T}, V)$ be a CTL* model, $U \subseteq \mathcal{S}_{\mathcal{T}}$ and ϕ a CTL* formula. Then*

(i) The game $\mathcal{G}_{\mathcal{M}}(U, \phi)$ is determined, and

(ii) Player $\exists$ wins $\mathcal{G}_{\mathcal{M}}(U, \phi)$ if and only if $\mathcal{M}, U \models \phi$. $\square$

4.2 Trails and Strategies

Paths through proof structures exhibit themselves considerable internal structure. In the following, we will make this structure explicit and show that to each trail ϑ of a proof structure Π corresponds a $\forall$-strategy of the game $\mathcal{G}_{\mathcal{S}}(\sigma_\vartheta, \phi)$. We then relate winningness of these strategies with the success condition for trails: we will see that the LTL game $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{A}\,\phi)$ is won by Player $\exists$ (and hence $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$) precisely if Π is successful. Unless otherwise stated we consider throughout this section an arbitrary but fixed LTL proof structure $\Pi = (\Gamma, \Delta, \gamma_r)$ for some system $\mathcal{S}$ and sequent $\gamma_r = \Xi \vdash \mathsf{A}(\phi)$.

4.2.1 Generative Paths and Admissibility

We start by defining, in a similar way as in [Dam94], the generation relations for the LTL proof rules in Figure 3.1. These relations describe dependencies among formulas as created by the application of a proof rule.

DEFINITION 4.2.1. The *generation relation* $\rightsquigarrow_{\gamma,\gamma'} \subseteq \Phi_\gamma \times \Phi_{\gamma'}$ is defined for each edge $(\gamma, \gamma') \in \Delta$ of Π by case analysis on the rule applied at γ:

$$\rightsquigarrow_{p \vdash \mathsf{A}(\Phi,q),\, p \vdash \mathsf{A}(\Phi)} \stackrel{\text{def}}{=} \mathsf{Id}(\Phi)$$

$$\rightsquigarrow_{p \vdash \mathsf{A}(\Phi,\phi_1 \vee \phi_2),\, p \vdash \mathsf{A}(\Phi,\phi_1,\phi_2)} \stackrel{\text{def}}{=} \{(\phi_1 \vee \phi_2, \phi_1),\ (\phi_1 \vee \phi_2, \phi_2)\} \cup \mathsf{Id}(\Phi)$$

$$\rightsquigarrow_{p \vdash \mathsf{A}(\Phi,\phi_1 \wedge \phi_2),\, p \vdash \mathsf{A}(\Phi,\phi_1)} \stackrel{\text{def}}{=} \{(\phi_1 \wedge \phi_2, \phi_1)\} \cup \mathsf{Id}(\Phi)$$

$$\rightsquigarrow_{p \vdash \mathsf{A}(\Phi,\phi_1 \wedge \phi_2),\, p \vdash \mathsf{A}(\Phi,\phi_2)} \stackrel{\text{def}}{=} \{(\phi_1 \wedge \phi_2, \phi_2)\} \cup \mathsf{Id}(\Phi)$$

$$\rightsquigarrow_{p \vdash \mathsf{A}(\Phi,\phi_1\,\mathsf{Z}\,\phi_2),\, p \vdash \mathsf{A}(\Phi,\mathsf{unf}(\phi_1\,\mathsf{Z}\,\phi_2))} \stackrel{\text{def}}{=} \{(\phi_1\,\mathsf{Z}\,\phi_2, \mathsf{unf}(\phi_1\,\mathsf{Z}\,\phi_2))\} \cup \mathsf{Id}(\Phi)$$

$$\rightsquigarrow_{p \vdash \mathsf{A}(\mathsf{X}\,\Phi),\, q \vdash \mathsf{A}(\Phi)} \stackrel{\text{def}}{=} \{(\mathsf{X}\,\phi, \phi) \mid \phi \in \Phi\}$$

$$\rightsquigarrow_{p \vdash \mathsf{A}(\Phi),\, q \vdash \mathsf{A}(\Phi)} \stackrel{\text{def}}{=} \mathsf{Id}(\Phi)$$

where $\mathsf{Id}(\Phi) = \{(\phi, \phi) \mid \phi \in \Phi\}$ is the identity relation on Φ. $\diamondsuit$

DEFINITION 4.2.2. (GENERATIVE PATHS) Let $\pi : \gamma_0\gamma_1 \cdots \gamma_j \cdots$ be a path in Π. We say that a (finite or infinite) sequence $\iota : \phi_0\phi_1 \cdots \phi_k \cdots$ of LTL formulas with $0 \leq |\iota| \leq |\pi|$ is a *generative path running along* π if either

- $\iota = \epsilon$, or

- $\Phi_{\gamma_0} = \{\phi_0\}$ and $\phi_i \rightsquigarrow_{\gamma_i,\gamma_{i+1}} \phi_{i+1}$ for all i with $i+1 < |\iota|$.

Denote by $I(\pi)$ the set of all generative paths and by $I^*(\pi)$ the set of finite generative paths running along π. We call the tree $I^*(\pi)$ the *internal pre-strategy* of π. $\diamondsuit$

In order to link up the game-theoretic notions of strategies and winningness to success of paths, it is convenient to introduce an alternative notion of success called admissibility which rests on generative paths. In a second step, we will then compare admissibility with success.

DEFINITION 4.2.3. (ADMISSIBILITY)

1. a path π in Π is called *admissible* if it is
   - finite and ends in an axiom, or
   - infinite and there is a generative path ι running along π such that
   $$\mathsf{inf}(\iota) \cap \mathsf{V}(\phi) \neq \varnothing.$$
2. a trail ϑ of Π is *admissible* if π_ϑ is admissible.
3. a proof structure Π is *admissible* in case all its Π-fair trails are. $\diamondsuit$

4.2.2 Internal Strategies of Trails

We now define the notion of internal (pseudo-)strategy of a trail ϑ, which is obtained by combining the suffixes of the run σ_ϑ induced by ϑ with the internal pre-strategy of the path π_ϑ induced by ϑ.

DEFINITION 4.2.4. Let $\vartheta\colon t_0 t_1 \cdots t_j \cdots$ be a trail of Π. Define

1. the sequence $\widetilde{\sigma_\vartheta}$ for $i \in \omega$ by
$$\widetilde{\sigma_\vartheta}(i) \stackrel{\text{def}}{=} h^\omega_\Sigma(\vartheta^i)$$
2. the trees T_ϑ and τ_ϑ by
$$T_\vartheta \stackrel{\text{def}}{=} \{\widetilde{\sigma_\vartheta} * \iota \mid \iota \in I^*(\pi_\vartheta)\} \qquad \text{and} \qquad \tau_\vartheta \stackrel{\text{def}}{=} \natural T_\vartheta$$
   where the operation $x * y$ is defined on words $x, y \in A^\infty$ coinductively by $x * \epsilon = \epsilon * x = \epsilon$ and $ay * bz = (a,b) \cdot (y * z)$. The tree T_ϑ is called the *internal pseudo-strategy* and τ_ϑ is called the *internal strategy* of the trail ϑ. $\diamondsuit$

Note that $\widetilde{\sigma_\vartheta}$ is a sequence of suffixes of σ_ϑ and that both T_ϑ and τ_ϑ are indeed trees. We will see shortly that τ_ϑ is a $\forall$-strategy for the game $\mathcal{G}(\sigma, \phi)$. Before that we record some basic properties of the internal (pseudo-)strategy of a trail ϑ in the form of two lemmas.

LEMMA 4.2.5. *Let ϑ be a trail of Π with $\pi_\vartheta : \gamma_0\gamma_1 \cdots \gamma_j \cdots$. Then for $0 \leq k < |\pi_\vartheta|$*

$$\psi \in \Phi_{\gamma_k} \iff \exists p \in T_\vartheta.\ |p| = k+1 \ \ \&\ \ p(k) = (\widetilde{\sigma_\vartheta}(k), \psi).$$

PROOF. By a routine induction on the length of nodes of T_ϑ. □

The previous lemma implies that the height of T_ϑ is $|\pi_\vartheta|$.

LEMMA 4.2.6. *There is a bijection between the paths of T_ϑ and those of τ_ϑ.*

PROOF. Note that whenever some node $n = n'c$ of T_ϑ has more than one child, we have $c = (\_, \phi_1 \vee \phi_2)$ and $n_{T_\vartheta} = \{nc_1, nc_2\}$ with $c_i = (\_, \phi_i)$. Since $c \neq c_1$ and $c \neq c_2$, the stutter removal operator preserves the branching structure of T_ϑ and is therefore a bijection between the paths of T_ϑ and those of τ_ϑ. □

DEFINITION 4.2.7. We say that a path π in Π is *closed* if it is either infinite or ends in a terminal sequent γ_t such that all $\psi \in \Phi_{\gamma_t}$ are assertions. Otherwise, π is called *open*. ♢

PROPOSITION 4.2.8. (TRAILS AND $\forall$-STRATEGIES I) *Let ϑ be a trail of Π. Then we have*

(i) τ_ϑ is a deterministic $\forall$-strategy for $\mathcal{G}(\sigma_\vartheta, \phi)$, and

(ii) τ_ϑ is complete *if and only if π_ϑ is closed.*

PROOF. (i) Let ϑ be a trail with $\pi_\vartheta\colon \gamma_0\gamma_1 \cdots \gamma_i \cdots$ and suppose

$$p\colon (\sigma_0, \psi_0)(\sigma_1, \psi_k) \cdots (\sigma_k, \psi_k) \in T_\vartheta$$

By the definitions of T_ϑ and $\widetilde{\sigma_\vartheta}$ we have $\sigma_j = \widetilde{\sigma_\vartheta}(j)$ for all $0 \leq j \leq k$, and for all $0 \leq i < k$

$$\sigma_{i+1} = \begin{cases} (\sigma_i)^1 & \text{if } \gamma_i \in \Gamma_{\mathsf{A(X)}} \\ \sigma_i & \text{otherwise} \end{cases}$$

and $\psi_i \rightsquigarrow_{\gamma_i, \gamma_{i+1}} \psi_{i+1}$. It follows that either $(\sigma_i, \psi_i) \triangleright (\sigma_{i+1}, \psi_{i+1})$ or $(\sigma_i, \psi_i) = (\sigma_{i+1}, \psi_{i+1})$. Thus $\natural p$ is a position of the game $\mathcal{G}(\sigma_\vartheta, \phi)$. Consequently, τ_ϑ is a tree prefix of $T_{\mathcal{G}(\sigma,\phi)}$. The fact that it is a deterministic $\forall$-strategy can be seen by inspecting the cases for the boolean operators in Definition 4.2.1.

(ii) "$\Rightarrow$": By contraposition. Suppose π_ϑ is open with $|\pi_\vartheta| = k+1$. Then there is a formula $\psi \in \Phi_{\pi_\vartheta(k)}$ that is not an assertion. By Lemma 4.2.5 there is a path $\mu \in T_\vartheta$ with $\mu(k) = (\widetilde{\sigma_\vartheta}(k), \psi)$. Hence, $\natural\mu$ is a path of τ_ϑ that is not a play of $\mathcal{G}(\sigma_\vartheta, \phi)$, so τ_ϑ is not complete.

"$\Leftarrow$": Suppose $\pi_\vartheta : \gamma_0 \cdots \gamma_j \cdots$ is closed. Let μ be a path of T_ϑ. By Lemma 4.2.6 it is sufficient to show that $\natural\mu$ is a play of $\mathcal{G}(\sigma_\vartheta, \phi)$. Suppose first that μ is finite, i.e., $\mu = n \cdot (\sigma, \psi)$ with $|\mu| = k+1$, and $\gamma_k = p \vdash \mathsf{A}(\Phi)$. By Lemma 4.2.5 we have $\psi \in \Phi$. We show that ψ is an assertion and thus $\natural\mu$ a play of $\mathcal{G}(\sigma_\vartheta, \phi)$. There are three cases:

(a) $|\mu| = |\pi|$ and γ_k is an axiom. Then ψ is an assertion since π_ϑ is closed by assumption.

(b) $|\mu| = |\pi|$ and γ_k is an anti-axiom. Then ψ is certainly an assertion.

(c) $|\mu| < |\pi|$. Then ψ must be an assertion, since otherwise by Definitions 4.2.1 and 4.2.4 μ would be extensible and hence not a path of T_ϑ.

If, on the other hand, μ is infinite, then $\natural\mu$ is an infinite path of τ_ϑ, hence a play of $\mathcal{G}(\sigma, \phi)$. $\square$

4.2.3 Winning and Losing Strategies

As a preparation to relating admissibility of trails and winningness of their internal strategies, we state some fundamental properties of finite plays in

LEMMA 4.2.9. (FINITE PLAYS) *Let ϑ be a trail of* Π. *Then*

(i) if π_ϑ ends in an axiom τ_ϑ is losing,

(ii) if π_ϑ ends in an anti-axiom τ_ϑ is winning, and

(iii) if π_ϑ does not end in an axiom all finite plays in τ_ϑ are won by $\forall$.

PROOF. Suppose $\pi : \gamma_0 \cdots \gamma_j \cdots$ is the path induced by a trail ϑ of Π. By Lemma 4.2.5, we can distinguish three situations from which a finite path $\mu \in T_\vartheta$ of length $|\mu| = k+1$ ending in some configuration (σ, q) with q an assertion may arise:

1. $\gamma_k = p \vdash \Phi, q$ is an axiom: since $\sigma(0) \models p$, also $\sigma(0) \models q$, so $\natural\mu$ is won by Player $\exists$.

2. $\gamma_k = p \vdash q$ is an anti-axiom: since $\sigma(0) \models p$, we have $\sigma(0) \not\models q$, so $\natural\mu$ is won by Player $\forall$.

3. $\gamma_k = p \vdash \Phi, q$ and rule $\mathsf{A}(bsf)$ is applied at γ_k: for similar reasons as in the previous case $\natural\mu$ is won by Player $\forall$.

These are the only cases giving rise to finite plays in τ_ϑ. Thus, point 1 proves (i), points 2 and 3 entail (iii) and (ii) follows from 2 and 3 together with Proposition 4.2.8(ii). □

PROPOSITION 4.2.10. (TRAILS AND $\forall$-STRATEGIES II) *Let ϑ be a trail of Π. Are equivalent:*

(i) ϑ is admissible,

(ii) τ_ϑ is losing, and

(iii) τ_ϑ is non-winning.

PROOF. (i)$\Rightarrow$(ii): Suppose ϑ is admissible. If π_ϑ is finite then τ_ϑ is losing by Lemma 4.2.9(i). If π_ϑ is infinite then $\widetilde{\sigma_\vartheta} * \iota$ is an infinite play won by Player $\exists$, where ι is the generative path witnessing the admissibility of π. The implication (ii)$\Rightarrow$(iii) is trivial. We show (iii)$\Rightarrow$(i) by contraposition. Suppose π_ϑ is inadmissible. If π_ϑ is finite then it ends in an anti-axiom and so τ_ϑ is winning by Lemma 4.2.9(ii). If, on the other hand, π_ϑ is infinite, then τ_ϑ is complete by Proposition 4.2.8. We also know, by virtue of Lemma 4.2.9(iii), that all finite plays in τ_ϑ are won by Player $\forall$. Moreover, by assumption no infinite play can be won by Player $\exists$. Thus τ_ϑ is a winning $\forall$-strategy. □

4.2.4 Represented Strategies

As the internal strategy of an admissible trail is loosing by the previous proposition, it can be seen as a failed attempt to produce a counter-example. Now the question naturally arises which types of strategies arise as internal strategies of trails and, more specifically, whether some winning $\forall$-strategy for a game $\mathcal{G}(\sigma, \phi)$ with $\sigma \in \mathcal{R}_\mathcal{S}(\Xi)$ is represented as a trail of Π, whenever Player $\forall$ wins that game (that is, whenever he has a winning strategy for the game).

DEFINITION 4.2.11. Let $\sigma \in \mathcal{R}_\mathcal{S}(\Xi)$ and let τ be a $\forall$-strategy for $\mathcal{G}_\mathcal{S}(\sigma, \phi)$.

- two positions $p, q \in \tau$ are called *step-equivalent,* written $p \simeq q$, if

$$\mathsf{cf}(p) = \mathsf{cf}(q) \quad \text{and} \quad \#\mathsf{X}(p) = \#\mathsf{X}(q),$$

where $\mathsf{cf}(p) = c$ if c is the current configuration in position p (that is, $p = p'c$ for some $p' \in \tau$) and $\#\mathsf{X}(p)$ denotes the number occurrences of X-formulas on p.

- τ is called *step-uniform* if it makes the same decisions for step-uniform positions, that is, for all $p, q \in \tau$:

$$p \simeq q \;\Rightarrow\; p_\tau/p = q_\tau/q$$

- a *trail* ϑ *of* Π *represents* τ if

$$\sigma_\vartheta = \sigma \quad \text{and} \quad \begin{cases} \tau_\vartheta = \tau & \text{if } \pi_\vartheta \text{ closed} \\ \tau_\vartheta \subset \tau & \text{if } \pi_\vartheta \text{ open} \end{cases}$$

We say that τ *is represented in* Π if there is a trail ϑ of Π representing τ. We call τ completely represented in Π if there is a trail ϑ representing τ such that $\tau_\vartheta = \tau$. ♢

Step-uniformity restricts the history-dependence of strategies by requiring them to make uniform decisions at uniform times (that is, after a given number of Next moves). In turns out that all complete, deterministic and step-uniform $\forall$-strategies are represented in Π.

PROPOSITION 4.2.12. (REPRESENTED $\forall$-STRATEGIES) *Suppose $\sigma \in \mathcal{R}_\mathcal{S}(\Xi)$ and let $\tau^\forall$ be a complete, deterministic and step-uniform $\forall$-strategy for $\mathcal{G}(\sigma, \phi)$. Then $\tau^\forall$ is represented in* Π.

PROOF. We construct two infinite sequences $\{\sigma_i\}_{i\in\omega}$, $\{\gamma_i\}_{i\in\omega}$ of suffixes of σ and elements of Γ^+, respectively, such that the following invariants are maintained:

I1. $\sigma_i(0) \models p_{\gamma_i}$, and

I2. for all $\iota \in I(\pi_i)$ with $|\iota| = i+1$ we have $\natural(\zeta_i * \iota) \in \tau^\forall$.

where $\pi_i = \gamma_0 \cdots \gamma_i$ and $\zeta_i = \sigma_0 \cdots \sigma_i$. We will then define a trail ϑ from these two sequences and show that the claim of the Proposition holds for that ϑ.

We start the construction with $\sigma_0 = \sigma$ and the root sequent $\gamma_0 = \Theta \vdash \mathsf{A}(\phi)$. Clearly, both invariants are fulfilled. Suppose that we have constructed the sequences up to some $k \geq 0$ and that both invariants hold for $i = k$. Consider two cases:

Case (1) γ_k is a terminal sequent. We complete the construction by defining σ_j and γ_j for $j > k$ by

$$\sigma_j = (\sigma_k)^{j-k} \quad \text{and} \quad \gamma_j = \begin{cases} \top & \text{if } \gamma_k \text{ is an axiom} \\ \bot & \text{otherwise} \end{cases}$$

Case (2): γ_k is not a terminal sequent. We construct σ_{k+1} and γ_{k+1} by case analysis on the rule R applied at γ_k. For rules $\mathsf{A}(bsf)$, $\mathsf{A}(\vee)$, $\mathsf{A}(\mathsf{U})$ and $\mathsf{A}(\mathsf{V})$, we set $\sigma_{k+1} = \sigma_k$ and we let γ_{k+1} be the unique successor sequent of γ_k in Π. The invariants are easily shown to be preserved. For rule $\mathsf{A}(sp)$ we set $\sigma_{k+1} = \sigma_k$ and choose γ_{k+1} to be some successor of γ_k such that (I1) is preserved. The existence of such a successor is guaranteed by the side condition of $\mathsf{A}(sp)$. (I2) is then trivially preserved. For rule $\mathsf{A}(\mathsf{X})$ we set $\sigma_{k+1} = (\sigma_k)^1$ and γ_{k+1} the only successor of γ_k. Again, both invariants are preserved.

The remaining and only interesting case is that of rule $\mathsf{A}(\wedge)$, so suppose rule $\mathsf{A}(\wedge)$ is applied at $\gamma_k = p \vdash \mathsf{A}(\Phi, \phi_1 \wedge \phi_2)$. Then there exists $\iota \in I(\pi_k)$ such that $|\iota| = k+1$ and $\iota = \phi \cdots (\phi_1 \wedge \phi_2)$, implying by (I2) that $p = p' \cdot (\sigma_k, \phi_1 \wedge \phi_2) = \natural(\zeta_k * \iota) \in \tau^\forall$. Since $\tau^\forall$ is complete and deterministic we have $p \cdot (\sigma_k, \phi_j) \in \tau^\forall$ for either $j = 1$ or $j = 2$. Set $\sigma_{k+1} = \sigma_k$ and $\gamma_{k+1} = p \vdash \mathsf{A}(\Phi, \phi_j)$. Invariant (I1) is trivially preserved. To see that (I2) is preserved, note that any position $q \in \tau^\forall$ with $q = \natural(\zeta_k * \iota')$ for some $\iota' \in I(\pi_k)$ of length $k+1$ and ending in $\phi_1 \wedge \phi_2$ is in fact step-equivalent with p. Since $\tau^\forall$ is step-uniform by assumption, it follows that $q \cdot (\sigma_k, \phi_j) = \natural(\zeta_{k+1} * (\iota' \cdot \phi_j)) \in \tau^\forall$. Hence, (I2) is preserved.

Now, define the trail ϑ for $j \in \omega$ by

$$\vartheta(j) = \sigma_j(0)[K \mapsto \gamma_j]$$

It is not difficult to see that ϑ is indeed a trail with

$$\sigma_\vartheta = \sigma \quad \text{and} \quad \widetilde{\sigma_\vartheta}(i) = \sigma_i \text{ for all } i \in \omega$$

Since any position $p \in \tau_\vartheta$ can be represented as $\natural(\zeta_k * \iota)$ for some $k \geq 0$ and $\iota \in I(\pi_k)$, we certainly have $\tau_\vartheta \subseteq \tau^\forall$. Moreover, as τ_ϑ is complete precisely if π_ϑ is closed (Lemma 4.2.8(ii)), we have $\tau_\vartheta = \tau^\forall$ if and only if π_ϑ is closed, showing that $\tau^\forall$ is represented in Π. □

The observation that any history-free winning strategy is complete and step-uniform immediately yields

COROLLARY 4.2.13. *Suppose* $\sigma \in \mathcal{R}_\mathcal{S}(\Xi)$ *and let* $\tau^\forall$ *be a deterministic, history-free winning* $\forall$*-strategy for* $\mathcal{G}(\sigma, \phi)$. *Then* $\tau^\forall$ *is completely represented in* Π. □

Note that the converse of Proposition 4.2.12 does not hold, even if we drop the completeness requirement: as the following example shows not every strategy represented in Π is necessarily step-uniform.

EXAMPLE 4.2.14. Figure 4.2 displays a path segment $\gamma_0\gamma_1\gamma_2\gamma_3$ of a proof structure.

$$\begin{tikzcd}
\boxed{\gamma_0 : p \vdash \mathsf{A}(\phi_1 \wedge \phi_2, \mathsf{G}(\phi_1 \wedge \phi_2))} \arrow[d] \\
\boxed{\gamma_1 : p \vdash \mathsf{A}(\phi_1, \mathsf{G}(\phi_1 \wedge \phi_2))} \arrow[d] \\
\boxed{\gamma_2 : p \vdash \mathsf{A}(\phi_1, \phi_1 \wedge \phi_2)} \arrow[d] \\
\boxed{\gamma_3 : p \vdash \mathsf{A}(\phi_1, \phi_2)}
\end{tikzcd}$$

Figure 4.2: A path segment leading to a non-step-uniform strategy

As a result of applying rule $\mathsf{A}(\wedge)$ to $\phi_1 \wedge \phi_2$ at γ_0 the disjunct ϕ_1 is selected, while at γ_2 the application of the same rule to $\phi_1 \wedge \phi_2$ selects ϕ_2. It is not hard to see that as a consequence the internal strategy τ_ϑ of any trail ϑ with π_ϑ containing this segment is not step-uniform. ♣

As the example suggests, proof structures can be constructed in a way such that multiple choices as in the example can be avoided and all strategies produced by its trails are step-uniform. A sufficient condition is to choose at any sequent γ a $\preccurlyeq$-maximal formula $\psi \in \Phi_\gamma$ and apply to ψ the rule corresponding to its top-level operator.

But let us after this short digression return to our main track.

4.2.5 Winningness and Admissibility

Putting together the results of the previous two sections, we get

PROPOSITION 4.2.15. (WINNINGNESS AND ADMISSIBLE TRAILS) *Let $\sigma \in \mathcal{R}_\mathcal{S}(\Xi)$. Then Player $\exists$ wins $\mathcal{G}_\mathcal{S}(\sigma, \phi)$ if and only if all trails inducing σ are admissible.*

PROOF. By contraposition using Proposition 4.2.10 and Corollary 4.2.13. □

The previous proposition can now easily be lifted to the level of proof structures as is recorded in

THEOREM 4.2.16. (WINNINGNESS AND ADMISSIBLE PROOF STRUCTURES) *Let Π be a LTL proof structure for $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}(\phi)$. Then Player $\exists$ wins $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{A}\,\phi)$ if and only if Π is admissible.*

PROOF. Note that Player $\exists$ wins $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{A}\,\phi)$ iff he wins $\mathcal{G}_{\mathcal{S}}(\sigma, \phi)$ for all $\sigma \in \mathcal{C}_{\mathcal{S}}(\Xi)$. The result then follows directly from Proposition 4.2.15. $\square$

4.2.6 Admissibility vs. Success

The previous section showed that the notion of admissibility of a proof structure Π for system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}(\phi)$ characterises the property that Player $\exists$ wins the game $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{A}\,\phi)$ (and hence $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$). In this section we study the relation between admissibility and success.

By examining the proof rules and the definition of the generation relation, it becomes clear that a successful path is also admissible. The converse implication does not hold for infinite paths in general as is demonstrated by the following counter-example.

EXAMPLE 4.2.17. Figure 4.3 shows a path π in a proof structure for property $\mathsf{A}(\mathsf{G}\,p \vee \mathsf{F}\,\mathsf{G}\,p)$. As the actual system is quite irrelevant for this example only the right-hand side of each sequent is shown. The dashed arrows indicate generative paths. Clearly, π is admissible, but not successful. ♣

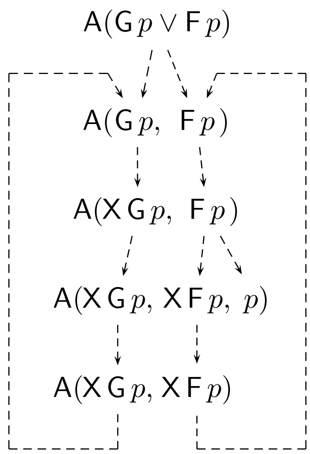

Figure 4.3: An admissible, but unsuccessful path

However, it turns out that if there is an admissible, but unsuccessful trail ϑ in a proof structure then, although τ_ϑ is a losing $\forall$-strategy, it can be transformed into a winning strategy.

LEMMA 4.2.18. *Let ϑ be an admissible, but unsuccessful trail of* Π. *Then*

(i) Player $\forall$ wins the game $\mathcal{G}_{\mathcal{S}}(\sigma_\vartheta, \phi)$, and

(ii) there is an inadmissible trail ϑ' in Π such that $\sigma_{\vartheta'} = \sigma_\vartheta$.

PROOF. (i) Suppose ϑ is an admissible, but unsuccessful trail of ϑ. It follows that σ_ϑ is a Ξ-run of $\mathcal{S}$ and that π_ϑ is infinite, admissible and unsuccessful.

We show that from τ_ϑ we can construct a winning $\forall$-strategy for $\mathcal{G}_{\mathcal{S}}(\sigma_\vartheta, \phi)$. Clearly, all finite plays are won by Player $\forall$, but we have to eliminate the "offending" infinite plays won by Player $\exists$. Loosely speaking, any play in τ_ϑ lost by Player $\forall$ is due to some "unlucky" choices, while the right (winning) choice is possible each time and even present elsewhere in τ_ϑ.

Let us call ψ-path a path in a (pre-)strategy on which V-formula ψ appears infinitely often (i.e., a play won by $\exists$) and let V be the set of V-formulas with a ψ-path appearing in τ_ϑ (and thus in T_ϑ). Let $\psi_1, \psi_2, \ldots, \psi_m$ be some linearisation of the partial order $(V, \succcurlyeq)$, that is, for all ψ_i, ψ_j with $i \leq j$ either $\psi_i \not\succcurlyeq \psi_j$ or they are incomparable w.r.t. the subformula order. Suppose $\psi_j = \phi_{j1} \mathsf{V}\, \phi_{j2}$ for each $1 \leq j \leq m$.

We construct a sequence $T_1, \ldots, T_{m+1}$ of trees starting from $T_1 = T_\vartheta$ and respecting the following two invariants:

J1. $\tau_i = \natural T_i$ is a complete $\forall$-strategy with all finite plays won by Player $\forall$

J2. if there is a ψ-path in T_i then $\psi \in \{\psi_i, \ldots, \psi_m\}$.

It follows that T_{m+1} contains no ψ-path and thus τ_{m+1} is winning. Clearly, both invariants hold for T_1.

In order to construct T_{i+1} from T_i, consider a ψ_i-path μ in T_i. Note that

- $(\sigma^k, \mathsf{unf}(\psi_i))$ is present on μ for all but finitely many k by Lemma 4.1.1
- since π_ϑ is infinite and there is a ψ_i-path in τ_ϑ, the unsuccessfulness of π_ϑ is due to an infinite number of occurrences of ϕ_{i2} on π_ϑ, implying that there are infinitely many j such that $\mathsf{cf}(n) = (\sigma^j, \phi_{i2})$ for some node $n \in T_\vartheta$

It follows from these two observation that there exist a $l \geq 0$, a finite prefix $p_\mu \in T_i$ of μ with $\mathsf{cf}(p_\mu) = (\sigma^l, \mathsf{unf}(\psi_i))$ and a node $q_\mu \in T_\vartheta$ with $\mathsf{cf}(q_\mu) = (\sigma^l, \phi_{i2})$. Now we replace in T_i the set of nodes $p_\mu \cdot (T_i/p_\mu)$ by

$$p_\mu \cdot (\sigma^l, \phi_{i2}) \cdot (T_\vartheta/q_\mu)$$

T_{i+1} is obtained from T_i by performing such a replacement for each ψ_i-path μ in T_i. Clearly, J1 is preserved. Since any of the trees T_ϑ/q_μ can contain ψ_j-paths at most for $j > i$, invariant J2 is also preserved.

(ii) Follows from (i) by Proposition 4.2.15. $\square$

PROPOSITION 4.2.19. (WINNINGNESS AND SUCCESSFUL TRAILS) *Let* $\sigma \in \mathcal{R}_{\mathcal{S}}(\Xi)$. *Then Player* $\exists$ *wins* $\mathcal{G}_{\mathcal{S}}(\sigma, \phi)$ *if and only if all trails of* Π *inducing* σ *are successful.*

PROOF. "$\Rightarrow$": By contraposition. Let $\sigma \in \mathcal{R}_{\mathcal{S}}(\Xi)$ and suppose there is an unsuccessful trail ϑ of Π inducing σ. We have to show that Player $\forall$ wins $\mathcal{G}_{\mathcal{S}}(\sigma, \phi)$. If ϑ is inadmissible this follows from Proposition 4.2.15, otherwise from Lemma 4.2.18. "$\Leftarrow$": By Proposition 4.2.15, since any successful trail is admissible. $\square$

An important consequence of this proposition is that any Ξ-computation σ following an unsuccessful path provides a counter-example to the statement $\mathcal{S}, \Xi \models \phi$, that is, $\sigma \not\models \phi$.

The next theorem shows that, although the notions of admissibility and success do not coincide on the level of individual paths or trails, they do on the level of proof structures.

THEOREM 4.2.20. (WINNINGNESS, ADMISSIBILITY AND SUCCESS) *Let* Π *be a proof structure for system* $\mathcal{S}$ *and sequent* $\Xi \vdash \mathsf{A}(\phi)$. *Are equivalent:*

(i) *Player* $\exists$ *wins* $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{A}\,\phi)$, *and*

(ii) Π *is admissible, and*

(iii) Π *is successful, i.e.,* $\Pi\colon \mathcal{S}, \Xi \Vdash \mathsf{A}\,\phi$.

PROOF. By Theorem 4.2.16 and Proposition 4.2.19. $\square$

We are now in a position to make up for the proof of Proposition 3.3.6. The proposition is restated here for convenience.

PROPOSITION 4.2.21. *Let* Π *be a proof structure for system* $\mathcal{S}$ *and sequent* $\Xi \vdash \mathsf{A}(\phi)$. *Suppose that there are no* U*-subformulas in* ϕ *and that all terminals in* Π *are axioms. Then* $\Pi\colon \mathcal{S}, \Xi \Vdash \mathsf{A}\,\phi$ *is successful.*

PROOF. Since there are no U-subformulas in ϕ and all terminals in Π are axioms all paths in Π are admissible. Hence Π is admissible. By Theorem 4.2.20 Π is successful. $\square$

4.3 Existence of a Proof Structure

Results obtained so far indicate that any *given* proof structure Π for a system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}(\phi)$ is successful precisely if Player $\exists$ wins the game $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{A}\,\phi)$. The next step is to show that such a proof structure does indeed *exist*, whenever Player $\exists$ wins $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{A}\,\phi)$, that is, $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$. For this purpose we need the ability to characterise the set of states satisfying a LTL formula $\mathsf{A}\,\psi$ by an assertion from $\mathcal{L}_\mu$. It is at this point that the need for the expressiveness of the fixed point operators in $\mathcal{L}_\mu$ arises.

LEMMA 4.3.1. *There is a function $\chi\colon CTL* \to \mathcal{L}_\mu$ such that for any system $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ and any ground-quantified CTL* formula ψ with free variables $Y \subseteq X$ we have that $\chi(\psi)$ is a formula of $\mathcal{L}_\mu$ with the same free variables Y such that*

$$\{s \in \Sigma \mid s \models \psi\} = \|\chi(\psi)\|$$

PROOF. By Proposition 2.4.4 we can transform ψ into an equivalent formula of the modal μ-calculus using the translation described by Dam [Dam94] (see also [Ref96]). The translation from the modal μ-calculus to our extended assertion language $\mathcal{L}_\mu$ is then straightforward. $\square$

NOTATION. We will write χ_ψ instead of $\chi(\psi)$.

LEMMA 4.3.2. *Let $\mathcal{S}$ be a system, Ξ a satisfiable assertion and $\mathsf{A}\,\phi$ a LTL formula. Suppose Player $\exists$ wins $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{A}\,\phi)$. Then there exists a proof structure Π for $\mathcal{S}$ and $\Xi \vdash \mathsf{A}(\phi)$.*

PROOF. We construct a sequence $\Pi_0, \Pi_1, \ldots, \Pi_n$ of pre-proof structures for $\mathcal{S}$ and ϕ, while maintaining the following invariant for all nodes $p \vdash \mathsf{A}\,\Phi$ of each Π_i:

$$p \models \mathsf{A}(\bigvee_{\phi \in \Phi} \phi) \text{ and } p \text{ is satisfiable.}$$

From the hypothesis it follows that $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$ and Ξ is satisfiable, so the invariant holds for the root sequent $\Xi \vdash \mathsf{A}(\phi)$, which makes up the initial pre-proof structure Π_0. Let us call a node of a pre-proof structure *open,* in case it has no successors, but is not a terminal. In order to construct Π_{i+1} from Π_i we pick an open node γ in Π_i and apply some rule(s) to it in the way described below. If some rule application generates a node that exists already, we loop back to that node. Finally, we will show that this procedure terminates, yielding a “canonical” proof structure $\Pi = \Pi_n$ for $\mathcal{S}$ and $\Xi \vdash \mathsf{A}(\phi)$.

