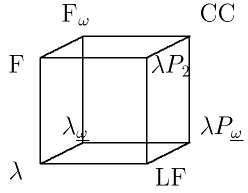
Théorie des types
II) Systèmes de Types Purs (PTS)

Joëlle Despeyroux

INRIA Sophia-Antipolis

Module
“Méthodes formelles et fiabilité du logiciel”
Dea d’informatique, Nice, automne 2000

Le λ-cube



Présentation uniforme de 8 λ-calculs typés
[H. Barendregt, S. Berardi, J.Terlow]

- Typage à la Church ($\lambda x : A.M$)
- λ-calculs vérifiant l’isomorphisme de Curry-Howard
- *Logical Frameworks*: cadre logique pour étudier (spécifications et preuves de propriétés) des logiques ou des langages.

Le cube est orienté; le calcul le plus riche est CC: le Calcul des Constructions.

On ajoute aux λ-calculs du cube des *types inductifs*, et des opérateurs de *récursion*.
Ex: CCind: le Calcul des Constructions + des opérateurs permettant de définir des types inductifs + des opérateurs de récursion (raisonnement par cas et point fixe).

Le λ-calcul simplement typé

types : $A ::= \alpha \mid A \rightarrow A'$

termes : $M ::= x \mid \lambda x : A.M' \mid (M\ M')$

contextes : $\Gamma ::= \ . \mid \Gamma, x : A$

reduction : $(\lambda x : A.M\ \ N) \hookrightarrow_\beta M[N/x]$

Un contexte est légal si tous les x_i sont distincts.

$$var\ \frac{x : A \in \Gamma}{\Gamma \vdash x : A}\ \Gamma \text{ légal} \qquad lam\ \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x : A.M\ \ :\ A \rightarrow B}$$

$$app\ \frac{\Gamma \vdash M : A \rightarrow B \qquad \Gamma \vdash N : A}{\Gamma \vdash (M\ N) : B}$$

Exemples:

$$\begin{array}{ll} & \vdash \lambda x : A.x : A \rightarrow A \\ y : B & \vdash \lambda x : A.y : A \rightarrow B \\ y : B, z : A & \vdash (\lambda x : A.y\ \ z) : B \\ & \vdash \lambda x : A.\lambda y : B.x : A \rightarrow B \rightarrow A \end{array}$$

Le terme $\lambda x.\ (x\ x)$ n'est pas bien typé.

Problème de **capture de variable**:
$\lambda x.\lambda x.x \equiv \lambda x'.\lambda x.x$ ou $\lambda x.\lambda x'.x$?

Solution: α-**conversion:** $\lambda x.x \equiv \lambda y.y$ et $\lambda x.\lambda x.x \equiv \lambda x.\lambda y.y$.

Autres réductions:

$$\begin{array}{ll} \lambda x : A.M \hookrightarrow_\alpha \lambda y : A.M[y/x] & - \text{ renommage} \\ \lambda x : A.(M\ x) \hookrightarrow_\eta M \text{ si } x \notin M & - \text{ Ex : } \lambda x : i.\ (S\ x) = S \end{array}$$

Le λ-calcul simplement typé: propriétés

Unicité du type: Pour tout contexte Γ et terme t, il existe au plus un type A tel que $\Gamma \vdash t : A$.

Il existe un algorithme qui étant donnés un contexte Γ et terme t, décide si t est typable dans Γ et donne son type, s'il existe.

Confluence: Si $t \hookrightarrow u_1$ et $t \hookrightarrow u_2$ alors il existe v tq $t \hookrightarrow v$.

Un terme *normal* est de la forme $\lambda x_1 : A_1. \ldots \lambda x_n : A_n.\ (x\ u_1 \ldots u_p)$.

t est *faiblement normalisable* si il existe v normal tq $t \hookrightarrow^* v$.

t est *fortement normalisable* si toute suite de réduction issue de t est finie.

Thm. Tout terme est fortement normalisable (pour β), et la forme normale est unique.

Thm. Tout terme est faiblement normalisable pour $\beta\eta$.

Déf. Un terme t de type $A_1 \rightarrow \ldots \rightarrow A_n \rightarrow B$ est en *forme β-normale η-longue* s'il est de la forme $\lambda x_1 : A_1. \ldots \lambda x_n : A_n.\ (x\ u_1 \ldots u_p)$, où les u_i sont en forme β-normale η-longue; Cette forme est unique.

