Types récursifs. Relations logiques.

Documents autorisés (en particulier le poly).

Les questions sont annotées par un niveau de difficulté, variant de (0) (facile) à (3) (difficile).

En comptant 5 minutes passées par problème facile, 10 minutes par problème de difficulté (1), 15 minutes par problème de difficulté (2), et 20 minutes par problème de difficulté (3), vous aurez besoin de 2h05 pour tout faire, ce qui vous laisse le temps nécessaire pour lire ce très long énoncé ...

Types récursifs

On se propose d'examiner une variante du système des types simples autorisant d'écrire des *types récursifs*. C'est déjà ce que propose CaML, où un type défini peut être défini en fonction de lui-même, par exemple :

```
type arbre = FEUILLE of int
           | NOEUD of arbre * arbre;;
type paradox = P of paradox -> paradox;;
type nat = O
         | S of nat;;
```

On va, pour représenter cette récursivité, ajouter au langage des types simples une construction $\mu X \cdot f(X)$, qui dénote en gros l'unique type τ que l'on définirait en CaML en écrivant `type` τ `=` $f(\tau)$`;;`. Par exemple, on coderait les types ci-dessus sous la forme de `arbre` $= \mu X \cdot \texttt{int} + (X \times X)$, `paradox` $= \mu X \cdot X \Rightarrow X$, `nat` $= \mu X \cdot \texttt{unit} + X$.

Formellement, les types sont :

$$F ::= b | X | F \Rightarrow G | \mu X \cdot F$$

où les b sont des types de base, les X des variables de type et où la variable de type X est liée dans $\mu X \cdot F$. On supposera que l'α-renommage est appliqué aux types implicitement.

On note $\sim$ la plus petite congruence sur les types telle que $\mu X \cdot F \sim F[X := \mu X \cdot F]$. (On rappelle qu'une congruence est une relation d'équivalence qui passe au contexte.)

Les règles de typage sont celles des types simples plus une règle portant sur les types récursifs. Appelons le système suivant *système* μ :

$$\frac{}{\Gamma, x : F \vdash x : F} (Var)$$

$$\frac{\Gamma \vdash u : F_1 \Rightarrow F_2 \quad \Gamma \vdash v : F_1}{\Gamma \vdash uv : F_2} (App) \qquad \frac{\Gamma, y : F_1 \vdash u[x := y] : F_2}{\Gamma \vdash \lambda x \cdot u : F_1 \Rightarrow F_2} (Abs)$$

$$\frac{\Gamma \vdash u : F \quad F \sim G}{\Gamma \vdash u : G} (TEq)$$

On ne considérera que la règle β, pas η, comme règle de réduction du λ-calcul dans cette partie.

1. (1) Montrer que tout λ-terme (non typé) u est de type $\mu X \cdot X \Rightarrow X$ dans n'importe quel contexte Γ envoyant chaque variable libre dans u vers $\mu X \cdot X \Rightarrow X$ dans le système μ.

2. (0) En déduire qu'il existe des λ-termes non fortement normalisants, et même non faiblement normalisants, dans le système μ. (On exhibera un contre-exemple.)

3. (3) On va réparer ce défaut en considérant un sous-système μ^+ du système μ, où dans $\mu X \cdot F$ la variable X ne peut avoir que des occurrences positives dans F. (Informellement, si l'on voit les types comme des formules, les occurrences positives sont celles qui apparaissent sous un nombre pair de négations, chaque implication $F \Rightarrow G$ comptant comme une négation de F.)

   Formellement, on dit que X *apparaît positivement* dans une variable de type Y si et seulement si $X = Y$; dans $F \Rightarrow G$ si X apparaît positivement dans G ou négativement dans F; dans $\mu Y \cdot F$ $(Y \neq X)$ si X apparaît positivement dans F. Et que, symétriquement, X *apparaît négativement* dans un type si et seulement si : ce type est $F \Rightarrow G$ et X apparaît négativement dans G ou positivement dans F; ou ce type est $\mu Y \cdot F$ $(Y \neq X)$ et X apparaît négativement dans F.

