Réductions closes

Documents autorisés (en particulier le poly).

Le but de ce problème est d'étudier une forme faible de réduction où l'α-conversion n'est pas nécessaire pour éviter les captures de variables. Ceci est dû à Maribel Fernández et Ian Mackie.

Le calcul que nous allons étudier est appelé le *λ_c-calcul*. Il s'agit d'un calcul à substitution explicites, dans lequel les variables ont des noms x, y, z, ..., comme dans le λ-calcul, et contrairement au $\lambda\sigma$-calcul.

Pour distinguer la substitution usuelle $s[x := t]$ des substitutions explicites, nous noterons ces dernières $M\{x := t\}$. L'opérateur $\_\{\_ := \_\}$ sera un constructeur du λ_c-calcul.

Une autre caractéristique du λ_c-calcul est qu'il est très fortement contraint. Par exemple, la λ-abstraction $\lambda x \cdot M$ ne sera un λ_c-terme valide que si x est libre dans M, et l'application MN ne sera un λ_c-terme valide que si M et N n'ont aucune variable libre en commun. En fait, toute variable apparaîtra exactement une fois dans tout λ_c-terme.

Si on s'en tenait là, on ne pourrait pas écrire grand-chose en λ_c-calcul. On a donc deux constructions supplémentaires:

- lorsque x n'est pas libre dans M, l'*effaceur* $E_x(M)$ est un terme où x est libre et dont la sémantique intuitive est: "effacer x, puis évaluer M"; si x n'est pas libre dans M, on ne peut pas écrire $\lambda x \cdot M$, mais on pourra écrire $\lambda x \cdot E_x(M)$.

- lorsque y et z sont deux variables distinctes, toutes deux libres dans M, le *copieur* $C_x^{y,z}(M)$ est un terme dans lequel x est libre mais y et z ne le sont plus. Sa sémantique intuitive est: "faire deux copies de x, mettre la première dans y, la seconde dans z, puis évaluer M". On ne peut pas écrire xx en λ_c-calcul, mais on peut à la place écrire $C_x^{y,z}(yz)$.

On définit la syntaxe des λ_c-termes M, N, P, ..., en même temps que l'ensemble des variables libres $\mathrm{fv}(M)$ dans M comme suit. L'ensemble des λ_c-termes est le plus petit tel que:

(a) Toute variable x est un λ_c-terme; on a $\mathrm{fv}(x) = \{x\}$;

(b) l'*abstraction* $\lambda x \cdot M$ est un λ_c-terme pour tout λ_c-terme M tel que $x \in \mathrm{fv}(M)$; on a $\mathrm{fv}(\lambda x \cdot M) = \mathrm{fv}(M) \setminus \{x\}$;

(c) l'*application* MN est un λ_c-terme pour tous λ_c-termes M et N tels que $\mathrm{fv}(M) \cap \mathrm{fv}(N) = \emptyset$; on a $\mathrm{fv}(MN) = \mathrm{fv}(M) \cup \mathrm{fv}(N)$;

(d) l'*effaceur* $E_x(M)$ est un λ_c-terme pour tout λ_c-terme M tel que $x \notin \mathrm{fv}(M)$; on a $\mathrm{fv}(E_x(M)) = \mathrm{fv}(M) \cup \{x\}$;

(e) le *copieur* $C_x^{y,z}(M)$ est un λ_c-terme pour tout λ_c-terme M et pour toutes variables $x \notin \mathrm{fv}(M)$, $y, z \in \mathrm{fv}(M)$, telles que $y \neq z$; on a $\mathrm{fv}(C_x^{y,z}(M)) = (\mathrm{fv}(M) \setminus \{y, z\}) \cup \{x\}$;

(f) la *substitution* $M\{x := N\}$ est un λ_c-terme pour tous λ_c-termes M et N et pour toute variable $x \in \mathrm{fv}(M)$ où $(\mathrm{fv}(M) \setminus \{x\}) \cap \mathrm{fv}(N) = \emptyset$; on a $\mathrm{fv}(M\{x := N\}) = (\mathrm{fv}(M) \setminus \{x\}) \cup \mathrm{fv}(N)$.