Let $\gamma = p \vdash \mathsf{A}(\Phi)$ be an open node of Π_i. We first consider the case, where rule $\mathsf{A}(\mathsf{X})$ is not applicable to γ. Let $\Phi = \Psi, \psi$ where ψ is not a Next formula. We proceed by case analysis on the structure of ψ. In case the top-level connective in ψ is a binary operator ($\wedge, \vee, \mathsf{V}$ or U), we simply apply the respective rule ($\mathsf{A}(\wedge)$,$\mathsf{A}(\vee)$, $\mathsf{A}(\mathsf{V})$ or $\mathsf{A}(\mathsf{U})$) at γ. Since these rules are all backwards sound, the invariant is preserved and Π_{i+1} is easily seen to be a pre-proof structure.

Now suppose $\psi = q$ is an assertion. Since γ is an open node, hence not a terminal sequent, we have the following two cases:

(a) $\models p \to \neg q$ and $\Psi \neq \varnothing$. We simply apply rule $\mathsf{A}(\mathit{bsf})$. The invariant is clearly preserved and Π_{i+1} is a pre-proof structure.

(b) $p \wedge q$ and $p \wedge \neg q$ are both satisfiable. Let r be an assertion equivalent to q such that sequent $p \wedge \neg r \vdash \mathsf{A}(\Psi, q)$ does not appear in Π_i. We continue the construction as follows

$$\mathsf{A}(sp)\ \frac{\gamma\colon\ p \vdash \mathsf{A}(\Psi, q)}{\dfrac{\gamma'\colon\ p \wedge r \vdash \mathsf{A}(\Psi, q)}{\checkmark} \qquad \mathsf{A}(\mathit{bsf})\ \dfrac{\gamma''\colon\ p \wedge \neg r \vdash \mathsf{A}(\Psi, q)}{\gamma'''\colon\ p \wedge \neg r \vdash \mathsf{A}(\Psi)}}$$

The application of rule $\mathsf{A}(sp)$ yields two new sequents γ' (an axiom) and γ''. Our choice of r avoids a potential loop back to a node where the Split rule is already applied. This ensures that condition (A-SPL) in the definition of a (pre-)proof structure is preserved. Clearly, the invariant holds for both γ' and γ''. Hence γ'' can not be an anti-axiom, so $\Psi \neq \varnothing$ and we can apply rule $\mathsf{A}(\mathit{bsf})$ at γ'' yielding node γ''' which also satisfies the invariant. In this way we obtain pre-proof structure Π_{i+1}.

For the case that rule $\mathsf{A}(\mathsf{X})$ is applicable to γ, we apply it in the following way:

$$\frac{p \vdash \mathsf{A}(\mathsf{X}\,\Phi)}{\chi_{\mathsf{A}(\Phi)} \vdash \mathsf{A}(\Phi)}$$

Since $p \models \mathsf{A}(\bigvee_{\phi\in\Phi} \mathsf{X}\,\phi)$ by the invariant, the side condition $p \models [\Lambda]\chi_{\mathsf{A}(\Phi)}$ of the rule holds. As p is satisfiable by the invariant and any state of $\mathcal{S}$ has a successor by assumption, $\chi_{\mathsf{A}\,\Phi}$ is satisfiable as well. Hence Π_{i+1} is a pre-proof structure. Moreover, $\chi_{\mathsf{A}(\Phi)} \models \mathsf{A}(\Phi)$ holds trivially, so the invariant is preserved.

Finally, we show that the above procedure terminates. Suppose for a contradiction that it does not terminate yielding an infinite pseudo-proof structure $\widehat{\Pi}$ by transfinite iteration. Now consider a spanning tree T for $\widehat{\Pi}$ rooted at γ_r. As any node of $\widehat{\Pi}$ is reachable from γ_r, all nodes of $\widehat{\Pi}$ must appear on T. Since $\widehat{\Pi}$ (and hence T) is finitely branching, there is an infinite branch π in T by König's Lemma. By Lemma 3.1.5, there must be an infinite number of applications of rule $\mathsf{A}(\mathsf{X})$ on π. But since there can only be a finite number of different right-hand sides of the form $\mathsf{A}(\mathsf{X}\,\Phi)$ appearing in sequents of $\widehat{\Pi}$, there must be two sequents γ and γ' containing same set of formulas $\mathsf{X}\,\Phi$ on π. By construction γ and γ' have the same and only successor sequent $\chi_{\mathsf{A}(\Phi)} \vdash \mathsf{A}(\Phi)$ which must thus appear twice on π, a contradiction, because π being a branch of the tree T is cycle-free. Hence, our construction terminates after a finite number of steps, yielding a proof structure Π. $\square$

THEOREM 4.3.3. (WINNINGNESS AND PROVABILITY) *Let $\mathcal{S}$ be a system, Ξ a satisfiable assertion and $\mathsf{A}\,\phi$ a LTL formula. Then Player $\exists$ wins $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{A}\,\phi)$ if and only if $\mathcal{S}, \Xi \Vdash \mathsf{A}\,\phi$ (that is, there exists a successful proof structure Π for $\mathcal{S}$ and $\Xi \vdash \mathsf{A}\,\phi$).*

PROOF. By Theorem 4.2.20 and Lemma 4.3.2. $\square$

4.4 Soundness and Completeness of Rule $\mathsf{A}(S)$

The final missing link in our correctness proof (third equivalence in Table 4.1) states that Rule A(S) is sound and complete relative to assertional validity for proving success of a proof structure.

PROPOSITION 4.4.1. (RELATIVE COMPLETENESS OF RULE $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$) *Let $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta)$ be a saturated system and let q and $p_1, \ldots, p_m$ be assertions. Then $\mathcal{S} \models \mathsf{A}(\mathsf{F}\,q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i)$ implies $\mathcal{S} \vdash \mathsf{A}(\mathsf{F}\,q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i)$.*

PROOF. The proof of relative completeness of Rule F-RESP in [MP91] can be adapted without difficulties, so we do not repeat it here. $\square$

From this proposition and Proposition 3.3.5 we immediately get

THEOREM 4.4.2. (SOUNDNESS AND RELATIVE COMPLETENESS OF RULE A(S)) *Let Π be a proof structure for a* saturated *system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{A}\,\phi$. Then $\Pi\colon \mathcal{S}, \Xi \Vdash \mathsf{A}\,\phi$ if and only if $\Pi\colon \mathcal{S}, \Xi \vdash \mathsf{A}\,\phi$.* $\square$

4.5 Main Result

The results of this chapter are summarised in

THEOREM 4.5.1. (SOUNDNESS AND RELATIVE COMPLETENESS FOR LTL) *Let $\mathcal{S}$ be a* saturated *system, Ξ a satisfiable assertion and $\mathsf{A}\,\phi$ a LTL formula. Then*

$$\mathcal{S}, \Xi \models \mathsf{A}\,\phi \textit{ if and only if } \mathcal{S}, \Xi \vdash \mathsf{A}\,\phi.$$

PROOF. The three steps depicted in Figure 4.1 are covered by Corollary 4.1.4 and Theorems 4.3.3 and 4.4.2, respectively. □

Chapter 5

ELL and CTL* Proof Structures

In this chapter we extend our proof system to CTL*. To this end, we first introduce ELL proof structures, the duals of LTL proof structures. The ELL proof rules can essentially be derived from the LTL rules by dualising the right-hand sides of sequents and side conditions. The symmetry is however not perfect. In particular, handling disjunction in the ELL system is more difficult than conjunction in the LTL system. We have two rules for disjunction which need to be applied in combination with the (ELL) Split rule, requiring a judicious choice of assertions. The notion of the associated system and trails of an ELL proof structure are analogous to the LTL case, and the success condition is dual to the one for LTL. We also present a success rule for ELL proof structure.

As expected, trails of ELL proof structures correspond to $\exists$-strategies in a similar way as LTL trails correspond to $\forall$-strategies. Most of the results about strategies can be transferred directly from the LTL case by duality. The existence of a "canonical" ELL proof structure and the strategies represented in it needs to be reviewed. The proof system for ELL is shown to be sound and relatively complete.

We extend our proof systems to apply to arbitrary CTL* formulas by combining LTL and ELL proof structures. The LTL and ELL rules dealing with assertions are extended to path-quantified formulas, which may appear arbitrarily nested inside CTL* formulas. For the case of a path-quantified formula, the side conditions for the extended rules require the construction of a new proof structure. As we prove statements of the form $\mathcal{S}, \Xi \models \phi$ and $\phi \approx \mathsf{A}\,\phi$ for any CTL* formula ϕ we can assume w.l.o.g. that any CTL* formula has a top-level path quantifier. A CTL* proof structure is then essentially a collection of ELL and LTL proof structures and it is a (S-)proof if the constituent proof structures are (S-)proofs. As the dependency among the latter proof structures is acyclic, path-quantified formulas can essentially be treated like assertions and the proof of soundness and relative completeness for CTL* directly lifts from the base cases for LTL and ELL.

5.1 ELL Proof Structures

Given a system $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$, an *ELL sequent* is of the form $p \vdash \mathsf{E}(\Phi)$, where p is an assertion and Φ is a non-empty, finite set of ground-quantified, path-quantifier-free CTL* formulas. A sequent $p \vdash \mathsf{E}(\Phi)$ is *valid* if $p \models \mathsf{E}(\bigwedge_{\phi\in\Phi} \phi)$, that is, from any state s satisfying p there is a s-computation of $\mathcal{S}$ satisfying the *conjunction* of the formulas appearing in Φ. We use abbreviations analogous to those used for LTL sequents.

An ELL proof structure is defined in the same way as a LTL proof structure, except that the form of the sequents and the set of rules is replaced:

DEFINITION 5.1.1. A *ELL proof structure* for a system $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ and sequent $\Xi \vdash \mathsf{E}\,\phi$ is a rooted graph

$$\Pi = (\Gamma, \Delta \subseteq \Gamma \times \Gamma, \gamma_r \in \Gamma),$$

where Γ is a finite set of sequents, $\gamma_r = \Xi \vdash \mathsf{E}\,\phi$ is the *root sequent* and for each node $\gamma \in \Gamma$ we have that

(E-SAT) p_γ is satisfiable,

(E-ACC) γ is reachable from γ_r,

(E-RUL) if γ has $n \geq 0$ successors $\{\gamma_1, \dots, \gamma_n\} = \{\gamma' \mid (\gamma, \gamma') \in \Delta\}$ then

$$R\,\frac{\gamma}{\gamma_1 \quad \cdots \quad \gamma_n}\,C_R$$

is the correct application of some rule R from Table 5.1, that is, the rules' side condition C_R is satisfied, and

(E-SPL) if $(\gamma, \gamma') \in \Delta$ then rule $\mathsf{E}(sp)$ *not* applied to both γ and γ'. $\Diamond$

We call a sequent γ an *axiom (anti-axiom)* if rule $\mathsf{E}(ax)$ ($\mathsf{E}(nx)$) is applied to γ. A sequent that is either an axiom or an anti-axiom is called *terminal*. The set of sequents where rule R is applied is again denoted by Γ_R.

Note that condition (E-SPL) ensures the temporal consistency of proof structures. In fact, Lemma 3.1.5 directly transfers to ELL proof structures: on any infinite path through an ELL proof structure, there is an infinite number of applications of Rule $\mathsf{E}(\mathsf{X})$.

$\mathsf{E}(ax)$	$\dfrac{p \vdash \mathsf{E}(q)}{.}$	$p \models q$
$\mathsf{E}(nx)$	$\dfrac{p \vdash \mathsf{E}(\Phi, q)}{.}$	$p \models \neg q$
$\mathsf{E}(bsf)$	$\dfrac{p \vdash \mathsf{E}(\Phi, q)}{p \vdash \mathsf{E}(\Phi)}$	$p \models q$
$\mathsf{E}(\vee_l)$	$\dfrac{p \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)}{p \vdash \mathsf{E}(\Phi, \phi_1)}$	
$\mathsf{E}(\vee_r)$	$\dfrac{p \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)}{p \vdash \mathsf{E}(\Phi, \phi_2)}$	
$\mathsf{E}(\wedge)$	$\dfrac{p \vdash \mathsf{E}(\Phi, \phi_1 \wedge \phi_2)}{p \vdash \mathsf{E}(\Phi, \phi_1, \phi_2)}$	
$\mathsf{E}(\mathsf{U})$	$\dfrac{p \vdash \mathsf{E}(\Phi, \phi_1 \,\mathsf{U}\, \phi_2)}{p \vdash \mathsf{E}(\Phi, \phi_2 \vee (\phi_1 \wedge \mathsf{X}(\phi_1 \,\mathsf{U}\, \phi_2)))}$	
$\mathsf{E}(\mathsf{V})$	$\dfrac{p \vdash \mathsf{E}(\Phi, \phi_1 \,\mathsf{V}\, \phi_2)}{p \vdash \mathsf{E}(\Phi, \phi_2 \wedge (\phi_1 \vee \mathsf{X}(\phi_1 \,\mathsf{V}\, \phi_2)))}$	
$\mathsf{E}(\mathsf{X})$	$\dfrac{p \vdash \mathsf{E}(\mathsf{X}\,\Phi)}{q \vdash \mathsf{E}\,\Phi}$	$p \models \langle\Lambda\rangle\, q$
$\mathsf{E}(sp)$	$\dfrac{p \vdash \mathsf{E}\,\Phi}{q_1 \vdash \mathsf{E}\,\Phi \quad \cdots \quad q_n \vdash \mathsf{E}\,\Phi}$	$p \models \bigvee_{i=1}^{n} q_i$

Table 5.1: ELL proof rules

5.1.1 *Some Remarks on the Rules*

The interpretation of right-hand side of sequents is dual to the one for LTL sequents. Note that no dualisation takes place on the left-hand side of sequents: the intended meaning of $p \vdash \mathsf{E}(\Phi)$ is still "*for all* states s satisfying p, ...". Besides the obvious substitution of E for A, the ELL rules can be systematically obtained from the LTL rules by

- swapping the format of axioms and anti-axioms w.r.t. their LTL counterparts, that is, ELL axioms are of the form $p \vdash \mathsf{E}(q)$ and anti-axioms of the form $p \vdash \mathsf{E}(\Phi, q)$,
- replacing all operators in the right-hand side of sequents by their duals; hence, rule $\mathsf{A}(op)$ becomes rule $\mathsf{E}(op^d)$, where op and op^d are dual operators (pairs of dual operators are $(\wedge, \vee)$, (V, U) and the self-dual (X, X)),
- negating assertions in side conditions that also occur on the right-hand side of the sequent (concretely, side conditions for axioms, anti-axioms and the predicate rule of the form $p \models q$ are replaced by $p \models \neg q$ and vice versa), and
- replacing the weakest precondition operator $[\Lambda]$ in the side condition of Rule $\mathsf{A}(\mathsf{X})$ by its dual $\langle\Lambda\rangle$ in the side condition of Rule $\mathsf{E}(\mathsf{X})$; observe that the assertion q is not negated as it appears on the left-hand side of the premise sequent

The Split Rule remains the same (up to substituting E for A) as it concerns only the left-hand side of sequents. There is one exception to this symmetry: rule $\mathsf{A}(\wedge)$ should become

$$\mathsf{E}(\vee)' \ \frac{p \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)}{p \vdash \mathsf{E}(\Phi, \phi_1) \quad p \vdash \mathsf{E}(\Phi, \phi_2)}$$

but there are two disjunction rules, $\mathsf{E}(\vee_l)$ and $\mathsf{E}(\vee_r)$, in our system. The reason is that the above rule is not backwards sound: $p \models \mathsf{E}(\Phi, \phi_1 \vee \phi_2)$ does not necessarily imply $p \models \mathsf{E}(\Phi, \phi_1)$ and $p \models \mathsf{E}(\Phi, \phi_2)$, hence threatening completeness of the proof system (see also discussion of A-sequent format in Section 3.1.1). The reader might object that none of our two rules $\mathsf{E}(\vee_l)$ and $\mathsf{E}(\vee_r)$ is backwards sound either. This is correct, but they form the building blocks for the following rule, which is derivable from $\mathsf{E}(\vee_l)$, $\mathsf{E}(\vee_r)$ and $\mathsf{E}(sp)$:

$$\mathsf{E}(\vee) \ \frac{p \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)}{q_1 \vdash \mathsf{E}(\Phi, \phi_1) \quad q_2 \vdash \mathsf{E}(\Phi, \phi_2)} \ p \models q_1 \vee q_2$$

This rule generalises rule $\mathsf{E}(\vee)'$ above. The reason for not including rule $\mathsf{E}(\vee)$ instead of $\mathsf{E}(\vee_l)$ and $\mathsf{E}(\vee_r)$ right from the beginning is that by condition (E-SAT) we cannot choose one of the q_i's to be equivalent to false, so $\mathsf{E}(\vee_l)$ and $\mathsf{E}(\vee_r)$ are still needed. The case is reminiscent of rule $\mathsf{A}(bsf)$ that in general needs to be applied in conjunction with the Split rule $\mathsf{A}(sp)$.

Observe that in general there are more choices to be made in the construction an ELL proof structure than in the construction of a LTL proof structure. In particular, the disjunction rule $\mathsf{E}(\vee)$ above requires a judicious choice of the two assertions q_1 and q_2.

5.1.2 Derived Rules

Some useful derived ELL proof rules are summarised in Table 5.2.

$\mathsf{E}(\vee)$	$\dfrac{p \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)}{q_1 \vdash \mathsf{E}(\Phi, \phi_1) \quad q_2 \vdash \mathsf{E}(\Phi, \phi_2)}$	$p \models q_1 \vee q_2$
$\mathsf{E}(\mathsf{U})'$	$\dfrac{p \vdash \mathsf{E}(\Phi, \phi_1 \,\mathsf{U}\, \phi_2)}{q_1 \vdash \mathsf{E}(\Phi, \phi_2) \quad q_2 \vdash \mathsf{E}(\Phi, \phi_1, \mathsf{X}(\phi_1 \,\mathsf{U}\, \phi_2))}$	$p \models q_1 \vee q_2$
$\mathsf{E}(\mathsf{V})'$	$\dfrac{p \vdash \mathsf{E}(\Phi, \phi_1 \,\mathsf{V}\, \phi_2)}{q_1 \vdash \mathsf{E}(\Phi, \phi_2, \phi_1) \quad q_2 \vdash \mathsf{E}(\Phi, \phi_2, \mathsf{X}(\phi_1 \,\mathsf{V}\, \phi_2))}$	$p \models q_1 \vee q_2$
$\mathsf{E}(\mathsf{F})$	$\dfrac{p \vdash \mathsf{E}(\Phi, \mathsf{F}\,\psi)}{q_1 \vdash \mathsf{E}(\Phi, \psi) \quad q_2 \vdash \mathsf{E}(\Phi, \mathsf{X}\,\mathsf{F}\,\psi)}$	$p \models q_1 \vee q_2$
$\mathsf{E}(\mathsf{G})$	$\dfrac{p \vdash \mathsf{E}(\Phi, \mathsf{G}\,\psi)}{p \vdash \mathsf{E}(\Phi, \psi, \mathsf{X}\,\mathsf{G}\,\psi)}$	
$\mathsf{E}(\mathsf{X})'$	$\dfrac{p \vdash \mathsf{E}(\mathsf{X}\,\Phi)}{q_1 \vdash \mathsf{E}(\Phi) \ \cdots \ q_n \vdash \mathsf{E}(\Phi)}$	$p \models \langle \Lambda \rangle \bigvee_{j=1}^{n} q_j$

Table 5.2: derived ELL rules

Due to the disjunctions appearing in unfoldings of temporal formulas, there also exists left and right single branch versions of rules $\mathsf{E}(\mathsf{V})'$, $\mathsf{E}(\mathsf{U})'$ and $\mathsf{E}(\mathsf{F})$ for the case where one of q_1 and q_2 is equivalent to false. For instance, rule $\mathsf{E}(\mathsf{V})'$ also appears in the following variants:

$$\mathsf{E}(\mathsf{V}_l)\ \frac{p \vdash \mathsf{E}(\Phi, \phi_1 \,\mathsf{V}\, \phi_2)}{p \vdash \mathsf{E}(\Phi, \phi_2, \phi_1)} \qquad \mathsf{E}(\mathsf{V}_r)\ \frac{p \vdash \mathsf{E}(\Phi, \phi_1 \,\mathsf{V}\, \phi_2)}{p \vdash \mathsf{E}(\Phi, \phi_2, \mathsf{X}(\phi_1 \,\mathsf{V}\, \phi_2))}$$

This said, rule $\mathsf{E}(\mathsf{G})$ is nothing else than rule $\mathsf{E}(\mathsf{V}_r)$ with $\phi_1 \stackrel{\text{def}}{=} \mathsf{false}$.

The following rule for the 'unless' operator (W) is *not* derivable, but sound and can thus be added to our system:

$$\mathsf{E}(\mathsf{W})\ \frac{p \vdash \mathsf{E}(\Phi, \phi_1 \,\mathsf{W}\, \phi_2)}{q_1 \vdash \mathsf{E}(\Phi, \phi_2) \quad q_2 \vdash \mathsf{E}(\Phi, \phi_1, \mathsf{X}(\phi_1 \,\mathsf{W}\, \phi_2))}\ p \models q_1 \vee q_2$$

It is based on the equivalence

$$\phi_1 \,\mathsf{W}\, \phi_2 \equiv \phi_2 \vee (\phi_1 \wedge \mathsf{X}(\phi_1 \,\mathsf{W}\, \phi_2)).$$

Again, we will use these rules freely in our examples.

5.2 Definition of ELL Success

By now it comes as no surprise that the notion of ELL success is dual to the one for LTL.

DEFINITION 5.2.1. (SUCCESSFUL PATH) A path π in an ELL proof structure Π is *successful* if it is

- finite, ending in an axiom, or
- infinite and for all $\psi \in \mathsf{U}(\phi)$ we have that

  $$\mathsf{inf}(\pi) \cap R_\psi \neq \varnothing$$

  where $R_\psi = \Gamma - Q_\psi$ with Q_ψ as in Definition 3.2.2. $\diamondsuit$

Note that for $\psi = \phi_1 \,\mathsf{U}\, \phi_2$ we have

$$R_\psi = \{\gamma \mid \Phi_\gamma \cap U_\psi = \varnothing \ \vee \ \phi_2 \in \Phi_\gamma\}.$$

In other words, an infinite path is successful if it is not the case that there is an U-formula $\psi = \phi_1 \,\mathsf{U}\, \phi_2$ which is unfolded infinitely often along the path with its "promise" ϕ_2 occurring only finitely many times on that path. Intuitively, such indefinite unfolding of ψ without the promised ϕ_2 also occurring infinitely often would correspond to postponing the fulfillment of the promise forever from some point on, which is in contradiction to the semantics of the until operator.

The system $\mathcal{S}^\Pi$ associated with an ELL proof structure Π is defined in the similar way as in Definition 3.2.7 with the difference that $\Gamma_{\mathsf{A}(\sharp)}$ is everywhere replaced by $\Gamma_{\mathsf{E}(\sharp)}$ for $\sharp \in \{ax, nx, \mathsf{X}\}$ (the sets Γ_{term} and Γ_{sys} are redefined accordingly). Also the notions of an ELL trail and its projections to system runs and paths in the proof structure remain the same up to a replacement of $\Gamma_{\mathsf{A}(\sharp)}$ by $\Gamma_{\mathsf{E}(\sharp)}$ as above. The definition of Π-fairness remains unchanged for ELL trails. The LTL Trail Lemma (Lemma 3.2.11) is then complemented by the following

Lemma 5.2.2. (ELL Trail Lemma) *Let* Π *be an ELL proof structure for* $\mathcal{S}$ *and* $\Xi \vdash \mathsf{E}(\phi)$. *We have:*

(i) all trails of Π *are infinite,*

(ii) for any state $s \models \Xi$ *there* exists *a* s*-run* σ *of* $\mathcal{S}$ *following some path* π *of* Π, *and*

(iii) a trail ϑ *of* Π *is* Π*-fair iff* σ_ϑ *is a computation of* $\mathcal{S}$.

Proof. (i) and (ii): By the side conditions of rules $\mathsf{E}(\mathsf{X})$ and $\mathsf{E}(sp)$. (iii) The same argument as in the LTL case applies also here. $\square$

Note in particular that point (ii) of this Lemma is weaker than the corresponding statement of the LTL Trail Lemma (Lemma 3.2.11), the latter saying that *any* run of $\mathcal{S}$ follows some path in a LTL proof structure. Clearly, this is due to the change of the side condition from a Hoare triple (for Rule $\mathsf{A}(\mathsf{X})$) to a possibility triple (for Rule $\mathsf{E}(\mathsf{X})$).

Definition 5.2.3. (Successful ELL Trail) An ELL trail ϑ is *successful* if π_ϑ is successful. $\diamondsuit$

Definition 5.2.4. (ELL Success Criterion) An ELL proof structure Π for system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{E}(\phi)$ is *successful* if for any state $s \models \Xi$ there is a successful Π-*fair* trail ϑ such that $\vartheta(0)|_X = s$. $\diamondsuit$

We proceed to a syntactic characterisation of the ELL success criterion.

Definition 5.2.5. Let $\Psi_{\mathsf{E}} \stackrel{\text{def}}{=} \mathsf{U}(\phi) \cup \{\bot\}$. The assertions K_ψ for $\psi \in \Psi_{\mathsf{E}}$ are defined by

$$K_\psi \stackrel{\text{def}}{=} \begin{cases} K \in \lceil Q_\psi \rceil & \text{for } \psi \in \mathsf{U}(\phi) \\ K = \lceil \bot \rceil & \text{for } \psi = \bot \end{cases}$$

$\diamondsuit$

Proposition 5.2.6. (ELL Success, Syntactically) *Let* Π *be an ELL proof structure. Then* *(i) a trail ϑ of Π is successful iff it satisfies the* ELL success formula

$$\Omega_{\mathsf{E}} \stackrel{def}{=} \bigwedge_{\psi \in \Psi_{\mathsf{E}}} \mathsf{G}\,\mathsf{F}\,\neg K_{\psi}$$

(ii) Π is successful iff $\mathcal{S}^{\Pi} \models \mathsf{E}_{\Pi}\,\Omega_{\mathsf{E}}$, where E_{Π} quantifies over Π-fair trails.

Proof. (i) Observe that for $\psi \in \mathsf{U}(\phi)$ and a state $t \in \Sigma^{\Pi}$ with $t(K) \in \{\top, \bot\}$ we have $t \models \neg K_{\psi}$, so any trail traversing an axiom and thus ending up with control K caught at $\top$ is successful. It is then easy to see that a trail ϑ is successful if and only if $\vartheta \models \Omega_{\mathsf{E}}$. (ii) Immediate from the definitions. □

5.3 A Proof Rule for ELL Success

For the time being let us disregard fairness conditions and work with saturated systems $\mathcal{S}$ only (fairness issues will be dealt with in Chapter 6). We are looking for an appropriate proof rule to establish that an ELL proof structure Π for a saturated system $\mathcal{S}$ is successful, i.e., $\mathcal{S}^{\Pi} \models \mathsf{E}_{\Pi}\,\Omega_{\mathsf{E}}$. We take a similar approach as for LTL success and first introduce a rule $(\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F}))$ for proving properties of the general form $\mathsf{E}(\bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i)$ over arbitrary saturated systems and then instantiate and modify it slightly yielding Rule E(S) for proving ELL success.

Remark (history variables) For completeness, the particular system under study possibly needs to be augmented with a *history variable*[1] prior the application of Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$ or E(S), respectively (see Section 5.5.4).

5.3.1 *Rule* $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$

We propose Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$ (see Figure 5.1) for arbitrary systems and existentially quantified conjunctions of recurrence properties (that is formulas of the form $\mathsf{E}(\bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i)$ with assertions r_i).

Let us now explain the "mechanics" of Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$. In contrast to Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$, we have a ranking function δ_i mapping program states to a well-founded domain $(W_i, \succ_i)$ for each $1 \leq i \leq m$. Just like Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ the present rule relies on auxiliary assertions α_i.

Condition R1 states that any initial state satisfies α_i for some $1 \leq i \leq m$. Condition R2 requires that from any α_i-state there is some transition leading

[1] A history variable [AL91] is a system variable that records information about the past behaviour of a system without affecting the original state components.

to an $\alpha_{i\oplus 1}$-state also satisfying r_i or preserving α_i and decreasing the ranking δ_i. Intuitively speaking, the ranking δ_i measures the distance to the next $\alpha_{i\oplus 1} \wedge r_i$-state.

> Let $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta)$ be a saturated system and let $r_1, \ldots, r_m$ be assertions. To apply this rule, find for $1 \leq i \leq m$:
>
> (a) a ranking function $\delta_i : \Sigma \to W_i$ mapping states of $\mathcal{S}$ into elements of a well-founded domain $(W_i, \succ_i)$, and
>
> (b) an assertion α_i,
>
> and check the validity of conditions R1 and R2 below, where $\alpha \stackrel{\text{def}}{=} \bigvee_{i=1}^m \alpha_i$ and $a \oplus b \stackrel{\text{def}}{=} ((a + b - 1) \,\mathsf{mod}\, m) + 1$.
>
> $$\begin{array}{ll} \text{R1.} & \Theta \to \alpha \\ \text{R2.} & \{\alpha_i \wedge \delta_i = w\} \ \langle \Lambda \rangle \ \{(\alpha_{i\oplus 1} \wedge r_i) \vee (\alpha_i \wedge \delta_i \prec_i w)\} \\ \hline & \mathcal{S} \vdash \mathsf{E}(\bigwedge_{i=1}^m \mathsf{G}\,\mathsf{F}\, r_i) \end{array}$$

Figure 5.1: Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$

Suppose that for a given proof structure Π we have identified rankings δ_i and assertions α_i satisfying conditions R1 and R2. Then starting from any initial state $s \models \Theta$ of $\mathcal{S}$, we can construct a s-run (= s-computation) of $\mathcal{S}$ satisfying $\bigwedge_{i=1}^m \mathsf{G}\,\mathsf{F}\, r_i$ in the following way. By R1 s satisfies α_i for some $1 \leq i \leq m$. Suppose that we have constructed the run prefix $\sigma_k : s_0 \cdots s_k$ with s_k satisfying α_j. Then by R2 there is a system transition leading to a state s' that either satisfies $\alpha_{i\oplus 1} \wedge r_i$ or still satisfies α_i and decreases the ranking δ_i. Set $s_{k+1} = s'$. By repeated application of this construction step we will eventually obtain a run prefix $\sigma_{k+l} : s_o \cdots s_k \cdots s_{k+l}$ with s_{k+l} satisfying $\alpha_{i\oplus 1} \wedge r_i$ by well-foundedness of the domain W_i. Then we repeat the same procedure for $\alpha_{i\oplus 1}$, constructing a prefix $\sigma_{k+l+n} : s_o \cdots s_k \cdots s_{k+l} \cdots s_{k+l+n}$ with s_{k+l} satisfying $\alpha_{i\oplus 2} \wedge r_{i\oplus 1}$, etc., ad infinitum. This yields a s-run σ of $\mathcal{S}$ satisfying $\bigwedge_{i=1}^m \mathsf{G}\,\mathsf{F}\, r_i$ as required. Thus, we have just proved

PROPOSITION 5.3.1. (SOUNDNESS OF RULE $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$) *Let $\mathcal{S}$ be a saturated system and let $r_1, \ldots, r_m$ be assertions. Then $\mathcal{S} \vdash \mathsf{E}(\bigwedge_{i=1}^m \mathsf{G}\,\mathsf{F}\, r_i)$ implies $\mathcal{S} \models \mathsf{E}(\bigwedge_{i=1}^m \mathsf{G}\,\mathsf{F}\, r_i)$.* $\square$

5.3.2 Rule $\mathsf{E}(S)$

The ELL success rule for saturated systems, called Rule E(S), is displayed in Figure 5.2. It is essentially obtained from Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$ by instantiation, taking $\mathcal{S}^\Pi$ for $\mathcal{S}$ and $\{\neg K_\psi \mid \psi \in \Psi_{\mathsf{E}}\}$ for $\{r_1, \ldots, r_m\}$. However, a slight modification has been made to condition E2 in Rule E(S) compared to its counterpart R2 in Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$: we have added $K_\top$ as a disjunct to the right-hand side of the possibility triple. The justification for this step is simple. If in the construction of a trail witnessing Ω_{E} (see proof of Proposition 5.3.1) we are able to reach $K_\top$-state at some point, then we can safely stop the construction. This is because we *know* that the trail prefix can be completed into a successful trail, so there is no need to prove it each time Rule E(S) is applied.

Let Π be an ELL proof structure, let $\Omega_{\mathsf{E}} = \bigwedge_{\psi \in \Psi_{\mathsf{E}}} \mathsf{G}\,\mathsf{F}\,\neg K_\psi$ as defined in Section 5.2 and suppose $K_1, \ldots, K_m$ enumerates $\{K_\psi \mid \psi \in \Psi_{\mathsf{E}}\}$. Find for $1 \le i \le m$:

(a) a ranking function $\delta_i : \Sigma^\Pi \to W$ mapping states of $\mathcal{S}^\Pi$ into elements of a well-founded domain $(W_i, \succ_i)$, and

(b) an assertion α_i,

such that conditions E1 and E2 below are valid, where $\alpha \overset{\text{def}}{=} \bigvee_{i=1}^m \alpha_i$ and $a \oplus b \overset{\text{def}}{=} ((a + b - 1) \,\mathsf{mod}\, m) + 1$.

$$\begin{array}{ll} \text{E1.} & \Theta^\Pi \to \alpha \\ \text{E2.} & \{\alpha_i \wedge \delta_i = w\} \left\langle \Lambda^\Pi \right\rangle \{K_\top \vee (\alpha_{i\oplus 1} \wedge \neg K_i) \vee (\alpha_i \wedge \delta_i \prec_i w)\} \\ \hline & \mathcal{S}^\Pi \vdash \mathsf{E}\,\Omega_{\mathsf{E}} \end{array}$$

Figure 5.2: Rule E(S) for proving success of ELL proof structures

Definition 5.3.2. Let Π be an ELL proof structure for $\mathcal{S}$ and $\Xi \vdash \mathsf{E}(\phi)$. We say that *Rule* $\mathsf{E}(S)$ *is applicable to* Π if $\mathcal{S}^\Pi \vdash \mathsf{E}\,\Omega_{\mathsf{E}}$, that is, there exist rankings δ_i and assertions α_i such that conditions E1 and E2 are valid. $\diamondsuit$

Definition 5.3.3. (ELL Proofs) Let Π be an ELL proof structure for $\mathcal{S}$ and $\Xi \vdash \mathsf{E}(\phi)$.

- Π is a *proof of* $\mathcal{S}, \Xi \models \mathsf{E}\,\phi$, written $\Pi : \mathcal{S}, \Xi \Vdash \mathsf{E}\,\phi$, if it is successful, and
- Π is a *S-proof of* $\mathcal{S}, \Xi \models \mathsf{E}\,\phi$, written $\Pi : \mathcal{S}, \Xi \vdash \mathsf{E}\,\phi$, if Rule E(S) is applicable to Π.

