Lambda-calcul simplement typé

version du 28 novembre 2001 – 10 h 47

Ω et Y contiennent des termes qui s'appliquent à eux-mêmes.

Le paradoxe du barbier est :

> Le barbier rase tous ceux qui ne se rasent pas eux-mêmes. *Qui rase le barbier* ?

Pour éviter les paradoxes, on cherche à éviter de tels termes.

On va donc typer les termes.

C'est aussi bien pour la programmation.

Le typage a donc deux objectifs :

– préserver la correction, rien de mauvais ne peut arriver,

– préserver la terminaison, toutes les réductions se terminent.

En λ-calcul la terminaison s'appelle la normalisation forte.

Les environnements

Il faut typer les variables libres, il faut donc faire des hypothèses sur les types de ces variables.

D'où la notion d'environnement.

Un environnement est un ensemble d'association de types à des variables.

$$\Gamma \equiv x_1 : \sigma_1, ..., x_n : \sigma_n$$

Les types

Un jugement est l'affirmation du type σ d'un terme M sous un certain environnement Γ :

$$\Gamma \vdash M : \sigma$$

Les types sont

– soit des variables α,

– soit des types applications $\sigma \to \tau$.

Les règles

(Var) $$\Gamma, x : \sigma \vdash x : \sigma$$

(Abs) $$\frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \to \tau}$$

(App) $$\frac{\Gamma \vdash M : \sigma \to \tau \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash M\,N : \tau}$$

Exercices

Typez les termes

$B \equiv \lambda xyz.x(yz)$,

$I \equiv \lambda x.x$,

$C \equiv \lambda xyz.xzy$,

$K \equiv \lambda xy.x$,

$S \equiv \lambda xyz.xz(yz)$.

Exercices (suite)

1. Typez $II \equiv (\lambda x.x)(\lambda x.x)$.
2. Typez **22** $\equiv (\lambda f x.f(fx))(\lambda f x.f(fx))$.
3. A-t-on le même type pour I (resp. **2**) dans chaque cas ?

Conclusion : Le système de types simples n'est pas assez général. Il ne permet d'affecter un type unique à chaque terme.

Si $\Gamma, x : \sigma \vdash M : \tau$ et $\Gamma \vdash N : \sigma$ alors $\Gamma \vdash M[x := N] : \tau$

Démonstration : Par induction sur M.

Si $\Gamma, x : \sigma \vdash M : \tau$ et $\Gamma \vdash N : \sigma$ alors $\Gamma \vdash M[x := N] : \tau$

Démonstration : Par induction sur M.

- Si $M \equiv y$ alors Γ contient $y : \tau$ et $M[x := N] \equiv M \equiv y$ donc le résultat qui est $\Gamma \vdash y : \tau$ est clair.

- Si $M \equiv x$ alors $\tau = \sigma$ et $M[x := N] \equiv N$ et le résultat est l'une des hypothèses.

Si $\Gamma, x : \sigma \vdash M : \tau$ et $\Gamma \vdash N : \sigma$ alors $\Gamma \vdash M[x := N] : \tau$

Démonstration : Par induction sur M.

- Si $M \equiv P\,Q$ alors par (App) pour un certain τ', on a

– $\Gamma, x : \sigma \vdash P : \tau' \to \tau$ duquel on tire par induction
$\Gamma \vdash P[x := N] : \tau' \to \tau$

– et $\Gamma, x : \sigma \vdash Q : \tau'$ duquel on tire par induction
$\Gamma \vdash Q[x := N] : \tau'$.

Donc par (App) $\Gamma \vdash (P\,Q)[x := N] : \tau$ puisque
$(P\,Q)[x := N] \equiv P[x := N]\,Q[x := N]$.

- Le cas $M \equiv \lambda x.P$ est similaire.

Réduction du sujet

Lemme Réduction du sujet

La β-réduction préserve le type.

Si $\Gamma \vdash M : \sigma$ et si $M \xrightarrow[\beta]{} N$ alors $\Gamma \vdash N : \sigma$.

