TD Système F, types de données

E. Lozes

9 mai 2007

Exercice 1 – Rappel système F_2
On rappelle les règles d'inférence de types propres au système F_2 (ajouter Ax, $\Rightarrow I$, $\Rightarrow E$ pour tout avoir) :

$$\frac{\Gamma \vdash u : \forall X.F}{\Gamma \vdash uF' : F[X := F']}(\forall E) \quad \frac{\Gamma \vdash u : F \quad X \notin v(\Gamma)}{\Gamma \vdash \Lambda X.u : \forall X.F}(\forall I)$$

1. On définit $\bot$ comme $\forall X.X$. Donner un terme preuve de $\forall X.\bot \Rightarrow X$.
2. Proposer un encodage du type $F \wedge F'$ (on pourra se souvenir de l'encodage de la paire $\langle u, v\rangle$ dans le lambda-calcul pur). Donner un terme preuve de $A \Rightarrow B \Rightarrow C \Rightarrow (A \wedge B) \Rightarrow C$. Montrer que les règles $\wedge E_1$, $\wedge E_2$ et $\wedge I$ sont admissibles en système F_2 si on adopte cet encodage.
3. On se propose d'encoder $A \vee B$ de sorte que le terme $i_1 u$, preuve de $A \vee B$ lorsque u est une preuve de A, est défini par $i_1 u = \Lambda P.\lambda k_1, k_2.k_1 u$. Retrouver la définition de $A \vee B$. Proposer un encodage de $\mathsf{case}\ u \left\{ \begin{array}{l} i_1 x \to v_1 \\ i_2 x \to v_2 \end{array} \right\}$, vérifier le typage et les réductions associées.

Exercice 2 – Vers les types récursifs
On se propose d'étudier une des propriétés clé pour la définition d'un système de type récursif, ie dans lequel on a une construction $\mu X.F(X)$ telle que $\mu X.F(X) = F(\mu X.F(X))$. Pour étudier la normalisation forte d'un système de types, l'approche général consiste à définir pour tout type F l'ensemble Red_F des réductibles de type F, et d'établir pour chaque F les trois propriétés suivantes :

(CR1) si $u \in RED_F$, alors $u \in SN$,
(CR2) si $u \in RED_F$ et $u \to u'$, alors $u' \in RED_F$,
(CR3) si u est *neutre*, et si tout u' tel que $u \to u'$ appartient à RED_F, alors $u \in RED_F$

1. Rappeller la définition de $Red_{\forall X.F}$. Comment est résolue l'imprédicativité ?
2. Proposer une définition de $Red_{\mu X.F}$, et en déduire qu'on pourrait être tenté d'établir que les candidats de réducibilité, ordonnés par inclusion, forment un treillis complet.

3. Montrer que si $(R_i)_{i\in I}$ est une famille de candidats de réducibilité, $\bigcap_{i\in I} R_i$ est un candidat de réducibilité (on fera attention au cas d'une famille vide). Vérifier que en revanche $\bigcup_{i\in I} R_i$ n'est pas clairement un candidat de réducibilité.
4. Montrer qu'un ensemble ordonné qui admet une borne sup pour tout ensemble est un treillis complet.

Exercice 3 – Les types de données

1. On définit le type Bool comme $\forall X.X \Rightarrow X \Rightarrow X$. Définir les termes true, false, et $\mathsf{ifthenelse}$ de type Bool et $\forall X.\mathsf{Bool} \Rightarrow X \Rightarrow X \Rightarrow X$ respectivement.
2. Rappeler le type des entiers de Church en système F.
3. Les entiers sont les termes libres sur la signature $0 : \mathsf{Nat}$ et $s : \mathsf{Nat} \Rightarrow \mathsf{Nat}$. De façon similaire, les objets $\mathsf{Bool} + \mathsf{Nat}$ sont les termes de la signature $i_{Bool} : \mathsf{Bool} \Rightarrow \mathsf{Bool} + \mathsf{Nat}$ et $i_{Nat} : \mathsf{Nat} \Rightarrow \mathsf{Bool} + \mathsf{Nat}$. On souhaite de même représenter le type X des termes libres construits sur une signature de fonctions de la forme générale $f : U_1 \Rightarrow .. \Rightarrow U_n \Rightarrow X \Rightarrow .. \Rightarrow X$. Si $\Sigma = f_1 : S_1, .., f_n : S_n$ est une telle signature, on pose $\mathsf{Type}(\Sigma) = \forall X.S_1 \Rightarrow .. \Rightarrow S_n \Rightarrow X$. Vérifier que c'est la construction appliquée pour les booléens, les entiers, le produit et l'union.
4. Proposer une définition du type des listes d'objets de type U et des arbres binaires d'objets de type U. Comment se traduisent les constructeurs f_i ?
5. Un type de données ne contient (presque) que des termes équivalents aux termes libres. Vérifier, dans le cas des entiers, qu'un λ-terme normalisé de type Nat est un entier de Church $\Lambda X \lambda x, f.f^n(x)$ (on pourra raisonner sur la forme normale de tête).

Exercice 4 – Types existentiels

On veut montrer que les types existentiels $\exists X.F$ sont admissibles dans le système F_2, au sens où on sait donner un encodage de $\exists X.F$ et des règles :

$$\frac{\Gamma \vdash V[X := U]}{\Gamma \vdash \exists X.V}(I\exists) \qquad \frac{\Gamma \vdash \exists X.V \quad \Gamma, V \vdash W \quad X \notin \mathsf{fv}(\Gamma, W)}{\Gamma \vdash W}(E\exists)$$

On propose d'encoder $\exists X.F$ en $\forall Y.(\forall X.F \Rightarrow Y) \Rightarrow Y$. Montrer que les règles ci-dessus sont admissibles au sens où on peut donner des termes $\langle U, v\rangle$ de type $\exists X.V$ et $\nabla X, x.w$ de type $(\exists X.V) \Rightarrow W$, pour certaines hypothèses de typage des sous-termes v et w que l'on précisera. Interpréter l'élimination des coupures pour $\exists$ sur le redex $(\nabla X, x.w)\langle U, v\rangle$.