λ-calcul typé

Lionel Vaux Auclair

I2M, université d'Aix-Marseille

M2 IMD

Types fonctionnels

On considère une grammaire de types fonctionnels simples :

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

où X varie dans un ensemble de types de base.

λ-termes simplement typés (à la Church)

- On fixe pour chaque A un ensemble infini $\mathcal{V}_A$ de variables, de sorte que les $\mathcal{V}_A$ forment une partition de $\mathcal{V}$.
- On annote x^A la variable x pour indiquer que $x \in \mathcal{V}_A$.

Définition

On définit la relation de typage, qu'on note $t : A$ pour « t est de type A » inductivement comme suit :

- si $x \in \mathcal{V}_A$ alors $x : A$;
- si $s : B$ alors $\lambda x^A.s : A \to B$;
- si $s : A \to B$ et $t : A$ alors $s\,t : B$.

Cette relation est *fonctionnelle* : s'il existe A tel que $t : A$ alors on dit que t est bien typé et de type A, et dans ce A est uniquement défini et on peut noter $\mathrm{type}(t) := A$.

Dans le cas contraire, on dit que t est mal typé.

Règles de typage (à la Church)

La définition précédente revient à se donner les règles suivantes :

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

La seule construction pour laquelle le typage peut échouer est l'application.

Exemples

$$
\begin{aligned}
I_A &:= \lambda x^A.x &&: \quad A \to A\\
K_{A,B} &:= \lambda x^A.\lambda y^B.x &&: \quad A \to B \to A\\
\underline{0}_{A,B} &:= \lambda x^A.\lambda y^B.y &&: \quad A \to B \to B\\
\underline{1}_{A,B} &:= \lambda f^{A\to B}.\lambda x^A.f\,x &&: \quad (A \to B) \to A \to B\\
\circ_{A,B,C} &:= \lambda g^{A\to B}.\lambda f^{B\to C}.\lambda x^A.f\,(g\,x) &&: \quad (A \to B) \to (B \to C) \to A \to C\\
S_{A,B,C} &:= \lambda f^{A\to B\to C}.\lambda g^{A\to B}.\lambda x^A.f\,x\,(g\,x) &&: \quad (A \to B \to C) \to (A \to B) \to A \to C\\
\underline{2}_A &:= \lambda f^{A\to A}.\lambda x^A.f\,(f\,x) &&: \quad (A \to A) \to A \to A
\end{aligned}
$$

Exemples

$$
\begin{aligned}
I_A &\coloneqq \lambda x^A.x && : \quad A \to A \\
K_{A,B} &\coloneqq \lambda x^A.\lambda y^B.x && : \quad A \to B \to A \\
\underline{0}_{A,B} &\coloneqq \lambda x^A.\lambda y^B.y && : \quad A \to B \to B \\
\underline{1}_{A,B} &\coloneqq \lambda f^{A\to B}.\lambda x^A.f\,x && : \quad (A \to B) \to A \to B \\
\circ_{A,B,C} &\coloneqq \lambda g^{A\to B}.\lambda f^{B\to C}.\lambda x^A.f\,(g\,x) && : \quad (A \to B) \to (B \to C) \to A \to C \\
S_{A,B,C} &\coloneqq \lambda f^{A\to B\to C}.\lambda g^{A\to B}.\lambda x^A.f\,x\,(g\,x) && : \quad (A \to B \to C) \to (A \to B) \to A \to C \\
\underline{2}_A &\coloneqq \lambda f^{A\to A}.\lambda x^A.f\,(f\,x) && : \quad (A \to A) \to A \to A
\end{aligned}
$$