Sémantique (interprétations de la logique)

Sémantique de Tarski: (tables de vérité)

A	B	$A \wedge B$	$A \vee B$	$A \Rightarrow B$	$\neg A$
t	t	t	t	t	f
f	t	f	t	t	t
t	f	f	t	f	
f	f	f	f	t	

Sémantique de Heyting-Kolmogorov:

- Une démonstration de $A \Rightarrow B$ est une fonction qui associe à toute démonstration de A une démonstration de B,
- Une démonstration de $A \wedge B$ est un couple formé d'une démonstration de A et d'une démonstration de B,
- Une démonstration de $A \vee B$ est un couple (i, p), où soit $i = 0$ et p est une démonstration de A, soit $i = 1$ et p est une démonstration de B,
- Une démonstration de $\neg A$ est une démonstration de $A \Rightarrow \bot$, où $\bot$ n'a aucune démonstration,
- Une démonstration de $\forall x.\ A$ est une fonction qui associe à tout objet t une démonstration de $A[t/x]$,
- Une démonstration de $\exists x.\ A$ est un couple (t, p), où t est un object, et p est une démonstration de $A[t/x]$.

L'isomorphisme de Curry-Howard

Les types:

Une formule de la logique *est représentée par un type* du λ-calcul:

- Les prop. atomiques $A, B, \cdots$ sont des types atomiques.
- À $A \Rightarrow B$ correspond $A \rightarrow B$, à $A \wedge B$ correspond $A \times B$.

Une preuve d'une formule A est un terme de type A:

Les termes:

logique	$\lambda -$ calcul
$\Gamma \vdash A \quad (A \in \Gamma)$	$x_1 : A_1 \ \cdots \ x_n : A_n] \vdash x_i : A_i$
$\dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}$	$\dfrac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x : A.\ t : A \rightarrow B}$
$\dfrac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}$	$\dfrac{\Gamma \vdash t : A \rightarrow B \quad \Gamma \vdash a : A}{\Gamma \vdash (t\ a) : B}$
$\dfrac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}$	$\dfrac{\Gamma \vdash a : A \quad \Gamma \vdash b : B}{\Gamma \vdash (a, b) : (A \times B)}$
$\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}$	$\dfrac{\Gamma \vdash t : (A \times B)}{\Gamma \vdash \mathrm{fst}\ t : A}$
$\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}$	$\dfrac{\Gamma \vdash t : (A \times B)}{\Gamma \vdash \mathrm{snd}\ t : B}$

Système F

types	A	$::= \alpha \mid A \rightarrow A' \mid \forall\alpha.A$
termes	M	$::= x \mid \lambda x : M.M' \mid \Lambda\alpha.M \mid (M\ M') \mid (M\ A)$
contextes	Γ	$::= . \mid \Gamma, x : A$

reductions $(\lambda x : A.M\ \ N) \hookrightarrow_\beta M[N/x]$
$(\Lambda\alpha.M\ \ A) \hookrightarrow_\beta M[A/\alpha]$

$$var\ \frac{x : A \in \Gamma}{\Gamma \vdash x : A}\ \Gamma\ legal$$

$$lam\ \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x : A.M\ :\ A \rightarrow B}$$

$$app\ \frac{\Gamma \vdash M : A \rightarrow B \qquad \Gamma \vdash N : A}{\Gamma \vdash (M\ N) : B}$$

$$Lam\ \frac{\Gamma \vdash M : A}{\Gamma \vdash \Lambda\alpha.M\ :\ \forall\alpha.A}\ \alpha \notin FV(\Gamma)$$

$$App\ \frac{\Gamma \vdash M : \forall\alpha.A}{\Gamma \vdash (M\ B) : A[B/\alpha]}\ B \in types$$

Exemples:
$\vdash \lambda x : A.x : A \rightarrow A$
$\vdash \Lambda\alpha.\lambda x : \alpha.x : \forall\alpha.\alpha \rightarrow \alpha$
$\vdash (\Lambda\alpha.\lambda x : \alpha.x\ \ A) : A \rightarrow A$

Termes et types dépendants

– les termes peuvent dépendre de termes et/ou de types

– les types peuvent dépendre de termes et/ou de types

– λ

$f : A \rightarrow B, x : A \vdash (f\ x) : B$
le terme $(f\ x)$ depend du terme x.

– F - polymorphisme

$f : \forall\alpha.\alpha \rightarrow \alpha \vdash (f\ A) : A \rightarrow A$
le terme $(f\ A)$ depend du type A.

