Au-delà des types simples

Lionel Vaux Auclair

I2M, université d'Aix-Marseille

M2 IMD

Enrichir le typage pour...

Représenter plus de fonctions calculables

Le système T

L'expressivité des termes simplement typés est très limitée.

Schwichtenberg a montré qu'on ne peut représenter que des polynômes un peu généralisés.

Le système T de Gödel est obtenu en ajoutant au λ-calcul simplement typé :

- un type de base $\overline{\mathbf{N}}$;
- des constantes de termes :

$$\begin{aligned} \overline{0} &: \overline{\mathbf{N}} \\ \overline{\mathsf{succ}} &: \overline{\mathbf{N}} \to \overline{\mathbf{N}} \\ \overline{\mathsf{rec}}_A &: A \to (\overline{\mathbf{N}} \to A \to A) \to \overline{\mathbf{N}} \to A \qquad \text{(pour chaque type } A\text{) ;} \end{aligned}$$

- des règles de réduction :

$$\overline{\mathsf{rec}}\, a\, f\, \overline{0} \rightsquigarrow a \qquad \text{et} \qquad \overline{\mathsf{rec}}\, a\, f\, (\overline{\mathsf{succ}}\, n) \rightsquigarrow f\ n\, (\overline{\mathsf{rec}}\, a\, f\ n)$$

Calculabilité dans le système T

- Les formes normales closes de type $\overline{\mathbf{N}}$ sont exactement les $\overline{n} := \overline{\mathsf{succ}}^n\,\overline{0}$.
- Avec $[\![\overline{\mathbf{N}}]\!] = \mathbf{N}$, l'interprétation ensembliste reste valide.
- On peut prouver que la réduction reste fortement normalisante : on ne code que des fonctions totales.
- Les fonctions $\mathbf{N}^k \to \mathbf{N}$ représentables sont exactement les fonctions prouvablement totales dans l'arithmétique de Peano.

C'est esquissé dans le chapitre 7 du *Proofs and types*.

PCF

On peut vouloir conserver un système de types, tout en retrouvant toutes les fonctions calculables.

Il suffit de rajouter au système T un opérateur de point fixe :

- une constante $Y_A : (A \to A) \to A$ pour chaque type A ;
- une règle de réduction $Y_A\, f \rightsquigarrow f\,(Y_A\, f)$.

C'est (à peu près) le langage PCF, un prototype de langage de programmation fonctionnelle.

(le pendant simplifié et théorique de la famille ML : SML, OCaml, F#)

Bien sûr on perd la normalisation ! $Y_A\, I_A \rightsquigarrow I_A\,(Y_A\, I_A) \rightsquigarrow Y_A\, I_A$.

Enrichir le typage pour...

Étendre la correspondance de Curry-Howard

Conjonction = produits

On peut étendre le système de types :

$$A, B, C ::= X \mid A \to B \mid A \times B \mid 1$$

et les termes :

$$s, t, \ldots ::= x \mid \lambda x.s \mid s\ t \mid \langle s, t \rangle \mid \pi_1 s \mid \pi_2 s \mid *$$

avec les règles :

$$\frac{s : A \qquad t : B}{\langle s, t \rangle : A \times B} \qquad \frac{s : A \times B}{\pi_1 s : A} \qquad \frac{s : A \times B}{\pi_2 s : B} \qquad \frac{}{* : 1}$$

et les réductions : $\pi_i \langle s_1, s_2 \rangle \rightsquigarrow s_i$.

- La réduction du sujet, la normalisation forte, l'interprétation fonctionnelle persistent.
- Modulo l'ajout de réductions *extensionnelles* (qui assurent par exemple que si $s : A \to B$ alors $s =_\beta \lambda x^A.t$), c'est le langage des catégories cartésiennes fermées,

Disjonction = sommes

On peut étendre le système de types :

$$A, B, C ::= X \mid A \to B \mid A \times B \mid 1 \mid A + B$$

et les termes :

$$s, t, \ldots ::= \cdots \mid \delta(s, x.t, y.u) \mid \iota_1 s \mid \iota_2 s$$

avec les règles :

$$\frac{s : A + B \qquad t : C \qquad u : C}{\delta(s, x^A.t, y^B.u) : C} \quad \frac{s : A}{\iota_1 s : A + B} \quad \frac{s : B}{\iota_2 s : A + B}$$

Et la réduction : $\delta(\iota_i s, x_1.t_1, x_2.t_2) \rightsquigarrow t_i[s/x_i]$.