On rappelle qu'il n'y a *pas* de règle de α-renommage.

Les règles de réduction sont:

(β)	$(\lambda x \cdot M)N$	$\to$	$M\{x := N\}$	si $\mathrm{fv}(\lambda x \cdot M) = \emptyset$
(Var)	$x\{x := N\}$	$\to$	N	
(App_1)	$(M_1 M_2)\{x := N\}$	$\to$	$(M_1\{x := N\})M_2$	si $x \in \mathrm{fv}(M_1)$
(App_2)	$(M_1 M_2)\{x := N\}$	$\to$	$M_1(M_2\{x := N\})$	si $x \in \mathrm{fv}(M_2)$
(Lam)	$(\lambda x \cdot M)\{y := N\}$	$\to$	$\lambda x \cdot (M\{y := N\})$	si $x \neq y, \mathrm{fv}(N) = \emptyset$
$(Copy_1)$	$C_x^{y,z}(M)\{x := N\}$	$\to$	$M\{y := N\}\{z := N\}$	si $\mathrm{fv}(N) = \emptyset$
$(Copy_2)$	$C_x^{y,z}(M)\{x' := N\}$	$\to$	$C_x^{y,z}(M\{x' := N\})$	si $x' \neq x, x' \neq y, x' \neq z, \mathrm{fv}(N) = \emptyset$
$(Erase_1)$	$E_x(M)\{x := N\}$	$\to$	M	si $\mathrm{fv}(N) = \emptyset$
$(Erase_2)$	$E_x(M)\{y := N\}$	$\to$	$E_x(M\{y := N\})$	si $x \neq y, \mathrm{fv}(N) = \emptyset$
$(Comp)$	$M\{x := N\}\{y := P\}$	$\to$	$M\{x := N\{y := P\}\}$	si $y \in \mathrm{fv}(N)$

On notera que $(\lambda x \cdot M)N$ n'est pas toujours un β-rédex: il ne l'est que si $\lambda x \cdot M$ est *clos*, c'est-à-dire n'a aucune variable libre. Les règles (Lam), $(Copy_1)$, $(Copy_2)$, $(Erase_1)$, $(Erase_2)$ ont des conditions similaires. A cause des conditions de formation des applications, $(M_1 M_2)\{x := N\}$ est toujours soit un App_1-rédex, soit un App_2-rédex, et ne peut pas être les deux en même temps.

I. Réduction, machines.

1. On rappelle que le combinateur de point fixe Y de Church est $Y = \lambda f \cdot (\lambda x \cdot f(xx))(\lambda x \cdot f(xx))$ en λ-calcul. Le combinateur correspondant en λ_c-calcul est

   $$Y_c = \lambda f \cdot C_f^{g,h}((\lambda x \cdot g(C_x^{y,z}(yz)))(\lambda x \cdot h(C_x^{y,z}(yz))))$$

   Y est-il fortement normalisant en λ-calcul? faiblement normalisant en λ-calcul? Mêmes questions pour Y_c en λ_c-calcul.

2. Montrer que si M se réduit en N en λ_c, alors $\mathrm{fv}(M) = \mathrm{fv}(N)$. (On ne traitera que les règles (β) et $(Erase_1)$ pour aller plus vite, et on admettra les autres cas.)

3. Montrer que si M se réduit en N par l'une quelconque des règles de réécriture de λ_c, et M est un λ_c-terme, alors N est aussi un λ_c-terme. (On ne traitera que les règles (App_1), $(Copy_1)$ et $(Comp)$ pour aller plus vite, et on admettra les autres cas.)