We say that $\mathcal{S}, \Xi \models \mathsf{E}\,\phi$ is *provable (S-provable)*, written $\mathcal{S}, \Xi \Vdash \mathsf{E}\,\phi$ ($\mathcal{S}, \Xi \vdash \mathsf{E}\,\phi$), if there exists a proof structure Π for $\mathcal{S}$ and $\Xi \vdash \mathsf{E}(\phi)$ such that $\Pi : \mathcal{S}, \Xi \Vdash \mathsf{E}\,\phi$ ($\Pi : \mathcal{S}, \Xi \vdash \mathsf{E}\,\phi$). $\diamondsuit$

Soundness of Rule E(S) follows from soundness of Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$ (Proposition 5.3.1) and from the discussion above:

Proposition 5.3.4. (Soundness of Rule E(S) for Proving ELL Success) *Let* Π *be an ELL proof structure for a* saturated *system* $\mathcal{S}$ *and sequent* $\Xi \vdash \mathsf{E}(\phi)$*. Then* $\Pi : \mathcal{S}, \Xi \vdash \mathsf{E}\,\phi$ *implies* $\Pi : \mathcal{S}, \Xi \Vdash \mathsf{E}\,\phi$ *is successful.* $\square$

Consider now the case where there is no U-subformula in ϕ. Then ϕ describes a safety property and we would expect that we do not need a well-foundedness argument to establish success. Indeed, the success formula for this case reads $\Omega_{\mathsf{E}} = \mathsf{G}\,\mathsf{F}\,\neg K_{\perp}$ and is satisfied by any trail not traversing an anti-axiom. In particular, any infinite path is successful. Since in the absence of fairness any run is a computation, the following proposition is a consequence of Lemma 5.2.2(ii):

Proposition 5.3.5. *Let* Π *be an ELL proof structure for a* saturated *system* $\mathcal{S}$ *and sequent* $\Xi \vdash \mathsf{E}(\phi)$ *such that* ϕ *does not contain any* U*-subformula and there is no anti-axiom in* Π*. Then* $\Pi : \mathcal{S}, \Xi \Vdash \mathsf{E}\,\phi$. $\square$

This should be compared with the corresponding result for LTL (Proposition 3.3.6), where no restriction to saturated systems was necessary. Here, we can in general only guarantee the existence of witnessing Ξ-*runs.* For systems with fairness constraints we still have to apply Rule E(S) in order to show the existence of the required *fair* Ξ-runs.

5.4 Example

Example 5.4.1. System $\mathcal{S}_5$ has a single natural number variable x and the following two transition relations:

$$\begin{aligned} \rho_{inc} &\stackrel{\text{def}}{=} x' = x + 1 \\ \rho_{zero} &\stackrel{\text{def}}{=} \exists y.\, y^2 = x \,\wedge\, x' = 0 \end{aligned}$$

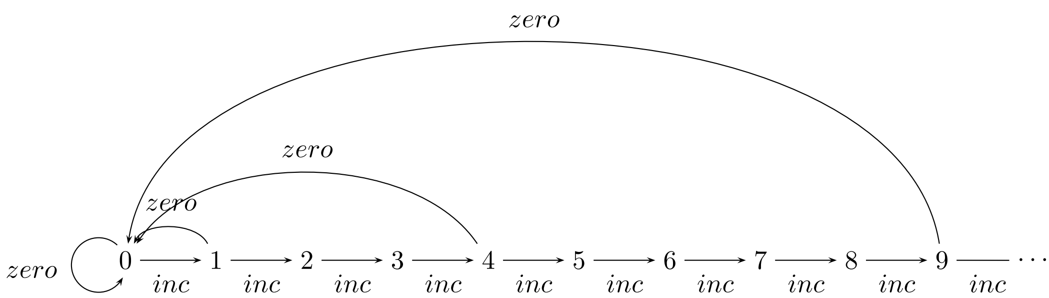

Figure 5.3: LTS induced by system $\mathcal{S}_5$.

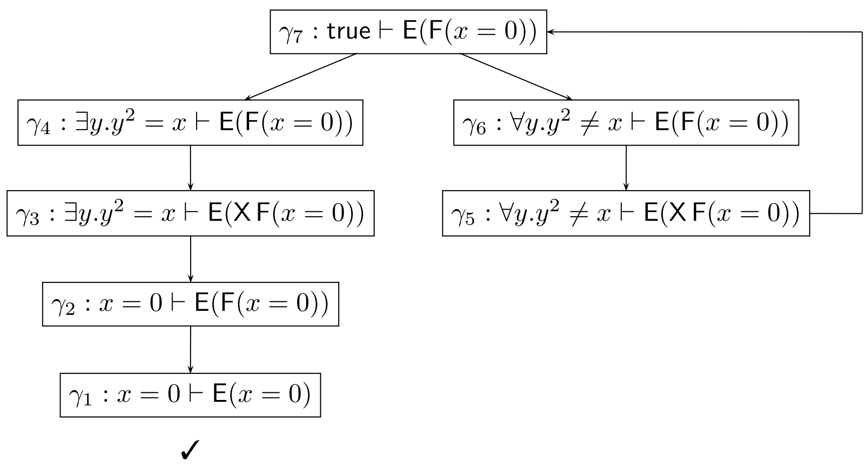

Figure 5.4: Proof structure Π_5 for system $\mathcal{S}_5$ and sequent $\mathsf{true} \vdash \mathsf{E}\,\mathsf{F}(x = 0)$.

The LTS induced by this system is shown in Figure 5.3. The statement that we want to verify for this system is $\mathcal{S}_5, \mathsf{true} \models \mathsf{E}\,\mathsf{F}(x = 0)$, that is, from any state it is possible to eventually reach the state where $x = 0$.

A possible proof structure Π_5 for this purpose is shown in Figure 5.3. Note that at the root sequent γ_7 we have applied rule $\mathsf{E}(sp)$, splitting cases according to whether or not x is a square number. At sequent γ_2 we use rule $\mathsf{E}(\mathsf{F}_l)$ and at γ_4 and γ_6 we use rule $\mathsf{E}(\mathsf{F}_r)$ (see Section 5.1.2). Let us now show that this proof structure is successful using Rule E(S).

Setting $\gamma_0 = \top$, suppose that the coding $\lceil\cdot\rceil$ is defined by $\lceil\gamma_i\rceil = i$ for $0 \le i \le 7$. Since there is only one U-subformula ($\mathsf{F}(x = 0)$), we write just F instead of $\mathsf{F}(x = 0)$ in α_F and K_F. In order to apply Rule E(S) we define the $\alpha_\perp$ and α_F by

$$\alpha_\perp \stackrel{\text{def}}{=} \mathsf{false} \quad \text{and} \quad \alpha_\mathsf{F} \stackrel{\text{def}}{=} \bigvee_{\gamma\in\Gamma^+} \widehat{p_\gamma}$$

and choose the single ranking

$$\delta(x, K) = (d(x), K)$$

for $\psi \in \{\perp, \mathsf{F}\}$ with the lexicographic ordering $\prec$ on $\mathbb{N}^2$, where

$$d(x) \stackrel{\text{def}}{=} \begin{cases} q(x) - x + 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$q(x) \stackrel{\text{def}}{=} \min\{m \in \mathbb{N} \mid \exists z.\, z^2 = m \wedge m \ge x\}$$

Thus, $d(x)$ measures the distance of from x to the smallest square number greater or equal to x.

Condition E1 holds trivially. Condition E2 for $\psi = \perp$ is also trivial. For $\psi = \mathsf{F}$, E2 boils down to showing

$$\{\widehat{p_\gamma} \wedge (\delta = (u, v))\}\left\langle\Delta^{\Pi_5}\right\rangle\{K_\top \vee (\alpha_\mathsf{F} \wedge (\delta \prec (u, v)))\}.$$

for each $\gamma \in \Gamma^+$. This means that every witnessing trail will have to reach $K_\top$ at some point. The preservation of α_F being immediate this reduces to

$$\{\widehat{p_\gamma} \wedge (\delta = (u, v))\}\left\langle\Delta^{\Pi_5}\right\rangle\{K_\top \vee (\delta \prec (u, v))\}.$$

For γ_0 and γ_1 we can obviously reach $K_\top$ in one step. In the other cases, we have to show that the ranking decreases. For $\gamma \in \{\gamma_2, \gamma_4, \gamma_6, \gamma_7\}$, it is a transition from $\Lambda^{\Pi_5}_{log}$ which makes the second component of the rank decrease while preserving the first (for γ_7 this may be either $(\gamma_7, =, \gamma_4)$ or $(\gamma_7, =, \gamma_6)$, depending on whether x is square or not). From γ_3 the transition $(\gamma_3, zero, \gamma_4)$ decreases both components of the rank if $x > 0$ and only the second if $x = 0$. For γ_5, the transition $(\gamma_5, inc, \gamma_7)$ decreases the distance $d(x)$, hence the ranking δ. ♣

5.5 Soundness and Completeness

5.5.1 Admissible Trails and Winning Strategies

DEFINITION 5.5.1. Let $\Pi = (\Gamma, \Delta, \gamma_r)$ be an ELL proof structure. The E-*generation relation* $\rightsquigarrow_{\gamma,\gamma'} \subseteq \Phi_\gamma \times \Phi_{\gamma'}$ is defined for each edge $(\gamma, \gamma') \in \Delta$ in a ELL proof structure by case analysis on the rule applied at γ:

$$\begin{array}{lcl}
\rightsquigarrow_{p \vdash \mathsf{E}(\Phi,q),\, p \vdash \mathsf{E}(\Phi)} & = & \mathsf{Id}(\Phi) \\
\rightsquigarrow_{p \vdash \mathsf{E}(\Phi,\phi_1 \vee \phi_2),\, p \vdash \mathsf{E}(\Phi,\phi_1)} & = & \{(\phi_1 \vee \phi_2, \phi_1)\} \cup \mathsf{Id}(\Phi) \\
\rightsquigarrow_{p \vdash \mathsf{E}(\Phi,\phi_1 \vee \phi_2),\, p \vdash \mathsf{E}(\Phi,\phi_2)} & = & \{(\phi_1 \vee \phi_2, \phi_2)\} \cup \mathsf{Id}(\Phi) \\
\rightsquigarrow_{p \vdash \mathsf{E}(\Phi,\phi_1 \wedge \phi_2),\, p \vdash \mathsf{E}(\Phi,\phi_1,\phi_2)} & = & \{(\phi_1 \wedge \phi_2, \phi_1),\ (\phi_1 \wedge \phi_2, \phi_2)\} \cup \mathsf{Id}(\Phi) \\
\rightsquigarrow_{p \vdash \mathsf{E}(\Phi,\phi_1 \,\mathsf{Z}\, \phi_2),\, p \vdash \mathsf{E}(\Phi,\mathsf{unf}(\phi_1 \,\mathsf{Z}\, \phi_2))} & = & \{(\phi_1 \,\mathsf{Z}\, \phi_2, \mathsf{unf}(\phi_1 \,\mathsf{Z}\, \phi_2))\} \cup \mathsf{Id}(\Phi) \\
\rightsquigarrow_{p \vdash \mathsf{E}(\mathsf{X}\,\Phi),\, q \vdash \mathsf{E}(\Phi)} & = & \{(\mathsf{X}\,\phi, \phi) \mid \phi \in \Phi\} \\
\rightsquigarrow_{p \vdash \mathsf{E}(\Phi),\, q \vdash \mathsf{E}(\Phi)} & = & \mathsf{Id}(\Phi)
\end{array}$$

where $\mathsf{Id}(\Phi) = \{(\phi, \phi) \mid \phi \in \Phi\}$ is the identity relation on Φ. ♢

The following notions are then defined in a similar way as their LTL counterparts:

- given a path π in an ELL proof structure an E-*generative path running along* π is determined as in Definition 4.2.2 but using the E-generation relation; the set of all E-generative paths running along π is denoted by $I_{\mathsf{E}}(\pi)$ and its subset of finite E-generative paths by $I^*_{\mathsf{E}}(\pi)$; the latter set is called *internal pre-strategy of* π
- given a trail ϑ of an ELL proof structure Π, the definitions of the i*nternal (pseudo-) strategy* T_ϑ *and* τ_ϑ *of* ϑ are the same as in Definition 4.2.4 except that $I^*(\pi_\vartheta)$ is replaced by $I^*_{\mathsf{E}}(\pi_\vartheta)$.

DEFINITION 5.5.2. (ADMISSIBILITY) Let Π be an ELL proof structure for system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{E}\,\phi$.

1. A *path* π in Π is *admissible* if

   - finite, ending in an axiom, or

- infinite and for all $\iota \in I_{\mathsf{E}}(\pi)$ we have that

$$\mathsf{inf}(\iota) \cap \mathsf{U}(\phi) = \varnothing$$

2. A *trail* ϑ of Π is *admissible* if π_ϑ is an admissible path.

3. Π is *admissible* if for all states s of $\mathcal{S}$ with $s \models \Xi$ there is a admissible Π-fair trail ϑ such that $\vartheta(0)|_X = s$. ♢

By duality an admissible path (trail) is also successful, but the converse does not hold for infinite paths in general. It comes as no surprise that we have

PROPOSITION 5.5.3. (TRAILS AND $\exists$-STRATEGIES) *Let* Π *be a proof structure for* $\mathcal{S}$ *and* $\Xi \vdash \mathsf{E}\,\phi$, *and let* ϑ *be a trail of* Π. *Then*

(i) τ_ϑ *is a deterministic* $\exists$*-strategy for* $\mathcal{G}_{\mathcal{S}}(\sigma_\vartheta, \phi)$,

(ii) τ_ϑ *is complete if and only if* π_ϑ *is closed, and*

(iii) τ_ϑ *is winning if and only if* ϑ *is admissible.*

PROOF. By duality. □

By this proposition and Lemma 5.2.2 we obtain the soundness result in

PROPOSITION 5.5.4. *Let* Π *be an ELL proof structure for* $\mathcal{S}$ *and* $\Xi \vdash \mathsf{E}\,\phi$. *If* Π *is admissible then Player* $\exists$ *wins* $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{E}\,\phi)$. □

In contrast to the LTL case, the converse direction does not hold in general, even in the absence of fairness: although we know by the ELL Trail Lemma that for each initial state of the system there is a run following *some* path, we cannot guarantee that this path is admissible. In fact, there need not be an admissible path at all in an ELL proof structure as the following simple example shows.

EXAMPLE 5.5.5. System $\mathcal{S}_6$ has a single natural number variable x and the two transition relations

$$\begin{aligned} \rho_{inc} &\stackrel{\text{def}}{=} x' = x + 1 \\ \rho_{zero} &\stackrel{\text{def}}{=} x' = 0 \end{aligned}$$

with initial condition $\Theta_6 \stackrel{\text{def}}{=} \mathsf{true}$. Figure 5.5 shows a possible ELL proof structure for this system and property $\phi_6 \stackrel{\text{def}}{=} \mathsf{E}\,\mathsf{F}(x = 0)$, which is clearly satisfied by $\mathcal{S}_6$.

The only path in Π_6 is inadmissible (and also unsuccessful). Obviously, the mistake was that we have chosen to apply Rule $\mathsf{E}(\vee_r)$ at γ_2 instead of using $\mathsf{E}(\vee)$ to split off the case where $x = 0$. ♢

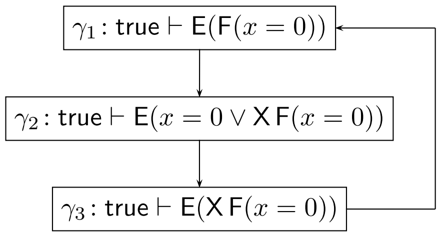

Figure 5.5: Proof structure Π_6 for $\mathcal{S}_6$ and $\phi_6 \stackrel{\text{def}}{=} \mathsf{E}\,\mathsf{F}(x = 0)$

5.5.2 Existence of an Admissible ELL Proof Structure

The construction of an admissible proof structure requires a judicious application of the Split Rule in combination with the two rules for disjunction (or the derived rule $\mathsf{E}(\vee)$). Therefore, we construct in (the proof of) the following lemma a "canonical" proof structure Π_K and show that it is admissible.

LEMMA 5.5.6. (EXISTENCE OF ADMISSIBLE ELL PROOF STRUCTURE) *Let Ξ be a satisfiable assertion and suppose Player $\exists$ wins the game $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{E}\,\phi)$. Then*

(i) there exists a proof structure Π_K for $\mathcal{S}$ and $\Xi \vdash \mathsf{E}(\phi)$,

(ii) any deterministic, history-free winning $\exists$-strategy $\tau^{\exists}$ for $\mathcal{G}_{\mathcal{S}}(\sigma, \phi)$ with $\sigma \in \mathcal{R}_{\mathcal{S}}(\Xi)$ is completely represented in Π_K

(iii) Π_K is admissible.

PROOF. (i) The construction is similar to the one for LTL proof structures in Lemma 4.3.2. We construct a finite sequence of pre-proof structures $\Pi_0, \ldots, \Pi_n$ such that Π_n is a proof structure for $\mathcal{S}$ and $\Xi \vdash \mathsf{E}(\phi)$. The initial pre-proof structure Π_0 consists of the root sequent $\Xi \vdash \mathsf{E}(\phi)$. Pre-proof structure Π_{i+1} is constructed from Π_i by applying some rule(s) to an open node (a non-terminal sequent with no successors) while maintaining the invariant

$$p \models \mathsf{E}(\bigwedge_{\phi \in \Phi} \phi) \text{ and } p \text{ satisfiable.}$$

The initial pre-proof structure Π_0 satisfies the invariant by assumption. In order to construct Π_{i+1} from Π_i, suppose first there is an open node $\gamma : p \vdash \mathsf{E}(\mathsf{X}\,\Phi)$ in Π_i where rule $\mathsf{E}(\mathsf{X})$ is applicable. We apply the rule in the following way:

$$\frac{\gamma : p \vdash \mathsf{E}(\mathsf{X}\,\Phi)}{\gamma' : \chi_{\mathsf{E}(\Phi)} \vdash \mathsf{E}(\Phi)}$$

where $\chi_{\mathsf{E}(\Phi)}$ is an assertion characterising the set of states where $\mathsf{E}(\Phi)$ holds (see Lemma 4.3.1). The side condition is satisfied and our invariant is preserved.

Now suppose there is an open node $\gamma : p \vdash (\Phi, \psi)$ in Π_i where rule $\mathsf{E}(\mathsf{X})$ is not applicable and ψ is a formula other than a X-formula. We proceed by case analysis on the top-level operator of ψ. The only interesting case is disjunction, all other cases are straightforward.

Suppose that $\psi = \phi_1 \vee \phi_2$. From the invariant it follows that for each $s \models p$ we have $s \models \mathsf{E}(\Phi, \phi_1)$ or $s \models \mathsf{E}(\Phi, \phi_2)$. We distinguish three cases:

(a) $p \models \neg\, \mathsf{E}(\Phi, \phi_2)$: it follows that $p \models \mathsf{E}(\Phi, \phi_1)$ and we apply rule $\mathsf{E}(\vee_l)$

$$\mathsf{E}(\vee_l)\ \frac{p \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)}{p \vdash \mathsf{E}(\Phi, \phi_1)}$$

clearly preserving the invariant.

(b) $p \models \neg\, \mathsf{E}(\Phi, \phi_1)$: this case is symmetrical to the previous one.

(c) Otherwise, there exist p-states s_1 and s_2 such that $s_1 \models \mathsf{E}(\Phi, \phi_1)$ and $s_2 \models \mathsf{E}(\Phi, \phi_2)$. We proceed by applying a combination of the Split and disjunction rules in the following way:

$$\mathsf{E}(sp)\ \frac{\gamma : p \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)}{\mathsf{E}(\vee_l)\ \dfrac{\gamma_1 : \chi_1 \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)}{\gamma_1' : \chi_1 \vdash \mathsf{E}(\Phi, \phi_1)} \qquad \mathsf{E}(\vee_r)\ \dfrac{\gamma_2 : \chi_2 \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)}{\gamma_2' : \chi_2 \vdash \mathsf{E}(\Phi, \phi_2)}}$$

where χ_1 and χ_2 are formulas equivalent to the characteristic predicates $\chi_{\mathsf{E}(\Phi,\phi_1)}$ and $\chi_{\mathsf{E}(\Phi,\phi_2)}$, respectively, chosen in a way as to avoid looping back to a sequent where the Split rule $\mathsf{E}(sp)$ is already applied. This ensures that condition (E-SPL) is satisfied. It is easy to see that the invariant holds for all four new sequents γ_1, γ_1', γ_2 and γ_2'.

Finally, by a similar argument as in (the proof of) Lemma 4.3.2 we can show that this construction terminates, thus yielding an ELL proof structure which we call Π_K. Observe that by the invariant Π_K can not contain any anti-axioms. We remark that case (c) above is the only place where rule $\mathsf{E}(sp)$ is applied in the construction of Π_K.

(ii) Let $\sigma \in \mathcal{R}_{\mathcal{S}}(\Xi)$ and suppose τ is a deterministic, history-free winning strategy for $\mathcal{G}_{\mathcal{S}}(\sigma, \phi)$. We show that τ is completely represented in Π_K. In a similar way as in Lemma 4.2.12, we construct two infinite sequences $\{\sigma_i\}_{i\in\omega}$ and $\{\gamma_i\}_{i\in\omega}$ of suffixes of σ and elements of Γ^+, respectively, while maintaining the following invariants:

J1. $\sigma_i(0) \models p_{\gamma_i}$, and

J2. for all $\iota \in I_{\mathsf{E}}(\pi_i)$ such that $|\iota| = i+1$ we have $\natural(\zeta_i * \iota) \in \tau$.

where $\pi_i = \gamma_0 \cdots \gamma_i$ and $\zeta_i = \sigma_0 \cdots \sigma_i$. Clearly both invariants hold for the initial choice $\sigma_0 = \sigma$ and $\gamma_0 = \Xi \vdash \mathsf{E}(\phi)$. Continuing the construction we determine σ_{k+1} and γ_{k+1} by case analysis on the rule R applied at γ_k. If γ_k is a terminal then it must be an axiom by construction of Π_K and we define $\sigma_j = (\sigma_k)^{k-j}$ and $\gamma_j = \top$ for $j > k$. From the remaining cases we pick $R \in \{\mathsf{E}(\vee_l), \mathsf{E}(\vee_r), \mathsf{E}(sp)\}$.

Suppose that rule $\mathsf{E}(\vee_l)$ is applied at $\gamma_k = p \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)$ and that it is not preceded by an application of $\mathsf{E}(sp)$ at γ_{k-1} (if $k > 0$). We set $\sigma_{k+1} = \sigma_k$ and $\gamma_{k+1} = p \vdash \mathsf{E}(\Phi, \phi_1)$. Since for any $\theta \in \Phi \cup \{\phi_1 \vee \phi_2\}$ there is a $\iota_\theta \in I_{\mathsf{E}}(\pi_k)$ of length $k+1$ such that $\iota_\theta \colon \phi \cdots \theta$ (see also Lemma 4.2.5), there is by (J2) a position $n_\theta = m_\theta \cdot (\sigma_k, \theta) = \natural(\zeta_k * \iota_\theta) \in \tau$. With τ being a winning strategy we must have $\sigma_k \models \theta$. By construction of Π_K we know that $p \models \neg\, \mathsf{E}(\Phi, \phi_2)$. As σ_k is a p-computation by (J1) we deduce that $\sigma_k \not\models \phi_2$. Since τ is winning it must move from any position of the form $m \cdot (\sigma_k, \phi_1 \vee \phi_2) = \natural(\zeta_k * \iota)$ (with $\iota \in I_{\mathsf{E}}(\pi_k)$ of length $k+1$) to position $m \cdot (\sigma_k, \phi_1 \vee \phi_2) \cdot (\sigma_k, \phi_1)$. Hence, (J2) is preserved. (J1) is trivially preserved. The case where $\mathsf{E}(\vee_l)$ is applied to γ_k without a preceding application of $\mathsf{E}(sp)$ is symmetrical.

Suppose that $\mathsf{E}(sp)$ is applied to γ_k. By construction of Π_K we know that $\gamma_k = p \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)$. Pick $\iota \in I(\pi_k)$ such that $|\iota| = k+1$ and $\iota = \phi \cdots (\phi_1 \vee \phi_2)$. From (J2) it follows that $n = \natural(\zeta_k * \iota) = m \cdot (\sigma_k, \phi_1 \vee \phi_2) \in \tau$ and since τ is deterministic and complete we have either $n \cdot (\sigma_k, \phi_1) \in \tau$ or $n \cdot (\sigma_k, \phi_2) \in \tau$. Suppose w.l.o.g. that $n \cdot (\sigma_k, \phi_1) \in \tau$. Since there are also positions $p \cdot (\sigma_k, \theta) \in \tau$ for all $\theta \in \Phi$ by (J2) and τ is winning we have $\sigma_k \models \theta$ as well as $\sigma_k \models \phi_1$, hence $\sigma_k(0) \models \chi_1$ (recall χ_1 is equivalent to $\chi_{\mathsf{E}(\Phi,\phi_1)}$). We set $\sigma_{k+1} = \sigma_k$ and $\gamma_{k+1} = \chi_1 \vdash \mathsf{E}(\Phi, \phi_1 \vee \phi_2)$ which appears as one of the successor sequents of γ_k by construction of Π_K. Hence (J1) is preserved. (J2) is trivially preserved. Again by construction of Π_K we know that the only successor sequent of γ_{k+1} is $\chi_1 \vdash \mathsf{E}(\Phi, \phi_1)$ which we choose as our γ_{k+2}. Set $\sigma_{k+2} = \sigma_{k+1}$. Invariant (J1) is trivially preserved, while (J2) is preserved because τ is *history-free* and the choice of the new configuration (σ_k, ϕ_1) at position $n = \natural(\zeta_k * \iota) = m \cdot (\sigma_k, \phi_1 \vee \phi_2)$ of τ is therefore independent of the particular ι picked above.

Having completed the construction of the sequences $\{\sigma_i\}_{i \in \omega}$ and $\{\gamma_i\}_{i \in \omega}$, we define ϑ for $j \in \omega$ by $\vartheta(j) = \sigma_j(0)[K \mapsto \gamma_j]$. It is not difficult to see that ϑ is a trail with $\sigma_\vartheta = \sigma$ and $\widetilde{\sigma_\vartheta}(i) = \sigma_i$. Since any position $m \in \tau$ can be represented as $\natural(\zeta_i * \iota)$ for some $i \geq 0$ and $\iota \in I_{\mathsf{E}}(\pi_i)$, we have $\tau_\vartheta \subseteq \tau$ by invariant (J2). Because all terminals of Π_K are axioms π_ϑ is closed, hence τ_ϑ

complete and therefore $\tau_\vartheta = \tau$.

(iii) Let $s \models \Xi$. By assumption Player $\exists$ wins the game $\mathcal{G}_\mathcal{S}(\sigma, \phi)$ for some $\sigma \in \mathcal{C}_\mathcal{S}(s)$. Hence, he has a deterministic, history-free winning strategy τ for this game (Proposition 4.1.2). By point (ii) there is a trail ϑ with $\sigma_\vartheta = \sigma$ and $\tau_\vartheta = \tau$, which is Π-fair by Lemma 5.2.2(iii). Thus, $\vartheta(0)|_X = s$ and ϑ is admissible by Proposition 5.5.3. Hence, Π_K is admissible. $\square$

5.5.3 Winningness and Successful Proof Structures

Before we state our main theorem on winningness and the existence of successful or admissible ELL proof structures, we have the following lemma, comparing admissibility and success.

LEMMA 5.5.7. (ADMISSIBILITY AND SUCCESS) *Let Π be an ELL proof structure for system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{E}(\phi)$. Suppose $s \models \Xi$ and there exists a successful, but inadmissible trail ϑ of Π with $\vartheta(0)|_X = s$. Then Player $\exists$ wins the game $\mathcal{G}_\mathcal{S}(\sigma_\vartheta, \phi)$.*

PROOF. Suppose ϑ is a successful, but inadmissible trail of Π. It follows that π_ϑ must be infinite. By observing that all finite plays in τ_ϑ are won by Player $\exists$ we can apply exactly the same procedure as in the proof of Lemma 4.2.18 (but using U-formulas instead of V-formulas) to construct from τ_ϑ a winning $\exists$-strategy for the game $\mathcal{G}_\mathcal{S}(\sigma_\vartheta, \phi)$. $\square$

THEOREM 5.5.8. (WINNINGNESS AND EXISTENCE OF ADMISSIBLE OR SUCCESSFUL ELL P.S.) *Let $\mathcal{S}$ be a system. Are equivalent:*

(i) Player $\exists$ wins the game $\mathcal{G}_\mathcal{S}(\Xi, \mathsf{E}\,\phi)$,

(ii) there exists an admissible proof structure Π for $\mathcal{S}$ and $\Xi \vdash \mathsf{E}(\phi)$, and

(iii) there exists a successful proof structure Π for $\mathcal{S}$ and $\Xi \vdash \mathsf{E}(\phi)$, that is, $\mathcal{S}, \Xi \Vdash \mathsf{E}\,\phi$.

PROOF. (i)$\Rightarrow$(ii): By Lemma 5.5.6. (ii)$\Rightarrow$(iii): Any admissible proof structure is also successful. (iii)$\Rightarrow$(i): Suppose Π is a successful proof structure for $\mathcal{S}$ and $\Xi \vdash \mathsf{E}(\phi)$ and let $s \models \Xi$. Then there is a successful Π-fair trail ϑ in Π with $\vartheta(0)|_X = s$. Then Player $\exists$ wins $\mathcal{G}_\mathcal{S}(\sigma_\vartheta, \phi)$ by Proposition 5.5.3 (i) and (iii), if ϑ is admissible, and by Lemma 5.5.7, otherwise. Hence, Player $\exists$ wins the game $\mathcal{G}_\mathcal{S}(\Xi, \mathsf{E}\,\phi)$. $\square$

5.5.4 Soundness and Completeness of Rule $\mathsf{E}(S)$

Rule E(S) is sound and relatively complete for showing success of proof structures for saturated systems as is stated in

THEOREM 5.5.9. (SOUNDNESS AND RELATIVE COMPLETENESS OF RULE E(S)) *Let $\mathcal{S}$ be a* saturated *system and let Π be a proof structure for $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{E}(\phi)$. Then $\Pi\colon \mathcal{S}, \Xi \Vdash \mathsf{E}\,\phi$ if and only if $\Pi\colon \mathcal{S}, \Xi \vdash \mathsf{E}(\phi)$.*

PROOF. Soundness of Rule E(S) was shown in Proposition 5.3.4. The proof of relative completeness is deferred to Chapter 6 (it follows from Lemma 6.3.7(ii)). $\square$

5.5.5 Main Result

THEOREM 5.5.10. (SOUNDNESS AND RELATIVE COMPLETENESS FOR ELL) *Let $\mathcal{S}$ be a* saturated *system, Ξ an assertion and $\mathsf{E}\,\phi$ an ELL formula. Then*

$$\mathcal{S}, \Xi \models \mathsf{E}\,\phi \quad \textit{if and only if} \quad \mathcal{S}, \Xi \vdash \mathsf{E}\,\phi$$

PROOF. By Corollary 4.1.4 and Theorems 5.5.8 and 5.5.9. $\square$

Note that the only reason that this theorem is restricted to saturated systems is that the same restriction appears in Theorem 5.5.9. This restriction will be lifted in Chapter 6 on proving success under fairness constraints.

5.6 A Proof System for Full CTL*

Extending our proof systems for LTL and ELL to full CTL* is now straightforward. Let $\mathcal{S}$ be a system, Ξ an assertion and ϕ a ground-quantified CTL* formula. We would like to verify $\mathcal{S}, \Xi \models \phi$. As a consequence of Proposition 2.4.3(iii) we know that $\mathcal{S}, \Xi \models \phi$ if and only if $\mathcal{S}, \Xi \models \mathsf{A}\,\phi$, so we may assume w.l.o.g. that ϕ has a top-level path quantifier. Therefore, there is no need for an intermediate "gluing" proof system as presented in [Kic96] (for the finite-state case) to resolve the top-level boolean combinations of state formulas.

In contrast to LTL and ELL formulas, the CTL* formula ϕ may have arbitrary path-quantified subformulas (of the form $\mathsf{Q}\,\psi$ for $\mathsf{Q} \in \{\mathsf{A}, \mathsf{E}\}$). Let us call ψ a *basic state formula* if it is either a literal or a path-quantified formula. As already observed by Emerson and Lei in [EL85], with respect to the LTL and ELL rules, path-quantified formulas can fortunately be treated very much like assertions. More precisely, we extend the axiom, anti-axiom and assertion rules ($\mathsf{Q}(ax)$, $\mathsf{Q}(nx)$ and $\mathsf{Q}(bsf)$) of the LTL and ELL proof systems to apply to all basic state formulas (see Table 5.3). The definitions of LTL and ELL proof structures are adapted to use the modified sets of LTL and ELL rules, but remain otherwise unchanged. The notion of a *proof* remains unchanged.

$\mathsf{A}(ax)$	$\dfrac{p \vdash \mathsf{A}(\Phi, \psi)}{.}$	$p \vdash \psi$	$\mathsf{E}(ax)$	$\dfrac{p \vdash \mathsf{E}(\psi)}{.}$	$p \vdash \psi$
$\mathsf{A}(nx)$	$\dfrac{p \vdash \mathsf{A}(\psi)}{.}$	$p \vdash \neg\psi$	$\mathsf{E}(nx)$	$\dfrac{p \vdash \mathsf{E}(\Phi, \psi)}{.}$	$p \vdash \neg\psi$
$\mathsf{A}(bsf)$	$\dfrac{p \vdash \mathsf{A}(\Phi, \psi)}{p \vdash \mathsf{A}(\Phi)}$	$p \vdash \neg\psi$	$\mathsf{E}(bsf)$	$\dfrac{p \vdash \mathsf{E}(\Phi, \psi)}{p \vdash \mathsf{E}(\Phi)}$	$p \vdash \psi$

Table 5.3: modified LTL and ELL rules; ψ is a basic state formula; $p \vdash q$ holds for an assertion q if $p \models q$.

The side conditions of rules $\mathsf{Q}(ax)$, $\mathsf{Q}(nx)$ and $\mathsf{Q}(bsf)$, we replace $p \models q$ and $p \models \neg q$ by $p \vdash \psi$ and $p \vdash \neg\psi$, respectively, where ψ is now a basic state formula. Recall that negation is a meta-level operator on all formulas but assertions, so the side conditions for path-quantified formulas have all the form $p \vdash \mathsf{Q}\,\theta$ and require the construction of a new LTL or ELL proof of $\mathcal{S}, p \models \mathsf{Q}\,\theta$ (the case depending on Q). For assertions the statement $p \vdash (\neg)q$ is to be interpreted as $p \models (\neg)q$ as hitherto.

As the type (LTL or ELL) of proof structure can be inferred from the top-level path quantifier of its root sequent, we will often just say "proof structure for system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{Q}\,\phi$".