- La réduction du sujet, la normalisation forte, l'interprétation fonctionnelle persistent.
- Les formes normales avec δ ne sont pas canoniques : il faut encore introduire des commutations ! C'est pas top. (*Proofs and types*, ch.10)
- Ça ne règle pas le cas de $\perp$, mais c'est une autre histoire...

Enrichir le typage pour...

Traiter la quantification

Types fonctionnels polymorphes

Les types du système F sont donnés par :

$$A, B, C, \ldots ::= X \mid A \to B \mid \forall X A.$$

On aura $s : \forall X A$ ssi $s : A[B/X]$ pour tout type B.
C'est :

- du *polymorphisme paramétrique* au sens de la programmation (Strachey) ;
- la quantification du second ordre en logique.

Le système F a été introduit simultanément et indépendamment avec ces deux motivations (Girard, 1972 ; Reynolds, 1974).

Le système F (à la Curry)

On étend le typage simple :

$$\frac{}{\Gamma, x : A \vdash x : A}\ (\mathit{var}) \qquad \frac{\Gamma, x : A \vdash s : B}{\Gamma \vdash \lambda x.s : A \to B}\ (\lambda)$$

$$\frac{\Gamma \vdash s : A \to B \qquad \Gamma \vdash t : A}{\Gamma \vdash s\,t : B}\ (\mathit{app})$$

avec le polymorphisme :

$$\frac{\Gamma \vdash s : A \qquad X \notin \mathrm{VL}(\Gamma)}{\Gamma \vdash s : \forall X A}\ (\forall_i) \qquad \frac{\Gamma \vdash s : \forall X A}{\Gamma \vdash s : A[B/X]}\ (\forall_e)$$

Exemples

$$\begin{aligned}
\vdash \lambda x.x &: \forall X(X \to X)\\
\vdash \lambda x.\lambda y.x &: \forall X \forall Y(X \to Y \to X)\\
\vdash \lambda x.\lambda y.y &: \forall X \forall Y(X \to Y \to Y)\\
\vdash \lambda f.\lambda x.f\,x &: \forall X \forall Y((X \to Y) \to X \to Y)\\
\vdash \lambda f.\lambda g.\lambda x.g\,(f\,x) &: \forall X \forall Y \forall Z((X \to Y) \to (Y \to Z) \to X \to Z)\\
\vdash \lambda f.\lambda g.\lambda x.f\,x\,(g\,x) &: \forall X \forall Y \forall Z((X \to Y \to Z) \to (X \to Y) \to X \to Z)\\
\vdash \lambda f.\lambda x.f\,(f\,x) &: \forall X((X \to X) \to X \to X)
\end{aligned}$$

Exemples

$$\vdash \lambda x.x : \forall X(X \to X)$$
$$\vdash \lambda x.\lambda y.x : \forall X \forall Y(X \to Y \to X)$$
$$\vdash \lambda x.\lambda y.y : \forall X \forall Y(X \to Y \to Y)$$
$$\vdash \lambda f.\lambda x.f\,x : \forall X \forall Y((X \to Y) \to X \to Y)$$
$$\vdash \lambda f.\lambda g.\lambda x.g\,(f\,x) : \forall X \forall Y \forall Z((X \to Y) \to (Y \to Z) \to X \to Z)$$
$$\vdash \lambda f.\lambda g.\lambda x.f\,x\,(g\,x) : \forall X \forall Y \forall Z((X \to Y \to Z) \to (X \to Y) \to X \to Z)$$
$$\vdash \lambda f.\lambda x.f\,(f\,x) : \forall X((X \to X) \to X \to X)$$

Problème

Le typage n'est plus guidé par la syntaxe des termes !