   Le système μ^+ est la restriction du système μ où, dans tout type $\mu X \cdot F$, X n'apparaît pas négativement dans F.

   On rappelle qu'un *candidat de réductibilité* est un ensemble de λ-termes S tel que :

(CR1) Si $u \in S$, alors $u \in SN$.

(CR2) Si $u \in S$ et $u \to u'$, alors $u' \in S$.

(CR3) Si u est neutre et pour tout u' tel que $u \to u'$, $u' \in S$, alors $u \in S$.

   On admettra d'autre part le théorème suivant, qui est une instance du théorème de point fixe de Tarski :
   Théorème (Tarski) Soit CR l'ensemble de tous les candidats de réductibilité, $f : CR \to CR$ une fonction monotone ($A \subseteq B \Rightarrow f(A) \subseteq f(B)$), alors f a un *plus petit point fixe* $lfp(f)$. (On rappelle qu'un point fixe de f est un candidat A tel que $f(A) = A$.)
   De plus, si f et g sont deux fonctions monotones telles que $f(A) \subseteq g(A)$ pour tout $A \in CR$, alors $lfp(f) \subseteq lfp(g)$.

   On définit alors une notion RED_F adaptée. Pour tout *contexte de candidats* C qui à chaque variable de type X associe un candidat de réductibilité, pour tout type F de μ^+, on définit RED_F^C par :

$$\begin{array}{rcl} RED_b^C & \hat{=} & SN \\ & & \text{(ensemble des termes fortement normalisants)} \\ RED_X^C & \hat{=} & C(X) \\ RED_{F\Rightarrow G}^C & \hat{=} & \{u \mid \forall v \in RED_F^C \cdot uv \in RED_G^C\} \\ RED_{\mu X \cdot F}^C & \hat{=} & lfp(S \text{ candidat} \mapsto RED_F^{C[X:=S]}) \end{array}$$

   Montrer que RED_F^C est bien défini. (La difficulté principale est de montrer que (S candidat $\mapsto RED_F^{C[X:=S]}$) est bien monotone. On montrera donc simultanément que : (1) RED_F^C est bien défini, (2) si X n'apparaît pas négativement dans F alors RED_F^C est monotone en $C(X)$, au sens où pour tout S tel que $C(X) \subseteq S$, $RED_F^C \subseteq RED_F^{C[X:=S]}$, (3) si X n'apparaît pas positivement dans F alors RED_F^C est antitone en $C(X)$, au sens où pour tout S tel que $C(X) \subseteq S$, $RED_F^C \supseteq RED_F^{C[X:=S]}$.)

4. (1) Justifier que RED_F^C est un candidat de réductibilité pour tout contexte de candidats C et pour tout type F. (Au lieu de refaire la preuve complètement, on pourra pour certains des cas à examiner indiquer où ils ont déjà été traités dans le cours.)

5. (3) En déduire que tout terme typé dans le système μ^+ est fortement normalisant. (Encore une fois, on pourra se référer à certaines preuves du cours pour les parties de la preuve qui ne changent pas ici.)

Théorème de Friedman

On considère dans cette partie le λ-calcul typé dans le système des types simples. Rappelons que les termes sont :

$$u, v, \ldots ::= x \mid \lambda x \cdot u \mid uv$$

et que les règles de typage sont :

$$\frac{}{\Gamma, x : F \vdash x : F}\,(Var)$$

$$\frac{\Gamma \vdash u : F_1 \Rightarrow F_2 \quad \Gamma \vdash v : F_1}{\Gamma \vdash uv : F_2}\,(App) \qquad \frac{\Gamma, y : F_1 \vdash u[x := y] : F_2}{\Gamma \vdash \lambda x \cdot u : F_1 \Rightarrow F_2}\,(Abs)$$

Une *structure applicative standard* est donnée par une famille d'ensembles E_b, pour tout type de base b. On définit alors E_F pour tout type F par récurrence structurelle sur F comme suit: quand $F = b$, E_b est donné, et quand $F = F_1 \Rightarrow F_2$, E_F est l'ensemble de toutes les applications (=fonctions totales) de E_{F_1} vers E_{F_2}.