DEFINITION 5.6.1. (CTL* PROOF STRUCTURES) Given a system $\mathcal{S}$, an assertion Ξ and a CTL* formula $\mathsf{Q}\,\phi$, a *CTL* proof structure* $\overline{\Pi}$ *for system* $\mathcal{S}$ *and sequent* $\Xi \vdash \mathsf{Q}\,\phi$ is a tuple $\overline{\Pi} = (\Pi_1, \ldots, \Pi_n)$ of LTL or ELL proof structures such that

- Π_i is a proof structure for $\mathcal{S}$ and $\Xi_i \vdash \mathsf{Q}_i\,\psi_i$, with $\Xi_1 \vdash \mathsf{Q}_1\,\psi_1 = \Xi \vdash \mathsf{Q}\,\phi$, and

- for all $1 < j \leq n$ there is a $1 \leq i < j$ such that $\Xi_j \vdash \mathsf{Q}_j\,\psi_j$ appears as a side condition of Π_i. $\diamondsuit$

DEFINITION 5.6.2. (CTL* PROOFS) Let $\overline{\Pi} = (\Pi_1, \ldots, \Pi_n)$ be a CTL* proof structure for system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{Q}\,\phi$. Then $\overline{\Pi}$ is a *proof (S-proof)* of $\mathcal{S}, \Xi \models \mathsf{Q}\,\phi$, written $\overline{\Pi} \colon \mathcal{S}, \Xi \Vdash \mathsf{Q}\,\phi$ ($\overline{\Pi} \colon \mathcal{S}, \Xi \vdash \mathsf{Q}\,\phi$), if all constituent proof structures Π_i are proofs (S-proofs).

We also say that the statement $\mathcal{S}, \Xi \models \mathsf{Q}\,\phi$ is *provable (S-provable)* and write $\mathcal{S}, \Xi \Vdash \mathsf{Q}\,\phi$ ($\mathcal{S}, \Xi \vdash \mathsf{Q}\,\phi$) if there is a CTL* proof structure $\overline{\Pi}$ such that $\overline{\Pi} \colon \mathcal{S}, \Xi \Vdash \mathsf{Q}\,\phi$ ($\overline{\Pi} \colon \mathcal{S}, \Xi \vdash \mathsf{Q}\,\phi$). ♢

5.6.1 Soundness and Completeness for CTL*

Let $\overline{\Pi} = (\Pi_1, \ldots, \Pi_n)$ be a CTL* proof structure. The matching of side conditions and root sequents appearing in the LTL and ELL proof structures Π_i constituting $\overline{\Pi}$ induces an acyclic dependency graph on the Π_i ($\overline{\Pi}$ itself is a linearisation of this graph). The quantifier depth of the temporal formula in the root sequent of the respective proof structure strictly decreases along each path in that graph. This observation forms the basis for

THEOREM 5.6.3. (SOUNDNESS AND RELATIVE COMPLETENESS FOR CTL*) *Let $\mathcal{S}$ be a saturated system, Ξ an assertion and $\mathsf{Q}\,\phi$ a CTL* formula. Then*

$$\mathcal{S}, \Xi \models \mathsf{Q}\,\phi \text{ if and only if } \mathcal{S}, \Xi \vdash \mathsf{Q}\,\phi$$

PROOF. By well-founded induction on the quantifier depth of $\mathsf{Q}\,\phi$. The base cases are covered by Theorem 4.5.1 for LTL and Theorem 5.5.10 for ELL. For the induction step, suppose $\mathsf{Q}\,\phi$ has path-quantifier depth n. It follows that the side conditions in the proof structure for $\mathcal{S}$ and $\Xi \vdash \mathsf{Q}\,\phi$ of the form $p \vdash \mathsf{Q}\,\psi$ have $\mathsf{qd}(\mathsf{Q}\,\psi) < n$. Hence by induction hypothesis $\mathcal{S}, p \models \mathsf{Q}\,\psi$ if and only if $\mathcal{S}, p \vdash \mathsf{Q}\,\psi$, so these side conditions can essentially be treated as if they were assertions. Thus, the results for the base cases lift to the induction step. □

5.7 Using Invariants in Proofs

Suppose we want to verify $\mathcal{S} \models \mathsf{Q}\,\phi$ for some system $\mathcal{S}$ and CTL* property $\mathsf{Q}\,\phi$. In the process of constructing a proof, we may reuse any previously proved invariant I of the system $\mathcal{S}$ in several ways. First, in the construction of a proof structure Π for $\mathcal{S}$ and $\mathsf{Q}\,\phi$ we may safely replace any side condition of a LTL or ELL proof rule of the form $p \models r$ by

$$I \wedge p \models r$$

This proceeding is sound, because I is also an invariant of the associated system $\mathcal{S}^\Pi$, so we know *a priori* that any state appearing on a trail will satisfy I.

Second, for the same reason it is sound to use invariant I to strengthen the assertion on the left-hand side of any implication or Hoare triple appearing as a verification condition in the success rules $\mathsf{A}(\mathrm{S})$ or $\mathsf{E}(\mathrm{S})$. In the same way we can also make use of invariants of the system $\mathcal{S}^\Pi$ itself. As an example, the assertion J_K defined by

$$J_K \stackrel{\mathrm{def}}{=} \bigvee_{\gamma \in \Gamma^+} (K = \lceil \gamma \rceil \to p_\gamma)$$

is an invariant of $\mathcal{S}^\Pi$. A type of formula occurring frequently as auxiliary assertion in applications is $\bigvee_{\gamma \in \Gamma_0} \widehat{p_\gamma}$ for some $\Gamma_0 \subseteq \Gamma^+$. By calling on the help of invariant J_K in the proof this assertion can be simplified to $K \in \lceil \Gamma_0 \rceil$.

Chapter 6

Proving Success under Fairness

In order to prove interesting liveness properties of reactive systems, it is important to be able to rely on the fair scheduling of system components (processes, transitions, ...). By definition a run is unfair, if some system "component" is indefinitely delayed though is it sufficiently often ready to progress. Liveness properties most frequently depend directly on the progress of individual components. So with pure non-deterministic scheduling (no fairness constraints), a much smaller number of liveness properties will hold of a system.

In our development of the local deductive model checking proof system, the LTL and ELL success rules do not account for fairness so far. It is important to note that this is the only reason why the soundness and relative completeness theorems for LTL and ELL (Theorems 4.5.1 and 5.5.10) and thus also the one for CTL* (Theorem 5.6.3) are restricted to saturated systems.

In this chapter, we will extend the success rules $\mathsf{A}(\mathrm{S})$ and $\mathsf{E}(\mathrm{S})$ to account for fairness constraints. As a consequence of Theorems 6.2.3 and 6.3.8, establishing soundness and relative completeness of the extended rule $\mathsf{A}(\mathrm{S})_{fair}$ and $\mathsf{E}(\mathrm{S})_{fair}$ for proving LTL and ELL success, respectively, the restriction to saturated systems in the above-mentioned soundness and completeness theorems for LTL, ELL and CTL* can be dropped. For CTL* we get

Theorem 6.0.1. (Soundness and Relative Completeness of CTL* Proof Structures) *Let $\mathcal{S}$ be system, Ξ an assertion and ϕ a CTL* formula. Then $\mathcal{S}, \Xi \models \phi$ if and only if $\mathcal{S}, \Xi \vdash \phi$.* □

Before we tackle the development of the extended rules, let us examine the possibility to express different types of fairness constraints in the temporal logic itself.

6.1 Expressing Fairness in CTL*

Generalised Fairness

Let us first consider a general form of (state-based) fairness. In LTL this can be expressed by the formula

$$\Omega \stackrel{\text{def}}{=} \bigwedge_{i=1}^{n} (\mathsf{F}\,\mathsf{G}\,p_i \vee \mathsf{G}\,\mathsf{F}\,r_i)$$

This formula satisfied by a run if for each i either the assertion p_i holds continuously from some point on or the assertion r_i holds infinitely often. Thus, if assertion p_i is seen to express non-readiness of some system component and r_i progress of that component, then formula Ω indeed expresses a generalised form of strong fairness. Weak and unconditional forms can be obtained by setting $p_i \stackrel{\text{def}}{=} \mathsf{false}$.

A simple way to verify a CTL* property ϕ of a saturated system $\mathcal{S}$ under such a fairness constraint Ω is to include the fairness constraint into the formula. This is achieved by replacing all path quantifiers by their relativised forms:

$$\begin{array}{lll} \mathsf{A}(\cdot) & \text{is replaced by} & \mathsf{A}_\Omega(\cdot) \stackrel{\text{def}}{=} \mathsf{A}(\Omega \rightarrow \cdot) \\ \mathsf{E}(\cdot) & \text{is replaced by} & \mathsf{E}_\Omega(\cdot) \stackrel{\text{def}}{=} \mathsf{E}(\Omega \wedge \cdot) \end{array}$$

Let ϕ^Ω be the formula obtained from ϕ by this transformation. If we denote by $\mathcal{S};\Omega$ the system with computations $\mathcal{C}_\mathcal{S} = \{\sigma \in \mathcal{R}_\mathcal{S} \mid \sigma \models \Omega\}$, then it is easy to see that $\mathcal{S};\Omega \models \phi$ precisely if $\mathcal{S} \models \mathsf{A}_\Omega\,\phi^\Omega$.

Weak and Strong Fairness

Let $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ be a system with fairness constraint $\mathcal{F} = (P, W, F)$. Unfortunately, weak and strong fairness as defined by $\mathcal{F}$ are not directly expressible in CTL* (in the sense of the existence of CTL* formula $\Omega_\mathcal{F}$ such that any run σ of $\mathcal{S}$ is fair w.r.t. $\mathcal{F}$ precisely if $\sigma \models \Omega_\mathcal{F}$). Expressing that some Λ_0-transition (with $\Lambda_0 \subseteq \Lambda$) is taken requires a relativised 'Next' operator. Such an operator is defined by

$$\sigma \models \mathsf{X}_{\Lambda_0}\,\phi \quad \text{if } (\sigma(0), \sigma(1)) \models \rho_{\Lambda_0}(\overline{x}, \overline{x}') \text{ and } \sigma^1 \models \phi$$

With this new operator a formula $\Omega_\mathcal{F}$ is definable in CTL* by

$$\Omega_\mathcal{F} \stackrel{\text{def}}{=} WF(W) \wedge SF(F)$$

where

$$
\begin{aligned}
\mathit{WF}(W) &\stackrel{\text{def}}{=} \textstyle\bigwedge_{\Lambda_w \in W} (\mathsf{F}\,\mathsf{G}\, en(\Lambda_w) \rightarrow \mathsf{G}\,\mathsf{F}\,\mathsf{X}_{\Lambda_w}\, \mathsf{true}) \\
\mathit{SF}(F) &\stackrel{\text{def}}{=} \textstyle\bigwedge_{\Lambda_f \in F} \big(\mathsf{G}\,\mathsf{F}\, en(\Lambda_f) \rightarrow \mathsf{G}\,\mathsf{F}\,\mathsf{X}_{\Lambda_f}\, \mathsf{true}\big)
\end{aligned}
$$

Note that $\mathit{WF}(W)$ and $\mathit{SF}(F)$ can equivalently be expressed in the following ways:

$$
\begin{aligned}
\mathit{WF}(W) &\equiv \textstyle\bigwedge_{\Lambda_w \in W} (\mathsf{G}\,\mathsf{F}\,\neg en(\Lambda_w) \vee \mathsf{G}\,\mathsf{F}\,\mathsf{X}_{\Lambda_w}\, \mathsf{true}) \\
&\equiv \textstyle\bigwedge_{\Lambda_w \in W} \mathsf{G}\,\mathsf{F}\,(\neg en(\Lambda_w) \vee \mathsf{X}_{\Lambda_w}\, \mathsf{true}) \\
\mathit{SF}(F) &\equiv \textstyle\bigwedge_{\Lambda_f \in F} \big(\mathsf{F}\,\mathsf{G}\,\neg en(\Lambda_f) \vee \mathsf{G}\,\mathsf{F}\,\mathsf{X}_{\Lambda_f}\, \mathsf{true}\big)
\end{aligned}
$$

Using the formula $\Omega_{\mathcal{F}}$ the verification that system $\mathcal{S}$ satisfies a CTL* formula ϕ, that is $\mathcal{S} \models \phi$, amounts to showing that $\mathcal{S}^- \models \mathsf{A}_{\Omega_{\mathcal{F}}}\, \phi^{\Omega_{\mathcal{F}}}$, where $\mathcal{S}^-$ is the saturated system underlying $\mathcal{S}$.

Discussion

Including fairness constraints into the property formulas has the advantage of great flexibility. While this method can be used for the verification of systems under generalised fairness, its use with the type of weak/strong fairness constraints we have introduced for transition systems would require the modification of the temporal logic and the proof system to include relativised Next operators X_Λ. However, while replacing Rule $\mathsf{E}(\mathsf{X})$ with a relativised version is no problem, the disjunctive semantics of LTL sequents creates some difficulties with generalising Rule $\mathsf{A}(\mathsf{X})$ to deal with these operators.

Another, more practical, disadvantage is that specification formulas including fairness constraints quickly grow to an unhandy size. This in turn leads to larger proof structures that are more difficult to survey. For these reasons we add the fairness constraints directly to the system specification and modify the success rules to account for fairness.

6.2 LTL Success under Fairness

In Section 3.3, we have introduced Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ for proving properties of the form

$$\mathsf{A}\left(\mathsf{F}\,q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i\right)$$

(where q and the p_i are assertions) for saturated systems. Rule $\mathsf{A}(\mathrm{S})$ for showing success of a LTL proof structure Π was then obtained by an appropriate instantiation of Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$. In this section, we follow a similar approach and first present Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})_{fair}$, an extension of Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ that accounts for fairness. Then this rule is slightly modified to deal with Π-fairness and instantiated to yield Rule $\mathsf{A}(\mathrm{S})_{fair}$, the LTL success rule under fairness.

6.2.1 Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})_{fair}$

Let $\mathcal{S}$ be a system with fairness constraint $\mathcal{F} = (P, W, F)$ and let q and $p_1, \ldots, p_m$ be assertions. Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})_{fair}$ for proving that the fair system $\mathcal{S}$ satisfies $\mathsf{A}(\mathsf{F}\,q \vee \bigvee_{i=1}^{m} p_i)$ is displayed in Figure 6.1. Just as Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ this rule is derived from Rule F-RESP of [MP91] for proving future response formulas of the form $\mathsf{G}(p \rightarrow \mathsf{F}\,q)$ under weak and strong (transition) fairness constraints (see also the discussion in Section 3.3.1).

Supposing that $\Lambda_1, \ldots, \Lambda_n$ enumerates $W \cup F$, this rule requires that we find an auxiliary assertion β_i for each p_i $(1 \leq i \leq m)$ just as Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ does and additionally an assertion β_{m+j} for each Λ_j $(1 \leq j \leq n)$, so we have $m + n$ auxiliary assertions.

Let us now take a look at the premises. Premises P1-P3 are the same as for Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$, with the index i ranging from 1 to $m + n$ in P2 and j from 1 to m in P3. The new conditions are P4-P6. These deal with the sets of fair transitions $\Lambda_k \in W \cup F$. Premise P4 states that from a β_{m+k}-state any Λ_k-transition either reaches a q-state or decreases the rank. These transitions are called "helpful" in [MP91], since they bring us nearer to a q-state. Conditions P5 and P6 deal with enabledness of fair transition sets. For the case where $\Lambda_k \in W$ P5 requires that β_{m+k} implies either q or enabledness of Λ_k. Finally, premise P6 covers the case where $\Lambda_k \in F$ and states that we have to prove that a modified system, call it $\mathcal{S}'$, satisfies the modified property $\mathsf{A}(\mathsf{F}(q \vee en(\Lambda_k)) \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i)$. System $\mathcal{S}'$ is the same as $\mathcal{S}$ except that the initial condition Θ is replaced by β_{m+k} and the fairness constraint $\mathcal{F} = (P, W, F)$ of $\mathcal{S}$ is replaced by $\mathcal{F}_k = (P, W, F_k)$, where $F_k = F - \{\Lambda_k\}$

removes the set Λ_k from the set F of strongly fair transition sets. Observe that although P6 requires the recursive application of Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})_{fair}$, this recursion is well-founded, because the set F is finite and F_k is smaller than F.

Let $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ be a system with $\mathcal{F} = (P, W, F)$, where $W \cup F = \{\Lambda_1, \ldots, \Lambda_n\}$. Let q and $p_1, \ldots, p_m$ be assertions. In order to apply this rule, find:

(a) a ranking function $\delta\colon \Sigma \to W$ mapping states of $\mathcal{S}$ into elements of a well-founded domain $(W, \succ)$, and

(b) assertions $\{\beta_1, \ldots, \beta_{m+n}\}$ (setting $\beta \stackrel{\text{def}}{=} \bigvee_{i=1}^{m+n} \beta_i$),

and check the validity of conditions P1-P6 below, where in P6 $\mathcal{F}_k \stackrel{\text{def}}{=} (P, W, F - \{\Lambda_k\})$.

P1.	$\Theta \to q \vee \beta$	
P2.	$\{\beta_i \wedge \delta{=}w\}\ \Lambda\ \{q \vee (\beta \wedge \delta \prec w) \vee (\beta_i \wedge \delta \preceq w)\}$	$i \in [1, m{+}n]$
P3.	$\{\beta_j \wedge \delta{=}w \wedge \neg p_j\}\ \Lambda\ \{q \vee (\beta \wedge \delta \prec w)\}$	$j \in [1, m]$
P4.	$\{\beta_{m+k} \wedge \delta{=}w\}\ \Lambda_k\ \{q \vee (\beta \wedge \delta \prec w)\}$	$k \in [1, n]$
P5.	$\beta_{m+k} \to q \vee en(\Lambda_k)$	$\Lambda_k \in W$
P6.	$\mathcal{S}, \beta_{m+k}; \mathcal{F}_k \vdash \mathsf{A}(\mathsf{F}(q \vee en(\Lambda_k)) \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i)$	$\Lambda_k \in F$
	$\mathcal{S} \vdash \mathsf{A}(\mathsf{F}\,q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i)$	

Figure 6.1: Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})_{fair}$

Soundness and Relative Completeness

THEOREM 6.2.1. (SOUNDNESS AND RELATIVE COMPLETENESS OF RULE $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})_{fair}$) *Let $\mathcal{S}$ be a system and let q and $p_1, \ldots, p_m$ be assertions. Then $\mathcal{S} \vdash \mathsf{A}(\mathsf{F}\,q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i)$ if and only if $\mathcal{S} \models \mathsf{A}(\mathsf{F}\,q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i)$.*

PROOF. "$\Rightarrow$" (soundness) By induction on the size of the set F. Suppose we have found the required assertions and the ranking and that premises P1-P6 hold. Consider a run $\sigma\colon s_0 \cdots s_j \cdots$ of the system $\mathcal{S}$. We have to show that σ is unfair or satisfies $\mathsf{F}\,q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i$. If σ satisfies $\mathsf{F}\,q$ then we are done.

Otherwise, q holds nowhere on σ, so β is invariant and the ranking δ does never increase on σ by P1 and P2. By well-foundedness the ranking δ is constant from some position l on in σ. Since β is the disjunction of all the β_i for $1 \leq i \leq m+n$, some β_i holds at s_l and continues to hold for all s_j with $j \geq l$ by P2. If $1 \leq i \leq m$, then also $s_j \models p_i$ for all $j \geq l$ by P3, so σ satisfies $\mathsf{F}\,\mathsf{G}\,p_i$ and we are done. Otherwise, $i = m+k$ for some $1 \leq k \leq n$ and by P4 Λ_k is never taken from position l on. If $\Lambda_k \in W$ it follows from P5 that Λ_k is enabled from l on, so σ is weakly unfair w.r.t. Λ_k. On the other hand, if $\Lambda_k \in F$ then by condition P6 and the induction hypothesis σ^l satisfies $\mathsf{F}(q \vee en(\Lambda_k)) \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i$ (since $s_l \models \beta_{m+k}$). If σ^l satisfies $\bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i$ then so does σ and we are done. Otherwise we have $\sigma^j \models \mathsf{F}\,en(\Lambda_k)$ for all $j \geq l$, since q holds nowhere on σ and β_{m+k} continues to hold from position l on. But this means that Λ_k is enabled infinitely often on σ^l and hence on σ, so σ is strongly unfair w.r.t. the set Λ_k. This establishes $\mathcal{S} \models \mathsf{A}(\mathsf{F}\,q \vee \bigvee_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,p_i)$ as required. Note that the base case (F empty) also clearly holds. We conclude that Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})_{fair}$ is sound.

"$\Leftarrow$" (relative completeness) The proof of relative completeness of Rule F-RESP of [MP91] goes through with the obvious minor modifications, so we do not repeat it here. $\square$

6.2.2 Rule $\mathsf{A}(S)_{fair}$ for LTL Success

Recall that in Proposition 3.3.2 we have characterised success of a LTL proof structure Π by

$$\mathcal{S}^{\Pi} \models \mathsf{A}_{\Pi}\,\widehat{\Omega}_{\mathsf{A}}$$

where the quantifier A_{Π} ranges over Π-fair trails and

$$\widehat{\Omega}_{\mathsf{A}} \stackrel{\text{def}}{=} \mathsf{F}\,K_{\top} \vee \Omega^0_{\mathsf{A}} \qquad \text{where} \quad \Omega^0_{\mathsf{A}} \stackrel{\text{def}}{=} \bigvee_{\psi \in \widehat{\Psi}_{\mathsf{A}}} \mathsf{F}\,\mathsf{G}\,K_{\psi}$$

In order to adapt Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ for proving LTL success, we have to modify it to deal with Π-fairness instead of the usual fairness constraints.

Definition 6.2.2. (Π-Enabledness) Let $\mathcal{S}$ be a system with fairness constraint $\mathcal{F} = (P, W, F)$, let Π be a proof structure and $\mathcal{S}^{\Pi}$ the system associated with Π with $\mathcal{F}^{\Pi} = (P^{\Pi}, W^{\Pi}, F^{\Pi})$. Define the assertion $en^{\Pi}(\Lambda_0^{\Pi})$ for $\Lambda_0^{\Pi} \in W^{\Pi} \cup F^{\Pi}$ by

$$en^{\Pi}(\Lambda_0^{\Pi}) \stackrel{\text{def}}{=} en(\Lambda_0)$$

where $\Lambda_0 \in W \cup F$ is the transition set of $\mathcal{S}$ inducing Λ_0^{Π}, that is, $\Lambda_0^{\Pi} = \pi_2^{-1}(\Lambda_0)$ (see also Definition 3.2.7). $\diamondsuit$

Let Π be a LTL proof structure for system $\mathcal{S}$ and let $\{K_\psi \mid \psi \in \widehat{\Psi}_{\mathsf{A}}\}$ and $K_\top$ as defined in Section 3.2.3. Furthermore, let $W^\Pi \cup F^\Pi = \{\Lambda_1^\Pi, \ldots, \Lambda_n^\Pi\}$ and $I = \widehat{\Psi}_{\mathsf{A}} \cup \{1, \ldots, n\}$. In order to apply this rule, find:

1. a ranking function $\delta : \Sigma^\Pi \to W$ mapping states of $\mathcal{S}^\Pi$ into elements of a well-founded domain $(W, \succ)$, and

2. assertions $\{\beta_i \mid i \in I\}$ (setting $\beta \stackrel{\text{def}}{=} \bigvee_{i \in I} \beta_i$),

and check the validity of conditions A1-A6 below, where $\Lambda_j = \pi_2(\Lambda_j^\Pi)$ in A5, A6 and $\mathcal{F}_k^\Pi = (P^\Pi, W^\Pi, F^\Pi - \{\Lambda_k^\Pi\})$ in A6.

A1.	$\Theta^\Pi \to q \vee \beta$	
A2.	$\{\beta_i \wedge \delta = w\}\ \Lambda^\Pi\ \{q \vee (\beta \wedge \delta \prec w) \vee (\beta_i \wedge \delta \preceq w)\}$	$i \in I$
A3.	$\{\beta_\psi \wedge \delta = w \wedge \neg K_\psi\}\ \Lambda^\Pi\ \{q \vee (\beta \wedge \delta \prec w)\}$	$\psi \in \widehat{\Psi}_{\mathsf{A}}$
A4.	$\{\beta_j \wedge \delta = w\}\ \Lambda_j^\Pi\ \{q \vee (\beta \wedge \delta \prec w)\}$	$j \in [1, n]$
A5.	$\beta_k \to q \vee en^\Pi(\Lambda_k^\Pi)$	$\Lambda_k^\Pi \in W^\Pi$
A6.	$\mathcal{S}^\Pi, \beta_k; \mathcal{F}_k^\Pi \vdash \mathsf{A}_\Pi \left(\mathsf{F}\left(q \vee en^\Pi(\Lambda_k^\Pi)\right) \vee \Omega_{\mathsf{A}}^0\right)$	$\Lambda_k^\Pi \in F^\Pi$
	$\mathcal{S}^\Pi \vdash \mathsf{A}_\Pi (\mathsf{F}\, q \vee \Omega_{\mathsf{A}}^0)$	

Figure 6.2: Rule $\mathsf{A}(\mathrm{S})_{fair}$: LTL success under fairness

This modified enabledness for $\Lambda_0^\Pi \in W^\Pi \cup F^\Pi$ means in fact enabledness of the original underlying transition set $\Lambda_0 \in W \cup F$ of $\mathcal{S}$. It might be helpful to see how weak and strong Π-fairness are expressed in LTL:

$$WF^\Pi(W^\Pi) \stackrel{\text{def}}{=} \bigwedge_{\Lambda_0^\Pi \in W^\Pi} \left(\mathsf{F}\,\mathsf{G}\, en^\Pi(\Lambda_0^\Pi) \to \mathsf{G}\,\mathsf{F}\,\mathsf{X}_{\Lambda_0^\Pi}\, \mathsf{true}\right)$$

$$SF^\Pi(F^\Pi) \stackrel{\text{def}}{=} \bigwedge_{\Lambda_0^\Pi \in F^\Pi} \left(\mathsf{G}\,\mathsf{F}\, en^\Pi(\Lambda_0^\Pi) \to \mathsf{G}\,\mathsf{F}\,\mathsf{X}_{\Lambda_0^\Pi}\, \mathsf{true}\right)$$

Rule $\mathsf{A}(\mathrm{S})_{fair}$ for showing LTL success under fairness (see Figure 6.2) is then obtained from Rule $\mathsf{A}(\mathsf{F}, \bigvee \mathsf{F}\,\mathsf{G})$ by

- instantiating $\mathcal{S}$ with $\mathcal{S}^\Pi$ and $p_1, \ldots, p_n$ with $\{K_\psi \mid \psi \in \widehat{\Psi}_{\mathsf{A}}\}$, and then
- replacing $en(\Lambda_k^\Pi)$ by $en^\Pi(\Lambda_k^\Pi)$ in P5 and P6.

Note that, since the rule invokes itself recursively, q remains uninstantiated at this point. In order to prove $\mathcal{S}^\Pi \models \mathsf{A}_\Pi \widehat{\Omega}_\mathsf{A}$ holds, we use the rule with q set to $K_\top$.

The notions of S-proof and S-provability (Definition 3.3.4) are modified to rely on the applicability of Rule $\mathsf{A}(\mathrm{S})_{fair}$ instead of Rule $\mathsf{A}(\mathrm{S})$. The proof of Theorem 6.2.1 can easily be adapted to show soundness and relative completeness of Rule $\mathsf{A}(\mathrm{S})_{fair}$ (with q unspecified). Together with Proposition 3.3.2 this yields

Theorem 6.2.3. (Soundness and Relative Completeness of Rule $\mathsf{A}(\mathrm{S})_{fair}$ for LTL Success) *Let* Π *be a LTL proof structure for a system* $\mathcal{S}$ *and sequent* $\Xi \vdash \mathsf{A}(\phi)$. *Then* $\Pi\colon \mathcal{S}, \Xi \Vdash \mathsf{A}\,\phi$ *if and only if* $\Pi\colon \mathcal{S}, \Xi \vdash \mathsf{A}\,\phi$. ☐

6.3 ELL Success under Fairness

In Section 5.3 we have introduced Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$ to prove properties of the form $\mathsf{E}\,(\bigwedge_{i=1}^m \mathsf{G}\,\mathsf{F}\,p_i)$, where $p_1, \ldots, p_m$ are assertions, for saturated systems. This rule was then instantiated to yield Rule $\mathsf{E}(\mathrm{S})$ for proving success of ELL proof structures.

Reducing Strong to Unconditional Fairness and Persistence

In order to fix some ideas for extending these rules to account for fairness constraints recall from Section 6.1 that proving an ELL property of the form $\mathsf{E}\,\psi$ for a system $\mathcal{S}$ with fairness constraint $\mathcal{F} = (P, W, F)$ amounts to showing that

$$\mathcal{S}^- \models \mathsf{E}\,(\Omega_\mathcal{F} \wedge \psi)$$

where $\mathcal{S}^-$ is the saturated system underlying $\mathcal{S}$ and $\Omega_\mathcal{F} = \mathit{WF}(W) \wedge \mathit{SF}(F)$ is the formula expressing the fairness constraint $\mathcal{F}$. Also recall that $\mathit{SF}(F)$ can be equivalently written as

$$\widehat{\mathit{SF}}(F) \stackrel{\text{def}}{=} \bigwedge_{\Lambda \in F} (\mathsf{F}\,\mathsf{G}\,\neg en(\Lambda) \vee \mathsf{G}\,\mathsf{F}\,\mathsf{X}_\Lambda\,\mathsf{true})$$

Consider a run σ of $\mathcal{S}^-$ satisfying $\Omega_\mathcal{F} \wedge \psi$, that is, a computation of $\mathcal{S}$ satisfying ψ. Then σ determines a (unique) partition (U, D) of F (that is, $U \cup D = F$ and $U \cap D = \varnothing$) such that

$$\sigma \models \mathit{UF}(U) \wedge \mathsf{F}\,\mathsf{G}\,\mathit{dis}(D)$$

where

$$UF(U) \stackrel{\text{def}}{=} \bigwedge_{\Lambda \in U} \mathsf{G}\,\mathsf{F}\,\mathsf{X}_\Lambda\,\mathsf{true}$$

$$dis(D) \stackrel{\text{def}}{=} \bigwedge_{\Lambda \in D} \neg en(\Lambda)$$

The partition (U, D) describes in what particular way σ satisfies the strong fairness formula $SF(F)$: each transition set $\Lambda \in D$ is disabled almost everywhere on σ (corresponding to the left disjunct for Λ in $\widehat{SF}(F)$), while any $\Lambda \in U$ is taken infinitely often on σ (corresponding to the right disjunct for Λ in $\widehat{SF}(F)$). The formula $UF(U)$ expresses *unconditional fairness* of each $\Lambda \in U$. Note also that $\mathsf{F}\,\mathsf{G}\,dis(D)$ is equivalent to $\bigwedge_{\Lambda \in D} \mathsf{F}\,\mathsf{G}\,\neg en(\Lambda)$.

If we extend our fairness constraints $\mathcal{F}$ to include unconditional fairness by defining $\mathcal{F} = (P, W, F, U)$, where W and F are defined as before and $U \subseteq P$ is a set of unconditionally fair transition sets, then we can formulate the statement

$$\mathcal{S}^- \models \mathsf{E}(\,WF(W) \wedge UF(U) \wedge \mathsf{F}\,\mathsf{G}\,dis(D) \wedge \psi)$$

equivalently as

$$\mathcal{S}; \mathcal{F}[U] \models \mathsf{E}(\mathsf{F}\,\mathsf{G}\,dis(D) \wedge \psi)$$

where $\mathcal{S}; \mathcal{F}[U]$ is the same as $\mathcal{S}$ except that the fairness constraint $\mathcal{F}$ of $\mathcal{S}$ is replaced by $\mathcal{F}[U] \stackrel{\text{def}}{=} (P, W, \varnothing, U)$. Clearly, $\mathcal{S}; \mathcal{F}[U] \models \mathsf{E}(\mathsf{F}\,\mathsf{G}\,dis(D) \wedge \psi)$ implies $\mathcal{S} \models \mathsf{E}\,\psi$, but the converse does not hold in general. However, we have the following

PROPOSITION 6.3.1. *Let $\mathcal{S}$ be a system with initial condition Θ and fairness constraint $\mathcal{F} = (P, W, F)$ and let $\mathsf{E}\,\psi$ be an ELL formula. Then $\mathcal{S} \models \mathsf{E}\,\psi$ if and only if there exist assertions Θ_j and partitions (U_j, D_j) of F for some $l \geq 1$ and $1 \leq j \leq l$ such that*

(i) $\Theta \rightarrow \bigvee_{j=1}^{l} \Theta_j$ *is valid, and*

(ii) $\mathcal{S}; \mathcal{F}[U_j], \Theta_j \models \mathsf{E}(\mathsf{F}\,\mathsf{G}\,dis(D_j) \wedge \psi)$ *for all* $1 \leq j \leq l$.

PROOF. "$\Leftarrow$" This direction is clear. "$\Rightarrow$" Let $\mathcal{S}$ be a system with fairness constraint $\mathcal{F} = (P, W, F)$. Suppose $\mathcal{S} \models \mathsf{E}\,\psi$. Define

$$\Psi(U, D) \stackrel{\text{def}}{=} WF(W) \wedge UF(U) \wedge dis(D) \wedge \psi$$

and let $P_1, \ldots, P_l$ with $P_j = (U_j, D_j)$ enumerate the set

$$\left\{ (U, D) \;\middle|\; \begin{array}{l} (U, D) \text{ partitions } F \text{ and there exists a} \\ \Theta\text{-run } \sigma \text{ of } \mathcal{S} \text{ such that } \sigma \models \mathsf{E}\,\Psi(U, D) \end{array} \right\}$$

Since F is finite, there are indeed finitely many P_j. Now we can define Θ_j for $1 \leq j \leq l$:

$$\Theta_j \stackrel{\text{def}}{=} \Theta \wedge \chi_{\mathsf{E}\,\Psi(P_j)}$$

Note first that although $\mathsf{E}\,\Psi(P_j)$ is, strictly speaking, not a CTL* formula (because of the relativised next operators), it is not difficult to see that there is characteristic assertion $\chi_{\mathsf{E}\,\Psi(P_j)}$ in $\mathcal{L}$. By the definition of the P_j and Θ_j the statement $\mathcal{S}^-, \Theta_j \models \mathsf{E}\,\Psi(P_j)$ holds, thus also $\mathcal{S}; \mathcal{F}[U_j], \Theta_j \models \mathsf{E}(\mathsf{F}\,\mathsf{G}\, dis(D_j) \wedge \psi)$ for each $1 \leq j \leq l$. Since $\mathcal{S} \models \mathsf{E}\,\psi$ by assumption, any initial state satisfies some Θ_j, hence $\Theta \to \bigvee_{j=1}^{l} \Theta_j$ is valid as required. $\square$

Let us return to the problem of proving ELL success. The ELL success formula Ω_{E} is of the form $\psi \stackrel{\text{def}}{=} \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\, r_i$. Let us for the moment disregard Π-fairness and stick with properties of this form over arbitrary systems. Let $\mathcal{S}$ be a system with fairness constraint $\mathcal{F} = (P, W, F)$. The above proposition allows us to reduce the problem of proving $\mathcal{S} \models \mathsf{E}\,\psi$ to showing

$$\mathcal{S}; \mathcal{F}[U_j], \Theta_j \models \mathsf{E}\left(\mathsf{F}\,\mathsf{G}\, dis(D_j) \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\, r_i\right)$$

for each $1 \leq j \leq l$ and an appropriate choice of assertions $\Theta_1, \ldots, \Theta_l$ and partitions $P_1, \ldots, P_l$ of F with $P_j = (U_j, D_j)$.