– $\lambda_{\underline{\omega}}$ - polymorphisme au niveau des types

$f = \lambda\alpha : types.\alpha \rightarrow \alpha$ avec $(f\ A) = A \rightarrow A$
le type $(f\ A)$ depend du type A.

2 constantes: $*$ et $\square$ (*Set/Prop* et *Type*, ou bien *Type* et *Kind*).

'$\alpha \in types$' devient $\vdash \alpha : *$.

$f = \lambda\alpha : *.\alpha \rightarrow \alpha \in kinds$

$Kinds = \{*, * \rightarrow *, * \rightarrow * \rightarrow *, \cdots\}$.

$kinds\ \ K\ \ ::= * \mid K \rightarrow K'$

'$k \in kinds$' devient $\vdash k : \square$. en particulier $\vdash * : \square$.

$\lambda_{\underline{\omega}}$. types dép. de types

termes $M \;::= x \mid c \mid \lambda x : M.M' \mid (M\ M') \mid M \rightarrow M'$
sortes $s \;\in \{*, \Box\} \subset \mathcal{C}$

$$ax\ \vdash * : \Box \qquad var\ \frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash x : A}\ x \notin \Gamma$$

$$aff\ \frac{\Gamma \vdash A : B \qquad \Gamma \vdash C : s}{\Gamma, x : C \vdash A : B}\ x \notin \Gamma$$

$$lam\ \frac{\Gamma, x : A \vdash M : B \qquad \Gamma \vdash (A \rightarrow B) : s}{\Gamma \vdash \lambda x : A.M\ :\ A \rightarrow B}$$

$$app\ \frac{\Gamma \vdash M : A \rightarrow B \qquad \Gamma \vdash N : A}{\Gamma \vdash (M\ N) : B}$$

$$prod\ \frac{\Gamma \vdash A\ :\ s \qquad \Gamma \vdash B\ :\ s}{\Gamma \vdash A \rightarrow B\ :\ s}$$

$$conv\ \frac{\Gamma \vdash A\ :\ B \qquad \Gamma \vdash B'\ :\ s \qquad B =_\beta B'}{\Gamma \vdash A\ :\ B'}$$

Exemples:
$\alpha : *, \beta : * \vdash \alpha \rightarrow \beta : *$
$\vdash \lambda\alpha : *.(\alpha \rightarrow \alpha) : * \rightarrow * : \Box$
$list \quad : * \rightarrow * : \Box$
$nil \quad : \Pi A : *.(list\ A)$
$cons \ : \Pi A : *.A \rightarrow (list\ A) \rightarrow (list\ A)$

LF. types dép. de termes

termes $M \;::= x \mid c \mid \lambda x : M.M' \mid (M\ M') \mid \Pi x : M.M'$

$$ax\ \vdash * : \Box \qquad var\ \frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash x : A}\ x \notin \Gamma$$

$$aff\ \frac{\Gamma \vdash A : B \qquad \Gamma \vdash C : s}{\Gamma, x : C \vdash A : B}\ x \notin \Gamma$$

$$lam\ \frac{\Gamma, x : A \vdash M : B \qquad \Gamma \vdash (\Pi x : A.B) : s}{\Gamma \vdash \lambda x : A.M\ :\ \Pi x : A.B}$$

$$app\ \frac{\Gamma \vdash M : \Pi x : A.B \qquad \Gamma \vdash N : A}{\Gamma \vdash (M\ N) : B[N/x]}$$

$$prod\ \frac{\Gamma \vdash A\ :\ * \qquad \Gamma \vdash B\ :\ s}{\Gamma \vdash \Pi x : A.B\ :\ s}$$

$$conv\ \frac{\Gamma \vdash A\ :\ B \qquad \Gamma \vdash B'\ :\ s \qquad B =_\beta B'}{\Gamma \vdash A\ :\ B'}$$

$Kinds = \{*, * \rightarrow *, A \rightarrow *, \cdots\}$.

