Majeure M1 **Cours Langages de Programmation**

COMPOSITION D'INFORMATIQUE

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) Soit $M = (\lambda f.\lambda g.\lambda x.gx + 3)(\lambda x.x + 4)(\lambda y.x + y)$. Est-ce que f, g, x, y sont libres dans M?

b) Calculer $M[f\backslash g]$, $M[g\backslash f]$, $M[x\backslash x + y]$, $M[y\backslash x + y]$.

c) Calculer la valeur de $(\lambda f.\lambda g.\lambda x.\lambda y.M) I\, J\, 3\ 2\ 1$ avec $I = \lambda x.x$, $J = \lambda x.2$ en expliquant tous les pas élémentaires de calcul.

Question 2 On considère PCF avec les listes et la fonctionnelle *foldR2* telle que

$$\mathit{foldR2}\ f(a_1 :: a_2 \cdots :: a_n :: \mathtt{nil})(b_1 :: b_2 \cdots :: b_n :: \mathtt{nil})\ c = f a_1 b_1 (f a_2 b_2 (\cdots (f a_n b_n c))) \qquad (n \geq 0)$$

a) Exprimer *foldR2* en PCF.

b) Déterminer son type en appliquant rigoureusement les règles de typage de PCF.

Question 3 Un appel de fonction est déplié (*inlining*) quand son invocation est remplacée par la substitution des arguments dans son corps de fonction: $(\lambda x.M)N$ remplacé par $M[x\backslash N]$.

a) Précisez quand cette opération est valide.

b) Quels sont les avantages et les inconvénients du dépliage.

Question 4 Récursion

a) Exprimer $\mathtt{letrec}\ x = M\ \mathtt{in}\ N$ en PCF. Exprimer $\mu x.M$ en fonction de $\mathtt{letrec}$.

b) Donner une règle de réduction pour $\mathtt{letrec}$.

c) On rajoute une constante Y à PCF telle que $YM \rightarrow M(YM)$. Exprimer $\mu x.M$ en fonction de Y et de M.

d) Montrer que Y est déjà exprimable par une expression bien typée de PCF.

e) Quelle est la valeur de $(\lambda f.\mu x.fx)(\lambda y.2)$?

Question 5 On considère PCF sans références, ni objets, mais avec les listes et les tableaux.

a) Quels sont les shémas de types des tableaux $[|\ \mathtt{nil}\ |]$ et $[|\ \lambda x.x\ |]$, et des fonctions $\lambda x.x[0]$ et $\lambda x.\lambda y.x[0] \leftarrow y$. Peut-on généraliser les variables de type de ces termes?

b) Les débordements de tableaux sont-elles les seules erreurs possibles à l'exécution? Montrer que les tableaux ne vérifient pas le théorème de progression?

c) Donner une règle de typage pour les tableaux qui permette d'exprimer les algorithmes de tri sur les tableaux comme des fonctions de type polymorphe, et qui vérifie le théorème de progression.

Question 6 On rajoute les exceptions à PCF. On étend l'ensemble des termes par:

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

Le terme $\texttt{try}\ M\ \texttt{with}\ N$ calcule M sauf si `fail` est exécuté dans M, auquel cas il renvoie N.

a) Un *contexte* est un terme $C[\ ]$ avec un trou (sous-terme manquant). Et $C[M]$ désigne le terme obtenu en mettant M à la place du trou. Un contexte est *actif* si son trou est en position évaluable. Ainsi $C[\ ] = [\ ]\underline{2}$ ou $C'[\ ] = (\lambda x.x)(2 + [\ ])$ sont des contextes actifs; mais $C[\,] = (\lambda x.[\ ])M$ n'en est pas un (puisqu'on ne calcule pas à l'intérieur d'une valeur).

Donner les règles de réduction pour calculer $\texttt{try}\ M\ \texttt{with}\ N$ avec l'aide d'un ensemble de contextes actifs soigneusement définis.

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

c) Donner sa règle de typage, ainsi que celle pour le terme `fail`.

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

e) On donne maintenant un argument à `fail`, qu'il est plus judicieux d'appeler maintenant `failwith`. Ainsi l'espace des termes est

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

Alors l'instruction $\texttt{try}\ M\ \texttt{with}\ \lambda x.N$ renvoie $N[x\backslash V]$ si $\texttt{failwith}\,V$ 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 7 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:

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

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.