$\leadsto$ Construisez les dérivations de typage correspondantes

Exemples

$$\begin{array}{rlcl}
I_A & := \lambda x^A.x & : & A \to A \\
K_{A,B} & := \lambda x^A.\lambda y^B.x & : & A \to B \to A \\
\underline{0}_B & := \lambda x^{B\to B}.\lambda y^B.y & : & (B \to B) \to B \to B \\
\underline{1}_{A,B} & := \lambda f^{A\to B}.\lambda x^A.f\,x & : & (A \to B) \to A \to B \\
\circ_{A,B,C} & := \lambda g^{A\to B}.\lambda f^{B\to C}.\lambda x^A.f\,(g\,x) & : & (A \to B) \to (B \to C) \to A \to C \\
S_{A,B,C} & := \lambda f^{A\to B\to C}.\lambda g^{A\to B}.\lambda x^A.f\,x\,(g\,x) & : & (A \to B \to C) \to (A \to B) \to A \to C \\
\underline{2}_A & := \lambda f^{A\to A}.\lambda x^A.f\,(f\,x) & : & (A \to A) \to A \to A
\end{array}$$

Exemples

$$
\begin{aligned}
I_A &:= \lambda x^A.x &&: \quad A \to A\\
K_{A,B} &:= \lambda x^A.\lambda y^B.x &&: \quad A \to B \to A\\
\underline{0}_B &:= \lambda x^{B\to B}.\lambda y^B.y &&: \quad (B \to B) \to B \to B\\
\underline{1}_B &:= \lambda f^{B\to B}.\lambda x^B.f\,x &&: \quad (B \to B) \to B \to B\\
\circ_{A,B,C} &:= \lambda g^{A\to B}.\lambda f^{B\to C}.\lambda x^A.f\,(g\,x) &&: \quad (A \to B) \to (B \to C) \to A \to C\\
S_{A,B,C} &:= \lambda f^{A\to B\to C}.\lambda g^{A\to B}.\lambda x^A.f\,x\,(g\,x) &&: \quad (A \to B \to C) \to (A \to B) \to A \to C\\
\underline{2}_A &:= \lambda f^{A\to A}.\lambda x^A.f\,(f\,x) &&: \quad (A \to A) \to A \to A
\end{aligned}
$$

Exemples

$$
\begin{aligned}
I_A &:= \lambda x^A.x && : \quad A \to A \\
K_{A,B} &:= \lambda x^A.\lambda y^B.x && : \quad A \to B \to A \\
\underline{0}_B &:= \lambda x^{B\to B}.\lambda y^B.y && : \quad (B \to B) \to B \to B \\
\underline{1}_B &:= \lambda f^{B\to B}.\lambda x^B.f\,x && : \quad (B \to B) \to B \to B \\
\circ_{A,B,C} &:= \lambda g^{A\to B}.\lambda f^{B\to C}.\lambda x^A.f\,(g\,x) && : \quad (A \to B) \to (B \to C) \to A \to C \\
S_{A,B,C} &:= \lambda f^{A\to B\to C}.\lambda g^{A\to B}.\lambda x^A.f\,x\,(g\,x) && : \quad (A \to B \to C) \to (A \to B) \to A \to C \\
\underline{2}_B &:= \lambda f^{B\to B}.\lambda x^B.f\,(f\,x) && : \quad (B \to B) \to B \to B
\end{aligned}
$$

Exemples

$$
\begin{array}{rcll}
I_A & := \lambda x^A.x & : & A \to A \\
K_{A,B} & := \lambda x^A.\lambda y^B.x & : & A \to B \to A \\
\underline{0}_B & := \lambda x^{B \to B}.\lambda y^B.y & : & (B \to B) \to B \to B \\
\underline{1}_B & := \lambda f^{B \to B}.\lambda x^B.f\,x & : & (B \to B) \to B \to B \\
\circ_{A,B,C} & := \lambda g^{A \to B}.\lambda f^{B \to C}.\lambda x^A.f\,(g\,x) & : & (A \to B) \to (B \to C) \to A \to C \\
S_{A,B,C} & := \lambda f^{A \to B \to C}.\lambda g^{A \to B}.\lambda x^A.f\,x\,(g\,x) & : & (A \to B \to C) \to (A \to B) \to A \to C \\
\underline{2}_B & := \lambda f^{B \to B}.\lambda x^B.f\,(f\,x) & : & (B \to B) \to B \to B \\
\underline{\Delta}_A & := \lambda x^A.x\,x & & \text{toujours mal typé !}
\end{array}
$$

