Majeure M1 **Cours Langages de Programmation**

CORRIGE DE LA COMPOSITION

Jean-Jacques Lévy
5 Décembre 2000, 3h

Avertissement On attachera une grande importance à la clarté, à la précision, à la concision de la rédaction. Seules les notes de cours sont autorisées.

Question 1 Variables libres et liées.

a) Seul x est libre dans $M = (\lambda f.\lambda g.\lambda x.gx + 3)(\lambda x.x + 4)(\lambda y.\underline{x} + y)$

b) $M[f\backslash g] = M[g\backslash f] = M[y\backslash x + y] = M$.
$M[x\backslash x + y] = (\lambda f.\lambda g.\lambda x.fx + 3)(\lambda x.x + 4)(\lambda z.x + y + z)$.

c) $(\lambda f.\lambda g.\lambda x.\lambda y.M)I\ J\ 3\ 2\ 1 \rightarrow (\lambda g.\lambda x.\lambda y.M)J\ 3\ 2\ 1 \rightarrow (\lambda x.\lambda y.M)\ 3\ 2\ 1$
$\rightarrow (\lambda y.(\lambda f.\lambda g.\lambda x.gx + 3)(\lambda x.x + 4)(\lambda y.3 + y))\ 2\ 1$
$\rightarrow (\lambda f.\lambda g.\lambda x.gx + 3)(\lambda x.x + 4)(\lambda y.3 + y)\ 1$
$\rightarrow (\lambda g.\lambda x.gx + 3)(\lambda y.3 + y)\ 1$
$\rightarrow (\lambda x.(\lambda y.3 + y)x + 3)\ 1 \rightarrow (\lambda y.3 + y)1 + 3)$
$\rightarrow 3 + 1 + 3 \rightarrow 4 + 3 \rightarrow 7$

Question 2
a) $\mathit{foldR2} = \mu\phi.\lambda f.\lambda x.\lambda y.$
$\quad$ $\texttt{ifnil}\ x\ \texttt{then}\ c\ \texttt{else}\ \texttt{ifnil}\ y\ \texttt{then}\ c\ \texttt{else}$
$\quad$ $f\ (\texttt{hd}\, x)\ (\texttt{hd}\, y)\ (\phi\ f\ (\texttt{tl}\, x)\ (\texttt{tl}\, y)\ c)$

b) $\mathit{foldR2} : (\alpha \rightarrow \beta \rightarrow \gamma \rightarrow \gamma) \rightarrow \texttt{list}(\alpha) \rightarrow \texttt{list}(\beta) \rightarrow \gamma \rightarrow \gamma$

Question 3 Récursion

a) $\texttt{letrec}\ x = M\ \texttt{in}\ N = \texttt{let}\ x = \mu x.M\ \texttt{in}\ N$
$\mu x.M = \texttt{letrec}\ x = M\ \texttt{in}\ x$.

b) $\texttt{letrec}\ x = M\ \texttt{in}\ N \rightarrow N[x\backslash\ \texttt{letrec}\ x = M\ \texttt{in}\ x]$.

c) $\mu x.M = Y(\lambda x.M)$

d) $Y = \lambda f.\mu x.fx$.

e) $(\lambda f.\mu x.fx)(\lambda y.2) = Y(\lambda y.2) \rightarrow (\lambda y.2)(Y(\lambda y.2)) \rightarrow 2$.

Question 4 PCF sans références, ni objets, mais avec les listes et les tableaux.

a) $[|\ \texttt{nil}\ |] : \texttt{array}(\texttt{list}(\alpha))$
$[|\ \lambda x.x\ |] : \texttt{array}(\alpha \rightarrow \alpha)$
$\lambda x.x[0] : \forall\alpha.\ \texttt{array}(\alpha) \rightarrow \alpha$
$\lambda x.\lambda y.x[0] \leftarrow y : \forall\alpha.\ \texttt{array}(\alpha) \rightarrow \alpha \rightarrow \texttt{void}$.

b) Non, le théorème de progression n'est pas exact comme pour le typage des références. En effet, on a $\texttt{let}\ r = [|\ \texttt{nil}\ |]\ \texttt{in}\ r[0] \leftarrow \underline{3} :: \texttt{nil}; (\texttt{hd}\, r[0])\underline{2}$ en ne faisant pas attention à la généralisation des types de tableaux.

c) On peut adopter la même règle que pour les références. On ne généralise pas les variables de type apparaissant dans le type d'un tableau dans l'évaluation de M dans $\mathtt{let}\ x = M\ \mathtt{in}\ N$.

Question 5 On rajoute les exceptions à PCF.

a) Définissons la classe de contextes actifs suivantes

$$\begin{array}{rcl} C[\,] & ::= & C[\,]N \mid MC[\,] \mid C[\,]+N \mid M+C[\,] \\ & \mid & \mathtt{ifz}\,C[\,]\ \mathtt{then}\ M\ \mathtt{else}\ N \\ & \mid & \mathtt{ifz}\,P\ \mathtt{then}\ C[\,]\ \mathtt{else}\ N \\ & \mid & \mathtt{ifz}\,P\ \mathtt{then}\ M\ \mathtt{else}\ C[\,] \end{array}$$

Alors les réductions sont $C[\mathtt{fail}] \to \mathtt{fail}$ et $\mathtt{try\ fail\ with}\ M \to M$ et $\mathtt{try}\ V\ \mathtt{with}\ M \to V$ quand $V \neq \mathtt{fail}$.

b) Rajouter les exceptions à l'espace des valeurs. Donner la sémantique *bigstep* pour $\vdash \mathtt{try}\ M\ \mathtt{with}\ N = V$.

c) Donner sa règle de typage, ainsi que celle pour le terme $\mathtt{fail}$.

d) Quelles modifications doit-on faire à l'algorithme W pour tenir compte de cette nouvelle instruction.

e) Supposons que $\mathtt{fail}$ prenne un argument. Ainsi

$$\begin{array}{rcll} M, N, P & ::= & \ldots & \\ & \mid & \mathtt{fail}\,M & \text{exception levée} \\ & \mid & \mathtt{try}\ M\ \mathtt{with}\ N & \text{gestionnaire de l'exception} \end{array}$$

Alors l'instruction $\mathtt{try}\ M\ \mathtt{with}\ \lambda x.N$ renvoie $N[x\backslash V]$ si $\mathtt{fail}\,V$ est exécuté dans M. Recommencer les questions précédentes avec cette nouvelle construction.

f) Donner une piste pour construire un système de type plus robuste?

Question 6 Codage des entiers unaires avec les objets dans PCF (sans les constantes $\underline{n}$ entières).

a) On représente les entiers par un objet à trois champs:

$$\begin{array}{rcl} \underline{\mathtt{0}} & = & \{\mathit{est\_zero} = \mathit{true};\ \mathit{pred} = \zeta x.\ldots;\ \mathit{succ} = \zeta x.\ldots\} \\ \underline{\mathtt{n+1}} & = & \{\mathit{est\_zero} = \mathit{false};\ \mathit{pred} = \zeta x.\ldots;\ \mathit{succ} = \zeta x.\ldots\} \end{array}$$

Donner une expression possible pour les deux méthodes *pred* et *succ* donnant le prédécésseur et le successeur de tout entier naturel. (On posera $\mathit{true} = \underline{0}$ et $\mathit{true} = \underline{1}$ pour obtenir des termes de corrects de PCF en l'absence de booléens).

b) Donner un codage (toujours dans les objets non-modifiables) qui minimise le nombre d'objets créés.