4. On construit une machine de Krivine pour évaluer les λ_c-termes. Un *état* de la machine est un couple M, R où M est un λ_c-terme, et R est une liste d'objets qui peuvent être soit

des λ_c-termes soit des *associations* $\{x := N\}$, où x est une variable et N un λ_c-terme. La sémantique d'un tel couple est donné par la fonction γ:

$$\gamma(M, []) = M \quad \gamma(M, N :: R) = \gamma(MN, R) \quad \gamma(M, \{x := N\} :: R) = \gamma(M\{x := N\}, R)$$

Un état M, R est *légal* ssi $\gamma(M, R)$ est un λ_c-terme. Les transitions de la machine sont:

$$\begin{array}{rcll} MN, R & \rightsquigarrow & M, N :: R & \\ M\{x := N\}, R & \rightsquigarrow & M, \{x := N\} :: R & \\ \lambda x \cdot M, N :: R & \rightsquigarrow & M, \{x := N\} :: R & \textit{Errata: ajouter ``si } \mathrm{fv}(\lambda x \cdot M) = \emptyset\textit{''} \\ & & \ldots & \end{array}$$

Compléter. Les deux premières transitions servent à chercher le rédex de tête. La troisième réalise la règle (β). On demande de compléter par les transitions correspondant aux autres règles de réduction. On devra avoir les trois *(Errata: les deux)* propriétés:

(a) si $M, R \rightsquigarrow M', R'$, et M, R est un état légal, alors $\gamma(M, R) \to^* \gamma(M', R')$ dans le λ_c-calcul et M', R' est un état légal;

(b) si M, R est un état légal *et final* (i.e., aucune transition ne s'applique), alors $\gamma(M, R)$ est une forme *normale de tête faible*, c'est-à-dire engendré par la grammaire:

$$\begin{array}{rcl} whnf & ::= & \lambda x \cdot N \quad | \quad xN_1 \ldots N_k \\ & | & C_x^{y,z}(whnf) \quad | \quad E_x(whnf) \end{array}$$

On ne demande pas une preuve de ces propriétés, mais, au moins pour la propriété (b), un argumentaire rapide montrant pourquoi $\gamma(M, R)$ ne peut pas être d'une autre forme.

Errata: les formes normales de tête faibles du λ_c-calcul sont beaucoup plus compliquées. On demande à la place de coder une machine de Krivine raisonnable, comme en cours et en TD, et on ne cherchera pas à montrer la propriété (b) ou une propriété analogue. Quoique quiconque qui aura réussi ce tour de force aura des points en plus.

II. Confluence du λ_c-calcul.

1. Soit σ_c le système de réécriture composé de toutes les règles de λ_c sauf (β).

   Pour tout λ_c-terme M où x est libre, on définit $|M|_x$ par:

$$|x|_x = 1 \quad |\lambda y \cdot M|_x = 1 + |M|_x \ (x \neq y) \quad |MN|_x = \begin{cases} 1 + |M|_x & \text{si } x \in \mathrm{fv}(M) \\ 1 + |N|_x & \text{si } x \in \mathrm{fv}(N) \end{cases}$$

$$|E_x(M)|_x = 1 \quad |E_y(M)|_x = 1 + |M|_x \ (x \neq y) \quad |M\{y := N\}|_x = \begin{cases} 1 + |M|_x & \text{si } x \in \mathrm{fv}(M) \\ & \textit{Errata: ajouter} \\ & \textit{``et si } x \notin \mathrm{fv}(N\textit{''} \\ 1 + |N|_x + |M|_y & \text{si } x \in \mathrm{fv}(N) \end{cases}$$

$$|C_x^{y,z}(M)|_x = 1 + |M|_y + |M|_z \quad |C_x^{y,z}(M)|_{x'} = 1 + |M|_{x'} \quad (x \neq x')$$

   $|M|_x$ est, en gros, la profondeur de l'(unique) occurrence de x dans M. Montrer que pour tout λ_c-terme M, pour toute variable $z \in \mathrm{fv}(M)$, si $M \to N$ en σ_c, alors $|M|_z \geq |N|_z$.