Exemples

$$
\begin{aligned}
I_A &:= \lambda x^A.x & &: \quad A \to A \\
K_{A,B} &:= \lambda x^A.\lambda y^B.x & &: \quad A \to B \to A \\
\underline{0}_B &:= \lambda x^{B\to B}.\lambda y^B.y & &: \quad (B \to B) \to B \to B \\
\underline{1}_B &:= \lambda f^{B\to B}.\lambda x^B.f\,x & &: \quad (B \to B) \to B \to B \\
\circ_{A,B,C} &:= \lambda g^{A\to B}.\lambda f^{B\to C}.\lambda x^A.f\,(g\,x) & &: \quad (A \to B) \to (B \to C) \to A \to C \\
S_{A,B,C} &:= \lambda f^{A\to B\to C}.\lambda g^{A\to B}.\lambda x^A.f\,x\,(g\,x) & &: \quad (A \to B \to C) \to (A \to B) \to A \to C \\
\underline{2}_B &:= \lambda f^{B\to B}.\lambda x^B.f\,(f\,x) & &: \quad (B \to B) \to B \to B \\
\underline{\Delta}_A &:= \lambda x^A.x\,x & & \quad \text{toujours mal typé !}
\end{aligned}
$$

$\leadsto$ Pourquoi ?

β-réduction typée

Lemme

Si $s : B$ et $t : A$ alors $s[t/x^A] : B$.

Démonstration: Par induction sur s. $\square$

Corollaire

Si $s : A$ et $s \rightarrow_\beta s'$ alors $s' : A$.

Notez que la réciproque ne tient pas.

β-réduction typée

Lemme

Si $s : B$ et $t : A$ alors $s[t/x^A] : B$.

Démonstration: Par induction sur s. $\square$

Corollaire

Si $s : A$ et $s \rightarrow_\beta s'$ alors $s' : A$.

Notez que la réciproque ne tient pas.

$\rightsquigarrow$ Trouvez des termes tels que $s \rightarrow_\beta t$ avec t bien typé et s mal typé

Interprétation ensembliste

- On fixe un ensemble $[\![X]\!]$ pour chaque X.
- On interprète chaque type de manière fonctionnelle :

$$[\![A \to B]\!] := [\![B]\!]^{[\![A]\!]}$$

  où F^E désigne l'ensemble des fonctions (totales) de E dans F.
- À tout $t : B$ avec $\mathrm{VL}(t) \in \{x_1^{A_1}, \ldots, x_n^{A_n}\}$, on associe une fonction $[\![t]\!]_{x_1,\ldots,x_n} : [\![A_1]\!] \times \cdots \times [\![A_n]\!] \to [\![B]\!]$:

$$\begin{aligned}
[\![x_i]\!]_{x_1,\ldots,x_n} : [\![A_1]\!] \times \cdots \times [\![A_n]\!] &\to [\![A_i]\!] \\
(a_1, \ldots, a_n) &\mapsto a_i \\
[\![\lambda x^A.s]\!]_{x_1,\ldots,x_n} : [\![A_1]\!] \times \cdots \times [\![A_n]\!] &\to [\![B]\!]^{[\![A]\!]} \\
(a_1, \ldots, a_n) &\mapsto (a \mapsto [\![s]\!]_{x_1,\ldots,x_n,x}(a_1, \ldots, a_n, a)) \\
[\![s\ t]\!]_{x_1,\ldots,x_n} : [\![A_1]\!] \times \cdots \times [\![A_n]\!] &\to [\![B]\!] \\
\vec{a} &\mapsto [\![s]\!]_{x_1,\ldots,x_n}(\vec{a})\big([\![t]\!]_{x_1,\ldots,x_n}(\vec{a})\big)
\end{aligned}$$

Sémantique dénotationnelle ensembliste

Évidemment, on veut (et on a) le résultat suivant :

Théorème

Si $s : A$, $s' : A$ *et* $s =_\beta s'$ *alors* $[\![s]\!]_{\vec{x}} = [\![s']\!]_{\vec{x}} \in [\![A]\!]^{[\![A_1]\!] \times \cdots \times [\![A_n]\!]}$.