Un *environnement* ρ est une fonction qui à chaque variable x associe un élément de $\bigcup_F E_F$.

Un environnement *respecte* un contexte de type $\Gamma \hat{=} x_1 : F_1, \ldots, x_n : F_n$ si et seulement si $\rho(x_i) \in E_{F_i}$ pour tout i, $1 \leq i \leq n$.

On définit une interprétation, dite *standard*, des λ-termes simplement typés en interprétant l'application syntaxique du λ-calcul comme étant la véritable application des fonctions à leurs arguments. Formellement, pour tout contexte Γ, pour tout environnement ρ respectant Γ, pour toute dérivation de typage π de $\Gamma \vdash u : F$, on définit $[\![\pi]\!]\rho$ par récurrence structurelle sur π :

$$\begin{aligned}
\left[\!\!\left[\frac{}{\Gamma, x : F \vdash x : F}\,(Var)\right]\!\!\right]\rho &\;\hat{=}\; \rho(x)\\
\left[\!\!\left[\frac{\begin{matrix}\vdots\,\pi_1 \\ \Gamma \vdash u : F_1 \Rightarrow F_2\end{matrix} \quad \begin{matrix}\vdots\,\pi_2 \\ \Gamma \vdash v : F_1\end{matrix}}{\Gamma \vdash uv : F_2}\,(App)\right]\!\!\right]\rho &\;\hat{=}\; f(v), \text{ où } f \hat{=} \left[\!\!\left[\begin{matrix}\vdots\,\pi_1 \\ \Gamma \vdash u : F_1 \Rightarrow F_2\end{matrix}\right]\!\!\right]\rho, \quad v \hat{=} \left[\!\!\left[\begin{matrix}\vdots\,\pi_2 \\ \Gamma \vdash v : F_1\end{matrix}\right]\!\!\right]\rho\\
\left[\!\!\left[\frac{\begin{matrix}\vdots\,\pi_1 \\ \Gamma, x : F_1 \vdash u : F_2\end{matrix}}{\Gamma \vdash \lambda x \cdot u : F_1 \Rightarrow F_2}\,(Abs)\right]\!\!\right]\rho &\;\hat{=}\; \text{la fonction qui à } v \in E_{F_1} \text{ associe } \left[\!\!\left[\begin{matrix}\vdots\,\pi_1 \\ \Gamma, x : F_1 \vdash u : F_2\end{matrix}\right]\!\!\right](\rho[x := v])
\end{aligned}$$

Par abus de langage, on résumera parfois une dérivation π de $\Gamma \vdash u : F$ au seul terme de preuve u, ce qui permettra d'écrire $[\![x]\!]\rho = \rho(x)$, $[\![uv]\!]\rho = [\![u]\!]\rho([\![v]\!]\rho)$, $[\![\lambda x \cdot u]\!]\rho = (v \mapsto [\![u]\!](\rho[x := v]))$.

Il est évident (on ne demande pas de le démontrer, mais vous devriez vous en convaincre) que cette interprétation est *correcte*, au sens où :

(i) si π dérive $\Gamma \vdash u : F$, et si ρ respecte Γ, alors $[\![\pi]\!]\rho$ est un élément de E_F;

(ii) si π et π' sont deux preuves $\beta\eta$-convertibles, alors $[\![\pi]\!]\rho = [\![\pi']\!]\rho$ pour tout ρ.

On va démontrer ici une forme de réciproque au point (ii). C'est un théorème de complétude dû à Harvey Friedman.

Pour ceci, on considère des familles de relations binaires R_F indicées par des types F. Informellement, pour chaque F, R_F est une relation entre termes d'une part et éléments de E_F d'autre part. On notera $u\ R_F\ a$ pour dire que u et a sont en relation par R_F.

On dit que $R \hat{=} (R_F)_{F \text{ type}}$ est une *relation logique* si et seulement si d'une part elle est compatible avec la convertibilité, autrement dit $u =_{\beta\eta} v$ et $u\ R_F\ a$ implique $v\ R_F\ a$; et si d'autre part:

Pour tous types F_1 et F_2, pour tout terme u, pour tout $f \in E_{F_1 \Rightarrow F_2}$, $u\ R_{F_1 \Rightarrow F_2}\ f$
si et seulement si
pour tout terme v, pour tout a in E_{F_1} tels que $v\ R_{F_1}\ a$, on a $uv\ R_{F_2}\ f(a)$.