- On est sortis de Curry–Howard.
- La typabilité et la vérification de type sont indécidables !

(J.B. Wells, 1998)

Le système F (à la Church)

On ajoute des constructeurs pour les règles du $\forall$:

$$\Lambda_2 \ni s, t, \ldots ::= x^A \mid \lambda x^A.s \mid s\,t \mid \Pi X.s \mid s[B]$$

avec le typage :

$$\frac{}{x^A : A}\,(var) \qquad \frac{s : B}{\lambda x^A.s : A \to B}\,(\lambda) \qquad \frac{s : A \to B \qquad t : A}{s\,t : B}\,(app)$$

$$\frac{s : A \qquad \left(X \notin \bigcup_{y^B \in \mathbf{VL}(s)} \mathrm{VL}_2(B)\right)}{\Pi X.s : \forall X A}\,(\forall_i) \qquad \frac{\Gamma \vdash s : \forall X A}{\Gamma \vdash s[B] : A[B/X]}\,(\forall_e)$$

où $\mathrm{VL}_2(B)$ est l'ensemble des variables de types libres dans B.

Le système F (à la Church)

On ajoute des constructeurs pour les règles du $\forall$:

$$\Lambda_2 \ni s, t, \ldots ::= x^A \mid \lambda x^A.s \mid s\,t \mid \Pi X.s \mid s[B]$$

avec le typage :

$$\frac{}{x^A : A}\ (var) \qquad \frac{s : B}{\lambda x^A.s : A \to B}\ (\lambda) \qquad \frac{s : A \to B \qquad t : A}{s\,t : B}\ (app)$$

$$\frac{s : A \qquad \left(X \notin \bigcup_{y^B \in \mathbf{VL}(s)} \mathrm{VL}_2(B)\right)}{\Pi X.s : \forall X A}\ (\forall_i) \qquad \frac{\Gamma \vdash s : \forall X A}{\Gamma \vdash s[B] : A[B/X]}\ (\forall_e)$$

où $\mathrm{VL}_2(B)$ est l'ensemble des variables de types libres dans B.

Et on étend la β-réduction avec $(\Pi X.s)[B] \rightsquigarrow s[B/X]$.

Exemples

$$
\begin{aligned}
\Pi X.\lambda x^X.x &: \forall X(X \to X) \\
\Pi X.\Pi Y.\lambda x^X.\lambda y^Y.x &: \forall X \forall Y(X \to Y \to X) \\
\Pi X.\Pi Y.\lambda x^X.\lambda y^Y.y &: \forall X \forall Y(X \to Y \to Y) \\
\Pi X.\Pi Y.\lambda f^{X \to Y}.\lambda x^X.f\,x &: \forall X \forall Y((X \to Y) \to X \to Y) \\
\Pi X.\Pi Y.\Pi Z.\lambda f^{X \to Y}.\lambda g^{Y \to Z}.\lambda x^X.g\,(f\,x) &: \forall X \forall Y \forall Z((X \to Y) \to (Y \to Z) \to X \to Z) \\
\Pi X.\Pi Y.\Pi Z.\lambda f^{X \to Y \to Z}.\lambda g^{X \to Y}.\lambda x^X.f\,x\,(g\,x) &: \forall X \forall Y \forall Z((X \to Y \to Z) \to (X \to Y) \to X \to Z) \\
\Pi X.\lambda f^{X \to X}.\lambda x^X.f\,(f\,x) &: \forall X((X \to X) \to X \to X)
\end{aligned}
$$