Démonstration: Il suffit de traiter le cas $s \to_\beta s'$, par induction sur s.
Le cas du redex est donné par :

Lemme

Si $s : B$ *et* $t : A$ *alors*

$$[\![s[t/x^A]]\!]_{x_1,\dots,x_n}(\vec{a}) = [\![s]\!]_{x_1,\dots,x_n,x}(\vec{a}, [\![t]\!]_{x_1,\dots,x_n}(\vec{a}))$$

qu'on montre par induction sur s. □

Sémantique dénotationnelle ensembliste

Évidemment, on veut (et on a) le résultat suivant :

Théorème

Si $s : A$, $s' : A$ *et* $s =_\beta s'$ *alors* $[\![s]\!]_{\vec{x}} = [\![s']\!]_{\vec{x}} \in [\![A]\!]^{[\![A_1]\!] \times \cdots \times [\![A_n]\!]}$.

On appelle sémantique dénationelle toute interprétation invariante par β-réduction : c'est un sujet de recherche très fructueux (et un cours d'option).

Ça existe aussi en non typé, mais l'interprétation ensembliste échoue : ça reviendrait à trouver un ensemble D avec $D = D^D$ (ce qui est impossible dès que D a au moins deux éléments).

λ-termes typables (à la Curry)

On peut aussi voir le typage comme un processus externe au calcul.

- Un contexte de typage est une famille finie Γ de types indexée par des variables, qu'on note par exemple $\Gamma = x_1 : A_1, \ldots, x_n : A_n$.
- Un jugement de typage est la donnée $\Gamma \vdash s : A$ d'un contexte Γ, d'un terme s et d'un type A, qu'on lit « dans le contexte Γ, s est typable de type A ».

Les règles de typage deviennent :

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

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

λ-termes typables (à la Curry)

On peut aussi voir le typage comme un processus externe au calcul.

- Un contexte de typage est une famille finie Γ de types indexée par des variables, qu'on note par exemple $\Gamma = x_1 : A_1, \ldots, x_n : A_n$.
- Un jugement de typage est la donnée $\Gamma \vdash s : A$ d'un contexte Γ, d'un terme s et d'un type A, qu'on lit « dans le contexte Γ, s est typable de type A ».

Les règles de typage deviennent :

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

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

Cette relation n'est plus fonctionnelle : on a par exemple $\vdash I : A \to A$ pour tout type A.

Le problème du typage

On parlera de terme pur pour les termes sans typage *a priori* des variables. On dit que t est typable s'il existe Γ et A avec $\Gamma \vdash t : A$.

Le problème du typage

On parlera de terme pur pour les termes sans typage *a priori* des variables. On dit que t est typable s'il existe Γ et A avec $\Gamma \vdash t : A$.

Savoir si un terme pur est typable (ou bien s'il admet un certain type dans un contexte fixé) peut être vu comme un problème de résolution de contraintes équationnelles sur sur les types :

- les types des variables sont des inconnues,
- les règles d'application imposent des contraintes.

Recherche d'un typage

Un arbre de typage de $\underline{2}$ est nécessairement de la forme :

$$
\dfrac{
  \dfrac{
    \dfrac{}{f : F, x : X \vdash f : F}\,(var)
    \qquad
    \dfrac{
      \dfrac{}{f : F, x : X \vdash f : F}\,(var)
      \quad
      \dfrac{}{f : F, x : X \vdash x : X}\,(var)
    }{f : F, x : X \vdash f\, x :}\,(app)
  }{f : F, x : X \vdash f\,(f\, x) :}\,(app)
}{
  \dfrac{f : F \vdash \lambda x. f\,(f\, x) :}{\vdash \lambda f. \lambda x. f\,(f\, x) :}\,(\lambda)
}\,(\lambda)
$$