Si $\Gamma \vdash M : \sigma$ et si $M \xrightarrow[\beta]{} N$ alors $\Gamma \vdash N : \sigma$.

Démonstration : Par induction sur la définition de $M \xrightarrow[\beta]{} N$.

– $M \equiv (\lambda x.M_1)M_2$

$\Gamma \vdash (\lambda x.M_1)M_2 : \sigma$ vient de $\Gamma, x : \tau \vdash M_1 : \sigma$ et de $\Gamma \vdash M_2 : \tau$.

D'autre part $N \equiv M_1[x := M_2]$

or d'après le lemme $\Gamma \vdash M_1[x := M_2] : \sigma$.

Si $\Gamma \vdash M : \sigma$ et si $M \xrightarrow[\beta]{} N$ alors $\Gamma \vdash N : \sigma$.

Démonstration : Par induction sur la définition de $M \xrightarrow[\beta]{} N$.

– $M \equiv (\lambda x.M_1)M_2$

$\Gamma \vdash (\lambda x.M_1)M_2 : \sigma$ vient de $\Gamma, x : \tau \vdash M_1 : \sigma$ et de $\Gamma \vdash M_2 : \tau$.

D'autre part $N \equiv M_1[x := M_2]$

or d'après le lemme $\Gamma \vdash M_1[x := M_2] : \sigma$.

– $M \equiv M_1 M_2$ avec $N \equiv N_1 M_2$ et $M_1 \xrightarrow[\beta]{} N_1$.

$\Gamma \vdash M$ vient de $\Gamma \vdash M_1 : \tau \to \sigma$ et $\Gamma \vdash M_2 : \tau$,

par induction $\Gamma \vdash N_1 : \tau \to \sigma$ et donc $\Gamma \vdash N_1 M_2 : \sigma$.

Si $\Gamma \vdash M : \sigma$ et si $M \xrightarrow[\beta]{} N$ alors $\Gamma \vdash N : \sigma$.

Démonstration : Par induction sur la définition de $M \xrightarrow[\beta]{} N$.

– $M \equiv (\lambda x.M_1)M_2$

$\Gamma \vdash (\lambda x.M_1)M_2 : \sigma$ vient de $\Gamma, x : \tau \vdash M_1 : \sigma$ et de $\Gamma \vdash M_2 : \tau$.

D'autre part $N \equiv M_1[x := M_2]$

or d'après le lemme $\Gamma \vdash M_1[x := M_2] : \sigma$.

– $M \equiv M_1 M_2$ avec $N \equiv N_1 M_2$ et $M_1 \xrightarrow[\beta]{} N_1$.

$\Gamma \vdash M$ vient de $\Gamma \vdash M_1 : \tau \to \sigma$ et $\Gamma \vdash M_2 : \tau$,

par induction $\Gamma \vdash N_1 : \tau \to \sigma$ et donc $\Gamma \vdash N_1 M_2 : \sigma$.

– Les autres cas sont similaires.

Commentaires

– Si un terme est d'un certain type, il n'en sortira pas par β-réduction.

– Si un terme a une forme normale et s'il a un type, alors sa forme normale a ce type.

L'isomorphisme de Curry-Howard

$$\text{(Var)}\quad \Gamma, x : \sigma \vdash x : \sigma$$

$$\text{(Abs)}\quad \frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \to \tau}$$

$$\text{(App)}\quad \frac{\Gamma \vdash M : \sigma \to \tau \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash M\,N : \tau}$$

$$\Gamma, p \vdash p$$

$$\Rightarrow I\quad \frac{\Gamma, p \vdash q}{\Gamma \vdash p \Rightarrow q}$$

$$\Rightarrow E\quad \frac{\Gamma \vdash p \Rightarrow q \qquad \Gamma \vdash p}{\Gamma \vdash q}$$

L'isomorphisme de Curry-Howard

Dans $\Gamma \vdash M : \sigma$,

– M : est une annotation qui est le «terme de preuve»,

– σ peut-être vu comme un type ou comme une proposition.