Exemples

$$\Pi X.\lambda x^X.x : \forall X(X \to X)$$
$$\Pi X.\Pi Y.\lambda x^X.\lambda y^Y.x : \forall X\forall Y(X \to Y \to X)$$
$$\Pi X.\Pi Y.\lambda x^X.\lambda y^Y.y : \forall X\forall Y(X \to Y \to Y)$$
$$\Pi X.\Pi Y.\lambda f^{X\to Y}.\lambda x^X.f\,x : \forall X\forall Y((X \to Y) \to X \to Y)$$
$$\Pi X.\Pi Y.\Pi Z.\lambda f^{X\to Y}.\lambda g^{Y\to Z}.\lambda x^X.g\,(f\,x) : \forall X\forall Y\forall Z((X \to Y) \to (Y \to Z) \to X \to Z)$$
$$\Pi X.\Pi Y.\Pi Z.\lambda f^{X\to Y\to Z}.\lambda g^{X\to Y}.\lambda x^X.f\,x\,(g\,x) : \forall X\forall Y\forall Z((X \to Y \to Z) \to (X \to Y) \to X \to Z)$$
$$\Pi X.\lambda f^{X\to X}.\lambda x^X.f\,(f\,x) : \forall X((X \to X) \to X \to X)$$

Oui, c'est lourd.

Δ

On type plus de monde :

$$
\dfrac{\dfrac{\dfrac{}{x : \forall X X}\ (var)}{x[\forall X X \to \forall X X] : \forall X X \to \forall X X}\ (inst) \qquad \dfrac{}{x : \forall X X}\ (var)}{\dfrac{x[\forall X X \to \forall X X]\, x : \forall X X}{\lambda x^{\forall X X}.x[\forall X X \to \forall X X]\, x : \forall X X \to \forall X X}\ (\lambda)}\ (app)
$$

Normalisation

Théorème

La réduction du système F est confluente.

Démonstration: Comme en λ-calcul. $\square$

Théorème

Les termes bien typés du système F sont fortement normalisables.

- La démonstration ne peut pas se faire par des moyens purement arithmétiques, pour de bonnes raisons qu'on évoquera ensuite.
- On utilise une technique de réductibilité.

On verra ça sur un autre exemple.

Qui est typable dans le système F ?

On a déjà énoncé que les termes typés sont fortement normalisables.

Lemme

Toute forme normale du λ-calcul pur est typable dans système F à la Curry.

Démonstration: Il suffit de typer toutes les variables avec le type $0 = \forall X X$. En effet :

- Si $x \in \mathrm{supp}(\Gamma)$, $\Gamma(x) = 0$ et $\Gamma \vdash t_i : A_i$ pour $1 \leq i \leq k$ alors $\Gamma \vdash x : A_1 \to \cdots \to A_n \to B$ pour n'importe quel type B,
  donc $\Gamma \vdash x\, t_1 \cdots t_k : B$, donc $\lambda x_1. \cdots \lambda x_n.x\, t_1 \cdots t_k$ est typable.
- Donc toute forme normale s est typable dans n'importe quel contexte de la forme $y_1 : 0, \ldots, y_l : 0$ avec $\{y_1, \ldots, y_l\} \supset \mathrm{VL}(s)$

$\square$

Par contre il existe des termes fortement normalisables mais non typables dans F.

(ils sont difficiles à trouver)

Codage des types de base : $\mathbf{B}$ et $\mathbf{N}$

- Les booléens $\underline{\text{vrai}}$ et $\underline{\text{faux}}$ sont exactement les formes normales closes (à la Curry) du type $\underline{\mathbf{B}} := \forall X(X \to X \to X)$.
- Les entiers de Church sont exactement les formes normales closes (à la Curry) du type $\underline{\mathbf{N}} := \forall X((X \to X) \to X \to X)$.

Il faut ajouter I.

- On peut coder le test sur les booléens, les opérations sur les entiers, la récursion primitive, etc., *avec les bons types !*

$$\underline{\text{rec}} : \forall X(X \to (\underline{\mathbf{N}} \to X \to X) \to \underline{\mathbf{N}} \to X)$$

- La réduction du système F simule celle du système T.

Codage des connecteurs : $\wedge$ et $\vee$

- On peut coder les types produits et sommes :

$$\begin{aligned} A \times B &:= \forall X((A \to B \to X) \to X) \\ 1 &:= \forall X(X \to X) \\ A + B &:= \forall X((A \to X) \to (B \to X) \to X) \\ 0 &:= \forall X X \end{aligned}$$

- On peut représenter les règles/constructeurs :
  - $\langle -, - \rangle : \forall X \forall Y (X \to Y \to X \times Y)$
  - $\iota_1 : \forall X \forall Y (X \to X + Y)$
  - ...
- La réduction du système F simule l'élimination des coupures en déduction naturelle intuitionniste.