Recherche d'un typage

Un arbre de typage de $\underline{2}$ est nécessairement de la forme :

$$
\dfrac{\dfrac{\dfrac{\dfrac{}{f : F, x : X \vdash f : F}\,(\mathit{var}) \qquad \dfrac{\dfrac{}{f : F, x : X \vdash f : F}\,(\mathit{var}) \quad \dfrac{}{f : F, x : X \vdash x : X}\,(\mathit{var})}{f : F, x : X \vdash f\,x : A}\,(\mathit{app})}{f : F, x : X \vdash f\,(f\,x) :}\,(\mathit{app})}{f : F \vdash \lambda x.f\,(f\,x) :}\,(\lambda)}{\vdash \lambda f.\lambda x.f\,(f\,x) :}\,(\lambda)
$$

avec A tel que $F = X \rightarrow A$

Recherche d'un typage

Un arbre de typage de $\underline{2}$ est nécessairement de la forme :

$$\dfrac{\dfrac{\dfrac{}{f : F, x : X \vdash f : F}\,(var) \qquad \dfrac{\dfrac{}{f : F, x : X \vdash f : F}\,(var) \quad \dfrac{}{f : F, x : X \vdash x : X}\,(var)}{f : F, x : X \vdash f\,x : A}\,(app)}{\dfrac{f : F, x : X \vdash f\,(f\,x) : B}{f : F \vdash \lambda x.f\,(f\,x) :}\,(\lambda)}\,(app)}{\vdash \lambda f.\lambda x.f\,(f\,x) :}\,(\lambda)$$

avec A tel que $F = X \to A$ et B tel que $F = A \to B$,

Recherche d'un typage

Un arbre de typage de $\underline{2}$ est nécessairement de la forme :

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{(var)}
\UnaryInfC{$f : F, x : X \vdash f : F$}
\AxiomC{}
\RightLabel{(var)}
\UnaryInfC{$f : F, x : X \vdash f : F$}
\AxiomC{}
\RightLabel{(var)}
\UnaryInfC{$f : F, x : X \vdash x : X$}
\RightLabel{(app)}
\BinaryInfC{$f : F, x : X \vdash f\, x : A$}
\RightLabel{(app)}
\BinaryInfC{$f : F, x : X \vdash f\,(f\, x) : B$}
\RightLabel{(λ)}
\UnaryInfC{$f : F \vdash \lambda x.f\,(f\, x) : X \to B$}
\RightLabel{(λ)}
\UnaryInfC{$\vdash \lambda f.\lambda x.f\,(f\, x) : F \to X \to B$}
\end{prooftree}
$$

avec A tel que $F = X \to A$ et B tel que $F = A \to B$,

Recherche d'un typage

Un arbre de typage de $\underline{2}$ est nécessairement de la forme :

$$\dfrac{\dfrac{}{f : F, x : X \vdash f : F}\,(var) \qquad \dfrac{\dfrac{}{f : F, x : X \vdash f : F}\,(var) \qquad \dfrac{}{f : F, x : X \vdash x : X}\,(var)}{f : F, x : X \vdash f\,x : A}\,(app)}{\dfrac{f : F, x : X \vdash f\,(f\,x) : B}{\dfrac{f : F \vdash \lambda x.f\,(f\,x) : X \to B}{\vdash \lambda f.\lambda x.f\,(f\,x) : F \to X \to B}\,(\lambda)}\,(\lambda)}\,(app)$$

avec A tel que $F = X \to A$ et B tel que $F = A \to B$,
donc $X = A = B$ et $F \to X \to B = (B \to B) \to (B \to B)$.

Recherche d'un typage

Un arbre de typage de $\underline{2}$ est nécessairement de la forme :

$$\dfrac{\dfrac{\dfrac{}{f : F, x : X \vdash f : F}\,(\mathit{var}) \qquad \dfrac{\dfrac{}{f : F, x : X \vdash f : F}\,(\mathit{var}) \qquad \dfrac{}{f : F, x : X \vdash x : X}\,(\mathit{var})}{f : F, x : X \vdash f\,x : A}\,(\mathit{app})}{\dfrac{f : F, x : X \vdash f\,(f\,x) : B}{f : F \vdash \lambda x.f\,(f\,x) : X \to B}\,(\lambda)}\,(\mathit{app})}{\vdash \lambda f.\lambda x.f\,(f\,x) : (B \to B) \to B \to B}\,(\lambda)$$

avec A tel que $F = X \to A$ et B tel que $F = A \to B$,
donc $X = A = B$ et $F \to X \to B = (B \to B) \to (B \to B)$.

