MPRI

Devoir du cours de récriture d'ordre supérieur

Jean-Pierre Jouannaud

On considère la signature monomorphe suivante (R est un type, $0, 1, D, sin, cos, +, \times$ sont des symboles de fonctions, X, F, G sont des symboles de variables libres) :

$$\begin{array}{rcl} R & : & * \\ 0,1 & : & R \\ D & : & (R \to R) \Rightarrow (R \to R) \\ sin, cos & : & R \Rightarrow R \\ +, \times & : & R \times R \Rightarrow R \\ X & : & R \\ F, G & : & R \Rightarrow R \end{array}$$

On se propose d'étudier les règles d'ordre supérieur définissant la dérivation formelle d'expressions fonctionnelles baties sur les opérations de la signature :

$$\begin{array}{rcl} D(\lambda x.X) & \to & \lambda x.0 \\ D(\lambda x.x) & \to & \lambda x.1 \\ D(\lambda x.sin(F(x)) & \to & \lambda x.cos(F(x) \times (D(\lambda x.F(x))\ x) \\ D(\lambda x.cos(F(x)) & \to & \lambda x. - sin(F(x) \times (D(\lambda x.F(x))\ x) \\ D(\lambda x.(F(x) + (G(x)) & \to & \lambda x.(D(\lambda x.F(x))\ x) + (D(\lambda x.G(x))\ x) \\ D(\lambda x.(F(x) \times (G(x)) & \to & \lambda x. \\ & & (D(\lambda y.(F(y))\ x) \times (G(x) + (F(x) \times (D(\lambda y.(G(y))\ x) \end{array}$$

Notons que la variable libre X de la première règle ne peut dépendre de x car étant d'arité nulle. Cette non dépendance provient de la définition des substitutions qui interdit la capture de variables. C'est donc une constante (vis à vis de x). Par contre, la variable F est d'arité 1. Elle devra être substituée lors du calcul d'une instance du membre gauche de règle par une abstraction de la forme $\lambda x.s$, où s est un terme pouvant dépendre de x.

On rappele que la substitution σ unifie l'équation $A = B$, où A et B désignent des expressions d'ordre supérieur arbitraires baties sur la signature ci-dessus, si

$$A\sigma \xleftrightarrow[\beta\eta]{*} B\sigma$$

1. Donner un exemple de terme dont la récriture nécessite l'utilisation du filtrage d'ordre supérieur.
2. Montrer rigoureusement que les règles précédentes (moins la première) n'ont pas de paire critique d'ordre supérieur (il est nul besoin de savoir unifier à l'ordre supérieur pour cela).
3. Quelles règles de l'arithmétique faut-il rajouter pour traiter la première règle ?

Comme ces règles utilisent le filtrage d'ordre supérieur, l'étude de leur terminaison nécessite un ordre de récriture normale d'ordre supérieur. Pour cela, nous allons donner une version plus générale du horpo sur les termes et les types, en même temps que nous la restreindrons pour définir un ordre modulo.

$$
\begin{array}{c}
s : \sigma \succ t : \tau \text{ ssi} \\
\mathbf{Type} : \sigma = \tau = * \text{ ou } \sigma \succeq \tau \\
\text{et}
\end{array}
$$

$$
\begin{array}{rcl}
\text{Clôture} & : & s \doteq f(\overline{s}) \text{ et } \exists u \in \mathcal{CC}(s,t) \text{ t.q. } u \succeq t \\
\text{Prec} & : & s \doteq f(\overline{s}),\ t \doteq g(\overline{t}), f >_{\mathcal{FS}} g \text{ et } A \\
\text{Prec@} & : & s \doteq f(\overline{s}),\ f \in \mathcal{F},\ t \doteq @(\overline{t}) \text{ et } A \\
\text{Prec}\lambda & : & s \doteq f(\overline{s}), f \in \mathcal{F}, t \doteq \lambda x : \alpha.v, x \notin \mathcal{V}ar(v) \text{ et } s \succ v \\
\text{Mul} & : & s \doteq f(\overline{s}),\ t \doteq g(\overline{t}), f =_{\mathcal{FS}} g \in Mul \text{ et } \overline{s} \succ_{mul} \overline{t} \\
\text{Lex} & : & s \doteq f(\overline{s}),\ t \doteq g(\overline{t}), f =_{\mathcal{FS}} g \in Lex, \overline{s} \succ_{lex} \overline{t} \text{ et } A \\
\text{Mul@} & : & s \doteq @(s_1, s_2), t \doteq @(\overline{t}) \text{ et } \mathcal{CC}(@(s_1,s_2),t) \succ_{mul} \overline{t} \\
\text{Mon}\lambda & : & s \doteq \lambda x : \alpha.u,\ t \doteq \lambda y : \beta.v, \alpha =_{\mathcal{T_S}} \beta \text{ et } u \succ v\{y \mapsto x\} \\
\text{Mon}\rightarrow & : & s \doteq \alpha \rightarrow \beta,\ t \doteq \alpha' \rightarrow \beta', \alpha =_{\mathcal{T_S}} \alpha' \text{ et } \beta \succ \beta' \\
\text{Reduction} & : & s \doteq @(\lambda x.u, v) \text{ et } u\{x \mapsto v\} \succeq t
\end{array}
$$

où

$@(\overline{t})$ est une application étendue de t ;