We now proceed as follows. In order to simplify the presentation, we first generalise our Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$ of Figure 5.1 to deal with properties of the form $\mathsf{F}\,\mathsf{G}\, q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\, r_i$ over saturated systems (Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$) and show its soundness and relative completeness. Then we show how to extend this rule to work with weak and unconditional fairness constraints (with $\mathcal{F}$'s of the form $(P, W, \varnothing, U)$), yielding Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$. Based on the reduction in Proposition 6.3.1, an auxiliary Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{fair}$ is then introduced for systems with the usual weak and strong fairness constraints. Finally, we will present variants of Rules $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$ and $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{fair}$ dealing with Π-fairness as required for ELL success.

Remark (history variables) Relative completeness results indicate that prior to the application of any rule in this section, it might be necessary to extend the system under study with a *history variable*.

6.3.1 *Rule* $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$

In order to simplify the presentation, we will first introduce a rule for saturated systems and properties of the form

$$\mathsf{E}\left(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i\right)$$

This rule, called $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$ and displayed in Figure 6.3, can then easily be extended to deal with fairness as sketched in the previous paragraph and detailed in the next section.

Let $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta)$ be a saturated system. Let q and $r_1, \dots, r_m$ be assertions. In order to apply this rule, find for each $0 \le i \le m$:

(a) a ranking function $\delta_i : \Sigma \to W_i$ mapping states of $\mathcal{S}$ into elements of a well-founded domain $(W_i, \succ_i)$, and

(b) an assertion α_i

and check the validity of conditions R1-R5 below (where in R1: $\alpha \overset{\text{def}}{=} \bigvee_{j=0}^{m} \alpha_i$ and in R3,R5: $1 \le i \le m$ and $i \oplus 1 \overset{\text{def}}{=} (i \,\mathsf{mod}\, m) + 1$).

R1. $\Theta \to \alpha$

R2. $\alpha_0 \to \neg q$

R3. $\alpha_i \to q$

R4. $\left\{\begin{array}{c} \alpha_0 \\ \wedge\ \ \delta_0 = u \end{array}\right\} \langle\Lambda\rangle \left\{\begin{array}{cc} & (\alpha_1 \wedge \delta_0 \preceq_0 u) \\ \vee & (\alpha_0 \wedge \delta_0 \prec_0 u) \end{array}\right\}$

R5. $\left\{\begin{array}{c} \alpha_i \\ \wedge\ \ \delta_0 = u \\ \wedge\ \ \delta_i = w \end{array}\right\} \langle\Lambda\rangle \left\{\begin{array}{cc} & (\alpha_{i\oplus 1} \wedge r_i \wedge \delta_0 \preceq_0 u) \\ \vee & (\alpha_i \wedge \delta_i \prec_i w \wedge \delta_0 \preceq_0 u) \\ \vee & (\alpha_0 \wedge \delta_0 \prec_0 u) \end{array}\right\}$

$\mathcal{S} \vdash \mathsf{E}(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i)$

Figure 6.3: Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$

The application of Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$ requires that we find for each $0 \le i \le m$ an auxiliary assertion α_i and a ranking function $\delta_i \colon \Sigma \to W_i$ mapping system states to elements of a well-founded domain $(W_i, \succ_i)$. For $1 \leq i \leq m$ these can be thought of as corresponding to assertion p_i, while α_0 and δ_0 are associated with q.

Before we discuss the premises in detail observe that setting q and α_0 to false and taking a trivial (constant) ranking for δ_0 yields exactly Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$, which is thus a special case of the present rule. Hence the assertions α_i and the rankings δ_i for $1 \leq i \leq m$ play a similar role as they do in Rule $\mathsf{E}(\bigwedge \mathsf{G}\,\mathsf{F})$. Intuitively speaking, for $1 \leq i \leq m$, in an α_i-segment the ranking δ_i decreases unless the target r_i is reached. New is that these "α_i-modes" are only active while q holds and whenever q does not hold a fall-back to the additional "α_0-mode" occurs and the ranking δ_0 decreases. As δ_0 does never increase, there can be only a finite number of fall-backs to $\neg q$-states.

Now let us examine the premises more closely. Premise R1 requires that any initial state also satisfies α_i for some $0 \leq i \leq m$. According to R2 and R3 the assertions α_0 and α_i (for $1 \leq i \leq m$) imply $\neg q$ and q, respectively.

Premise R4 states that from an α_0 state, it is possible to reach an α_1-state with δ_0 not increasing or again an α_0-state with the ranking δ_0 decreasing. The final premise R5 says that for the other modes α_i (for $1 \leq i \leq m$) there are three possibilities: from an α_i-state we may

1. advance to an $\alpha_{i \oplus 1} \wedge r_i$-state, with δ_0 not increasing, or
2. reach an α_i-state, with δ_i decreasing and δ_0 not increasing, or
3. fall back to a α_0-state with δ_0 decreasing.

Note that by premises R4 and R5 the ranking δ_0 is not allowed to increase.

Soundness and Relative Completeness

Theorem 6.3.2. (Soundness and Relative Completeness of Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$) *Let $\mathcal{S}$ be a saturated system, and let q and $r_1, \ldots, r_m$ be assertions. Then $\mathcal{S} \vdash \mathsf{E}(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i)$ if and only if $\mathcal{S} \models \mathsf{E}(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i)$.* □

Proof. Let $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta)$ be a saturated system and let q and $r_1, \ldots, r_m$ be assertions.

Soundness. Suppose we have found intermediate assertions α_i and ranking functions δ_i for $0 \leq i \leq m$ such that premises R1-R5 are valid. We say that a transition $s \xrightarrow{\lambda} s'$ is a *witness for a possibility triple* $\{p\}\ \langle \Lambda_0 \rangle\ \{q\}$, if $s \models p$, $s' \models q$ and $\lambda \in \Lambda_0$. Furthermore, we say a run (prefix) $\sigma\colon s_0 s_1 \cdots s_k \cdots$

is *constructed according to some set T of possibility triples*, if every transition on σ is a witness for some triple in T.

By inspecting the premises R1-R5, it is not hard to see that from any state $s \models \Theta$ a s-run σ can be constructed according to R4 and R5 and that α invariantly holds along these runs. Suppose $\sigma\colon s_0 s_1 \cdots s_k \cdots$ is such a run. We have to show that $\sigma \models \mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i$.

Suppose first that $\sigma \not\models \mathsf{F}\,\mathsf{G}\,q$. Since σ was constructed according to R4 and R5, the ranking δ_0 never increases along σ. By R4 and R5 it decreases whenever a transition is made to a state where q does not hold. Since $\sigma \not\models \mathsf{F}\,\mathsf{G}\,q$, there are infinitely many positions on σ where q does not hold. This implies that δ_0 decreases infinitely often, contradicting the well-foundedness of $\succ_0$. Hence, $\sigma \models \mathsf{F}\,\mathsf{G}\,q$.

It remains to show that also $\sigma \models \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i$. Since $\sigma \models \mathsf{F}\,\mathsf{G}\,q$, there is a position k_0 such that $s_j \models q$ for all $j \geq k_0$. Since α_0 implies $\neg q$, we also know that $s_j \not\models \alpha_0$ for all $j \geq k_0$. On the other hand, α holds invariantly along σ by R1,R4 and R5, so $s_{k_0} \models \alpha_i$ for some $1 \leq i \leq m$. We show that there is a $k_1 > k_0$ such that $s_{k_1} \models \alpha_{i\oplus 1} \wedge r_i$. This follows by well-foundedness of $\succ_i$ from the fact that all transitions $s_j \xrightarrow{\lambda_j} s_{j+1}$ for $j \geq k_0$ are witnesses for R5. Now we can repeat this argument to show that there is a $k_2 > k_1$ such that $s_{k_2} \models \alpha_{i\oplus 2} \wedge r_{i\oplus 1}$ and so on, ad infinitum. Hence, also $\sigma \models \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i$. Since our initial state was arbitrary we have $\mathcal{S} \models \mathsf{E}(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i)$ as required.

Relative completeness. Suppose $\mathcal{S} \models \mathsf{E}(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i)$ holds. We will define auxiliary assertions α_i and rankings δ_i for $0 \leq i \leq m$ and show that premises R1-R5 are valid.

Before we do so, however, we have to extend our system $\mathcal{S}$ with a (natural number) history variable H, yielding system $\widehat{\mathcal{S}}$ which is defined as follows:

$$\widehat{X} \stackrel{\text{def}}{=} X \cup \{H\} \qquad\qquad H \notin X$$

$$\widehat{\Theta} \stackrel{\text{def}}{=} \Theta \wedge \left(\begin{array}{lllll} & H = 0 & \wedge & \neg q \\ \vee & H = 1 & \wedge & q \end{array} \right)$$

$$\widehat{\rho_\lambda} \stackrel{\text{def}}{=} \rho_\lambda \wedge \bigvee_{i=0}^{m} \left(H = i \wedge \begin{array}{llllll} & H' = 0 & \wedge & \neg q' & & \\ \vee & H' = 1 & \wedge & q' & \wedge & i = 0 \\ \vee & H' = H & \wedge & q' \wedge \neg r' & \wedge & i > 0 \\ \vee & H' = H \oplus 1 & \wedge & q' \wedge r' & \wedge & i > 0 \end{array} \right)$$

The idea behind variable H is to record the actual "mode". Note that H is indeed a history variable as it does not affect the original state components, neither by modifying enabledness nor by making other variables depend on

H. Therefore, we also have $\widehat{\mathcal{S}} \models \mathsf{E}(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i)$. Now, let us define the intermediate assertions α_i:

$$\begin{array}{lllllllll} \alpha_0 & \stackrel{\text{def}}{=} & \chi_0 & \wedge & \neg q & \wedge & H = 0 & & \\ \alpha_i & \stackrel{\text{def}}{=} & \chi_0 & \wedge & q & \wedge & H = i & & \text{for } 1 \leq i \leq m \end{array}$$

where

$$\chi_0 \stackrel{\text{def}}{=} \chi_{\mathsf{E}\,\Psi_0} \quad \text{with} \quad \Psi_0 \stackrel{\text{def}}{=} \mathsf{F}\,\mathsf{G}\,q \wedge \textstyle\bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i$$

$$\chi_1 \stackrel{\text{def}}{=} \chi_{\mathsf{E}\,\Psi_1} \quad \text{with} \quad \Psi_1 \stackrel{\text{def}}{=} \mathsf{G}\,q \wedge \textstyle\bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i$$

Observe that each α_i implies χ_0, reflecting the fact that each state of a run σ witnessing Ψ_0 satisfies χ_0. However, this is not sufficient, as a run where χ_0 holds invariantly does not necessarily satisfy Ψ_0. It is for this reason that we need the ranking functions δ_i. Intuitively speaking, they make sure that each "target" r_i can be reached repeatedly while at the same time $\neg q$ is met only finitely often. The ranking functions $\delta_i \colon \widehat{\Sigma} \longrightarrow (\mathbb{N}, >)$, which all map extended states to the standard well-founded domain of natural numbers, are defined by

$$\delta_0(s) \stackrel{\text{def}}{=} \begin{cases} \min\{|\sigma| \mid \sigma \colon s \cdots s' \text{ a } \chi_0\text{−segment}, s' \models \chi_1\} & \text{if } s \models \chi_0 \wedge \neg\chi_1 \\ 0 & \text{otherwise} \end{cases}$$

$$\delta_i(s) \stackrel{\text{def}}{=} \begin{cases} \min\{|\sigma| \mid \sigma \colon s \cdots s' \text{ a } \chi_1\text{−segment}, |\sigma| > 1, s' \models r_i\} & \text{if } s \models \chi_1 \\ \delta_0(s) & \text{otherwise} \end{cases}$$

where $1 \leq i \leq m$ and a *q-segment* is a segment of a run such that all states appearing on it satisfy the assertion q. Note that the ranking functions are well-defined, since all the sets of which we take the minima are always non-empty. We say a segment σ *realises* ranking $\delta_j(s)$ for some $0 \leq j \leq m$, if $\delta_j(s) = |\sigma|$.

For a $\chi_0 \wedge \neg\chi_1$-state s the ranking $\delta_0(s)$ gives the minimal length of a χ_0-segment the last state of which satisfies χ_1, whereas it yields zero on any other states. In a similar way, for χ_1-states the ranking functions δ_i for $1 \leq i \leq m$ measure the least distance to a r_i-state reachable on a χ_1-segment. On the other hand, for a state s not satisfying χ_1 the ranking $\delta_i(s)$ equals $\delta_0(s)$. The idea is that we are not interested in the fulfillment of the r_i until we have reached a χ_1-state. Thus, from a $(\chi_0\wedge)\neg\chi_1$-state our primary goal is to reach χ_1. Note the equivalence $\Psi_0 \equiv \mathsf{F}\,\Psi_1$, underlining our idea of ignoring r_i until q has become stable.

We proceed to the verification of premises R1-R5. Premises R2 and R3 follow immediately from the definition of the α_i. The remaining premises are:

R1. Since $\Theta \models \mathsf{E}\,\Psi_0$ by hypothesis, any state s_0 that satisfies Θ also satisfies χ_0. Hence, $\widehat{\Theta}$ implies $\alpha_0 \vee \alpha_1$.

For premises R4 and R5, note that the first transition on the segment realising the respective ranking provides the witnessing transition required by the premise. In other words, we always follow the shortest way to reach the respective goal.

R4. Suppose $s \models \alpha_0$ and $\delta_0(s) = u$, realised by the χ_0-segment $\sigma\colon ss' \cdots s''$. Note that by the definition of δ_0 we have always $u > 1$. It follows that $s \models \chi_0 \wedge \neg q$ and $s(H) = 0$, as well as $s' \models \chi_0$. We distinguish two cases:

(a) $s' \not\models q$. This implies that $s'(H) = 0$ and $s' \not\models \chi_1$. Therefore, $\delta_0(s') < u$ and $s' \models \alpha_0$ as required.

(b) $s' \models q$. In this case $s'(H) = 1$. For the ranking δ_0 we have $\delta_0(s') = u - 1$ if $s \not\models \chi_1$ and $\delta_0(s') = 0$ otherwise. In both cases $\delta_0(s') \leq u$ and also $s' \models \alpha_1$.

R5. Suppose $s \models \alpha_i$, $\delta_0(s) = u$ and $\delta_i(s) = w$ for some $1 \leq i \leq m$. There are two main cases:

(a) $s \not\models \chi_1$. Let $\sigma\colon ss' \cdots s''$ be a χ_0-segment realising $\delta_0(s) = u$. Note that $u > 1$. We have three sub-cases:

(i) $s' \not\models q$. It follows that $s' \not\models \chi_1$, hence $u > 2$ and $\delta_0(s') = u - 1 < u$. By the definition of $\widehat{\rho_\lambda}$ we have $s'(H) = 0$ (a fall-back). Therefore, $s' \models \alpha_0 \wedge (\delta_0 < u)$.

(ii) $s' \models q \wedge \neg r_i$. Since s' is on a χ_0-segment we have $s' \models \chi_0 \wedge q$ and by the definition of $\widehat{\rho_\lambda}$ we stay in $s'(H) = i$. Thus, $s' \models \alpha_i$. We also have $s' \not\models \chi_1$, otherwise we would also have $s \models \chi_1$, since $s \models q$. Hence, $u > 2$ and $\delta_0(s') = u - 1 < u$. But as neither s nor s' satisfies χ_1 we have $\delta_i(s) = w = \delta_0(s) = u > \delta_0(s') = \delta_i(s')$. Hence, $s' \models \alpha_i \wedge (\delta_i < w) \wedge (\delta_0 < u)$ as required.

(iii) $s' \models q \wedge r_i$. By the definition of $\widehat{\rho_\lambda}$ we have $s'(H) = i \oplus 1$, thus $s' \models \alpha_{i\oplus 1} \wedge r_i$. For the ranking the same argument as in (ii) above shows that $\delta_0(s') \leq u$. Therefore, $s' \models \alpha_{i\oplus 1} \wedge r_i \wedge (\delta_0 \leq u)$ in this case.

(b) $s \models \chi_1$. Let $\sigma\colon ss' \cdots s''$ be a χ_1-segment realising $\delta_i(s) = u$. Then we have $s' \models \chi_1$ and therefore also $s' \models \chi_0 \wedge q$, since χ_1 implies χ_0 as well as q. As s and s' both satisfy χ_1 we get $\delta_0(s) = \delta_0(s') = 0$.

(i) $s' \not\models r_i$. It follows that $s(H) = i$ and $w > 2$, implying $s' \models \alpha_i$ and $\delta_i(s') < w$. Hence, $s' \models \alpha_i \wedge \delta_i < w \wedge (\delta_0 \leq u)$.

(ii) $s' \models r_i$. Here the "mode" is switched to $s'(H) = i \oplus 1$ and therefore $s' \models \alpha_{i\oplus 1} \wedge r_i \wedge (\delta_0 \leq u)$.

Proof. This completes the proof of *semantic* completeness. For *syntactic* completeness it remains to show that the auxiliary assertions and the rankings are expressible in our assertion language $\mathcal{L}$ over the (acceptable) structure $\mathcal{A}$ we are working in (see Section 2.3.1). The assertions α_i are already formulated in $\mathcal{L}$. It is not difficult to see that the rankings δ_i can also be expressed in $\mathcal{L}$ using the coding scheme for finite sequences supported by the structure $\mathcal{A}$. We conclude that Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$ is complete relative to validity in $\mathcal{L}$. $\square$

6.3.2 Rules $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$ and $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{fair}$

Let us now extend Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$ to account for fairness. Recalling the discussion at the beginning of this section, showing that

$$\mathcal{S} \models \mathsf{E}\left(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i\right)$$

holds for a system $\mathcal{S}$ with fairness constraint $\mathcal{F} = (P, W, F)$ can be reduced by Proposition 6.3.1 to proving

$$\mathcal{S}; \mathcal{F}[U_j], \Theta_j \models \mathsf{E}\left(\mathsf{F}\,\mathsf{G}\,(q \wedge dis(D_j)) \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i\right) \qquad (*)$$

for an appropriate choice of assertions Θ_j and partitions (U_j, D_j) of F. The system $\mathcal{S}; \mathcal{F}[U_j]$ is obtained from $\mathcal{S}$ by substituting the fairness constraint $\mathcal{F}$ with $\mathcal{F}[U_j] = (P, W, \varnothing, U_j)$, thus replacing the strong fairness constraint F of $\mathcal{F}$ by the unconditional fairness constraint U_j.

This reduction is implemented in Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{fair}$ (see Figure 6.4), which invokes Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$ (displayed in Figure 6.5), to prove the statements of the form $(*)$ above.

Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$ is a modified version of Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$. In addition to the latter rule this new rule requires that we also find pairs of assertions α_j and ranking functions δ_j for $m+1 \leq j \leq m+n$, each such pair

Let $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ be a system with fairness constraints $\mathcal{F} = (P, W, F)$. Let q and $r_1, \ldots, r_m$ be assertions. Find:

(a) assertions $\Theta_1, \ldots, \Theta_l$, and

(b) partitions (U_j, D_j) of F for $1 \leq j \leq l$,

such that condition F1 below is valid and F2 can be established using Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$.

$$\begin{array}{ll} \text{F1.} & \Theta \rightarrow \bigvee_{j=1}^{l} \Theta_j \\ \text{F2.} & \mathcal{S}; \mathcal{F}[U_j], \Theta_j \vdash \mathsf{E}\,(\mathsf{F}\,\mathsf{G}\,(q \wedge dis(D_j)) \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i) \qquad 1 \leq j \leq l \\ \hline & \mathcal{S} \vdash \mathsf{E}(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i) \end{array}$$

Figure 6.4: Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{fair}$

corresponding to an element of the set $\{\Lambda_1, \ldots, \Lambda_n\} = W \cup U$ of weakly and unconditionally fair transition sets.

DEFINITION 6.3.3. Let $\alpha_0, \alpha_1, \ldots, \alpha_l$ be assertions and $\delta_0, \delta_1, \ldots, \delta_l$ be ranking functions as required for the application of Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$ of Figure 6.5. Define assertions $\alpha_i^{u,w}$, $\alpha_i^{\leq u}$, $\alpha_i^{<w}$ and $\alpha_i^{<w,\leq u}$ for $0 \leq i \leq l$ by

$$\begin{array}{lcl} \alpha_i^{u,w} & \stackrel{\text{def}}{=} & \alpha_i \wedge \delta_0 = u \wedge \delta_i = w \\ \alpha_i^{\leq u} & \stackrel{\text{def}}{=} & \alpha_i \wedge \delta_0 \preceq_0 u \\ \alpha_i^{<w} & \stackrel{\text{def}}{=} & \alpha_i \wedge \delta_i \prec_i w \\ \alpha_i^{<w,\leq u} & \stackrel{\text{def}}{=} & \alpha_i \wedge \delta_i \prec_i w \wedge \delta_0 \preceq_0 u \end{array}$$

$\diamondsuit$

Premises R1-R5 are the same as in Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$. Using the notation introduced in the definition above, R5 now reads:

$$\text{R5. } \{\alpha_i^{u,w}\}\ \langle\Lambda\rangle\ \left\{(\alpha_{i\oplus 1}^{\leq u} \wedge r_i) \vee \alpha_i^{<w,\leq u} \vee \alpha_0^{<u}\right\}$$

where $0 \leq i \leq m$. Note that superscripting an α_i with $\leq u$ expresses (independently of i) that δ_0 does not increase, while a superscript $< w$ states the decrease of δ_i.

Given a system $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ with fairness constraint $\mathcal{F} = (P, W, \varnothing, U)$ and assertions q and $r_1, \ldots, r_m$. Suppose $\Lambda_1, \ldots, \Lambda_n$ enumerates $W \cup U$.

In order to apply this rule, find for each $0 \leq i \leq m + n$:

(a) a ranking function $\delta_i : \Sigma \rightarrow W_i$ mapping states of $\mathcal{S}$ into elements of a well-founded domain $(W_i, \succ_i)$,

(b) an assertion α_i,

and check the validity of conditions R1-R7 below.

$$
\begin{array}{ll}
\text{R1.} & \Theta \rightarrow \alpha \\
\text{R2.} & \alpha_0 \rightarrow \neg q \\
\text{R3.} & \bigvee_{i=1}^{m+n} \alpha_i \rightarrow q \\
\text{R4.} & \{\alpha_0^{u,u}\}\ \langle \Lambda \rangle\ \left\{(\alpha_1^{\leq u} \vee \alpha_0^{<u}\right\} \\
\text{R5.} & \{\alpha_i^{u,w}\}\ \langle \Lambda \rangle\ \left\{(\alpha_{i\oplus 1}^{\leq u} \wedge r_i) \vee \alpha_i^{<w,\leq u} \vee \alpha_0^{<u}\right\} \\
\text{R6.} & \left\{\alpha_j^{u,w}\right\}\ \langle \Lambda_{j-m} \mid \Lambda \rangle\ \left\{\alpha_{j\oplus 1}^{\leq u} \mid (\alpha_{j\oplus 1}^{\leq u} \wedge d_j) \vee \alpha_j^{<w,\leq u} \vee \alpha_0^{<u}\right\} \\
\text{R7.} & \{\alpha_k^{u,w}\}\ \langle \Lambda_{k-m} \mid \Lambda \rangle\ \left\{\alpha_{k\oplus 1}^{\leq u} \mid \alpha_k^{<w,\leq u} \vee \alpha_0^{<u}\right\} \\
\hline
 & \mathcal{S} \vdash \mathsf{E}(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i)
\end{array}
$$

The side conditions are: $1 \leq i \leq m$ in R5, $m + 1 \leq j \leq m + n$ and $\Lambda_{j-m} \in W$ in R6, and $m + 1 \leq k \leq m + n$ and $\Lambda_{k-m} \in U$ in R7.

The assertion α is defined by $\alpha \stackrel{\text{def}}{=} \bigvee_{i=0}^{m+n} \alpha_i$, assertions $\alpha_i^{u,w}$, $\alpha_i^{\leq u}$, $\alpha_i^{<w}$ and $\alpha_i^{<w,\leq u}$ are as in Definition 6.3.3, and $d_j \stackrel{\text{def}}{=} \neg en(\Lambda_{j-m})$ and the operation $\oplus$ is given by $a \oplus b \stackrel{\text{def}}{=} ((a + b - 1) \bmod (m + n)) + 1$.

Figure 6.5: Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$

The new premises dealing with weak and unconditional fairness are R6 and R7, respectively. Note that, in a very similar way as the assertions r_i of R5, each element $W \cup U$ is associated with a new "target" that is visited infinitely often on any run σ constructed according to R4-R7. Namely, each $\Lambda_w \in W$ must be taken or disabled infinitely often, and each $\Lambda_u \in U$ must be taken infinitely often on such a run σ (thus making sure that σ, in addition to witnessing $\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i$, is indeed a computation of $\mathcal{S}$). For this reason, premises R6 and R7 are very similar to R5. The difference is that the "target" now involves taking transition sets in W or U. In order to account for this new type of target, we introduce a generalised form of possibility triple.

DEFINITION 6.3.4. (GENERALISED POSSIBILITY TRIPLE) Let $\mathcal{S}$ be a system with variables X and set of transitions Λ. Let p, q_1 and q_2 be assertions over X and let $\Lambda_1, \Lambda_2 \subseteq \Lambda$. Define

$$\{p\}\ \langle \Lambda_1 \mid \Lambda_2 \rangle\ \{q_1 \mid q_2\} \stackrel{\text{def}}{=} p \rightarrow (\langle \Lambda_1 \rangle\, q_1 \vee \langle \Lambda_2 \rangle\, q_2)$$

$\diamondsuit$

A generalised possibility triple of the form $\{p\}\ \langle \Lambda_1 \mid \Lambda_2 \rangle\ \{q_1 \mid q_2\}$ states that from a p-state there is a Λ_1-transition leading to a q_1-state *or* a Λ_2-transition leading to a q_2-state.

Consider first the new premise R7 concerning unconditional fairness. For $m+1 \leq k \leq m+n$ and $\Lambda_{k-m} \in U$, it reads

$$\text{R7. } \{\alpha_k^{u,w}\}\ \langle \Lambda_{k-m} \mid \Lambda \rangle\ \left\{\alpha_{k\oplus 1}^{\leq u} \mid \alpha_k^{<w,\leq u} \vee \alpha_0^{<u}\right\}$$

stating that in α_k-mode, there is

- a Λ_{k-m}-transition leading to the successor mode $\alpha_{k\oplus 1}$ with the ranking δ_0 not increasing, *or*
- an arbitrary system transition either preserving α_k-mode with δ_k decreasing (and δ_0 not increasing) or causing a fall-back to α_0 with δ_0 decreasing.

In the former case, taking an Λ_{k-m}-transition means reaching the target for k.

Premise R6 dealing with weak fairness combines the forms of R5 and R7. For $m+1 \leq j \leq m+n$ and $\Lambda_{j-m} \in W$, it is given by

$$\text{R6. } \left\{\alpha_j^{u,w}\right\}\ \langle \Lambda_{j-m} \mid \Lambda \rangle\ \left\{\alpha_{j\oplus 1}^{\leq u} \mid (\alpha_{j\oplus 1}^{\leq u} \wedge \neg en(\Lambda_{j-m})) \vee \alpha_j^{<w,\leq u} \vee \alpha_0^{<u}\right\}$$

In addition to R7, there is a second possibility to hit the target for j, namely by taking any system transition leading to a state in the successor mode $\alpha_{j\oplus 1}$ where Λ_{j-m} is disabled.

After this discussion it is not difficult to see that, provided we can find the required assertions α_i and ranking functions δ_i such that all premises hold, any Θ-run σ constructed according to R4-R7 is weakly fair w.r.t. all $\Lambda_w \in W$ and unconditionally fair w.r.t. all $\Lambda_u \in U$, that is, σ is a Θ-computation of $\mathcal{S}$ (w.r.t. $\mathcal{F}$). It is then a tedious, but not a difficult affair to adapt the proof of soundness and relative completeness of Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})$ to Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$.

Theorem 6.3.5. (Soundness and Relative Completeness of Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$) *Let $\mathcal{S}$ be a system with fairness constraint $\mathcal{F} = (P, W, \varnothing, U)$ and let q and $r_1, \ldots, r_m$ be assertions. Then $\mathcal{S} \vdash \mathsf{E}\,(\mathsf{F}\,\mathsf{G}\, q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\, r_i)$ by Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$ if and only if $\mathcal{S} \models \mathsf{E}\,(\mathsf{F}\,\mathsf{G}\, q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\, r_i)$.* □

The following result then follows directly from the previous one and Proposition 6.3.1:

Theorem 6.3.6. (Soundness and Relative Completeness of Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{fair}$) *Let $\mathcal{S}$ be a system and let q and $r_1, \ldots, r_m$ be assertions. Then $\mathcal{S} \vdash \mathsf{E}\,(\mathsf{F}\,\mathsf{G}\, q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\, r_i)$ by Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{fair}$ if and only if $\mathcal{S} \models \mathsf{E}\,(\mathsf{F}\,\mathsf{G}\, q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\, r_i)$.* □

6.3.3 Rules $\mathsf{E}(S)_{fair}$ and $\mathsf{E}(S)^{\top}_{wuf}$ for ELL Success

In order to derive rules for proving ELL success from Rules $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{fair}$ and $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$, we will instantiate the system $\mathcal{S}$ with the system $\mathcal{S}^{\Pi}$ associated with a proof structure Π and the assertions $p_1, \ldots, p_m$ with $\{\neg K_{\psi} \mid \psi \in \Psi_{\mathsf{E}}\}$. Additionally, we will need to make two modifications, one concerning Π-fairness and the other concerning the special role of the control location $\top$ in the associated system $\mathcal{S}^{\Pi}$. The resulting Rules $\mathsf{E}(\mathrm{S})_{fair}$ and $\mathsf{E}(\mathrm{S})^{\top}_{wuf}$ are displayed in Figures 6.6 and 6.7, respectively.

We now examine the mentioned modifications in more detail. First, we have to deal with Π-fairness instead of the usual type fairness constraints. This modification concerns the enabledness assertions only. More precisely, in Rule $\mathsf{E}(\mathrm{S})^{\top}_{wuf}$

- assertion $dis^{\Pi}(D^{\Pi})$ instantiates q of Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$, where

$$dis^{\Pi}(D^{\Pi}) \stackrel{\text{def}}{=} \bigwedge_{\Lambda_0^{\Pi} \in D^{\Pi}} \neg en^{\Pi}(\Lambda_0^{\Pi})$$

Let Π be an ELL proof structure for system $\mathcal{S}$ and let $\Omega_{\mathsf{E}} = \bigwedge_{\psi \in \Psi_{\mathsf{E}}} \mathsf{G}\,\mathsf{F}\,\neg K_\psi$ as defined in Section 5.2. Find:

(a) assertions $\Theta_1, \ldots, \Theta_l$, and

(b) partitions (U_j^Π, D_j^Π) of F^Π for $1 \leq j \leq l$,

such that condition C1 below is valid and C2 can be established by Rule $\mathsf{E}(\mathrm{S})^\top_{wuf}$.

$$\frac{\begin{array}{ll} \text{C1.} & \Theta^\Pi \to \bigvee_{j=1}^{l} \Theta_j \\ \text{C2.} & \mathcal{S}^\Pi, \Theta_j \vdash_{(U_j^\Pi, D_j^\Pi)} \mathsf{E}_\Pi\, \Omega_{\mathsf{E}} \quad 1 \leq j \leq l \end{array}}{\mathcal{S}^\Pi \vdash \mathsf{E}_\Pi\, \Omega_{\mathsf{E}}}$$

Figure 6.6: Rule $\mathsf{E}(\mathrm{S})_{fair}$

- assertion $d_j \stackrel{\text{def}}{=} \neg en^\Pi(\Lambda^\Pi_{j-m})$ in E6 of $\mathsf{E}(\mathrm{S})^\top_{wuf}$ replaces $\neg en(\Lambda_{j-m})$ in R6 of $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$.

The second modification is based on the fact that any trail prefix $t_0 \cdots t_m$ with $t_m(K) = \lceil \top \rceil$ can be extended to a (successful) Π-fair trail, so it is not necessary to prove this every time we apply the ELL success rules. For this reason we have added a disjunct $K_\top$ on the right hand side of each possibility triple in E4-E7, saving us from proving anything about modes and rankings whenever $K_\top$ can be reached by a Λ^Π-transition (see also Section 5.3.2). Because of this second modification the conclusion of Rule $\mathsf{E}(\mathrm{S})^\top_{wuf}$ reads

$$\mathcal{S}^\Pi, \Xi \vdash_{(U^\Pi, D^\Pi)} \mathsf{E}_\Pi\, \Omega_{\mathsf{E}}$$

rather than

$$\mathcal{S}^\Pi; \mathcal{F}^\Pi[U^\Pi], \Xi \vdash \mathsf{E}_\Pi(\mathsf{F}\,\mathsf{G}\, dis^\Pi(D^\Pi) \wedge \Omega_{\mathsf{E}})$$

and premise C2 of Rule $\mathsf{E}(\mathrm{S})_{fair}$ accordingly reads $\mathcal{S}^\Pi, \Theta_j \vdash_{(U_j^\Pi, D_j^\Pi)} \mathsf{E}_\Pi\, \Omega_{\mathsf{E}}$. Note that, in contrast to the first modification, this one is not necessary but convenient.

Let Π be an ELL proof structure for system $\mathcal{S}$ and let $\mathcal{S}^\Pi$ with fairness constraint $\mathcal{F}^\Pi = (P^\Pi, W^\Pi, F^\Pi)$ be the system associated with Π. Let (U^Π, D^Π) be a partition of F^Π and suppose that $\Lambda_1^\Pi, \ldots, \Lambda_m^\Pi$ enumerates $W^\Pi \cup U^\Pi$. Let Ξ be an assertion and $\Omega_{\mathsf{E}} = \bigwedge_{\psi \in \Psi_{\mathsf{E}}} \mathsf{G}\,\mathsf{F}\,\neg K_\psi$ as defined in Section 5.2. Suppose $K_1, \ldots, K_n$ enumerates $\{K_\psi \mid \psi \in \Psi_{\mathsf{E}}\}$. In order to apply this rule, find

(a) a ranking function $\delta_i : \Sigma^\Pi \to W_i$ mapping states of $\mathcal{S}^\Pi$ into elements of a well-founded domain $(W_i, \succ_i)$, and

(b) an assertion α_i,

for each $0 \le i \le m+n$ and check the validity of conditions E1-E7:

E1. $\Xi \to \alpha$

E2. $\alpha_0 \to \neg dis^\Pi(D^\Pi)$

E3. $\bigvee_{i=1}^{m+n} \alpha_i \to dis^\Pi(D^\Pi)$

E4. $\{\alpha_0^{u,u}\}\ \langle \Lambda^\Pi \rangle\ \{K_\top \vee \alpha_1^{\le u} \vee \alpha_0^{<u}\}$

E5. $\{\alpha_i^{u,w}\}\ \langle \Lambda^\Pi \rangle\ \left\{K_\top \vee \left(\alpha_{i\oplus 1}^{\le u} \wedge \neg K_i\right) \vee \alpha_i^{\le u,<w} \vee \alpha_0^{<u}\right\}$

E6. $\left\{\alpha_j^{u,w}\right\}\ \left\langle \Lambda_{j-m}^\Pi \mid \Lambda^\Pi \right\rangle\ \left\{\alpha_{j\oplus 1}^{\le u} \mid K_\top \vee \left(\alpha_{j\oplus 1}^{\le u} \wedge d_j\right) \vee \alpha_j^{\le u,<w} \vee \alpha_0^{<u}\right\}$

E7. $\{\alpha_k^{u,w}\}\ \left\langle \Lambda_{k-m}^\Pi \mid \Lambda^\Pi \right\rangle\ \left\{\alpha_{k\oplus 1}^{\le u} \mid K_\top \vee \alpha_k^{\le u,<w} \vee \alpha_0^{<u}\right\}$

$\mathcal{S}^\Pi, \Xi \vdash_{(U^\Pi, D^\Pi)} \mathsf{E}_\Pi\, \Omega_{\mathsf{E}}$

The side conditions are: $1 \le i \le m$ in E5, $m+1 \le j \le m+n$ and $\Lambda_{j-m}^\Pi \in W^\Pi$ in E6, and $m+1 \le k \le m+n$ and $\Lambda_{k-m}^\Pi \in U^\Pi$ in E7.