L'isomorphisme de Curry-Howard

On obtient le tableau de correspondance :

types	propositions
termes	preuves
réduction	simplification des preuves

Normalisation forte

Définition : Un terme M est fortement normalisable si toute suite de réductions $M \xrightarrow[\beta]{} M_1 \xrightarrow[\beta]{} \ldots \xrightarrow[\beta]{} M_n \ldots$ est finie.

Théorème : Si $\Gamma \vdash M : \tau$ alors M est fortement normalisable.

Normalisation forte (démonstration)

Lemme : Si A, B et $\vec{C}$ sont fortement normalisables

et $AB\vec{C}$ n'est pas fortement normalisable

alors il y a un D tel que

1. $A \xrightarrow[\beta]{} \lambda x.D$
2. $D[x := B]\vec{C}$ n'est pas fortement normalisable.

Normalisation forte (démonstration)

Démonstration :

Puisque A, B et $\vec{C}$ sont fortement normalisables, la réduction infinie qui part de $AB\vec{C}$ est de la forme

$$AB\vec{C} \underset{\beta}{\longrightarrow\!\!\!\!\!\rightarrow} (\lambda x.D)B_1\vec{C_1} \underset{\beta}{\longrightarrow} D[x := B_1]\vec{C_1} \underset{\beta}{\longrightarrow\!\!\!\!\!\rightarrow} \ldots$$

Mézalors les résultat vient immédiatement du fait que

$$D[x := B]\vec{C} \underset{\beta}{\longrightarrow\!\!\!\!\!\rightarrow} D[x := B_1]\vec{C_1}.$$

Normalisation forte (démonstration)

Lemme : Si P est typé

et si M et P sont fortement normalisables,

alors $M[x := P]$ est fortement normalisable.

Normalisation forte (démonstration)

Démonstration :

Par induction sur le triplet $(type(P), h(M), taille(M))$, où

– où $type(P)$ est la complexité du type de P,

– $h(M)$ (ou hauteur) est la longueur de la plus longue réduction qui commence en M,

– $taille(M)$ est la taille de M.

Les cas où M est une abstraction $\lambda y.N$ ou de la forme $y\vec{R}$ sont faciles et laissés au lecteur.

– $M = (\lambda y.L)Q\vec{R}$

Par induction, $L[x := P]$, $Q[x := P]$ et $\vec{R}[x := P]$ sont fortement normalisables.

Par le lemme précédent, il suffit de montrer que

$L[x := P][y := Q[x := P]]\ \vec{R}[x := P] \equiv D[x := P]$

où $D \equiv L[y := Q]\ \vec{R}$ est fortement normalisable .

Or $h(D) < h(M)$ donc par induction $M[x := P]$ est fortement normalisable.

– $M = xL\vec{Q}$

Par induction, $L' \equiv L[x := P]$ et $\vec{Q'} \equiv \vec{Q}[x := P]$ sont fortement normalisables.

Par le lemme précédent, il suffit de montrer que si $P \xrightarrow[\beta]{} \lambda y.P_1$

alors $M' \equiv P_1[y := L']\vec{Q'}$ est fortement normalisable.

Puisque $type(L') < type(P)$, on peut appliquer l'hypothèse de récurrence et donc $P_1[y := L']$ est fortement normalisable.

Et ainsi par induction

– parce que $M' \equiv (z\vec{Q'})[z := P_1[y := L']]$ est fortement normalisable
– et que $type(P_1) < type(P)$,

on conclut que $M[x := P]$ est fortement normalisable.

Normalisation forte (démonstration)

Le théorème est prouvé par induction structurelle sur le terme dont on cherche à prouver la normalisation forte.

– Le terme est une variable ou une abstraction c'est clair.

– Le terme est une application $P\,Q \equiv (zQ)[z := P]$

alors c'est évident par le lemme précédent et par induction structurelle.