Recherche d'un typage

Un arbre de typage de $\underline{2}$ est nécessairement de la forme :

$$\dfrac{\dfrac{}{f : F, x : X \vdash f : F}\,(var) \qquad \dfrac{\dfrac{}{f : F, x : X \vdash f : F}\,(var) \qquad \dfrac{}{f : F, x : X \vdash x : X}\,(var)}{f : F, x : X \vdash f\,x : A}\,(app)}{\dfrac{\dfrac{f : F, x : X \vdash f\,(f\,x) : B}{f : F \vdash \lambda x.f\,(f\,x) : X \to B}\,(\lambda)}{\vdash \lambda f.\lambda x.f\,(f\,x) : (B \to B) \to B \to B}\,(\lambda)}\,(app)$$

avec A tel que $F = X \to A$ et B tel que $F = A \to B$,
donc $X = A = B$ et $F \to X \to B = (B \to B) \to (B \to B)$.

Chercher si s est typable, c'est chercher un typage des variables tel que s soit typé.

Entiers de Church

Lemme

On a $\vdash \underline{n} : A$ *pour tout* $n \in \mathbf{N}$*, ssi* A *est de la forme* $\mathbf{N}_B := (B \to B) \to B \to B$*.*

Démonstration:

- Pour le sens direct, il suffit de regarder le cas $n = 2$ (déjà vu).
- Pour la réciproque, on prouve par récurrence sur n que $f : B \to B, x : B \vdash f^n\, x : B$.

$\square$

Entiers de Church

Lemme

On a $\vdash \underline{n} : A$ *pour tout* $n \in \mathbf{N}$*, ssi* A *est de la forme* $\mathbf{N}_B := (B \to B) \to B \to B$.

Démonstration:

- Pour le sens direct, il suffit de regarder le cas $n = 2$ (déjà vu).
- Pour la réciproque, on prouve par récurrence sur n que $f : B \to B, x : B \vdash f^n\, x : B$.

$\square$

$\rightsquigarrow$ Vérifiez que $\underline{\text{add}} : \mathbf{N}_B \to \mathbf{N}_B \to \mathbf{N}_B$.

Entiers de Church

Lemme

On a $\vdash \underline{n} : A$ *pour tout* $n \in \mathbf{N}$, *ssi* A *est de la forme* $\mathbf{N}_B := (B \to B) \to B \to B$.

Démonstration:

- Pour le sens direct, il suffit de regarder le cas $n = 2$ (déjà vu).
- Pour la réciproque, on prouve par récurrence sur n que $f : B \to B, x : B \vdash f^n\, x : B$.

$\square$

$\rightsquigarrow$ Vérifiez que $\underline{\mathrm{add}} : \mathbf{N}_B \to \mathbf{N}_B \to \mathbf{N}_B$.

$\rightsquigarrow$ En posant $\underline{\mathrm{mult}} := \lambda a.\lambda b.\lambda f.\lambda x.a\,(b\,f)\,x$, vérifiez que $\underline{\mathrm{mult}}$ représente la multiplication et que $\underline{\mathrm{mult}} : \mathbf{N}_B \to \mathbf{N}_B \to \mathbf{N}_B$.

Entiers de Church

Lemme

On a $\vdash \underline{n} : A$ pour tout $n \in \mathbf{N}$, ssi A est de la forme $\mathbf{N}_B := (B \to B) \to B \to B$.

Démonstration:

- Pour le sens direct, il suffit de regarder le cas $n = 2$ (déjà vu).
- Pour la réciproque, on prouve par récurrence sur n que $f : B \to B, x : B \vdash f^n\, x : B$.

$\square$

$\rightsquigarrow$ Vérifiez que $\underline{\mathrm{add}} : \mathbf{N}_B \to \mathbf{N}_B \to \mathbf{N}_B$.