$\succeq \doteq \succ \cup =$, où $=$ est l'équivalence sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$ engendrée par permutation des arguments des symboles de Mul ;

$=_{\mathcal{T}}$ et $=_{\mathcal{T_S}}$ sont les restriction de $=$ aux termes et aux types ;

$A \doteq \forall v \in \overline{t}\ s \succ v$ ou $\exists u \in \mathcal{CC}(s,t)$ t.q. $u \succeq v$, où la clôture par calcul $\mathcal{CC}(,)$ est définie comme suit :

$\mathcal{CC}(\lambda x : \sigma.u, t) \doteq \{u\}$ si $x \notin \mathcal{V}ar(t)$ sinon $\emptyset$

$\mathcal{CC}(\alpha \rightarrow \beta, \tau) \doteq \{\beta\}$

$\mathcal{CC}(@(s_1, s_2), t) \doteq \{s_1, s_2\}$ if s_1 is not an abstraction nor a variable, and $\{s_2\}$ sinon

$\mathcal{CC}(f(\overline{s}), t) \doteq \mathcal{CC}'(f(\overline{s}), \emptyset)$, où $f \in \mathcal{F}$ et

$\mathcal{CC}'(u = f(\overline{u}), \mathcal{V})$, pour $f \in \mathcal{F}$ et $\mathcal{V} \cap \mathcal{V}ar(u) = \emptyset$, est le plus petit ensemble de termes bien typés contenant $\mathcal{V} \cup \overline{u}$, et clos pour les opérations suivantes :

- sous terme : soit $v \in \mathcal{CC}'(u, \mathcal{V})$, et w sous-terme de type minimum de v tel que $\mathcal{V}ar(v) \subseteq \mathcal{V}ar(u)$; alors $w \in \mathcal{CC}'(u, \mathcal{V})$;
- precedence : soit g such that $f >_{\mathcal{F}} g$, and $\overline{v} \in \mathcal{CC}'(u, \mathcal{V})$; then $g(\overline{v}) \in \mathcal{CC}'(u, \mathcal{V})$;
- recursion : soit $\overline{v} \in \mathcal{CC}'(u, \mathcal{V})$ t.q. $f(\overline{v})$ est bien typé et $\overline{u}(\succ)_{stat_f}\overline{v}$; alors $g(\overline{v}) \in \mathcal{CC}'(u, \mathcal{V})$ pour tout $g =_{\mathcal{F}} f$;
- application : soit $v : \sigma \rightarrow \tau \in \mathcal{CC}'(u, \mathcal{V})$ et $w : \sigma \in \mathcal{CC}'(u, \mathcal{V})$; alors $@(v, w) \in \mathcal{CC}'(u, \mathcal{V})$;
- abstraction : $x \notin \mathcal{V}ar(u) \cup \mathcal{V}$ et $v \in \mathcal{CC}'(u, \mathcal{V} \cup \{x\})$; alors $\lambda x.v \in \mathcal{CC}'(u, \mathcal{V})$;
- reduction : soit $v \in \mathcal{CC}'(u, \mathcal{V})$, et $v \succeq w$; alors $w \in \mathcal{CC}'(u, \mathcal{V})$;
- affaiblissement : soit $x \notin \mathcal{V}ar(u, v) \cup \mathcal{V}$; alors $v \in \mathcal{CC}'(u, \mathcal{V} \cup \{x\})$ ssi $v \in \mathcal{CC}'(u, \mathcal{V})$.

On admettra que l'ordre ainsi défini est un ordre de récriture normale d'ordre supérieur.

3. Pourquoi est-il nécessaire de restreindre la définition de $\mathcal{CC}(s)$ dans le cas où s est une application ? Montrer que sinon la β-stabilité n'est pas satisfaite.
4. Argumentez pourquoi la nouvelle définition de la clôture par calcul dans le cas des termes dont le symbole de tête est dans $\mathcal{F}$ ne détruit pas la bonne fondation de horpo (quel est l'impact sur les preuves des propriétés des interprétations ?).
5. Montrer que les règles de dérivation formelle sont fortement normalisantes en utilisant l'ordre précédent.