Plus succinctement, R est une relation logique si et seulement

$$u\ R_F\ a \wedge u =_{\beta\eta} v \quad \Rightarrow \quad v\ R_F\ a \tag{1}$$

$$u\ R_{F_1 \Rightarrow F_2}\ f \quad \Leftrightarrow \quad (\forall v, a \cdot v\ R_{F_1}\ a \Rightarrow uv\ R_{F_2}\ f(a)) \tag{2}$$

(On remarquera la similarité avec la notion de réductibilité.)

1. (2) Montrer que si $x_1 : F_1, \ldots, x_n : F_n \vdash u : F$ est dérivable, et si $v_i \; R_{F_i} \; a_i$ pour tout i, $1 \leq i \leq n$, alors $u[x_1 := v_1, \ldots, x_n := v_n] \; R_F \; [\![u]\!][x_1 := a_1, \ldots, x_n := a_n]$. (On rappelle que $[x_1 := v_1, \ldots, x_n := v_n]$ dénote la substitution envoyant chaque x_i vers v_i, et que $[x_1 := a_1, \ldots, x_n := a_n]$ est l'environnement envoyant chaque x_i vers a_i.)

2. (0) En déduire que si pour tout F, R_F est une relation *injective modulo* $\beta\eta$ (autrement dit $u \; R_F \; a$ et $v \; R_F \; a$ impliquent $u =_{\beta\eta} v$), alors pour tous termes clos u et v de même type F, tels que $[\![u]\!][] = [\![v]\!][]$, on a en fait $u =_{\beta\eta} v$.

3. (2) On note $[u]$ la classe d'équivalence d'un terme quelconque u modulo $\beta\eta$. Pour tout type F, posons Λ_F l'ensemble des termes clos de type F modulo $\beta\eta$; formellement, $\Lambda_F \hat{=} \{[u] \mid \vdash u : F \text{ dérivable}\}$.

   On construit une interprétation standard $(E_F)_{F \text{ type}}$ en posant $E_b \hat{=} \Lambda_b$ pour tout type de base b.

   On définit maintenant une famille de fonctions $r_F : E_F \to \Lambda_F$ et $i_F : \Lambda_F \to E_F$ par récurrence sur F comme suit :

$$
\begin{array}{rcl@{\qquad}rcl}
r_b(a \in E_b) & \hat{=} & a \in \Lambda_b & i_b([u] \in \Lambda_b) & \hat{=} & [u] \in E_b \\
r_{F \Rightarrow G}(f \in E_{F \Rightarrow G}) & \hat{=} & \lambda x \cdot r_G(f(i_F([x]))) & i_{F \Rightarrow G}([u] \in \Lambda_{F \Rightarrow G}) & \hat{=} & (a \mapsto i_G([u] \; r_F(a)))
\end{array}
$$

   où la notation $\lambda x \cdot [u]$ dénote $[\lambda x \cdot u]$ dans la définition de $r_{F \Rightarrow G}$ (x est une variable quelconque fixée), et où $[u] \; r_F(a)$ (définition de $i_{F \Rightarrow G}$) dénote $[uv]$, en posant $[v] \hat{=} r_F(a)$.

   Montrer que (r_F, i_F) définit une *rétraction* de E_F sur Λ_F au sens où $r_F \circ i_F$ est la fonction identité id_{Λ_F}.

4. (2) Montrer que la famille de relations R_F définies par $u \; R_F \; a$ si et seulement si $a = i_F([u])$ est une relation logique.

5. (1) Déduire des questions précédentes qu'il existe une structure applicative standard dans laquelle pour tous termes u et v tels que $\Gamma \vdash u : F$ et $\Gamma \vdash v : F$ soient dérivables, $u =_{\beta\eta} v$ si et seulement si pour tout environnement ρ qui respecte Γ, $[\![u]\!]\rho = [\![v]\!]\rho$.