$\rightsquigarrow$ En posant $\underline{\mathrm{mult}} := \lambda a.\lambda b.\lambda f.\lambda x.a\,(b\,f)\,x$, vérifiez que $\underline{\mathrm{mult}}$ représente la multiplication et que $\underline{\mathrm{mult}} : \mathbf{N}_B \to \mathbf{N}_B \to \mathbf{N}_B$.

$\rightsquigarrow$ Peut-on faire la même chose avec la version de $\underline{\mathrm{mult}}$ vue dans la fiche d'activité ?

Typage et variables libres

Si $\Gamma = x_1 : A_1, \ldots, x_n : A_n$, on note $\mathrm{supp}(\Gamma) := \{x_1, \ldots, x_n\}$ le support de Γ.

Lemme

Si $\Gamma \vdash s : A$ *alors* $\mathrm{VL}(s) \subset \mathrm{supp}(\Gamma)$.

Démonstration: Par induction sur la dérivation. $\square$

Lemme

Si $\Gamma, x : A \vdash s : B$ *et* $x \notin \mathrm{VL}(s)$ *alors* $\Gamma \vdash s : B$.

Démonstration: Par induction sur la dérivation. $\square$

Lemme

Si $\Gamma \vdash s : B$ *et* $x \notin \mathrm{supp}(\Gamma)$ *alors* $\Gamma, x : A \vdash s : B$.

Démonstration: Par induction sur la dérivation. $\square$

Réduction du sujet

La typabilité reste stable par β-réduction :

Théorème

Si $\Gamma \vdash s : A$ *et* $s \to_\beta s'$ *alors* $\Gamma \vdash s' : A$*.*

Démonstration: Par induction sur la dérivation. Le cas du redex est donné par :

Lemme

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

qu'on prouve également par induction sur la dérivation. $\square$

Church *vs* Curry (typé *vs* typable)

- On a vu que le type d'un terme pur typable n'est pas nécessairement unique.
- De même, il peut y avoir plusieurs dérivations d'un jugement de typage.

  mais la structure de l'arbre est toujours la même : c'est celle du terme
- Une dérivation de typage à la Curry détermine un terme bien typé à la Church.
- Un terme bien typé $t : A$ à la Church et un ensemble $\{x_1^{A_1}, \dots, x_n^{A_n}\} \ni \mathrm{VL}(t)$ déterminent réciproquement une dérivation de $x_1 : A_1, \dots, x_n : A_n \vdash t : A$.

Church *vs* Curry (typé *vs* typable)

C'est donc à peu près la même chose mais :

- la validité du typage à la Church est clairement décidable (et linéaire en la taille du terme et des types) ;
- la typabilité (savoir si un terme admet un type) et la vérification de typage (savoir si un jugement est dérivable) à la Curry sont décidables aussi, mais c'est moins évident ;

  et ça peut changer drastiquement si on étend le système de types
- il est souvent « évident » mais pas facile à prouver que ce qu'on définit sur les dérivations de typage ou les termes typés ne dépend en fait que du terme pur sous-jacent.

  par exemple la sémantique

Correspondance de Curry–Howard (bis)

Une démonstration en déduction naturelle minimale implicative (c.-à-d. avec les seules règles (ax), ($\Rightarrow_i$) et ($\Rightarrow_e$)), n'est rien d'autre qu'une dérivation de typage à la Curry.

Mieux : la transformation associée à la β-réduction par le théorème de réduction du sujet est exactement l'élimination des coupures.

Correspondance de Curry–Howard (bis)

Une démonstration en déduction naturelle minimale implicative (c.-à-d. avec les seules règles (ax), $(\Rightarrow_i)$ et $(\Rightarrow_e)$), n'est rien d'autre qu'une dérivation de typage à la Curry.

Mieux : la transformation associée à la β-réduction par le théorème de réduction du sujet est exactement l'élimination des coupures.
Bénéfices :

- on bénéficie de tous les résultats de la riche littérature sur le λ-calcul et la β-réduction pour étudier la structure des preuves et l'élimination des coupures ;
- la logique apporte des garanties sur le comportement des « programmes ».