Cohérence de l'arithmétique du second ordre

- Il est assez facile de voir que les fonctions représentables dans F sont prouvablement totales dans l'arithmétique du second ordre.

  À la louche : l'arithmétique de Peano avec quantification sur les ensembles d'entiers.
- La réciproque est valide. Voir le ch.15 de *Proofs and types* pour avoir une idée.
- On peut ramener la cohérence de l'arithmétique du second ordre à celle de sa version intuitionniste *via* une traduction par doubles négations. *Cf.* un DM.
- On peut coder cette version dans le système F.

  C'est un peu tordu à cause du premier ordre.
- La cohérence de l'arithmétique du second ordre se déduit de celle de F :

  Il n'y a pas de terme clos de type 0.
- Et c'est une conséquence de la normalisation !

  C'était la conjecture de Takeuti, résolue par Girard.

Enrichir le typage pour...

Caractériser des propriétés calculatoires de λ-termes

Types avec intersection

On autorise les fonctions à exiger que leur argument satisfassent plusieurs types :

$$A, B, \ldots ::= X \mid \tilde{A} \to B$$
$$\tilde{A}, \tilde{B}, \ldots ::= A_1 \cap \cdots \cap A_n$$

avec les règles :

$$\frac{}{\Gamma, x : A_1 \cap \cdots \cap A_n \vdash x : A_i}\ (var) \qquad \frac{\Gamma, x : \tilde{A} \vdash s : B}{\Gamma \vdash \lambda x.s : \tilde{A} \to B}\ (\lambda)$$

$$\frac{\Gamma \vdash s : A_1 \cap \cdots \cap A_n \to B \qquad \Gamma \vdash t : A_1 \quad \cdots \quad \Gamma \vdash t : A_n}{\Gamma \vdash s\, t : B}\ (app)$$

Types avec intersection

On autorise les fonctions à exiger que leur argument satisfassent plusieurs types :

$$A, B, \ldots ::= X \mid \tilde{A} \to B$$
$$\tilde{A}, \tilde{B}, \ldots ::= A_1 \cap \cdots \cap A_n$$

avec les règles :

$$\frac{}{\Gamma, x : A_1 \cap \cdots \cap A_n \vdash x : A_i}\ (var) \qquad \frac{\Gamma, x : \tilde{A} \vdash s : B}{\Gamma \vdash \lambda x.s : \tilde{A} \to B}\ (\lambda)$$

$$\frac{\Gamma \vdash s : A_1 \cap \cdots \cap A_n \to B \qquad \Gamma \vdash t : A_1 \quad \cdots \quad \Gamma \vdash t : A_n}{\Gamma \vdash s\, t : B}\ (app)$$

- On note $\Gamma \vdash t : A_1 \cap \cdots \cap A_n$ si $\Gamma \vdash t : A_i$ pour $1 \leq i \leq n$.
- C'est un genre de conjonction mais pas au sens de Curry–Howard : il n'y a pas de version à la Church.
- On note ω l'intersection vide. En particulier $t : \omega$ pour tout t.

Exemples

Si $A \in \tilde{A}$ (*i.e.* $\tilde{A} = A_1 \cap \cdots \cap A_n$ et $A = A_i$) :

$$\dfrac{\dfrac{}{x : \tilde{A} \vdash x : A}\ (var)}{\vdash I = \lambda x.x : \tilde{A} \to A}\ (\lambda)$$

Exemples

Si $A \in \tilde{A}$ (*i.e.* $\tilde{A} = A_1 \cap \cdots \cap A_n$ et $A = A_i$) :

$$\dfrac{\dfrac{}{x : \tilde{A} \vdash x : A}\ (var)}{\vdash I = \lambda x.x : \tilde{A} \to A}\ (\lambda)$$

$$\dfrac{\dfrac{\dfrac{}{x : (\omega \to A) \vdash x : \omega \to A}\ (var)}{x : (\omega \to A) \vdash x\,x : A}\ (app)}{\vdash \Delta = \lambda x.x\,x : (\omega \to A) \to A}\ (\lambda)$$