Exemples:
$even \quad : nat \rightarrow * : \Box$
$even_0 \quad : (even\ 0)$
$even_S \quad : \Pi n : nat.(even\ n) \rightarrow (even\ (S\ (S\ n)))$

$list_A \quad : nat \rightarrow * : \Box$
$nil_A \quad : (list_A\ 0)$
$cons_A \quad : \Pi n : nat.A \rightarrow (list_A\ n) \rightarrow (list_A\ (S\ n))$

Le λ-cube

termes $M \ ::= x \mid c \mid \lambda x : M.M' \mid (M\ M') \mid \Pi x : M.M'$

$$ax \ \ \vdash * : \Box \qquad var \ \frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash x : A} \ x \notin \Gamma$$

$$aff \ \frac{\Gamma \vdash A : B \qquad \Gamma \vdash C : s}{\Gamma, x : C \vdash A : B} \ x \notin \Gamma$$

$$lam \ \frac{\Gamma, x : A \vdash M : B \qquad \Gamma \vdash (\Pi x : A.B) : s}{\Gamma \vdash \lambda x : A.M \ : \ \Pi x : A.B}$$

$$app \ \frac{\Gamma \vdash M : \Pi x : A.B \qquad \Gamma \vdash N : A}{\Gamma \vdash (M\ N) : B[N/x]}$$

$$prod \ \frac{\Gamma \vdash A \ : \ s_1 \qquad \Gamma \vdash B \ : \ s_2}{\Gamma \vdash \Pi x : A.B \ : \ s_2}$$

$$conv \ \frac{\Gamma \vdash A \ : \ B \qquad \Gamma \vdash B' \ : \ s \qquad B =_\beta B'}{\Gamma \vdash A \ : \ B'}$$

$\{(s_1, s_2)\} \subseteq \{(*, *), (*, \Box), (\Box, *), (\Box, \Box)\}$

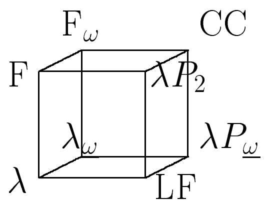
Exemple:

$list \ : * \to nat \to * : \Box$

$nil \ : \Pi A : *.(list\ A\ 0)$

$cons \ : \Pi A : *.\Pi n : nat.A \to (list\ A\ n) \to (list\ A\ (S\ n))$

Les λ-calculs du cube



λ	$(*, *)$				Church (40)
F	$(*, *)$	$(\Box, *)$			Girard (72), Reynolds (74)
LF	$(*, *)$		$(*, \Box)$		de Bruijn (70), HarperHP (87)
$\lambda_{\underline{\omega}}$	$(*, *)$			$(\Box, \Box)$	(polyrec) de Lavalette (91)
λP_2	$(*, *)$	$(\Box, *)$	$(*, \Box)$		Longo & Monggi (88)
F$_\omega$	$(*, *)$	$(\Box, *)$		$(\Box, \Box)$	Girard (72)
$\lambda P_{\underline{\omega}}$	$(*, *)$		$(*, \Box)$	$(\Box, \Box)$	
CC	$(*, *)$	$(\Box, *)$	$(*, \Box)$	$(\Box, \Box)$	Coquant & Huet (88)

Isomorphisme de Curry-Howard:

face gauche : logique propositionelle
face droite : types dép. de termes $(*, \Box)$. Calcul des prédicats
haut : termes dép. types $(\Box, *)$. Ordre sup. Polymorphisme.
fond : types dép. de types $(\Box, \Box)$

$CCind = CC + induction.$

Les Systèmes de Types Purs (PTS)

termes $M ::= x \mid c \mid \lambda x : M.M' \mid (M\ M') \mid \Pi x : M.M'$
Un PTS est défini par la donnée d'un triplet $(\mathcal{S}, \mathcal{A}, \mathcal{R})$ où $\mathcal{S} \subset \mathcal{C}$, $\mathcal{A}$ est un ens d'axiomes $(c : s)$ où $c \in \mathcal{C}$ et $s \in \mathcal{S}$, et $\mathcal{R}$ est un ens de triplets (s_1, s_2, s_3) avec $s_i \in \mathcal{S}$.

$$ax \quad \vdash c : s \quad (c : s) \in \mathcal{A}$$

$$var \quad \frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash x : A}\ x \notin \Gamma$$

$$aff \quad \frac{\Gamma \vdash A : B \qquad \Gamma \vdash C : s}{\Gamma, x : C \vdash A : B}\ x \notin \Gamma$$

$$lam \quad \frac{\Gamma, x : A \vdash M : B \qquad \Gamma \vdash (\Pi x : A.B) : s}{\Gamma \vdash \lambda x : A.M \ :\ \Pi x : A.B}$$

$$app \quad \frac{\Gamma \vdash M : \Pi x : A.B \qquad \Gamma \vdash N : A}{\Gamma \vdash (M\ N) : B[N/x]}$$

$$prod \quad \frac{\Gamma \vdash A\ :\ s_1 \qquad \Gamma \vdash B\ :\ s_2}{\Gamma \vdash \Pi x : A.B\ :\ s_3}\ (s_1, s_2, s_3) \in \mathcal{R}$$

$$conv \quad \frac{\Gamma \vdash A\ :\ B \qquad \Gamma \vdash B'\ :\ s \qquad B =_\beta B'}{\Gamma \vdash A\ :\ B'}$$

Le λ-cube est un sous-ens des PTS, où $(s_1, s_2) = (s_1, s_2, s_2)$.