Normalisation

Théorème

Si $\Gamma \vdash s : A$ *alors* s *admet une forme normale.*

Démonstration: Comme la preuve d'élimination des coupures. $\square$

Exemple d'application : la formule de Peirce

Lemme

En général, la formule $P := ((A \to B) \to A) \to A$ n'est pas prouvable en logique minimale.

Démonstration: Il suffit de montrer qu'il n'y a pas de terme de ce type, clos (sans variable libre) et en forme normale, lorsque A et B sont atomiques et distinctes.
Soit t irréductible, avec $\vdash t : P$.

- On peut écrire $t = \lambda x_1. \cdots \lambda x_n.x\, t_1 \cdots t_k$.
- Comme t est clos et A est atomique, on a nécessairement $n = 1$ et $x = x_1$, donc $x : (A \to B) \to A \vdash x\, t_1 \cdots t_k : A$.
- Nécessairement, $k = 1$ et $x : (A \to B) \to A \vdash t_1 : A \to B$.
- On peut écrire $t_1 = \lambda y_1. \cdots \lambda y_m.y\, u_1 \cdots u_l$.
- Le seul cas possible est $t_1 = x\, u_1$ avec $x : (A \to B) \to A \vdash u_1 : A \to B$.

Exemple d'application : la formule de Peirce

Lemme

En général, la formule $P := ((A \to B) \to A) \to A$ n'est pas prouvable en logique minimale.

Démonstration: Il suffit de montrer qu'il n'y a pas de terme de ce type, clos (sans variable libre) et en forme normale, lorsque A et B sont atomiques et distinctes.
Soit t irréductible, avec $\vdash t : P$.

- On peut écrire $t = \lambda x_1. \cdots \lambda x_n.x\, t_1 \cdots t_k$.
- Comme t est clos et A est atomique, on a nécessairement $n = 1$ et $x = x_1$, donc $x : (A \to B) \to A \vdash x\, t_1 \cdots t_k : A$.
- Nécessairement, $k = 1$ et $x : (A \to B) \to A \vdash t_1 : A \to B$.
- On peut écrire $t_1 = \lambda y_1. \cdots \lambda y_m.y\, u_1 \cdots u_l$.
- Le seul cas possible est $t_1 = x\, u_1$ avec $x : (A \to B) \to A \vdash u_1 : A \to B$.
- Il suffit de démontrer par récurrence sur la taille de v qu'on n'a jamais $x : (A \to B) \to A \vdash v : A \to B$.

$\square$

Exemple d'application : le type des entiers

Lemme

Si $\vdash s : \boldsymbol{N}_X$ *où* X *est un type atomique, alors* $s =_\beta I$ *ou il existe* $n \in \boldsymbol{N}$ *tel que* $s =_\beta \underline{n}$*.*

Démonstration: Il suffit de regarder le cas où s est en forme normale. □

L'identité est une version de $\underline{1}$: pour tous t et u, $I\; t\; u =_\beta t\; u =_\beta \underline{1}\; t\; u$.

Normalisation forte

On dit qu'un λ-terme t est fortement normalisable s'il n'a pas de suite infinie de réductions $t = t_0 \to_\beta t_1 \to_\beta t_2 \to_\beta \cdots$.

- Un terme fortement normalisable est nécessairement normalisable.
- La réciproque est fausse (voir $\underline{0}\,\Omega$)

Théorème

Si $\Gamma \vdash s : A$ alors s est fortement normalisable.

Démonstration:

Normalisation forte

On dit qu'un λ-terme t est fortement normalisable s'il n'a pas de suite infinie de réductions $t = t_0 \to_\beta t_1 \to_\beta t_2 \to_\beta \cdots$.

- Un terme fortement normalisable est nécessairement normalisable.
- La réciproque est fausse (voir $\underline{0}\,\Omega$)

Théorème

Si $\Gamma \vdash s : A$ alors s est fortement normalisable.

Démonstration:

$\rightsquigarrow$ Exercice.

$\square$

La réciproque est fausse, car il y a des formes normales non typables : voir Δ.