Exemples

Si $A \in \tilde{A}$ (*i.e.* $\tilde{A} = A_1 \cap \cdots \cap A_n$ et $A = A_i$) :

$$\dfrac{\dfrac{}{x : \tilde{A} \vdash x : A}\ (var)}{\vdash I = \lambda x.x : \tilde{A} \to A}\ (\lambda)$$

$$\dfrac{\dfrac{\dfrac{}{x : (\omega \to A) \vdash x : \omega \to A}\ (var)}{x : (\omega \to A) \vdash x\, x : A}\ (app)}{\vdash \Delta = \lambda x.x\, x : (\omega \to A) \to A}\ (\lambda)$$

$$\dfrac{\dfrac{x : \omega \vdash I : \tilde{A} \to A}{\vdash \underline{0} = \lambda x.I : \omega \to \tilde{A} \to A}\ (\lambda)}{\vdash \underline{0}\, \Omega : \tilde{A} \to A}\ (app)$$

Substitution

Lemme

Si $\Gamma, x : \tilde{A} \vdash s : B$ *et* $\Gamma \vdash t : \tilde{A}$ *alors* $\Gamma \vdash s[t/x] : B$.

Démonstration: Par induction sur s, comme pour les types simples. $\square$

Substitution

Lemme

Si $\Gamma, x : \tilde{A} \vdash s : B$ *et* $\Gamma \vdash t : \tilde{A}$ *alors* $\Gamma \vdash s[t/x] : B$.

Démonstration: Par induction sur s, comme pour les types simples. $\square$

Lemme

Si $\Gamma \vdash s[t/x] : B$ *alors il existe* $\tilde{A}$ *tel que* $\Gamma, x : \tilde{A} \vdash s : B$ *et* $\Gamma \vdash t : \tilde{A}$.

Démonstration: Par induction sur s :

- Si $s = x$: on prend $\tilde{A} = B$.
- Si $s = y \neq x$: on prend $\tilde{A} = \omega$.
- Si $s = \lambda x.u$: on applique l'hypothèse d'induction directement.
- Si $s = u_1\, u_2$: l'hypothèse d'induction donne $\tilde{A}_1$ et $\tilde{A}_2$ et on pose $\tilde{A} = \tilde{A}_1 \cap \tilde{A}_2$.

$\square$

Sémantique dénotationnelle

Théorème

Si $s \rightarrow_\beta s'$ *alors* $\Gamma \vdash s : A$ *ssi* $\Gamma \vdash s' : A$.

Démonstration: Par induction sur s, en utilisant les lemmes de substitution pour le redex. $\square$

Corollaire

En posant

$$[\![s]\!]_{\vec{x}} = \{(\tilde{A}_1, \ldots, \tilde{A}_n, B) \text{ t.q. } x_1 : A_1, \ldots, x_n : A_n \vdash s : B\}$$

on obtient une sémantique dénotationnelle du λ*-calcul pur.*

Un terme est typable ssi sa sémantique est non vide.

Caractérisation de la normalisabilité de tête

Lemme

Toute forme normale de tête est typable.

Démonstration: On a

$$\Gamma, x : \overbrace{\omega \to \cdots \to \omega}^{k} \to A \vdash x\, t_1 \cdots t_k : A$$

pour tous termes $t_1, \ldots, t_k$ et tout type A. $\square$

Corollaire

Tout terme normalisable de tête est typable (i.e. sa sémantique est non vide)

Réciproquement :

Théorème

Tout terme typable est normalisable de tête.

Caractérisation de la normalisabilité de tête

Lemme

Toute forme normale de tête est typable.

Démonstration: On a

$$\Gamma, x : \overbrace{\omega \to \cdots \to \omega}^{k} \to A \vdash x\, t_1 \cdots t_k : A$$

pour tous termes $t_1, \ldots, t_k$ et tout type A. $\square$

Corollaire

Tout terme normalisable de tête est typable (i.e. sa sémantique est non vide)

Réciproquement :

Théorème

Tout terme typable est normalisable de tête.