The assertion α is defined by $\alpha \stackrel{\text{def}}{=} \bigvee_{i=0}^{m+n} \alpha_i$, assertions $\alpha_i^{u,w}$, $\alpha_i^{\le u}$, $\alpha_i^{<w}$ and $\alpha_i^{<w,\le u}$ are as in Definition 6.3.3; $dis^\Pi(D^\Pi)$ and d_j are defined by $dis^\Pi(D) \stackrel{\text{def}}{=} \bigwedge_{\Lambda_0^\Pi \in D^\Pi} \neg en^\Pi(\Lambda_0^\Pi)$ and $d_j \stackrel{\text{def}}{=} \neg en^\Pi(\Lambda_{j-m}^\Pi)$ with $en^\Pi(\cdot)$ as in Definition 6.2.2; the operation $\oplus$ is given by $a \oplus b \stackrel{\text{def}}{=} ((a+b-1) \bmod (m+n)) + 1$.

Figure 6.7: Rule $\mathsf{E}(\mathrm{S})_{wuf}^\top$

A Special Case: no Strong Fairness. If the strong fairness constraint F of the system (and hence F^{Π}) is empty, the only partition of F^{Π} is $(\varnothing, \varnothing)$, so we can directly apply Rule $\mathsf{E}(\mathrm{S})^{\top}_{wuf}$ by instantiating Ξ with Θ^{Π} and both U^{Π} and D^{Π} with $\varnothing$. We can also replace $dis^{\Pi}(\varnothing)$, being an empty conjunction, by true and "turn off" α_0 by setting it to false, making E2 and E4 hold trivially. The help of α_0 to ensure the persistence of $dis^{\Pi}(\varnothing)$ is not needed, since $\mathsf{F\,G\,true}$ is a tautology.

Soundness and Relative Completeness

LEMMA 6.3.7. *Let Π be a ELL proof structure for a system $\mathcal{S}$, let Ξ be an assertion and (U^{Π}, D^{Π}) a partition of F^{Π}, the strong fairness constraint of the associated system $\mathcal{S}^{\Pi}$. Then*

(i) $\mathcal{S}^{\Pi}, \Xi \vdash_{(U^{\Pi}, D^{\Pi})} \mathsf{E}_{\Pi}\, \Omega_{\mathsf{E}}$ *by Rule* $\mathsf{E}(S)^{\top}_{wuf}$ *implies* $\mathcal{S}^{\Pi}, \Xi \models \mathsf{E}_{\Pi}\, \Omega_{\mathsf{E}}$, *and*

(ii) $\mathcal{S}^{\Pi}; \mathcal{F}^{\Pi}[U^{\Pi}], \Xi \models \mathsf{E}_{\Pi} \left(\mathsf{F\,G}\, dis^{\Pi}(D^{\Pi}) \wedge \Omega_{\mathsf{E}}\right)$ *implies* $\mathcal{S}^{\Pi}, \Xi \vdash_{(U^{\Pi}, D^{\Pi})} \mathsf{E}_{\Pi}\, \Omega_{\mathsf{E}}$ *by Rule* $\mathsf{E}(S)^{\top}_{wuf}$.

PROOF. Let us call $\mathsf{E}(\mathrm{S})_{wuf}$ the rule obtained from Rule $\mathsf{E}(\mathrm{S})^{\top}_{wuf}$ by substituting false for $K_{\top}$ (hence ignoring the second modification above). It is straightforward to adapt the proof of soundness and relative completeness of Rule $\mathsf{E}(\mathsf{F\,G}, \bigwedge \mathsf{G\,F})_{wuf}$ (Theorem 6.3.5) to Rule $\mathsf{E}(\mathrm{S})_{wuf}$, showing that the latter rule it is sound and relatively complete for proving

$$\mathcal{S}^{\Pi}; \mathcal{F}[U^{\Pi}], \Xi \models \mathsf{E}_{\Pi} \left(\mathsf{F\,G}\, dis^{\Pi}(D^{\Pi}) \wedge \Omega_{\mathsf{E}}\right)$$

Then (ii) follows immediately from relative completeness of Rule $\mathsf{E}(\mathrm{S})_{wuf}$.

For (i) suppose that all premises of Rule $\mathsf{E}(\mathrm{S})^{\top}_{wuf}$ are valid and consider a state t_0 of $\mathcal{S}^{\Pi}$ satisfying Ξ. Then in the process of constructing a trail starting in t_0 according to E4-E7, we will either reach point where we have already constructed $t_0 \cdots t_k$ and $t_k \models K_{\top}$, in which case this prefix can certainly by extended to a successful Π-fair trail ϑ, or we will never reach a state satisfying $K_{\top}$, in which case it follows from the soundness of Rule $\mathsf{E}(\mathrm{S})_{wuf}$ that the constructed trail ϑ is Π-fair and successful. In any case there is a successful Π-fair trail starting in t_0, so point (i) holds. □

We modify the notions of S-proof and S-provability (Definition 5.3.3) to rest on the applicability of Rule $\mathsf{E}(\mathrm{S})_{fair}$ instead of Rule $\mathsf{E}(\mathrm{S})$.

THEOREM 6.3.8. (SOUNDNESS AND RELATIVE COMPLETENESS OF RULE $\mathsf{E}(\mathrm{S})_{fair}$ FOR ELL SUCCESS) *Let Π be a ELL proof structure for a system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{E}(\phi)$. Then $\mathcal{S}, \Xi \Vdash \mathsf{E}\,\phi$ if and only $\mathcal{S}, \Xi \vdash \mathsf{E}\,\phi$.*

Proof. Note that Proposition 6.3.1 is easily adapted to associated systems $\mathcal{S}^{\Pi}$ under Π-fairness. The result then follows by this modified proposition together with Lemma 6.3.7. $\square$

Chapter 7

Application: The Bakery Protocol

In this chapter we will illustrate the use of our proof system by proving some properties of Leslie Lamport's Bakery Algorithm for mutual exclusion [Lam74]. The algorithm is based on the idea of a ticket machine, where people entering a (big) bakery draw a ticket with a number on it that indicates their turn to buy their Sunday morning croissants.

We will state and prove the properties of mutual exclusion (mutually exclusive access of the clients to the croissants), accessibility (eventual access, once having a ticket) and possible unboundedness (the possibility of unbounded growth of the ticket numbers) of this algorithm.

7.1 Program Specification

We will consider here a version of the algorithm with two processes competing for access to their respective critical sections. The programs for the two processes P_1 and P_2 are given in graphical form in Figure 7.1.

Each process i has two variables: a control variable π_i, ranging over three control locations, called N_i (non-critical section), T_i (trying section) and C_i (critical section), and a natural number data variable y_i, indicating the ticket number. There are basically three possible actions in each process, one corresponding to each control location. Transition t_i draws a ticket by setting its own number y_i to the number of the other process incremented by one, while moving from the non-critical to the trying section. Transition e_i enters the critical section of process i, if allowed to do so by the ticket number y_i, that is, the other process' number is zero or greater than y_i. Finally, transition l_i leaves the critical section by resetting the ticket number to zero. An additional transition i (not shown in the figure) allows idling steps at any point of a computation. These are mainly to model the activity of the processes in their respective critical and non-critical sections.

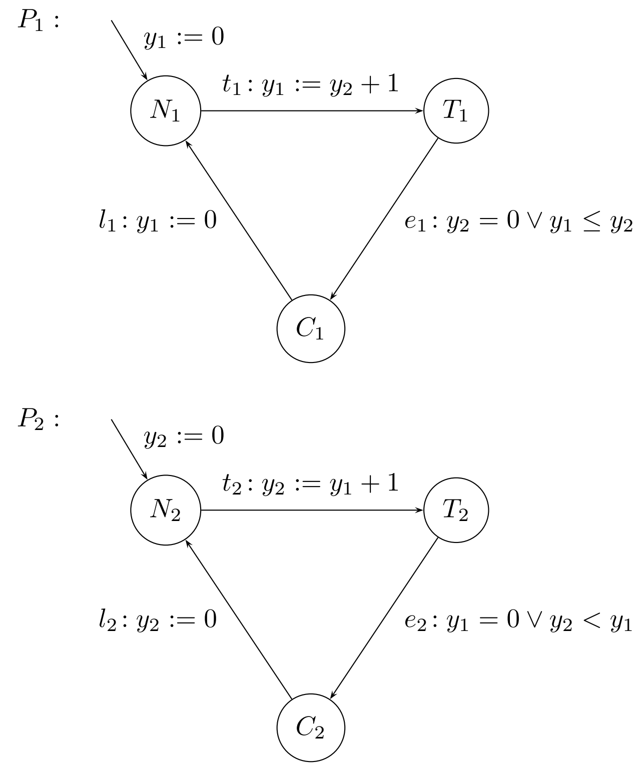

Figure 7.1: Graphical specification of the bakery protocol transitions

The formal specification of the algorithm, system $\mathcal{S}_{bak}$, is displayed in Figure 7.2. For the sake of brevity we identify control locations L_i with the assertion $\pi_i = L_i$ in formulas and just write L_i for $\pi_i = L_i$ and L'_i for $\pi'_i = L_i$, where $L_i \in \{N_i, T_i, C_i\}$. The program starts in a state, where both processes are in their respective non-critical sections N_i and the initial ticket numbers are zero. The fairness constraint W declares the two sets $\{e_1, l_1\}$ and $\{e_2, l_2\}$ as weakly fair. This means that each process can neither indefinitely delay entering its critical section, if it is continually possible to do so, nor stay in its critical section forever (t_i is always enabled in process i's critical section).

$$
\begin{aligned}
X &\stackrel{\text{def}}{=} \{\pi_1, \pi_2, y_1, y_2\} \\[1em]
\Lambda &\stackrel{\text{def}}{=} \{i, t_1, e_1, l_1, t_2, e_2, l_2\} \\[1em]
\rho_i &\stackrel{\text{def}}{=} \mathsf{pres}(\pi_1, \pi_2, y_1, y_2) \\
\rho_{t_1} &\stackrel{\text{def}}{=} N_1 \wedge T_1' \wedge y_1' = y_2 + 1 \wedge \mathsf{pres}(\pi_2, y_2) \\
\rho_{e_1} &\stackrel{\text{def}}{=} T_1 \wedge C_1' \wedge (y_2 = 0 \vee y_1 \leq y_2) \wedge \mathsf{pres}(\pi_2, y_1, y_2) \\
\rho_{l_1} &\stackrel{\text{def}}{=} C_1 \wedge N_1' \wedge y_1' = 0 \wedge \mathsf{pres}(\pi_2, y_2) \\
\rho_{t_2} &\stackrel{\text{def}}{=} N_2 \wedge T_2' \wedge y_2' = y_1 + 1 \wedge \mathsf{pres}(\pi_1, y_1) \\
\rho_{e_2} &\stackrel{\text{def}}{=} T_2 \wedge C_2' \wedge (y_1 = 0 \vee y_2 < y_1) \wedge \mathsf{pres}(\pi_1, y_1, y_2) \\
\rho_{l_2} &\stackrel{\text{def}}{=} C_2 \wedge N_2' \wedge y_2' = 0 \wedge \mathsf{pres}(\pi_1, y_1) \\[1em]
\Theta &\stackrel{\text{def}}{=} N_1 \wedge N_2 \wedge y_1 = 0 \wedge y_2 = 0 \\[1em]
\mathcal{F} &\stackrel{\text{def}}{=} (P, W, \varnothing) \\
P &\stackrel{\text{def}}{=} \{\{i, t_1, t_2\}, \{e_1, l_1\}, \{e_2, l_2\}\} \\
W &\stackrel{\text{def}}{=} \{\{e_1, l_1\}, \{e_2, l_2\}\}
\end{aligned}
$$

Figure 7.2: The system specification $\mathcal{S}_{bak} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$, where the assertion $\mathsf{pres}(\overline{x}) \stackrel{\text{def}}{=} \overline{x}' = \overline{x}$ describes the variables that are preserved.

7.2 Property Specification

We show that the bakery algorithm satisfies the following properties:

MUX $\phi_{mux} \stackrel{\text{def}}{=} \mathsf{A\,G}(\neg C_1 \vee \neg C_2)$

mutual exclusion; the two processes are never in their respective critical sections at the same time

ACC $\phi^i_{acc} \stackrel{\text{def}}{=} \mathsf{A\,G}(T_i \rightarrow \mathsf{F}\,C_i)$

accessibility; whenever a process tries to access its critical section then it will eventually succeed

UNB $\phi^i_{unb} \stackrel{\text{def}}{=} \mathsf{A\,G\,E\,F}(y_i \geq B)$

unboundedness; from any point in a computation there is a continuation such that variable y_i grows beyond bound B; the parameter B is a fixed, but arbitrary natural number, so this means that y_i may grow without bound

While the first two are essential properties that should be satisfied by any mutual exclusion algorithm, the third property is a particularity of the bakery algorithm. It is the possibility of unbounded growth of the ticket variables makes it an infinite state system.

7.3 Verification of Mutual Exclusion

For the verification of the mutual exclusion property

$$\phi_{mux} = \mathsf{A\,G}(\neg C_1 \vee \neg C_2)$$

we propose two different approaches, the first based on a generic proof structure for invariance and the second on a more refined style of proof structure taking the structure of the system into account.

7.3.1 A Generic Proof of Invariance

Figure 7.3 shows the generic proof structure Π_{INV} for some given system $\mathcal{S} = (X, \Sigma, \{\rho_\lambda \mid \lambda \in \Lambda\}, \Theta, \mathcal{F})$ and invariance property $\mathsf{A\,G}\,p$ (with p an assertion). The verification conditions generated by this proof structure are:

I1.	$\Theta \rightarrow \psi$	from $\mathsf{A}(sp)$at γ_0
I2.	$\psi \rightarrow p$	from $\mathsf{A}(ax)$ at γ_3
I3.	$\{\psi\}\,\Lambda\,\{\psi\}$	from $\mathsf{A}(\mathsf{X})$ at γ_2

The instantiation of this proof structure requires that we find an inductive assertion ψ (I3), strengthening p (I2) and implied by the initial condition Θ (I1). As there is no U-subformula in $\mathsf{A\,G}\,p$ and no anti-axiom in the proof structure, it is successful (Proposition 4.2.21).

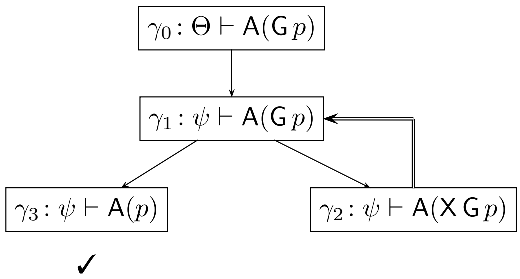

Figure 7.3: Generic LTL proof structure Π_{INV} for invariance properties.

Note that I1-I3 are exactly the premises of the general invariance rule INV of [MP91, MP95]. Rule INV is shown to be sound and relatively complete in [MP91]. This means that any invariance can be proved using proof structure Π_{INV}. The generic proof structure Π_{INV} is necessarily very coarse in the sense that it does not reflect the structure of the system at hand. For our concrete example, we choose a different style of proof structure exhibiting some of the (abstract) structure of a system.

7.3.2 A Refined Style of Invariance Proofs

A more detailed proof structure for $\mathcal{S}_{bak} \vdash \phi_{mux}$ is shown in Figures 7.4 and 7.5, where the intermediate assertions ψ_1,ψ_2 and ψ_3 are defined by

$$\begin{aligned}
\psi_1 &\stackrel{\text{def}}{=} N_1 \wedge N_2 \\
\psi_2 &\stackrel{\text{def}}{=} \neg N_1 \wedge \neg C_2 \wedge (y_2 = 0 \vee y_1 \leq y_2) \\
\psi_3 &\stackrel{\text{def}}{=} \neg C_1 \wedge \neg N_2 \wedge (y_1 = 0 \vee y_2 < y_1)
\end{aligned}$$

The construction starts as with proof structure Π_{INV} above by an application of Rule $\mathsf{A}(wk)$ at the root sequent γ_5, yielding sequent γ_4 generalising the statement to be proven. This rule application generates the side condition

$$\Theta \rightarrow \psi_1 \vee \psi_2 \vee \psi_3$$

that is, I1 above where $\psi \stackrel{\text{def}}{=} \psi_1 \vee \psi_2 \vee \psi_3$. As the initial condition Θ implies N_1 and N_2 this is clearly a valid assertion.

At γ_4 we apply Rule $\mathsf{A}(sp)$ in order to split the cases, one for each ψ_i. Each of these cases is represented in Figure 7.4 by a "macro node" labeled ψ_i. The internal structure of macro node ψ_i is displayed in Figure 7.5. All incoming edges of node ψ_i in Figure 7.4 in fact point to sequent γ_0^i in Figure 7.5. At this sequent Rule $\mathsf{A}(\mathsf{G})$ is applied, yielding sequents γ_1^i and γ_2^i. In order to show that the latter is an axiom, the validity of the assertion

$$\psi_i \rightarrow \neg C_1 \vee \neg C_2$$

has to be shown. Clearly, these assertions are valid for all $1 \leq i \leq 3$. Edges leaving the macro node ψ_i in Figure 7.4 are in fact leaving sequent γ_1^i in Figure 7.5, where the derived Rule $\mathsf{A}(\mathsf{X})'$ (creating multiple successor nodes) is applied, leaving us with the Hoare triple

$$\{\psi_i\}\ \Lambda\ \{\psi_1 \vee \psi_2 \vee \psi_2\}$$

to be discharged. This assertion is not valid for all $1 \leq i \leq 3$ as it stands. We need the help of an additional (inductive) invariant J_0, defined by

$$J_0 \stackrel{\text{def}}{=} (N_1 \leftrightarrow y_1 = 0) \wedge (N_2 \leftrightarrow y_2 = 0)$$

It is easy to see (and prove) that J_0 is indeed an invariant of $\mathcal{S}_{bak}$. We use J_0 to strengthen the left-hand side of our Hoare triples, turning them into

$$\{J_0 \wedge \psi_i\}\ \Lambda\ \{\psi_1 \vee \psi_2 \vee \psi_2\}$$

for $1 \leq i \leq 3$. Recall from Section 5.7 that it is safe to use previously proven invariants to strengthen the premise of a Hoare triple in this way. A little calculation shows that all of these triples are indeed valid assertions.

Comparing with Invariance Diagrams

It is interesting to observe that the complete graph G consisting of the macro nodes ψ_1,ψ_2 and ψ_3 (here considered as "black boxes") and the edges connecting them is an invariance verification diagram as presented by Manna and Pnueli in [MP94] (see Section 2.6.4).

It is easy to see that any invariance diagram D for $\mathcal{S}$ and p can be cast into a (partial) proof structure by interpreting each node ψ_i of D as the

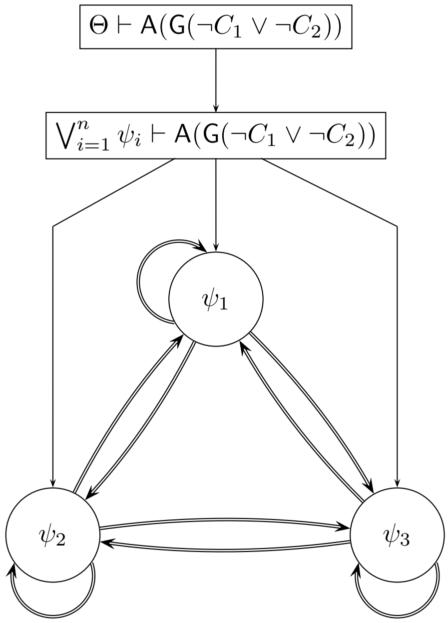

Figure 7.4: Proof structure Π_{mux} for $\mathcal{S}_{bak} \vdash \phi_{mux}$

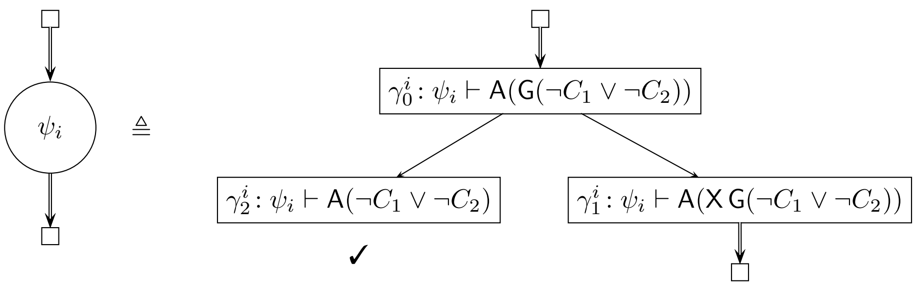

Figure 7.5: "Macro nodes" ψ_i for proof structure Π_{mux}

"macro node" displayed in Figure 7.5 and completed to a full proof structure Π_D for $\mathcal{S} \vdash \mathsf{A}(\mathsf{G}\,p)$ by adding two additional sequents in the way indicated by our example in Figure 7.4. The verification conditions generated by Π_D are exactly the Hoare triples generated by D *plus* the additional implications (1) $\Theta \to \bigvee_{i=1}^{n} \psi_i$, and (2) $\psi_i \to p$ for $1 \leq i \leq n$, which are also part of the invariance diagram rule (tough not represented in D).

Finally note that the verification conditions to be discharged for such a proof structure Π_D (or, equivalently the conditions associated with diagram D) are essentially the same as the ones generated by Π_{INV} with $\psi \stackrel{\text{def}}{=} \bigvee_{i=1}^{n} \psi_i$. However, as the graph D need not be complete (as it is accidentally the case in our example), it represents generally a more precise abstract view of the system $\mathcal{S}$ under study than it is the case for Π_{INV}.

7.4 Verification of Accessibility

We prove the accessibility property for Process 1, that is,

$$\phi^1_{acc} = \mathsf{A}\,\mathsf{G}(T_1 \to \mathsf{F}\,C_1).$$

For this purpose we propose proof structure Π_{acc} for $\mathcal{S}_{bak} \vdash \phi^1_{acc}$, displayed in Figure 7.6. Edges leaving $\mathsf{A}(\mathsf{X})$-sequents are drawn with double lines. Some of these are labeled to emphasise a particular underlying system transition.

The upper half of the proof structure (sequents $\gamma_0, \ldots, \gamma_6$) is rather uninteresting: we first generalise the root sequent yielding γ_1 and then split cases at γ_4. All verification conditions for this part hold trivially. Its primary purpose is to transform the root sequent $\gamma_0 : \Theta \vdash \mathsf{A}(\mathsf{G}(\neg T_1 \vee \mathsf{F}\,C_1))$ into the sequent $\gamma_7 : T_1 \vdash \mathsf{A}(\mathsf{F}\,C_1)$[1]. Here the proof starts to become interesting.

The key step in the construction of Π_{acc} is the application of Rule $\mathsf{A}(sp)$ at sequent γ_7 with the following choice of left-hand side assertions for the three successor sequents (writing p_i for p_{γ_i}):

$$\begin{aligned} p_8 &\stackrel{\text{def}}{=} T_1 \wedge \neg en(e_1) \wedge T_2 \\ p_{11} &\stackrel{\text{def}}{=} T_1 \wedge \neg en(e_1) \wedge C_2 \\ p_{14} &\stackrel{\text{def}}{=} en(e_1) \end{aligned}$$

The verification condition to discharge is $T_1 \to p_8 \vee p_{11} \vee p_{14}$. This does not hold as it stands, but we can again safely call on the help of invariant J_0 and

[1] Of course, we could have started from the (root) sequent $T_1 \vdash \mathsf{A}(\mathsf{F}\,C_1)$ right away, since $T_1 \models \mathsf{A}(\mathsf{F}\,C_1)$ implies $\Theta \models \mathsf{A}(\mathsf{G}(\neg T_1 \vee \mathsf{F}\,C_1))$, but we have chosen to present here a complete example without "shortcuts".

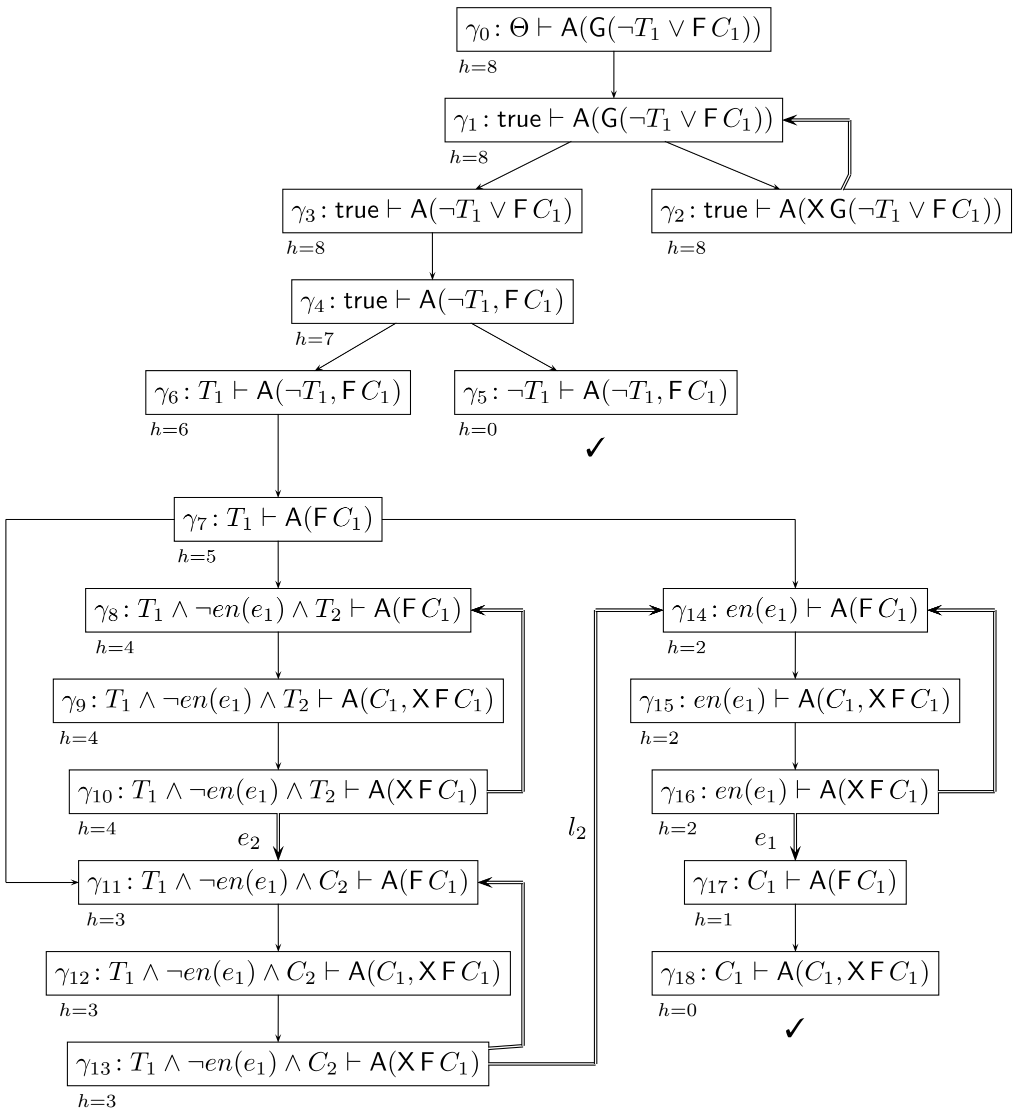

Figure 7.6: Proof structure Π_{acc} for $\mathcal{S}_{bak} \vdash \phi^1_{acc}$

show that

$$J_0 \wedge T_1 \to p_8 \vee p_{11} \vee p_{14}$$

is valid. This is indeed the case, since T_1 implies $(T_1 \vee \neg en(e_1)) \vee en(e_1)$ and $\neg en(T_1)$ implies $y_2 > 0$, hence $T_2 \vee C_2$ using J_0.

Note that at sequents γ_{10},γ_{13} and γ_{16} we have in fact applied the derived Rule $\mathsf{A(X)}'$ to create two successor nodes in each case. At γ_{10} the side condition is

$$\{T_1 \wedge \neg en(e_1) \wedge T_2\} \ \Lambda \ \{(T_1 \wedge \neg en(e_1) \wedge T_2) \vee (T_1 \wedge \neg en(e_1) \wedge C_2)\}$$

that is $\{p_8\} \ \Lambda \ \{p_8 \vee p_{11}\}$, which is easily seen to be valid by observing that the idle transition i obviously preserves p_8 and that transition e_2 is enabled in p_8-states and leads to a p_{11}-state. All other transitions are disabled in p_8-states. A similar argument shows that the side condition for the application of Rule $\mathsf{A(X)}'$ at γ_{13}, namely $\{p_{11}\} \ \Lambda \ \{p_{11} \vee p_{14}\}$ holds, this time with l_2 leading from p_{11} to p_{14}. For sequent γ_{16} the side condition is

$$\{en(e_1)\} \ \Lambda \ \{en(e_1) \vee C_1\}$$

It is clear that transition e_1 leads to C_1. Transitions i, t_2 and l_2 preserve the enabledness of e_1. All other transitions are disabled if e_1 is enabled and thus lead trivially to $en(e_1) \vee C_1$. In particular, consider transition e_2. We have to show that $en(e_1) \wedge \rho_{e_2} \to en(e_1) \vee C_1$. But $en(e_1) \wedge \rho_{e_2}$ implies that $y_1 = 0$ or $y_2 = 0$, but also that neither N_1 nor N_1 holds, contradicting invariant J_0.

7.4.1 Proving Success for Π_{acc}

Preparing the application of Rule $\mathsf{A}(\mathrm{S})_{fair}$ to prove Π_{acc} successful, we name the weakly fair sets by $\Lambda_{w_1} \stackrel{\text{def}}{=} \{e_1, l_1\}$ and $\Lambda_{w_2} \stackrel{\text{def}}{=} \{e_2, l_2\}$ and note that the only V-subformula in ϕ^1_{acc} is the G-subformla, so we refer to it as G for short. As there is no strong fairness constraint, we can take $I \stackrel{\text{def}}{=} \{\bullet, \mathsf{G}, w_1, w_2\}$ for the set indexing the required auxiliary assertions. We set $\gamma_{19} \stackrel{\text{def}}{=} \top$ and then define the coding $\lceil\cdot\rceil$ as usual by $\lceil\gamma_i\rceil = i$ for $0 \leq i \leq 19$.

Choosing the auxiliary quantities.

We divide the (pseudo-) sequents into three groups

$$\begin{aligned} \Gamma_{\mathsf{G}} &\stackrel{\text{def}}{=} \{\gamma_0, \ldots, \gamma_7\} \cup \{\gamma_{17}, \gamma_{18}, \top\} \\ \Gamma_{w_1} &\stackrel{\text{def}}{=} \{\gamma_{14}, \gamma_{15}, \gamma_{16}\} \\ \Gamma_{w_2} &\stackrel{\text{def}}{=} \{\gamma_8, \ldots, \gamma_{13}\} \end{aligned}$$

and then define the auxiliary assertions by $\beta_\bullet \stackrel{\text{def}}{=} \mathsf{false}$ and, for $\sharp \in \{\mathsf{G}, w_1, w_2\}$,

$$\beta_\sharp \stackrel{\text{def}}{=} \bigvee_{\gamma \in \Gamma_\sharp} \widehat{p_\gamma}$$

We also need a ranking, which we define on extended states by

$$\delta(\pi_1, \pi_2, y_1, y_2, K) \stackrel{\text{def}}{=} h(K)$$

where $h(\lceil\top\rceil) = 0$ and $h(\lceil\gamma\rceil)$ is defined for sequents $\gamma \in \Gamma$ as indicated in Figure 7.6.

What is the intuition leading to this choice? Note that there are four non-trivial strongly connected subgraphs in Π_{acc}, namely $S_1 = \{\gamma_1, \gamma_2\}$, $S_2 = \{\gamma_8, \gamma_9, \gamma_{10}\}$, $S_3 = \{\gamma_{11}, \gamma_{12}, \gamma_{13}\}$ and $S_4 = \{\gamma_{14}, \gamma_{15}, \gamma_{16}\}$. Consider the three cases, where on some trail ϑ control K remains from some point on caught in

- $\lceil S_1\rceil$: Then ϑ is successful (since $t(K) \in \lceil S_1\rceil$ implies $t \models K_\mathsf{G}$ for all $t \in \Sigma^{\Pi_{acc}}$),
- $\lceil S_2 \cup S_3\rceil$: Then ϑ is Π-unfair w.r.t. $\Lambda_{w_2}^{\Pi_{acc}}$ (since the only transitions that can be taken along edges (γ_{10}, γ_8) and $(\gamma_{13}, \gamma_{11})$ are $(\gamma_{10}, i, \gamma_8)$ and $(\gamma_{13}, i, \gamma_{11})$, respectively, while for $\gamma \in S_2 \cup S_3$, p_γ implies that Λ_{w_2} is enabled, hence $en^\Pi(\Lambda_{w_2}^{\Pi_{acc}})$, and
- $\lceil S_4\rceil$: Then ϑ is Π-unfair w.r.t. $\Lambda_{w_1}^{\Pi_{acc}}$ (since neither $(\gamma_{16}, e_1, \gamma_{14})$ nor $(\gamma_{16}, l_1, \gamma_{14})$ can be taken along the edge $(\gamma_{16}, \gamma_{14})$, but, for $\gamma \in S_4$, p_γ implies enabledness of Λ_{w_1}, hence $en^\Pi(\Lambda_{w_1}^{\Pi_{acc}})$.

This means that there are no "harmful" cycles in the proof structure Π_{acc} (for which we would have to show that no computation can follow them).

Recall from the discussion of "mechanics" of Rule $\mathsf{A}(\mathrm{S})_{fair}$ that if its premises hold then some auxiliary assertion $\beta_\sharp$ will eventually become stable on a trail unless $K_\top$ is reached. If this assertion is β_G then by premise A3 the trail is successful and if it is β_{w_1}or β_{w_2} then by A4 and A5 the trail is Π-unfair w.r.t. $\Lambda_{w_1}^{\Pi_{acc}}$ or $\Lambda_{w_2}^{\Pi_{acc}}$, respectively. These roles of the $\beta_\sharp$ perfectly matches our observations above and suggests that we choose

- β_G such that it is implied by $\widehat{p_{\gamma_1}}$ and by $\widehat{p_{\gamma_2}}$,
- β_{w_2} such that it is implied by each $\widehat{p_\gamma}$ for $\gamma \in S_2 \cup S_3$, and
- β_{w_1} such that it is implied by each $\widehat{p_\gamma}$ for $\gamma \in S_4$.