Propriétés des PTS

Thm *(Subst)* $(\Gamma, x : A, \Gamma_1 \vdash B : C)$ & $(\Gamma \vdash M : A) \Rightarrow$
$\Gamma, \Gamma_1[M/x] \vdash B[M/x] : C[M/x]$.
Thm *(Aff.)* Si $\Gamma \subset \Gamma_1$, alors $\Gamma \vdash M : A \Rightarrow \Gamma_1 \vdash M : A$.
Thm *(Inversion)*

1. $\Gamma \vdash c : A \Rightarrow \exists s \in \mathcal{S}.\ A = s$ & $(c : s) \in \mathcal{A}$
2. $\Gamma \vdash x : A \Rightarrow \exists s \in \mathcal{S}. \exists A' = A.\ \Gamma \vdash A' : s$ & $(x : A') \in \Gamma$
3. $\Gamma \vdash \lambda x : A.M\ :\ C \Rightarrow \exists s \in \mathcal{S}. \exists B.$
   $C = \Pi x : A.B$ & $\Gamma, x : A \vdash M : B$ & $\Gamma \vdash (\Pi x : A.B) : s$
4. $\Gamma \vdash (M\ N) : C \Rightarrow \exists A, B.$
   $C = B[N/x]$ & $\Gamma \vdash M : \Pi x : A.B$ & $\Gamma \vdash N : A$
5. $\Gamma \vdash \Pi x : A.B\ :\ C \Rightarrow \exists (s_1, s_2, s_3) \in \mathcal{R}.$
   $C = s_3$ & $\Gamma \vdash A\ :\ s_1$ & $\Gamma, x : A \vdash B\ :\ s_2$

Thm *(SRT)* $\Gamma \vdash M : A$ & $M \hookrightarrow_* M' \Rightarrow \Gamma \vdash M' : A$.
Thm *(Unicité du type)* $\Gamma \vdash M : A$ & $\Gamma \vdash M : B \Rightarrow A = B$. (*)
Thm *(SN)* $\Gamma \vdash M : A \Rightarrow SN(M)$ & $SN(A)$.

(*) L'*unicité du type* n'est vraie que pour les PTS 'singly sorted'.
Def. Un PTS est *'singly sorted'* ssi:
$(c : s), (c, s') \in \mathcal{A} \Rightarrow s = s'$
$(s_1, s_2, s_3), (s_1, s_2, s_3') \in \mathcal{R} \Rightarrow s_3 = s_3'$

Déduction Naturelle dans LF

$$\Rightarrow_I\!: \frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B} \qquad \Rightarrow_E\!: \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}$$

$$\begin{array}{ll} Prop & : * \\ \Rightarrow & : Prop \rightarrow Prop \rightarrow Prop \\ T & : Prop \rightarrow * \\ \Rightarrow_i & : \Pi A, B : Prop.(T(A) \rightarrow T(B)) \rightarrow T(A \Rightarrow B) \\ \Rightarrow_e & : \Pi A, B : Prop.T(A \Rightarrow B) \rightarrow T(A) \rightarrow T(B) \end{array}$$

Exemples:

$\vdash (\Rightarrow_i \ A\ A\ (\lambda x : T(A).x)) : T(A \Rightarrow A)$ représente $\dfrac{A \vdash A}{\vdash A \Rightarrow A}$

$\vdash (\Rightarrow_i \cdots (\Rightarrow_i \cdots (\lambda x : T(A).\lambda y : T(B).x)) : T(A \Rightarrow (B \Rightarrow A))$

représente $\dfrac{\dfrac{A, B \vdash A}{A \vdash B \Rightarrow A}}{\vdash A \Rightarrow (B \Rightarrow A)}$