Prouvons-le.

Candidats

On note :

$$N := \{x\, t_1 \cdots t_k \mid k \in \mathbf{N},\ t_1, \ldots, t_k \in \Lambda\}$$
$$H := \{s \in \Lambda \mid s \text{ normalisable de tête}\}$$

Un candidat est un ensemble A de termes tel que :

- $N \subset A \subset H$;
- si $(s[t_0/x])\, t_1 \cdots t_n \in A$ alors $(\lambda x.s)\, t_0\, t_1 \cdots t_n \in A$.

En particulier, H est un candidat.

Réductibilité

Si A et B sont des ensembles de termes alors on pose $A \to B := \{s \mid \forall t \in A,\, s\,t \in B\}$.

Lemme

Si A et B sont des candidats alors $A \to B$ et $A \cap B$ aussi.

Démonstration: Pour $A \cap B$ c'est direct. Pour $A \to B$:

- si $s \in N$ alors, pour tout $t \in A$, $s\,t \in N$ donc $s\,t \in B$: donc $N \subset A \to B$;
- si $s \in A \to B$, on fixe une variable x quelconque et alors $x \in N \subset A$ donc $s\,x \in B$ donc $s\,x$ est normalisable de tête ; on déduit facilement que s aussi ; donc $A \to B \subset H$;
- si $u' = (s[t_0/x])\,t_1 \cdots t_n \in A \to B$ on doit montrer que $u = (\lambda x.s)\,t_0\,t_1 \cdots t_n \in A \to B$: pour tout $t \in A$, on a $u'\,t = (s[t_0/x])\,t_1 \cdots t_n\,t \in B$; or $u\,t = (\lambda x.s)\,t_0\,t_1 \cdots t_n \in B$ par l'hypothèse sur B, et donc $u \in A \to B$.

$\square$

Adéquation

On interprète les types par des candidats : (en particulier $\omega^* = \Lambda$)

$$X^* := H$$
$$(\tilde{A} \to B)^* := \tilde{A}^* \to B^*$$
$$(A_1 \cap \cdots \cap A_n)^* := A_1^* \cap \cdots \cap A_n^*$$

Lemme

Si $x_1 : \tilde{A}_1, \ldots, x_n : \tilde{A}_n \vdash s : B$ alors pour tous termes $t_1 \in \tilde{A}_1^*, \ldots, t_n \in \tilde{A}_n^*$ on a $s[t_1/x_1] \cdots [t_n/x_n] \in B^*$.

Démonstration: Par induction sur s, en notant $s' = s[t_1/x_1] \cdots [t_n/x_n]$:

- si $s = x_i$: $B^* \supset \tilde{A}_i^*$, $s' = t_i$ et on conclut directement ;
- si $s = uv$: l'hypothèse d'induction donne $u' \in \tilde{A}^* \to B^*$ et $v' \in \tilde{A}^*$ donc $u'\, v' \in B^*$;
- si $s = \lambda x.u$ et $B = \tilde{C} \to D$: pour tout $t \in \tilde{C}^*$, l'hypothèse d'induction donne $u'[t/x] \in D^*$ et comme D^* est un candidat on a $(\lambda x.u)\, t \in D^*$; et donc $s \in \tilde{C}^* \to D^*$.

□

Caractérisation de la normalisabilité de tête

Corollaire

Si $\Gamma \vdash s : A$ *alors* $s \in H$*.*

Démonstration: Il suffit de remarquer que $x_i \in N \subset \tilde{A}_i^*$. □

Théorème

Un terme s *est typable avec intersections ssi* s *est normalisable de tête ssi* $[\![s]\!] \neq \emptyset$*.*

Ça s'adapte :

- à la normalisabilité : il faut interdire ω ;
- à la normalisabilité forte : il faut restreindre encore la forme des types.

Chapitres 3 et 4 du Krivine.

À suivre. . .