Since $\beta = \beta_{\mathsf{G}} \vee \beta_{w_1} \vee \beta_{w_2}$ has to be invariant along trails unless $K_{\top}$ is reached, we have to make a choice as where to put the remaining control points of $\mathcal{S}^{\Pi_{acc}}$, namely γ_i for $i \in \{0, 3, \ldots, 7, , 17, 18\}$ and $\top$. We add them to Γ_{G} yielding the definition of β_{G} given above. For β_{w_1} and β_{w_2} note that $\Gamma_{w_1} = S_4$ and $\Gamma_{w_2} = S_2 \cup S_3$. We have set $\beta_{\bullet}$ to false, since it is not needed.

Finding the right ranking is then easy, as with our choice of the auxiliary assertions $\beta_{\sharp}$ it can remain constant within each of the strongly connected subgraphs $S_1, \ldots, S_4$. This is because for a state $t \in \Sigma^{\Pi_{acc}}$ we have

- $t(K) \in S_1$ implies $t \models K_{\mathsf{G}}$,
- $\Lambda^{\Pi_{acc}}_{w_2}$ is not taken along an edge in S_2 or S_3, and
- $\Lambda^{\Pi_{acc}}_{w_2}$ is not taken along an edge in S_4.

Transitions along the remaining edges bring us closer to an axiom and the ranking can be easily defined in a way as to decrease along these, if necessary.

Verification of premises A1-A6 of Rule $\mathsf{A}(\boldsymbol{S})_{fair}$.

The initial condition $\Theta^{\Pi_{acc}} = \widehat{p_{\gamma_0}}$ certainly implies β_{G}, hence β, so condition A1 holds. Also, as already indicated above, β_{w_1} implies $en^{\Pi}(\Lambda^{\Pi_{acc}}_{w_1})$ and β_{w_2} implies $en^{\Pi}(\Lambda^{\Pi_{acc}}_{w_2})$, so condition A5 holds as well. Premise A6 holds trivially as the strong fairness constraint is empty. It remains to show A2-A4. Note that by our particular choice of the auxiliary assertions, β is trivially preserved across Λ^{Π}-transitions, as $\rho^{\Pi_{acc}}_{(\gamma,\lambda,\gamma')} \to \widehat{p_{\gamma}}'$ for any $(\gamma, \lambda, \gamma') \in \Lambda^{\Pi_{acc}}$. So in checking A2-A4, we are primarily concerned with the ranking.

A2 A look at Figure 7.6 confirms that the ranking never increases along an edge in the proof structure and hence along a $\Lambda^{\Pi_{acc}}$-transition. The ranking decreases as required along edges where the $\beta_{\sharp}$-mode changes and that are not leading to $\top$. Namely, these are the edges leaving γ_7 and γ_{16} as well as edge $(\gamma_{13}, \gamma_{14})$.

A3 We have to show that the ranking δ decreases along all $\Lambda^{\Pi_{acc}}$-transitions from β_{G}-states not satisfying K_{G} and not reaching $K_{\top}$. The concerned sequents are $\gamma_3, \ldots, \gamma_7$ in the upper part and γ_{17} in the lower part of the proof structure. Clearly, δ decreases along all edges leaving these sequents.

A4 From a β_{w_1}-state $(\gamma_{16}, e_1, \gamma_{17})$ is the only $\Lambda^{\Pi_{acc}}_{w_1}$-transition that can be taken and it decreases δ from 2 to 1. There are two $\Lambda^{\Pi_{acc}}_{w_1}$-transitions that can be taken from β_{w_2}-states, namely $(\gamma_{10}, e_2, \gamma_{11})$ and $(\gamma_{13}, l_2, \gamma_{14})$. These both decrease δ as a glance at Figure 7.6 shows.

This completes the verification of accessibility for Process 1.

7.5 Verification of Unboundedness

In this section, we show the property of unboundedness for Process 1:

$$\phi^1_{unb} \stackrel{\text{def}}{=} \mathsf{A}\,\mathsf{G}\,\mathsf{E}\,\mathsf{F}(y_1 \geq B)$$

A CTL* proof structure for $\mathcal{S}_{bak}$ and this property is composed of the LTL proof structure Π_{unb1} shown in Figures 7.7 and the ELL proof structure Π_{unb2} of 7.8.

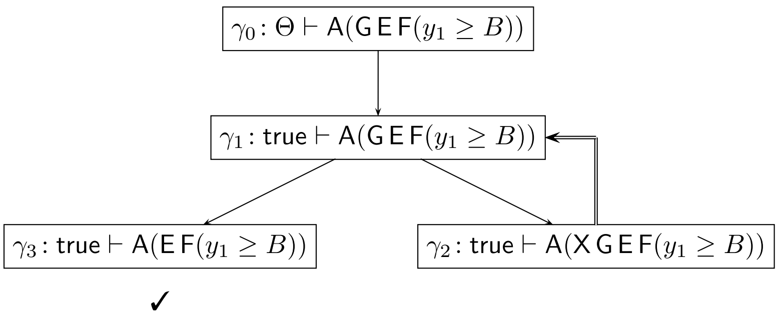

Figure 7.7: Proof structure Π_{unb1} for $\mathcal{S}_{bak} \vdash \phi^1_{unb}$

The LTL proof structure Π_{unb1} reduces the root sequent $\Theta \vdash \mathsf{A}(\mathsf{G}\,\mathsf{E}\,\mathsf{F}(y_1 \geq B))$ to the axiom $\mathsf{true} \vdash \mathsf{A}(\mathsf{E}\,\mathsf{F}(y_1 \geq B))$. The justification of the latter requires the construction of a proof for $\mathcal{S}_{bak}, \mathsf{true} \models \mathsf{E}(\mathsf{F}(y_1 \geq B)))$. The other side conditions of Π_{unb1} are trivially satisfied.

Proof structure Π_{unb2} for $\mathcal{S}_{bak}$ and $\mathsf{true} \vdash \mathsf{E}(\mathsf{F}(y_1 \geq B))$ is to show that from any state there is a computation where y_1 grows beyond bound B.

7.5.1 Checking the Side Conditions for Π_{unb2}

We use the three disjuncts of invariant $\psi = \psi_1 \vee \psi_2 \vee \psi_3$ from Section 7.3 to split cases at the root sequent γ_0 of Π_{unb2}. The corresponding side condition, the left-hand side of which is strengthened with ψ, reads

$$\psi \rightarrow (\psi_1 \wedge y_1 < B) \vee (\psi_2 \wedge y_1 < B) \vee (\psi_2 \wedge y_1 < B) \vee y_1 \geq B$$

which is clearly valid. Note that at sequents γ_1,γ_3 and γ_5 the derived Rule $\mathsf{E}(\mathsf{F}_r)$ is applied and at γ_7 its companion, Rule $\mathsf{E}(\mathsf{F}_l)$, is applied. Sequent γ_8 is an ELL axiom.

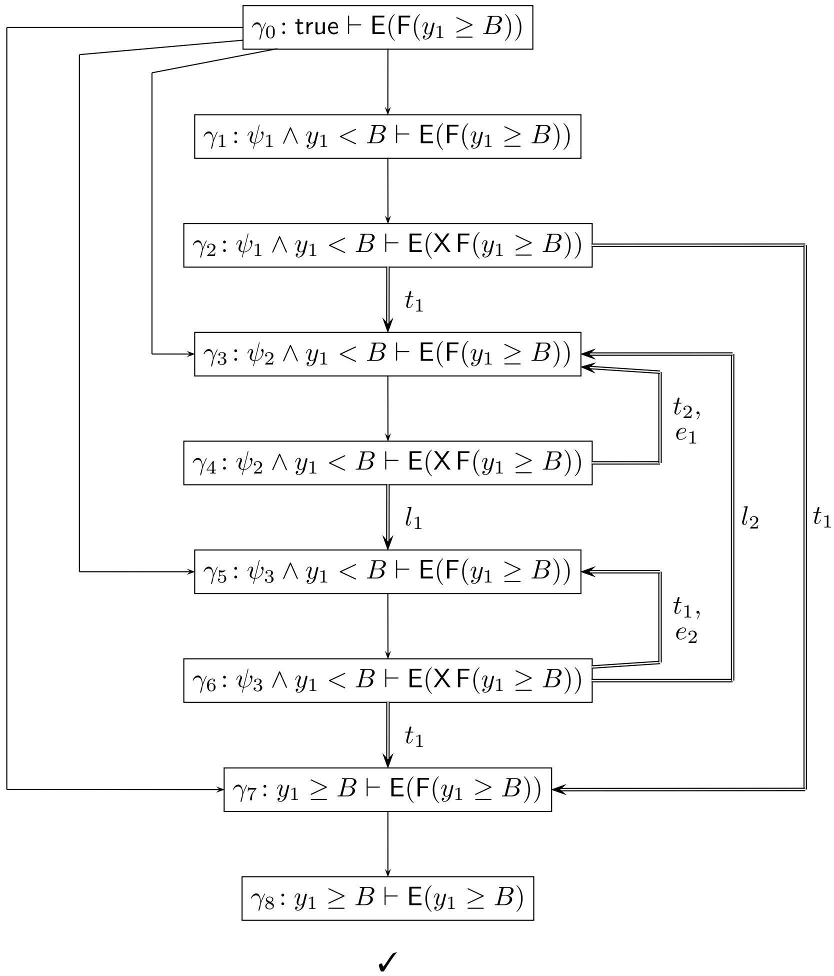

Figure 7.8: Proof structure Π_{unb2} for $\mathcal{S}_{bak}, \mathsf{true} \vdash \mathsf{E}(\mathsf{F}(y_1 \geq B))$

At the remaining sequents γ_2, γ_4 and γ_6 derived Rule $\mathsf{E(X)}'$ is applied to create two or three successor sequents in each case. We pick γ_6 and take a closer look at its side condition

$$\psi_3 \wedge y_1 < B \rightarrow \langle \Lambda \rangle \, ((\psi_2 \wedge (y_1 < B)) \vee (\psi_3 \wedge (y_1 < B)) \vee (y_1 \geq B))$$

Recall that ψ_3 was defined by $\neg C_1 \wedge \neg N_2 \wedge (y_1 = 0 \vee y_2 < y_1)$. Its first part, the assertion $\neg C_1 \wedge \neg N_2$, is equivalent to

$$(N_1 \wedge \neg N_2) \vee (T_1 \wedge T_2) \vee (T_1 \wedge C_2).$$

We proceed by case analysis according to the latter assertion to show that

1. $N_1 \wedge \neg N_2 \wedge (y_1 = 0 \vee y_2 < y_1) \wedge (y_1 < B) \rightarrow \langle t_1 \rangle \, \psi_3$
2. $T_1 \wedge T_2 \wedge (y_1 = 0 \vee y_2 < y_1) \wedge (y_1 < B) \rightarrow \langle e_2 \rangle \, (\psi_3 \wedge (y_1 < B))$
3. $T_1 \wedge C_2 \wedge (y_1 = 0 \vee y_2 < y_1) \wedge (y_1 < B) \rightarrow \langle l_2 \rangle \, (\psi_2 \wedge (y_1 < B))$

All of these can easily shown to be valid with a little calculation. As an informal justification note that the transition to be taken is enabled in each case. The fact that t_1 and e_2 preserve ψ_3 in (1,2) and that l_1 leads from ψ_3 to ψ_2 has already been established in Section 7.3. Furthermore, observe that e_2 and l_2 do not modify y_1, so its value certainly stays below B if this was the case before the respective transition. For (1) note that $\langle t_1 \rangle \, \psi_3$ implies $\langle t_1 \rangle \, ((\psi_3 \wedge y_1 < B) \vee (y_1 \geq B))$.

7.5.2 Proving Success for Π_{unb1} and Π_{unb2}

To show that proof structure Π_{unb1} is successful, all we need to do is to prove that its sequent γ_3 is indeed an axiom, in other words, that proof structure Π_{unb2} is successful. To this end, we can directly apply Rule $\mathsf{E}(\mathrm{S})_{wuf}$ with Ξ instantiated to $\Theta^{\Pi_{unb2}}$ and partition $(\varnothing, \varnothing)$, as there are no strong fairness constraints. The assertion $dis^{\Pi}(\varnothing)$, being an empty conjunction, then boils down to true.

We set $\gamma_9 \stackrel{\text{def}}{=} \top$ and define as usual $\lceil \gamma_i \rceil = i$. The formula $\mathsf{F}(y_1 \geq B)$ is the only U-subformula appearing in this proof structure, so we refer to it as F for short. We then have $\Psi_{\mathsf{E}} = \{\bot, \mathsf{F}\}$ and $W^{\Pi_{unb2}} = \{\Lambda_{w_1}^{\Pi_{unb2}}, \Lambda_{w_2}^{\Pi_{unb2}}\}$. We associate index 1 with $\bot$, 2 with F, 3 with $\Lambda_{w_1}^{\Pi_{unb2}}$ and 4 with $\Lambda_{w_2}^{\Pi_{unb2}}$. Then $K_1 \stackrel{\text{def}}{=} K_\bot \equiv \mathsf{false}$ and $K_2 \stackrel{\text{def}}{=} K_{\mathsf{F}} \equiv K \in \{0, \ldots, 7\}$, so

$$\begin{array}{lclcl} \neg K_1 & \equiv & \neg K_\bot & \equiv & \mathsf{true} \\ \neg K_2 & \equiv & \neg K_{\mathsf{F}} & \equiv & K \in \{8, 9\} \end{array}$$

Auxiliary assertions and rankings

We choose the auxiliary assertions $\alpha_0, \dots, \alpha_4$ as follows:

$$\begin{array}{lllllll} \alpha_0 & \stackrel{\text{def}}{=} & \mathsf{false} & \quad & \alpha_1 & \stackrel{\text{def}}{=} & \bigvee_{i\in[0,6]} \widehat{p_{\gamma_i}} \\ \alpha_2 & \stackrel{\text{def}}{=} & \widehat{p_{\gamma_7}} & & \alpha_3 & \stackrel{\text{def}}{=} & \widehat{p_{\gamma_8}} \\ \alpha_4 & \stackrel{\text{def}}{=} & \mathsf{false} & & & & \end{array}$$

The only non-trivial ranking function is δ_1, which we define by

$$\delta_1(\pi_1, \pi_2, y_1, y_2, K) \stackrel{\text{def}}{=} (B \dot{-} \max(y_1, y_2), (\pi_1, \pi_2), h(K))$$

ordered lexicographically, where $\dot{-}$ is natural number subtraction (with $b \dot{-} a \stackrel{\text{def}}{=} 0$ for $b < a$) and

$$h(K) \stackrel{\text{def}}{=} \begin{cases} 2 & \text{if } K = 0 \\ 1 & \text{if } K \text{ is odd} \\ 0 & \text{otherwise} \end{cases}$$

The ordering relation on locations is defined for $i = 1, 2$ by $N_i < C_i < T_i$, and the pairs (π_1, π_2) ordered point-wise. Denote the lexicographic ordering on the range of δ_1 by $\prec$. Clearly, its inverse $\succ$ is well-founded. The other ranking functions δ_0,δ_2,δ_3 and δ_4 are trivial, defined, e.g., by $\delta_i \stackrel{\text{def}}{=} 0$.

The idea is that any witnessing trail that can be constructed according to E4-E7 reaches $\top$ at some point. This is the reason why we can "afford" to set α_4 to false.

Verification of premises E1-E7 of Rule $\mathsf{E}(\boldsymbol{S})_{wuf}$.

Premise E1 is equivalent to $\widehat{p_{\gamma_0}} \rightarrow \bigvee_{i\in[0,8]} \widehat{p_{\gamma_i}}$ and is certainly satisfied. Premises E2, E3 and E4 of Rule $\mathsf{E}(\mathrm{S})_{wuf}$ are trivially valid. This is also the case for E7, since there is no strong fairness constraint. It remains to show premises E5 and E6. The latter is trivial for $j = 4$. For $j = 3$, it is

$$\begin{array}{ll} & \{\alpha_3\} \left\langle \Lambda^{\Pi_{unb2}} \right\rangle \{K_\top \vee \dots\} \\ \equiv & \{K = 8 \wedge y_1 \geq B\} \left\langle \Lambda^{\Pi_{unb2}} \right\rangle \{K = 9 \vee \dots\} \end{array}$$

which is valid. It is for instance implied by $\{K = 8\} \langle(\gamma_8, i, \gamma_9)\rangle \{K = 9\}$.

Premise E5 for $i = 2$ follows from the validity of the following assertion:

$$\begin{array}{ll} & \{\alpha_2\} \left\langle \Lambda^{\Pi_{unb2}} \right\rangle \{\alpha_3 \wedge \neg K_2\} \\ \equiv & \{K = 7 \wedge y_1 \geq B\} \langle(\gamma_7, =, \gamma_8)\rangle \{K = 8 \wedge y_1 \geq B\} \end{array}$$

Most of the work is in establishing premise E5 for $i = 1$. After some simplification, it boils down to

$$\{\alpha_1 \wedge \delta_1 = (u,v,w)\} \left\langle \Lambda^{\Pi_{unb2}} \right\rangle \{K_\top \vee (\alpha_1 \wedge \delta_1 \prec (u,v,w)) \vee \alpha_2\}$$

or, equivalently, for all $0 \leq k \leq 6$:

$$\{\widehat{p_{\gamma_k}} \wedge \delta_1 = (u,v,w)\} \left\langle \Lambda^{\Pi_{unb2}} \right\rangle \{(\bigvee_{i \in [0,6]} \widehat{p_{\gamma_i}} \wedge \delta_1 \prec (u,v,w)) \vee \widehat{p_{\gamma_7}}\}$$

since $K_\top$ can not be reached from $K \in [0,6]$. The results of the verification of these conditions are summarised in Table 7.1.

$\Lambda^{\Pi_{unb2}}$-transition	departing from $\widehat{p_{\gamma_k}} \wedge \ldots$	$B \dot{-} \max(y_1, y_2)$	(π_1, π_2)	$h(K)$	δ_1
$(\gamma_0, =, \gamma_1)$	$\psi_1 \wedge (y_1 < B)$	$=$	$=$	$\downarrow$	$\downarrow$
$(\gamma_0, =, \gamma_3)$	$\psi_2 \wedge (y_1 < B)$	$=$	$=$	$\downarrow$	$\downarrow$
$(\gamma_0, =, \gamma_5)$	$\psi_3 \wedge (y_1 < B)$	$=$	$=$	$\downarrow$	$\downarrow$
$(\gamma_0, =, \gamma_7)$	$y_1 \geq B$	nc	nc	nc	nc
$(\gamma_1, =, \gamma_2)$	true	$=$	$=$	$\downarrow$	$\downarrow$
$(\gamma_2, t_1, \gamma_7)$	$y_2 + 1 \geq B$	nc	nc	nc	nc
$(\gamma_2, t_1, \gamma_3)$	$y_2 + 1 < B$	$\downarrow$	nc	nc	$\downarrow$
$(\gamma_3, =, \gamma_4)$	true	$=$	$=$	$\downarrow$	$\downarrow$
$(\gamma_4, t_2, \gamma_3)$	N_2	$\downarrow$	nc	nc	$\downarrow$
$(\gamma_4, e_1, \gamma_3)$	$T_1 \wedge T_2$	$=$	$\downarrow$	nc	$\downarrow$
$(\gamma_4, l_1, \gamma_5)$	$C_1 \wedge T_2$	$=$	$\downarrow$	nc	$\downarrow$
$(\gamma_5, =, \gamma_6)$	true	$=$	$=$	$\downarrow$	$\downarrow$
$(\gamma_6, t_1, \gamma_7)$	$N_1 \wedge (y_2 + 1 \geq B)$	nc	nc	nc	nc
$(\gamma_6, t_1, \gamma_5)$	$N_1 \wedge (y_2 + 1 < B)$	$\downarrow$	nc	nc	$\downarrow$
$(\gamma_6, e_2, \gamma_5)$	$T_1 \wedge T_2$	$=$	$\downarrow$	nc	$\downarrow$
$(\gamma_6, l_2, \gamma_3)$	$T_1 \wedge C_2$	$=$	$\downarrow$	nc	$\downarrow$

Table 7.1: Behaviour of ranking δ_1 along $\Lambda^{\Pi_{unb2}}$-transitions from α_1-states – $\downarrow$ means strict decrease, $=$ remaining constant, and 'nc' "don't care"

In this table an entry for $\Lambda^{\Pi_{unb2}}$-transition $(\gamma_k, \lambda, \gamma_j)$ (where $\lambda \in \Lambda \cup \{=\}$) with formula θ in the second column should be read for $j \neq 7$ as

$$\{\widehat{p_{\gamma_k}} \wedge \theta \wedge \delta_1 = (u,v,w)\} \left\langle (\gamma_k, \lambda, \gamma_j) \right\rangle \{\delta_1 \prec (u,v,w)\}$$

and for $j = 7$ as

$$\{\widehat{p_{\gamma_k}} \wedge \theta\} \ \langle(\gamma_k, \lambda, \gamma_7)\rangle \ \{\mathsf{true}\}$$

Note that adding $\widehat{p_{\gamma_j}}$ on the right-hand side of these possibility triples is redundant, as these assertions are already part of $\rho^{\Pi_{unb2}}_{(\gamma_k,\lambda,\gamma_j)}$.

Additionally, the table indicates how the individual components of the ranking δ_1 behave along this transition. Observe that

- we need not care about the ranking for transitions leading to $\widehat{p_{\gamma_7}}$,
- for transitions of the form $(\gamma, =, \gamma')$ the first two components of the ranking obviously do not change, while the third, $h(K)$, decreases,
- for transitions of the form (γ, e_i, γ') and (γ, l_i, γ') the first component remains constant, while the second decreases (moving π_i from T_i to C_i and from C_i to N_i, respectively), and
- for transitions of the form (γ, t_i, γ') with $\gamma' \neq \gamma_7$ the first component of the ranking decreases.

We pick two cases and take a closer look at them.

1. $(\gamma_4, l_1, \gamma_5)$: The possibility triple for this case is equivalent to

$$\begin{array}{l} \{K = 4 \wedge C_1 \wedge T_2 \wedge (y_2 = 0 \vee y_1 \leq y_2) \wedge y_1 < B \wedge \delta_1 \prec (u, v, w)\} \\ \langle(\gamma_4, l_1, \gamma_5)\rangle \\ \{\delta_1 \prec (u, v, w)\} \end{array}$$

This follows from the valid assertion

$$\begin{array}{lll} J_0 \wedge K = 4 \wedge C_1 \wedge T_2 \wedge (y_2 = 0 \vee y_1 \leq y_2) \wedge y_1 < B \\ \rightarrow \ \exists \pi'_1, \pi'_2, y'_1, y'_2, K'. \\ \qquad\qquad K = 4 \wedge \psi_2 \wedge y_1 < B \\ \qquad \wedge \ \ C_1 \wedge N'_1 \wedge y'_1 = 0 \wedge \mathsf{pres}(\pi_2, y_2) \\ \qquad \wedge \ \ K' = 5 \wedge \neg C'_1 \wedge \neg N'_2 \wedge (y'_1 = 0 \wedge y'_2 < y'_1) \wedge y'_1 < B \\ \qquad \wedge \ \ \max(y'_1, y'_2) = \max(y_1, y_2) \wedge (\pi'_1, \pi'_2) < (C_1, T_2) \end{array}$$

The obvious witnessing instantiation for the primed variables is $\pi'_1 = N_1$, $\pi'_2 = T_2$, $y'_1 = 0$, $y'_2 = y_2$ and $K' = 5$. Note in particular that by the use of invariant J_0 we can deduce that $y_1 \leq y_2$ (since T_2), so $\max(y'_1, y'_2) = y'_2 = y_2 = \max(y_1, y_2)$. For the second component of the ranking we have $(\pi'_1, \pi'_2) = (N_1, T_1) < (C_1, T_2)$. Thus, the overall ranking does indeed decrease along $(\gamma_4, l_1, \gamma_5)$.

2. $(\gamma_6, t_1, \gamma_5)$: Here, the assertion to show valid is

$$\left\{ \begin{array}{rl} & K = 6 \wedge N_1 \wedge \neg N_2 \wedge (y_1 = 0 \vee y_2 < y_1) \\ \wedge & (y_1 < B) \wedge (y_2 + 1 < B) \wedge \delta_1 = (u, v, w) \end{array} \right\}$$
$$\langle(\gamma_6, t_1, \gamma_5)\rangle$$
$$\{\delta_1 \prec (u, v, w)\}$$

This one follows from

$$\begin{array}{l} J_0 \wedge K = 6 \wedge N_1 \wedge \neg N_2 \wedge (y_1 = 0 \vee y_2 < y_1) \wedge (y_1 < B) \wedge (y_2 + 1 < B) \\ \quad \rightarrow \quad \exists \pi_1', \pi_2', y_1', y_2', K'. \\ \qquad\qquad \begin{array}{rl} & K = 5 \wedge \psi_3 \wedge y_1 < B \\ \wedge & N_1 \wedge T_1' \wedge y_1' = y_2 + 1 \wedge \mathsf{pres}(\pi_2, y_2) \\ \wedge & K' = 5 \wedge \neg C_1' \wedge \neg N_2' \wedge (y_1' = 0 \wedge y_2' < y_1') \wedge y_1' < B \\ \wedge & \max(y_1, y_2) < \max(y_1', y_2') < B \end{array} \end{array}$$

which is easily seen to be valid by instantiating the primed variables in the only possible way by $\pi_1' = T_1$, $\pi_2' = \pi_2$, $y_1' = y_2 + 1$, $y_2' = y_2$ and $K' = 5$. Using invariant J_0 we deduce $y_1 = 0$ from N_1, so we have $\max(y_1, y_2) = y_2$ while $\max(y_1', y_2') = y_2 + 1 < B$. Thus, there is a strict decrease in the first component of the ranking.

This concludes our (sketch of) the proof that Π_{unb2} and hence Π_{unb1} are successful. We conclude that $\mathcal{S}_{bak} \models \phi^1_{unb}$.

Chapter 8

Conclusions and Related Work

8.1 Summary and Discussion

The aim of this thesis was to design a tableau-based proof system for the model checking of CTL* properties of infinite state fair reactive systems and to explain its soundness and completeness in terms of model checking games. In this section, we will first recapitulate how this goal was achieved and then discuss some selected issues.

Proof Structures

The present work generalises the finite state local model checking technique for CTL* proposed by Bhat, Cleaveland and Grumberg in [BCG95] to infinite-state systems equipped with a quite general type of fairness constraints. The sequent format is extended to deal with infinite sets of states described by assertions and the LTL proof rules of [BCG95] are generalised accordingly. In their work, proving $s \models \mathsf{E}\,\psi$ is reduced to showing $s \not\models \mathsf{A}\,\neg\psi$ by constructing an unsuccessful LTL proof structure. While this reduction is appropriate from an algorithmic point of view, deductive proofs call for a more direct approach and we therefore introduce a separate set of rules for ELL. Each rule system includes a Split rule implementing case analysis. A simple local condition imposes a mild restriction on its application that ensures the temporal consistency of proof structures. A proof system for CTL* is obtained from the combined rule systems by extending the terminal and predicate rules to account for path-quantified subformulas. The side conditions for path-quantified formulas involve the construction of a new LTL or ELL proof structure.

Success Criteria

As in the finite state case, the local rules serving the construction of proof structures are complemented with a global success criterion, identifying the proof structures that are acceptable as proper proofs. The success criteria for our two types of proof structures are complicated by the fact that a path in a proof structure is no longer followed by exactly one run as in the finite state case. In particular, there may be paths that are followed by no run at all. In order to account for this situation, we have defined the success criteria for LTL and ELL proof structures on a derived system, called the associated system, obtained as a combination of the original system and the proof structure at hand. A run of the associated system, called a trail of the proof structure, combines a system run with a path through the proof structure. The notions of success and fairness are then lifted to trails and success of LTL and ELL proof structures is then defined w.r.t. successful Π-fair trails. A syntactic characterisation of the two success criteria as temporal properties of the associated system exhibits the duality of LTL and ELL success and provides the starting point for the design of proof rules for success.

Success Rules

A success rule for LTL could be derived from a proof rule for future response properties as described in [MP91]. On the other hand, the ELL success rule is new. Both of these rules rely on a well-foundedness argument, although in a different way. First introduced in a basic version for saturated systems, the success rules are extended in Chapter 6 to account for fairness constraints. While weak fairness is relatively easy to deal with in each case, it is strong fairness that makes up most of the complexity of these rules. The LTL success rule $\mathsf{A}(\mathrm{S})_{fair}$ invokes itself recursively to prove a similar property of a modified (associated) system with a smaller strong fairness constraint, while Rule $\mathsf{E}(\mathrm{S})_{fair}$ requires a choice to be made, splitting the proof into several cases according to the way witnesses are supposed to satisfy the strong fairness constraint. The individual cases are proved using Rule $\mathsf{E}(\mathrm{S})^{\top}_{wuf}$.

Soundness and Completeness via Games

A novel approach is followed in the proof of soundness and completeness of our proof system. Due to the expressiveness of our assertion language the best we can expect is to show completeness relative to the validity of assertions. The novelty is that we use a game-theoretic argument for the main parts of the proof, which proceeds in three stages. First, we have characterised the

CTL* satisfaction relation in terms of the existence of winning strategies in CTL* model checking games. This characterisation is not a priori related to proof structures and has an interest of its own. In a second step, we have identified a close connection between LTL (ELL) trails and $\forall$-strategies ($\exists$-strategies) and subsequently between non-winningness (winningness) and LTL (ELL) admissibility. Admissibility is then compared to success and a successful proof structure for a system $\mathcal{S}$ and sequent $\Xi \vdash \mathsf{Q}\,\phi$ is shown to exist precisely if Player $\exists$ wins the game $\mathcal{G}_{\mathcal{S}}(\Xi, \mathsf{Q}\,\phi)$. The final step consists in showing that the success rules are sound and relatively complete. We think that a game-theoretic analysis can provide interesting insights into the inner workings of tableau proof systems such as the one presented here.

Discussion

Deductive local model checking applies to any ground-quantified CTL* formula and any reactive system that can be described as a fair transition system. As with algorithmic local methods only the part of the state space that is relevant to the property to be proved needs to be represented in a proof structure. Infinite-state systems are not the only domain of application of our proof system. It is equally useful for finite state systems that are too large to be model checked automatically.

As the modal μ-calculus subsumes CTL* in expressive power and several proof systems have been proposed for it (e.g., [BS92, And93, RH96, GBK97]), the question may arise why we need a specialised proof system for CTL*. One problem with translating CTL* into the modal μ-calculus is that the translation is double exponential [Dam94], indicating that CTL* can be a lot more concise than the μ-calculus (this is especially true for certain path formulas). Another difficulty is that μ-calculus formulas are generally harder to understand than CTL* formulas. The combined effect of these two problems is that the translation might completely obscure the meaning of the original formula, thus making it hard to prove using a proof system that was designed for the modal μ-calculus. CTL* is undoubtedly the temporal logic with the best trade-off between expressiveness and readability, and this fact alone justifies the design of a proof system for this logic.

Unlike with the deductive approach of [MP91], there is no need to transform the property formula into some canonical form prior to starting a proof. Similar to the translation into another logic, the problem with canonisation is that the original property may be obscured and a proof more difficult to find as a consequence. One could argue that canonisation has merely been replaced by a reduction of the original formula to the (uniform) success formula that has to be shown to hold for each proof structure. However, the proof structure itself is constructed from the original property formula and the presence of its subformulas and their unfolding forms in the sequents can provide a certain guidance for its construction. Moreover, the graphical representation of a proof structure often provides considerable help in guessing the auxiliary quantities required for the application of the success rule, given that the success criterion is formulated in terms of fairness constraints on (essentially) the original system and successful paths in the proof structure. Nonetheless, the application of the success rules is probably the most difficult part of a proof. But it should also be kept in mind that for a quite large class of LTL formulas, namely for all those with no occurrences of Until operators, any proof structure without anti-axioms is successful by construction, thus making the application of Rule $\mathsf{A}(\mathrm{S})_{fair}$ needless.

The price to be paid for the generality of our approach is that it is no longer possible to construct proofs in a fully automatic way. Human insight into the system and property to be proved is required to successfully complete a proof. This insight is brought into a proof in the form of choices that have to be made at specific points. For instance, the application of the Next rules requires choosing a new assertion for the successor sequent. Choosing these assertions as general as possible increases the chance of being able to loop back to that sequent later in the proof. Using the success rules also involves choosing the intermediate assertions and ranking functions. Making the right choices can lead to very compact proofs. It is not a question of whether it is good or bad that such choices need to be made, but whether the insight of the designer (and insight can be expected!) can be transformed naturally into a successful[1] proof. More substantial case studies are needed to obtain conclusive answers to this question.

Strong fairness is a difficult issue, which is often ignored altogether. Nonetheless, in many situations weak fairness alone is not sufficient to guarantee the required progress of individual system components for essential liveness properties to hold. Although we have proposed success rules dealing with weak as well as strong fairness, we feel that the treatment of strong fairness in the ELL success rule $\mathsf{E}(\mathrm{S})_{fair}$ is not completely satisfactory. It requires a choice to be made prior to the application of Rule $\mathsf{E}(\mathrm{S})^{\top}_{wuf}$ as to how trails witnessing the ELL success formula Ω_{E} should satisfy the strong fairness constraint F^{Π} of $\mathcal{S}^{\Pi}$. We would prefer if this choice could be eliminated or made in a more "dynamic" way as part of the application of Rule $\mathsf{E}(\mathrm{S})^{\top}_{wuf}$.

An important issue that is only marginally covered in this thesis is the possibility that a property fails to hold and the extraction of counterexamples. An inherent problem of all deductive systems is the distinction between

[1] here in a non-technical sense

our inability to find a proof and the case where the statement we try to prove is wrong. One possibility to follow in case we are unable to prove $\mathcal{S}, \Theta \vdash \phi$ is to try proving $\mathcal{S}, \Xi \vdash \neg\phi$ for some Ξ such that $\Xi \rightarrow \Theta$, that is, proving the contrary of the original property for a subset of the initial states. A better approach would try to use the already (possibly only partially) constructed proof structure for $\mathcal{S}$ and $\Theta \vdash \phi$ for such an effort and show that it is unsuccessful. A possible solution starts from the observation that a LTL or ELL proof structure Π for $\mathcal{S}$ and $\Xi \vdash \mathsf{Q}\,\phi$ is unsuccessful iff $\mathcal{S}^{\Pi} \not\models \mathsf{Q}_{\Pi}\,\Omega_{\mathsf{Q}}$ (where Q stands for A or E) iff there is an assertion ζ implying Θ^{Π} such that $\mathcal{S}^{\Pi}, \zeta \models \overline{\mathsf{Q}}_{\Pi}\neg\Omega_{\mathsf{Q}}$ (where $\overline{\mathsf{Q}}$ is the dual of Q). As the negation of Ω_{Q} has the same form as $\Omega_{\overline{\mathsf{Q}}}$, it is then not difficult to adapt Rule $\mathsf{A}(\mathrm{S})_{fair}$ for proving that a ELL proof structure is unsuccessful and likewise Rules $\mathsf{E}(\mathrm{S})_{fair}$ and $\mathsf{E}(\mathrm{S})^{\top}_{wuf}$ for proving that a LTL proof structure is unsuccessful. The case is more involved for CTL*, since a linear counterexample does not always exist. This topic certainly deserves further attention.

8.2 Related Work

8.2.1 Finite-State Model Checking

We have already discussed the paper [BCG95] which provided the starting point for the development of our deductive local model checking technique.