2. En utilisant cette construction, on définit la traduction suivante:

$$[x] = x \quad [\lambda x \cdot M] = [M] \quad [MN] = app([M], [N])$$
$$[E_x(M)] = [M] \quad [C_x^{y,z}(M)] = [M] \quad [M\{x := N\}] = f_{|M|_x}([M], [N])$$

vers les termes du premier ordre sur la signature app (d'arité 2), f_n, $n \geq 1$ (d'arité 2).

Montrer que si $M \to N$ en σ_c, alors $[M] \to^+ [N]$ dans un système de réécriture σ_c' que l'on déterminera. Montrer que σ_c' termine, et en déduire que σ_c termine. (Ceci devrait vous rappeler la preuve de terminaison de σ du poly.)

3. On admettra que σ_c est localement confluent. (Comme on l'a dit en cours, les machines sont bien meilleures à montrer ce genre de résultat que les êtres humains.) Montrer que σ_c est confluent. On demande en particulier le nom du théorème qui permet de conclure.

4. On définit le λ_c^*-calcul sur le langage des λ_c-termes, augmenté de la construction $(\lambda^* x \cdot M)N$. Formellement, l'ensemble des λ_c^*-termes est le plus petit vérifiant les conditions **(a)–(f)** de la page 1 (avec "λ_c-terme" remplacé par "λ_c^*-terme" partout) et tel que

   (g) $(\lambda^* x \cdot M)N$ est un λ_c^*-terme pour tous λ_c^*-termes M et N tels que $\mathrm{fv}(M) = \{x\}$; on a $\mathrm{fv}((\lambda^* x \cdot M)N) = \mathrm{fv}(N)$.

   Les règles de réduction du λ_c^* sont celles de σ_c, plus les règles

$$\begin{array}{cccc} (\beta^*) & (\lambda^* x \cdot M)N & \to & M\{x := N\} \\ (App^*) & ((\lambda^* x \cdot M)N)\{x := P\} & \to & (\lambda^* x \cdot M)(N\{x := P\}) \end{array}$$

   qui remplacent la règle (β) du λ_c-calcul. Montrer que le λ_c^*-calcul termine. Indication: on pourra exhiber une traduction simple du λ_c^*-calcul dans le λ_c-calcul qui envoie toute réduction du λ_c^*-calcul vers une réduction dans σ_c. (La définition du λ_c^*-calcul devrait vous rappeler le théorème des développements finis présenté en cours — et qui n'est pas dans le poly. La démonstration de terminaison est infiniment plus simple.)

   Errata: il fallait évidemment lire $((\lambda^* x \cdot M)N)\{y := P\} \to (\lambda^* x \cdot M)(N\{y := P\})$ *pour* (App^*)*, avec* y *libre dans* N*; ceci ne prête pas à conséquence.*

5. Si M est un λ_c^*-terme, son effacement $E(M)$ est le λ_c-terme obtenu en effaçant toutes les étoiles; autrement dit $E((\lambda^* x \cdot M)N) = (\lambda x \cdot E(M))(E(N))$, les autres cas étant évidents.

   Une *décoration* d'un λ_c-terme M est un λ_c^*-terme M' quelconque tel que $E(M') = M$. Autrement dit, on obtient les décorations de M en décidant d'ajouter ou pas une étoile au-dessus de chaque λ qui débute un β-rédex dans M.

   On notera que tout λ_c^*-terme sans étoile est β^*-normal (n'a pas de rédex pour la règle (β^*)). Réciproquement, tout λ_c^*-terme qui est β^*-normal est sans étoile, et est donc un λ_c-terme.

   Soit $\to_1$ la relation de réduction définie sur les λ_c-termes par $M \to_1 N$ si et seulement s'il existe une décoration M' de M tel que $M' \to^* N$ dans λ_c^*. (N.B.: comme N est un λ_c-terme, il est sans étoile.)

Montrer que si $M \to N$ dans le λ_c-calcul, alors $M \to_1 N$, et que si $M \to_1 N$ alors $M \to^* N$ dans le λ_c-calcul.

6. On admettra que λ_c^* est localement confluent. Montrer que $\to_1$ est fortement confluente.

7. En déduire que le λ_c-calcul est confluent.