Déduction Naturelle dans LF *(suite)*

$$\neg_I : \frac{\Gamma, A \vdash \bot}{\Gamma \vdash \neg A} \qquad \neg_E : \frac{\Gamma \vdash A \quad \Gamma \vdash \neg A}{\Gamma \vdash \bot}(\bot_I) \qquad \bot_E : \frac{\Gamma \vdash \bot}{\Gamma \vdash A}$$

$$\wedge_I : \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \qquad \wedge_{E1} : \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A} \qquad \wedge_{E2} : \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}$$

$$\vee_{I1} : \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \qquad \vee_{I2} : \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}$$

$$\vee_E : \frac{\Gamma \vdash A \vee B \quad \Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma \vdash C}$$

$$\begin{array}{ll} tt, \bot & : Prop \\ \neg & : Prop \rightarrow Prop \\ \wedge, \vee & : Prop \rightarrow Prop \rightarrow Prop \end{array}$$

$$\begin{array}{ll} \neg_i & : \Pi A : Prop.(\Pi B : Prop.T(A) \rightarrow T(B)) \rightarrow T(\neg A) \\ \neg_e & : \Pi A : Prop.T(A) \rightarrow T(\neg A) \rightarrow \Pi B : Prop.T(B) \\ tt_i & : T(tt) \\ \bot_e & : \Pi A : Prop.T(\bot) \rightarrow T(A) \\ \wedge_i & : \Pi A, B : Prop.T(A) \rightarrow T(B) \rightarrow T(A \wedge B) \\ \wedge_{e1} & : \Pi A, B : Prop.T(A \wedge B) \rightarrow T(A) \\ \wedge_{e2} & : \Pi A, B : Prop.T(A \wedge B) \rightarrow T(B) \\ \vee_{i1} & : \Pi A, B : Prop.T(A) \rightarrow T(A \vee B) \\ \vee_{i2} & : \Pi A, B : Prop.T(B) \rightarrow T(A \vee B) \\ \vee_e & : \Pi A, B, C : Prop.T(A \vee B) \rightarrow (T(A) \rightarrow T(C)) \rightarrow \\ & \quad (T(B) \rightarrow T(C)) \rightarrow T(C) \end{array}$$

Déduction Naturelle dans LF *(suite)*

$$\forall_i : \frac{\Gamma \vdash P}{\Gamma \vdash \forall x.P}\ x \notin \Gamma \qquad \forall_e : \frac{\Gamma \vdash \forall x.P}{\Gamma \vdash P[a/x]}$$

Mieux écrit: $\forall_i : \dfrac{\Gamma \vdash P(t)}{\Gamma \vdash \forall x.P(x)}\ x \notin \Gamma \qquad \forall_e : \dfrac{\Gamma \vdash \forall x.P(x)}{\Gamma \vdash P(a)}$

On représente $\forall x.P(x)$ par $\forall(\lambda x : i.P(x))$.
La substitution est modélisée par l'application méta.

$$\begin{array}{ll}
\forall & : (var \rightarrow Prop) \rightarrow Prop \\
\forall_i & : \Pi P : var \rightarrow Prop.(\Pi x : var.T(P\ x)) \rightarrow T(\forall P) \\
\forall_e & : \Pi P : var \rightarrow Prop.T(\forall P) \rightarrow \Pi x : var.T(P\ x)
\end{array}$$

Plus de 'side condition' sur les variables
var objet = var méta

syntaxe abstraite d'ordre supérieur (hoas)

$$\exists_I : \frac{\Gamma \vdash A(t)}{\Gamma \vdash \exists x.\ A(x)} \qquad \exists_E : \frac{\Gamma \vdash \exists x.\ A(x) \qquad \Gamma, A \vdash B}{\Gamma \vdash B}\ (x \notin \Gamma, B)$$

$$\begin{array}{ll}
\exists & : (var \rightarrow Prop) \rightarrow Prop \\
\exists_i & : \Pi A : var \rightarrow Prop.\Pi x : var.T(A\ x) \rightarrow T(\exists A) \\
\exists_e & : \Pi A : var \rightarrow Prop.\Pi B : Prop.T(\exists A) \rightarrow \\
& \quad (\Pi x : var.T(A\ x) \rightarrow T(B)) \rightarrow T(A)
\end{array}$$

Bibliographie

References

[Bar91] Henk Barendregt. Lambda calculi with types. In Samson Abramsky, D. M. Gabbai, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2. Oxford University Press, 1991.

[HHP93] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *Journal of the Association for Computing Machinery*, 40(1):143–184, January 1993.

[PW99] Christine Paulin and Benjamin Werner. Introduction to Coq. Lectures notes, DEA Paris University and Types summer school'99, August 1999.