In recent work by Biere, Clarke and Zhu [BCZ99], developed in parallel with ours but independently, they propose an interesting tableau-based method for ELL that combines local and global model checking techniques for finite-state systems. Their sequents have the form $S \vdash \mathsf{E}(\Phi)$, where S is a *finite* set of states and their rules are similar to our ELL rules. They also have a Split rule, although not needed for completeness in their case:

$$\frac{S \vdash \mathsf{E}(\Phi)}{S_1 \vdash \mathsf{E}(\Phi) \quad S_2 \vdash \mathsf{E}(\Phi)} \quad S_1 \cup S_2 = S$$

The side condition requires that the two cases S_1 and S_2 exactly cover the original set S. This rule can therefore not be used for weakening. Their Next rule is also a restricted version of ours:

$$\frac{S \vdash \mathsf{E}(\mathsf{X}\,\Phi)}{\mathsf{img}(S) \vdash \mathsf{E}(\Phi)}$$

where $\mathsf{img}(S) \stackrel{\text{def}}{=} \{s' \mid \exists s \in S.\, s \rightarrow s'\}$ with transition relation $\rightarrow$. This is the set-theoretic equivalent of the strongest post-condition. For total transition relations, this is a particular way to satisfy (the set-theoretical equivalent of) the possibility triple appearing as the side condition of our rule $\mathsf{E}(\mathsf{X})$. Using $\mathsf{img}(S)$ in the successor sequent is certainly more suitable for algorithmic purposes than using a subset of $\mathsf{img}(S)$ (corresponding to a choice of successor states). As a consequence of these restrictions (w.r.t. to our system) and the finiteness of the sets S, any state appearing in a sequent of a proof structure is reachable from an initial state and any path in the proof structure is followed by *at least one* run of the system. Therefore, a witnessing run can always be extracted from a successful path (they do not consider fairness). The interest of this method lies in the ability to represent the finite sets appearing in the sequents by BDDs and to have efficient algorithms manipulating them. BDD techniques have hitherto been used only for global model checking. This technique thus combines advantages of local and global model checking.

8.2.2 Deductive and Semi-Algorithmic Methods

Manna and Pnueli's Proof System

The proof system described in [MP91, MP95] has already been sketched in Section 2.6.3. The advantage of this system is the small number (three) of basic rules, which are shown to be relatively complete. The price to be paid is that formulas have to be brought into canonical form by a complex translation [LPZ85, MP90] prior to the proof. The drawbacks of canonisation have already been discussed above.

Diagram-Based Methods

Some of the diagram-based methods have been sketched in Section 2.6.4. The verification diagrams of [MP94] are a diagrammatic form of some of the proof rules in [MP91]. We have already compared them with our approach in the previous chapter (Section 7.3.2), where we showed that invariance diagrams can be considered as a condensed form of proof structures. Other types of verification diagrams can be translated to proof structures in a similar way.

Generalised verification diagrams (GVDs) [BMS95, MBSU98] (and the theses [Uri98, Sip99]) are a direct proof method as is ours (in contrast to methods that are driven by the search for a counterexample such as DMC). It consists of first constructing an abstraction of the system (the GVD) which is then model checked algorithmically. An advantage of separating these two steps is that an abstraction can possibly be used for the verification of several properties. As with all abstraction methods, if the model checking phase fails to establish the property then this result is not conclusive and the abstraction needs to be refined until the model checker succeeds (provided the property holds). On the other hand, in the construction of a proof structure the system and its property are explored hand-in-hand and the presence of the temporal formulas in the sequents may provide some guidance for its construction. Moreover, at least for LTL properties, any proof structure is successful, provided the property holds. If we are unable to complete the construction of an LTL proof structure then we can still try to extract a counterexample (a run following an unsuccessful path) from the pre-proof structure constructed so far. The construction of an ELL proof structure might however fail due to badly chosen assertions, even if the property holds.

Deductive model checking (DMC) [SUM99] (and the theses [Uri98, Sip99]) differs from our method in that it is indirect, that is, driven by the search for a counterexample. Whereas the GVD method starts on the system side (by building an abstraction), the DMC method starts on the side of the formula, that is, from the tableau of the negation of the LTL formula φ to be verified (called the initial falsification diagram). It then proceeds by refinement of falsification diagrams, while maintaining the invariant that all counterexamples to φ present in the system (if any) are represented in each falsification diagram. The process is stopped if the language accepted by a falsification diagram can be seen to be empty.

As an early diagram-based method $\forall$-automata were proposed in [MP89] as an alternative to temporal logic verification. These are finite-state ω-automata that accept an infinite sequence σ if *all* runs of the automaton on σ are accepting. This is reminiscent of the success criterion of LTL proof structures, where *all* Π-fair trails are required to be successful, that is, for any computation σ *all* trails projecting to σ have to be successful.

It should be noted that $\forall$-automata, GVDs and DMC (the latter two being based on a form of Müller automaton) have all the expressive power of ω-regular languages [Tho90] or, equivalently, extended temporal logic (ETL) [Wol83].

Fix and Grumberg's Proof System for CTL

The first proof system for CTL is proposed by Fix and Grumberg in [FG96]. They use transitions systems with weak fairness constraints as their computational model. Sequents are of the form $P\,Sat\,p \rightarrow \phi$, where P is a program, p is an assertional pre-condition and ϕ is a CTL formula (where path quantification is over weakly fair runs). A sequent is valid if the initial state of every computation tree of P satisfies the implication $p \rightarrow \phi$. They present a set of rules for proving the validity of sequents. The rule to be applied to a sequent is determined by the top-level connective of the CTL formula (negated forms are also included). Each rule reduces its conclusion to a set of assertions and/or simpler temporal properties. The proof system is shown to be relatively complete.

Their rule for $P\,Sat\,p \rightarrow \neg\,\mathsf{A}(\phi_1\,\mathsf{U}\,\phi_2)$ reduces this sequent to proving $P\,Sat\,p \rightarrow \mathsf{E}\,\mathsf{G}(\neg\phi_2)$ or $P\,Sat\,p \rightarrow \neg\,\mathsf{E}(\neg\phi_2\,\mathsf{U}(\neg\phi_1 \wedge \neg\phi_2))$. It is interesting to see how their rule for $P\,Sat\,p \rightarrow \mathsf{E}\,\mathsf{G}\,\phi$ ensures that a *(weakly) fair* run satisfying $\mathsf{G}\,\phi$ exists. Their rule mechanism to achieve this is based in the observation that a fair run is composed of fair segments. On a fair segment each fair transition is either disabled in some state or taken somewhere (see also the proof of completeness of Rule F-RESP in [MP91]). They introduce a function $g\colon \Sigma \rightarrow \{0,1\}^n$ mapping states to a bit vector with one bit for each fair transition of the system. The premises are designed in such a way that g records the transitions that have been granted (that is, disabled or taken) on a segment. The end points of segments are indicated by an auxiliary assertion I that is required to hold infinitely often on a run witnessing $\mathsf{G}\,\phi$. Assertion I is required to imply $g = \overline{0}$ (n zeros), indicating that all fair transitions have been granted. In this way, g implements a particular form of ranking function.

It is interesting to compare this mechanism with the modes and rankings of our Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$ for proving properties of the form $\mathsf{E}(\mathsf{F}\,\mathsf{G}\,q \wedge \bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,r_i)$ (Section 6.3.2). Suppose that we have no assertions r_i and an empty unconditional fairness constraint. By setting $\alpha_0 \stackrel{\text{def}}{=} \mathsf{false}$ we can actually "turn off" the fall-backs and thus obtain a rule for properties of the form $\mathsf{E}\,\mathsf{G}\,q$, albeit only for assertions q. Only premises R1, R3 and R6 remain nontrivial for this variant of Rule $\mathsf{E}(\mathsf{F}\,\mathsf{G}, \bigwedge \mathsf{G}\,\mathsf{F})_{wuf}$. The existence of a (weakly) fair run witnessing $\mathsf{G}\,q$ is ensured by the modes α_i and rankings δ_i, one such pair for each $\Lambda_i \in W$. Whereas their ranking g measures the distance to the end of a fair interval, our mode α_i "remembers" the Λ_i to be granted next and δ_i measures the distance to the point where Λ_i is granted. Premise R6 requires that the mode is switched from α_i to $\alpha_{i\oplus 1}$ upon granting Λ_i, thereby ensuring that all elements of W are granted in a cyclic manner.

Considering the capability required of their function g to record the granting of transitions, it is not surprising that they also need a history variable in the proof of relative completeness of their rule.

Other Proof Systems

In [HGD95, BDG+98] a proof system for first-order ACTL is proposed. ACTL is the sublogic of CTL, where only universal path quantifiers are allowed. They generate a first-order *success formula*, the validity of which is sufficient to conclude that the property holds. Their system is not relatively complete, since it does not include a well-foundedness argument to show that U-formulas fulfill their promises. However, their primary goal is not completeness, but to obtain a high degree of automation by separating control from data aspects.

Several proof systems the modal μ-calculus have been proposed. Bradfield and Stirling [BS92, Bra91] describe a tableau system for the propositional μ-calculus that was obtained by generalisation from the finite-state system in [SW91]. Andersen has extended the Winskel's rewriting version of the latter system [Win91] to the infinite-state case [And93]. Rathke and Hennessy describe a proof system for a first-order version of the modal μ-calculus [RH96, Rat97].

A compositional proof systems for sequential value-passing CCS processes and the first-order μ-calculus is presented by Gurov, Berezin and Kapron [GBK97] and a separate system handling parallel composition is introduced in [BG97] (see also Gurov's thesis [Gur98], where both systems are described). Mads Dam [Dam98] also described a compositional proof system for the first-order μ-calculus.

8.2.3 Model Checking Games

Model checking games were introduced by Stirling in a series of papers [Sti95, Sti96a, Sti97]. It turns out that a successful tableau constructed according to the rules in [SW91, Sti96b] can be seen as a winning strategy for such a game[2]. This is in contrast to our system, where a *trail (path)* of a proof structure corresponds to a strategy (pre-strategy) for CTL* games.

A efficient local model checking algorithm for the μ-calculus based on games is presented in [SS98]. It constructs a winning strategy for the model checking game corresponding to the property to be verified. By playing against the machine (and loosing each play) these games can help the user understand *why* a property hold or fails. This algorithm has been incorporated in to the Edinburgh Concurrency Workbench [MS].

Model checking games for CTL* have been proposed only very recently in (as yet) unpublished work by Lange and Stirling [LS00]. The difference with our CTL* games is that while our games are played along runs of the system, their games are state-based with configurations of the form $p, s \vdash [\varphi], \Phi$, where p is the current *pathplayer* (either $\exists$ or $\forall$), s is a state, φ is the formula in *focus* and Φ is a set of formulas. They give a set of rules that define the legal moves and the player who possibly needs to make a choice (of a subformula or successor state) in that move. Disregarding the focus, these

[2] This view of tableau as strategies leads to a spectacular simplification of the original proof of soundness and completeness in [SW91].

rules are very similar to our rules for LTL and ELL, but they are combined into one system with the pathplayer in the configuration indicating which part of the system is currently being used. The pathplayer is reset when a path-quantified formula appears in the focus. The formulas in focus in a configuration and its successor are usually related by a generation relation (in our terminology), but there is a special rule allowing the pathplayer's opponent to change the focus. This is necessary, because plays move from state to state and not along a run as in our case, and gives the opponent a chance to redo previous moves. Their work is too recent to give a more in-depth comparison, but the precise relationship of their games with ours and especially with proof structures will provide a nice topic for further investigation.

It is interesting to consider our CTL* games and strategies in the light of the work on abstract games presented by Perdita Stevens [Ste98a, Ste98b]. An abstract version of our CTL* games could have configurations (R, ϕ), where R is a *set* of runs and ϕ a CTL* formula. The rules remain essentially the same except that the games are now played along all the runs in R simultaneously. A Next move would thus proceed from a configuration $(R, \mathsf{X}\,\psi)$ to (R', ψ), where $R' = \{\sigma^1 \mid \sigma \in R\}$. Unlike concrete plays, finite abstract plays may end in a draw.

Consider the abstract game $\mathcal{G}_{\mathcal{S}}(R_0, \mathsf{A}\,\phi)$ where $\mathsf{A}\,\phi$ is a LTL formula and R_0 is the set of Ξ-computations following a path π in a LTL proof structure for $\mathcal{S}$ and $\Xi \vdash \mathsf{A}\,\phi$. Then the path π can be transformed into an *abstract* $\forall$-*strategy* τ_π for that game by defining

$$T_\pi \stackrel{\text{def}}{=} \{R_\pi * \iota \mid \iota \in I^*(\pi)\} \qquad \tau_\pi \stackrel{\text{def}}{=} \natural T_\pi$$

where

$$R_\pi(i) \stackrel{\text{def}}{=} \{\widetilde{\sigma_\vartheta}(i) \mid \exists\ \Pi\text{−fair trail } \vartheta.\, \pi_\vartheta = \pi\}$$

Note that $R_\pi(0) = R_0$. This strategy treats all computation following π in a uniform way, such that no (abstract) plays end in a draw. In summary, paths in proof structures seem to correspond to strategies in abstract games, while trails correspond to strategies of concrete games. The details remain to be checked.

8.3 Directions for Future Work

Some possible tracks for future work that come to mind are listed below.

Tool support

For the practical application of our method tool support is essential. The number of verification conditions alone makes proofs by hand error-prone. A suitable tool would consist of a graphical front-end based on a graph editor for the construction of proof structures and a theorem prover to discharge the verification conditions at the back-end. The front-end can be based on a graph editor with application of proof rules driven by the syntax of formulas. Clicking on a formula on the right-hand side of a sequent could apply the rule corresponding to its top-level connective. Clicking on the left-hand side would invoke the Split rule. The front-end would also manage the verification conditions generated by the rules and provide an interface to the theorem prover. High-level tactics and techniques for the automatic generation of invariants (see e.g., [BBM97]) could assist the user in the construction of proof structures. For proving success, auxiliary assertions could be assigned to each sequent in the proof structure and the Hoare or possibility triples to be discharged could be associated to edges with the proof tool trying to identify and eliminate the trivial conditions. We plan to implement such a tool with system specifications based on the high-level concurrent object language SOL and its twisted system semantics as described in the thesis of Krzysztof Worytkiewicz [Wor00].

Refinement of Proof Structures and SCS-Based Success Rules

After the partial construction of a proof structure it may happen that we would like to make it more “fine-grained”, in order to simplify the subsequent application of a success rule, for example. Therefore, it could be helpful to be able to refine an already or partially constructed proof structure, instead of starting the construction from scratch. The design of refinement rules similar to the ones used in DMC [SUM99] could be considered.

Another point of investigation is in alternative success rules based on an examination of each unsuccessful strongly connected subgraph (SCS), similar to the ones used in DMC [SUM99] and GVDs [BMS95, MBSU98]. These could in some situations provide a more “user-friendly” way of proving success. A possible problem is that the number of SCS grows exponentially with the size of a proof structure. However, for many practical applications the number of SCS might be small enough to make such an approach a suitable alternative.

Real-time systems

Real-time systems (see [AH92] for a survey) are intrinsicly infinite state due to the unbounded progress of time. In the clocked transition system (CTS) model of [KMP98], there are several clock variables one of which is the (global) master clock T. Time is advanced by a special *tick* transition, while system transitions do not modify the clock variables. Progress (of time) is guaranteed by the so-called non-zenoness condition, which replaces the fairness constraints found in discrete systems. A CTS run is zeno if time stops at some point or converges to an upper bound. Only non-zeno runs (with time diverging) are considered as computations. A CTS is non-zeno if every computation prefix can be extended to a computation. This condition can be expressed as the CTL formula

$$\forall \epsilon > 0. \forall t.\ \mathsf{A}\,\mathsf{G}(T = t \rightarrow \mathsf{E}\,\mathsf{F}(T \geq t + \epsilon))$$

where path quantification is to be understood over all runs [Sip99]. This formula can be verified using our proof system.

However, for the proper verification of properties of real-time systems path-quantification ranges over computations only (that is, the non-zeno runs). The local rules for the construction of proof structures can be used for this purpose as they stand, but the success rules need to be reviewed and adapted for non-zenoness.

Model Checking Games and Counterexamples

The precise connection between our CTL* games and the version presented in [LS00] should be investigated. Our own CTL* games are played along runs and, while appropriate for proving soundness and completeness of our proof system, seem not very suitable for the extraction of winning strategies that can help the users understand why a property holds or not. The connection of the CTL* games of [LS00] to proof structures should be carefully examined. In particular, it should be checked whether winning strategies for their CTL* games are represented in some form in our proof structures.

Bibliography

[AFK88] K.R. Apt, N. Francez, and S. Katz. Appraising fairness in languages for distributed programming. *Distributed Computing*, 2:226–241, 1988.

[AH92] R. Alur and T. Henzinger. Logics and models for real-time: a survey. In J. W. de Bakker, C. Huizing, W. P. de Roever, and G. Rozenberg, editors, *Proceedings of the REX Workshop "Real-Time: Theory in Practice"*, volume 600 of *Lecture Notes in Computer Science*, pages 74–106. Springer-Verlag, 1992.

[AL91] Martín Abadi and Leslie Lamport. The existence of refinement mappings. *Theoretical Computer Science*, 82(2):253–284, May 1991.

[And93] Henrik Reif Andersen. *Verification of Temporal Properties of Concurrent Systems.* PhD thesis, Computer Science Department, Aarhus University, June 1993.

[AS85] Bowen Alpern and Fred B. Schneider. Defining liveness. *Information Processing Letters*, 21:181–185, October 1985.

[BBM97] Nikolaj S. Bjorner, Anca Browne, and Zohar Manna. Automatic generation of invariants and intermediate assertions. *Theoretical Computer Science*, 173(1):49–87, 1997.

[BBP89] B. Banieqbal, H. Barringer, and A. Pnueli, editors. *Temporal Logic in Specification*, volume 398 of *Lecture Notes in Computer Science*. Springer-Verlag, 1989.

[BCG95] G. Bhat, R. Cleaveland, and O. Grumberg. Efficient on-the-fly model checking for CTL*. In *Logic in Computer Science, LICS 95*, pages 388–397, 1995.

[BCM92] J. R. Burch, E. M. Clarke, and K. L. McMillan. Symbolic model checking: 10^{20} states and beyond. *Information and Computation*, 98:142–170, 1992.

[BCZ99] Armin Biere, Edmund Clarke, and Yunshan Zhu. Combining local and global model checking. *Electronic Notes in Theoretical Computer Science*, 23(2), 1999.

[BDG+98] Jürgen Bohn, Werner Damm, Orna Grumberg, Hardi Hungar, and Karen Laster. First-order CTL model checking. In *Foundations of Software Technology and Theoretical Computer Science, FSTTCS '98, Chennai, India*, volume 1530, pages 283–294. Springer-Verlag, 1998. Preliminary version appeared in [HGD95].

[BG97] Sergey Berezin and Dilian Gurov. A compositional proof system for the modal μ-calculus and CCS. Technical Report CMU-CS-97-105, Carnegie Mellon University, Pittsburgh, PA, 1997.

[BMS95] I. A. Browne, Z. Manna, and H. B. Sipma. Generalized temporal verification diagrams. In *Foundations of Software Technology and Theoretical Computer Science, FSTTCS '95, Bangalore, India*, volume 1026 of *Lecture Notes in Computer Science*, pages 484–498. Springer-Verlag, 1995.

[Bra91] Julian Charles Bradfield. *Verifying Temporal Properties of Systems with Applications to Petri Nets.* PhD thesis, University of Edinburgh, July 1991.

[Bry86] Randal E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(6):677–691, August 1986.

[BS92] Julian Bradfield and Colin Stirling. Local model checking for infinite state spaces. *Theoretical Computer Science*, 96:157–174, 1992.

[BVW] Orna Bernholtz, Moshe Y. Vardi, and Pierre Wolper. An automata-theoretic approach to branching-time model checking. In *CAV '94*, Lecture Notes in Computer Science, pages 142–155. Springer-Verlag.

[BW90] J. C. M. Baeten and W. P. Weijland. *Process Algebra*, volume 18 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, 1990.

[CE81] E. M. Clarke and E. A. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Proceedings Workshop on Logics of Programs*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71, Springer, Berlin, 1981.

[CES86] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8:244–263, 1986.

[CGH97] E. M. Clarke, O. Grumberg, and H. Hamaguchi. Another look at symbolic ltl model checking. *Formal Methods in System Design*, 10(1), 1997.

[CGL93] Edmund M. Clarke, Orna Grumberg, and David E. Long. Verification tools for finite-state concurrent systems. In J. W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *A Decade of Concurrency: Reflections and Perspectives*, volume 803 of *Lecture Notes in Computer Science*, pages 124–175. Springer-Verlag, 1993.

[CGP99] Edmund M. Clarke, Orna Grumberg, and Doron Peled. *Model Checking*. The MIT Press, 1999.

[CMP93] E. Chang, Z. Manna, and A. Pnueli. The safety-progress classification. In Friedrich L. Bauer, Wilfried Brauer, and Helmut Schwichtenberg, editors, *Logic and Algebra of Specification*, volume 94 of *NATO ASI Series F: Computer and System Sciences*, pages 143–202. Springer-Verlag, 1993.

[CS87] Berardo Costa and Colin Stirling. Weak and strong fairness in CCS. *Information and Computation*, 73(3):207–244, 1987.

[Dam94] Mads Dam. CTL* and ECTL* as fragments of the modal μ-calculus. *Theoretical Computer Science*, 126:77–96, 1994.

[Dam95] Mads Dam. Proving properties of dynamic process networks. In *CONCUR '95*, volume 962 of *Lecture Notes in Computer Science*, pages 12–26. Springer-Verlag, 1995.

[Dam98] Mads Dam. Proving properties of dynamic process networks. *Information and Computation*, 140:95–114, 1998. Full version of [Dam95].

[DF95] Jürgen Dingel and Thomas Filkorn. Model checking for infinite state systems using data abstraction, assumption-commitment style reasoning and theorem proving. In *CAV '95*, volume 939 of *Lecture Notes in Computer Science*. Springer-Verlag, 1995.

[DGG97] Dennis Dams, Orna Grumberg, and Rob Gerth. Abstract interpretation of reactive systems. *ACM Transactions on Programming Languages and Systems*, 19(2):253–291, 1997.

[dN87] Rocco de Nicola. Extensional equivalences for transition systems. *Acta Informatica*, 24:211–237, 1987.

[EH86] E. A. Emerson and J. Y. Halpern. "Sometimes" and "not never" revisited: on branching vs. linear time temporal logic. *Journal of the ACM*, 33(1):151–178, January 1986.

[EL85] E. A. Emerson and C. L. Lei. Modalities for model checking: branching time strikes back (extended abstract). In *Proceedings of the 12th Annual ACM Symposium on the Principles of Programming Languages*, pages 84–96, 1985.

[Eme90] E.A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of theoretical computer science*, volume B, pages 995–1072. Elsevier, 1990.

[FG96] Limor Fix and Orna Grumberg. Verification of temporal properties. *Journal of Logic and Computation*, 6(3):343–361, 1996.

[Fra86] N. Francez. *Fairness*. Springer, New York, 1986.

[GBK97] Dilian Gurov, Sergey Berezin, and Bruce Kapron. A modal mu-calculus and a proof system for value passing processes. *Electronic Notes in Theoretical Computer Science*, 5, 1997.

[GPVW95] Rob Gerth, Doron Peled, Moshe Y. Vardi, and Pierre Wolper. Simple on-the-fly automatic verification of linear time temporal logic. Unpublished manuscript, January 1995.

[GS97] Susanne Graf and Hassen Saide. Construction of abstract state graphs with pvs. In Orna Grumberg, editor, *Computer-Aided Verification, CAV 97*, volume 1254 of *Lecture Notes in Computer Science*, pages 72–83. Springer-Verlag, 1997.

[Gur98] Dilian Gurov. *Specification and Verification of Communicating Systems with Value Passing.* PhD thesis, University of Victoria, Canada, 1998.

[HGD95] Hardi Hungar, Orna Grumberg, and Werner Damm. What if model checking must be truly symbolic? In *TACAS '95*, volume 987 of *Lecture Notes in Computer Science*. Springer-Verlag, 1995.

[Hoa85] C. A. R. Hoare. *Communicating Sequential Processes.* Prentice Hall International Series on Computer Science. Prentice-Hall, 1985.

[Kic96] Alexander Kick. *Generierung von Gegenbeispielen und Zeugen bei der Modellprüfung.* PhD thesis, Fakultät für Informatik, Universität Karlsruhe, Germany, 1996. (in german).

[KMP93] Y. Kesten, Z. Manna, and A. Pnueli. Temporal verification of simulation and refinement. In J. W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *A Decade of Concurrency: Reflections and Perspectives*, volume 803 of *Lecture Notes in Computer Science*, pages 273–346. Springer-Verlag, 1993.

[KMP98] Y. Kesten, Z. Manna, and A. Pnueli. Verification of clocked and hybrid systems. In G. Rozenberg and F. W. Vaandrager, editors, *Lectures on Embedded Systems*, volume 1494 of *Lecture Notes in Computer Science*, pages 4–73. Springer-Verlag, 1998.

[Koz83] D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[KRP95] Y. Kesten, L. Raviv, and A. Pnueli. OBDD's LTL MC: Model checking of linear tl, using obdd's. Technical Report CS95-13, Weizmann Institute of Science, 1995.

[Kur94] Robert P. Kurshan. *Computer-aided verification of coordinating processes.* Princeton University Press, 1994.

[Kwi89] Marta Z. Kwiatkowska. Survey of fairness notions. *Information and Software Technology*, 31(7):371–386, September 1989.

[Lam74] Leslie Lamport. A new solution of Dijkstra's concurrent programming problem. *Communications of the ACM*, 17(8):435–455, 1974.

[Lam94] Leslie Lamport. Temporal logic of actions. *ACM Transactions on Programming Languages and Systems*, 16(3):872–923, May 1994.

[LP85] O. Lichtenstein and A. Pnueli. Checking that finite state concurrent programs satisfy their linear specification. In *Proceedings of the 12th Annual ACM Symposium on the Principles of Programming Languages*, pages 97–107, 1985.

[LPS81] D. Lehmann, A. Pnueli, and J. Stavi. Impartiality, justice and fairness: the ethics of concurrent termination. In *Proceedings of the 8th ICALP*, volume 115, pages 264–277. Springer-Verlag, July 1981.

[LPZ85] O. Lichtenstein, A. Pnueli, and L. Zuck. The glory of the past. In *Proceedings of Conference on Logics of Programs*, volume 193 of *Lecture Notes in Computer Science*, pages 196–218. Springer-Verlag, 1985.

[LS00] Martin Lange and Colin Stirling. Model checking games for ctl*. submitted to ICTL 2000, 2000.

[LT87] N. Lynch and M. Tuttle. Hierarchical correctness proofs for distributed algorithms. In *Proceedings of the 6th Annual ACM Symposium on Principles of Distributed Computing*, pages 137–151. ACM Press, 1987. Extended version in Technical Report MIT/LCS/TR-387, Lab for Computer Science, MIT.

[MBSU98] Zohar Manna, Anca Browne, Henny B. Sipma, and Tomás E. Uribe. Visual abstractions for temporal verification. In *AMAST '98*, volume 1548 of *Lecture Notes in Computer Science*, pages 28–41. Springer-Verlag, 1998.

[Mil89] Robin Milner. *Communication and Concurrency*. Prentice Hall International Series in Computer Science. Prentice Hall, 1989.

[Mil99] Robin Milner. *Communicating and Mobile Systems: the π-Calculus*. Cambridge University Press, 1999.

[Mos74] Yiannis N. Moschovakis. *Elementary Induction on Abstract Structures*, volume 77 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Company, 1974.

[MP89] Zohar Manna and Amir Pnueli. Specification and verification of concurrent programs by ∀-automata. In *[BBP89]*, pages 124–164. Springer-Verlag, 1989.

[MP90] O. Maler and A. Pnueli. Tight bounds on the complexity of cascaded decomposition of automata. In *Proceedings of the 31th IEEE Symposium on the Foundations of Computer Science*, pages 672–682, 1990.

[MP91] Z. Manna and A. Pnueli. Completing the temporal picture. *Theoretical Computer Science*, 83(1):97–139, 1991.

[MP92] Zohar Manna and Amir Pnueli. *The Temporal Logic of Reactive and Concurrent Systems.* Springer Verlag, 1992.

[MP94] Zohar Manna and Amir Pnueli. Temporal verification diagrams. In Masami Hagiya and John C. Mitchell, editors, *Theoretical Aspects of Computer Software, Proceedings TACS 94*, volume 789 of *Lecture Notes in Computer Science*, pages 726–765. Springer-Verlag, 1994.

[MP95] Zohar Manna and Amir Pnueli. *Temporal Verification of Reactive Systems – Safety.* Springer Verlag, 1995.

[MS] Faron Moller and Perdita Stevens. *The Edinburgh Concurrency Workbench user manual.* Laboratory for Foundations of Computer Science, Edinburgh University. available at `http://www.dcs.ed.ac.uk/home/cwb/doc`.

[Par76] David Park. Finiteness is mu-ineffable. *Theoretical Computer Science*, 3(2):173–181, 1976.

[Pnu92] Amir Pnueli. System specification and refinement in temporal logic. In R. Shyamasundar, editor, *Proceedings of the 12th Conference on Foundations of Software Technology and Theoretical Computer Science, New Delhi, India, December 1992*, volume 652 of *Lecture Notes in Computer Science*, pages 1–38, 1992.

[QS82] J. Queille and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *International Symposium on Programming*, volume 137 of *Lecture Notes in Computer Science*, pages 337–351. Springer-Verlag, 1982.

[Rat97] Julian Rathke. *Symbolic Techniques for Value-Passing Calculi.* PhD thesis, University of Sussex, United Kingdom, 1997.

[Ref96] Frank Reffel. Übersetzung von CTL* Formeln in den μ-Kalkül. Master's thesis, Fakultät für Informatik, Universität Karlsruhe, Germany, 1996. (in german).

[Rei85] W. Reisig. *Petri Nets.* EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1985.

[RH96] Julian Rathke and Matthew Hennessy. Local model checking for a value-based μ-calculus. Technical Report 05/96, University of Sussex, 1996.

[RSS95] S. Rajan, N. Shankar, and M. K. Srivas. An integration of model checking with automated proof checking. In *CAV '95*, volume 939 of *Lecture Notes in Computer Science*, pages 84–97. Springer-Verlag, 1995.

[SdRG89] F. A. Stomp, W.-P. de Roever, and R. T. Gerth. The μ-calculus as an assertion language for fairness arguments. *Information and Computation*, 82:278–322, 1989.

[Sip99] Henny Sipma. *Diagram-based verification of discrete, real-time and hybrid systems.* PhD thesis, Stanford University, 1999.

[SNW96] Vladimiro Sassone, Mogens Nielsen, and Glynn Winskel. Models of concurrency: towards a classification. *Theoretical Computer Science*, 170(1–2):297–348, 1996.

[SS98] Perdita Stevens and Colin Stirling. Practical model-checking using games. In *TACAS '98*, volume 1384 of *Lecture Notes in Computer Science*, pages 85–101. Springer-Verlag, 1998.

[Sta88] E. W. Stark. Proving entailment between conceptual state specifications. *Theoretical Computer Science*, 56:135–154, 1988.

[Ste98a] Perdita Stevens. Abstract games for infinite state processes. In *CONCUR '98*, volume 1466 of *Lecture Notes in Computer Science*, pages 147–162. Springer-Verlag, 1998.

[Ste98b] Perdita Stevens. Abstract interpretation of games. In *VMCAI '98*, 1998.

[Sti89] Colin Stirling. Comparing linear and branching time temporal logics. In *Temporal Logic in Specification*, volume 398 of *Lecture Notes in Computer Science*. Springer-Verlag, 1989.

[Sti92] Colin Stirling. Modal and temporal logics. In S. Abramsky, Dov M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2: Background: Computational Structures, pages 477–563. Oxford University Press, 1992.

[Sti95] Colin Stirling. Local model checking games. In *CONCUR '95*, volume 962 of *Lecture Notes in Computer Science*, pages 1–11. Springer-Verlag, 1995.

[Sti96a] Colin Stirling. Games and modal μ-calculus. In *TACAS '96*, volume 1055 of *Lecture Notes in Computer Science*, pages 298–312. Springer-Verlag, 1996.

[Sti96b] Colin Stirling. Modal and temporal logics for processes. volume 1043 of *Lecture Notes in Computer Science*, pages 149–237. Springer-Verlag, 1996.

[Sti97] Colin Stirling. Bisimulation, model checking and other games. Notes for Mathfit instructional meeting on games and computation, Edinburgh, June 1997.

[SUM99] Henny Sipma, Tomás E. Uribe, and Zohar Manna. Deductive model checking. *Formal Methods in System Design*, 15:49–74, 1999.

[SW91] Colin Stirling and David Walker. Local model checking in the modal μ-calculus. *Theoretical Computer Science*, 89:161–177, 1991.

[Tho90] W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 133–191. Elsevier Science Publishers , Amsterdam, 1990.

[TL94] Wolfgang Thomas and Helmut Lescow. Logical specifications of infinite computations. In J. W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *A Decade of Concurrency - Reflections and Perspectives*, volume 803 of *Lecture Notes in Computer Science*, pages 583–621. Springer-Verlag, 1994.

[Uri98] Tomás E. Uribe. *Abstraction-Based Deductive-Algorithmic Verification of Reactive Systems.* PhD thesis, Stanford University, December 1998.

[vG90] R. J. van Glabbeek. The linear time - branching time spectrum. In J. C. M. Baeten and J. W. Klop, editors, *CONCUR 90*, volume 458, pages 278–297. Springer-Verlag, 1990.

[vG93] R. J. van Glabbeek. The linear time - branching time spectrum II – the semantics of sequential systems with silent moves. In E. Best, editor, *CONCUR '93*, volume 715, pages 66–81. Springer-Verlag, 1993.

[VW86] M. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proceedings of the IEEE Symposium on Logic in Computer Science*, pages 322–331, 1986.

[Win91] Glynn Winskel. A note on model checking the modal μ-calculus. *Theoretical Computer Science*, 83:157–167, 1991.

[Wol83] P. Wolper. Temporal logic can be more expressive. *Information and Control*, 56(1, 2):72–93, 1983.

[Wor00] Krzysztof Worytkiewicz. *Components and Synchronous Communication in Categories of Processes.* PhD thesis, Department of Computer Science, Swiss Federal Institute of Technology, Lausanne, Switzerland, 2000.

Curriculum Vitae

January 25, 1967 Born in St. Gallen, Switzerland

1981-1985 Cantonal School (Gymnasium), Heerbrugg, Matura Type B

1986-1992 studies in Computer Science at the Swiss Federal Institute of Technology (ETH), Zurich, engineer diploma in Computer Science

specialisation in computer architecture, compiler construction, operating systems and control theory

diploma thesis on "Network Computing with Non-Blocking Remote Procedure Calls" (in german) supervised by Prof. Dr. W. Gander

June 1992-March 1994 scientific co-worker at the Institute for Scientific Computing at ETH Zurich in the group of Prof. Dr. W. Gander and Dr. P. Arbenz

Sept.-Oct. 1992 visitor in the group of Prof. Dr. Jack Dogarra at the University of Tennessee, Knoxville

May 1994-May 2000 Ph.D. at the Swiss Federal Institute of Technology, Lausanne, under supervision of Prof. Dr. Claude Petitpierre

current address:
Computer Networking Laboratory
Department of Computer Science
Swiss Federal Institute of Technology
EPFL-DI-LTI
1015 Lausanne
Switzerland

e-mail: `christoph.sprenger@epfl.ch`