Examen de λ-calcul 2014: types de données

Correction.

Documents autorisés (en particulier le poly).

J'ai mis une indication de barême en début de chaque question. Elle n'est absolument pas contractuelle, et pourra varier. La morale est : ne passez pas trop de temps sur les questions faciles (à 0,5 pt.) dont le total ne vous fournira au mieux que 5 points sur 20.

On rappelle les règles :

$$\begin{array}{rrcll} (\beta) & (\lambda x \cdot u)v & \to & u[x := v] & \\ (\eta) & \lambda x \cdot ux & \to & u & \text{si } x \text{ n'est pas libre dans } u \end{array}$$

closes par passage au contexte. Une *β-réduction* est une réécriture où chaque étape utilise la règle (β) seule. Une *$\beta\eta$-réduction* est une réduction où chaque étape utilise la règle (β) ou la règle (η).

On admettra sans preuve les résultats suivants :

(i) Si un terme clos u se β-réduit en un terme v, alors v est clos lui aussi.

(ii) La relation de $\beta\eta$-réduction (et pas seulement la β-réduction) est confluente.

(iii) Si un terme u se $\beta\eta$-réduit en un terme w, alors on peut repousser toutes les η-réductions : il existe un terme u' tel que u se β-réduit en u', et u' se η-réduit en w.

1 Un peu de λ-calcul pur

On notera $\to_{\beta\eta}$ la relation de $\beta\eta$-réduction.

1. [0,5 pt.] Montrer que si u se η-réduit en u', alors u et u' ont exactement les mêmes variables libres.

   Par récurrence sur la profondeur du η-rédex contracté. Dans le cas de base, $u = \lambda x \cdot tx$ avec $x \notin \mathrm{fv}(t)$ (c'est important !), donc $\mathrm{fv}(u) = (\mathrm{fv}(t) \cup \{x\}) \smallsetminus \{x\} = \mathrm{fv}(t)$, et $u' = t$ donc $\mathrm{fv}(u') = \mathrm{fv}(t) = \mathrm{fv}(u)$.

   Les cas de récurrence sont des appels triviaux à l'hypothèse de récurrence.

2 Ensembles saturés

Par définition, un ensemble S de λ-termes est *saturé* si et seulement si S est stable par β-réduction *inverse* : si u se β-réduit en une étape en u' et que $u' \in S$, alors $u \in S$.

On notera $\Rrightarrow$ l'opérateur suivant : $S \Rrightarrow T = \{u \mid \forall v \in S \,.\, uv \in T\}$.

2. [0,5 pt.] Montrer que si S et T sont des ensembles saturés, alors $S \Rrightarrow T$ est saturé lui aussi. Toutes les hypothèses sont-elles nécessaires ?

   ***Supposons que** $u \to_\beta u'$ **et** $u' \in S \Rrightarrow T$. **Par définition, pour tout terme** v **dans** S, $u'v$ **est dans** T. **Comme** $uv \to_\beta u'v$ **et que** T **est saturé,** $uv \in S$. **Puisque** v **est arbitraire,** u **est dans** $S \Rrightarrow T$.*

   Toutes les hypothèses ne sont pas nécessaires : nous n'avons pas eu besoin de l'hypothèse selon laquelle** S **est saturé.

3. [0,5 pt.] Montrer que toute intersection $\bigcap_{i \in I} S_i$ d'ensembles saturés est saturée (y compris pour $I = \emptyset$, pour lequel cette intersection est vue comme l'ensemble Λ de tous les λ-termes).

   C'est trivial. Si** $u \to_\beta u'$ **et** u' **est dans** $\bigcap_{i \in I} S_i$, u' **est dans chaque** S_i. **Par définition,** u **est aussi dans chaque** S_i, **donc dans leur intersection.

On considérera les types du système F :

$$
\begin{array}{rcll}
F, G, \cdots & ::= & \alpha & \text{variables de types} \\
 & | & F \Rightarrow G & \text{implications} \\
 & | & \forall \alpha \cdot F & \text{quantifications universelles.}
\end{array}
$$

On définit, comme pour $RED_F^{\mathcal{C}}$ dans le cours :

$$
R_\alpha^{\mathcal{C}} = \mathcal{C}(\alpha) \quad R_{F \Rightarrow G}^{\mathcal{C}} = \{u \mid \forall v \in R_F^{\mathcal{C}} \,.\, uv \in R_G^{\mathcal{C}}\} \quad R_{\forall \alpha \cdot F}^{\mathcal{C}} = \bigcap_{S \text{ saturé}} R_F^{\mathcal{C}[\alpha \mapsto S]}
$$

mais **attention !** On ne supposera plus ici que $\mathcal{C}$ est un contexte de candidats. On supposera simplement que pour toute variable de type α, $\mathcal{C}(\alpha)$ est un ensemble *saturé* de λ-termes. On appellera un tel $\mathcal{C}$ un *contexte saturé*.

On pourra utiliser, sans preuve, les résultats suivants :

(iv) Pour tout type F, si $\mathcal{C}$ est un contexte saturé, alors $R_F^{\mathcal{C}}$ est un ensemble saturé. C'est une conséquence facile des questions 2 et 3.

(v) $R_{F[\alpha:=G]}^{\mathcal{C}} = R_F^{\mathcal{C}[\alpha \mapsto R_G^{\mathcal{C}}]}$.

(vi) Si α n'est pas libre dans F, alors $R_F^{\mathcal{C}}$ est indépendant de $\mathcal{C}(\alpha)$, c'est-à-dire que pour tout ensemble saturé S, $R_F^{\mathcal{C}[\alpha \mapsto S]} = R_F^{\mathcal{C}}$.

(vii) Comme cas particulier de (vi), pour tout type F *clos*, c'est-à-dire sans variable de type libre, $R_F^{\mathcal{C}}$ est indépendant de $\mathcal{C}$. On notera dans ce cas R_F l'ensemble saturé $R_F^{\mathcal{C}}$, où $\mathcal{C}$ est un contexte saturé arbitraire.

Attention ! (de nouveau) On ne confondra pas la notation R_F employée ici avec la notion RED_F utilisée en cours pour les types simples (même si elle est très proche).

On rappelle les règles du système F, pour éviter toute ambiguïté :

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

$$\frac{\Gamma, x : F \vdash u : G}{\Gamma \vdash \lambda x \cdot u : F \Rightarrow G} (\Rightarrow I) \qquad \frac{\Gamma \vdash u : F \Rightarrow G \quad \Gamma \vdash v : F}{\Gamma \vdash uv : G} (\Rightarrow E)$$

$$\frac{\Gamma \vdash u : F}{\Gamma \vdash u : \forall \alpha \cdot F} (\forall I) \qquad \frac{\Gamma \vdash u : \forall \alpha \cdot F}{\Gamma \vdash u : F[\alpha := G]} (\forall E)$$

où α n'est pas libre dans Γ

La différence principale (avec F_2) est qu'on a le même terme en haut et en bas, dans les règles $(\forall I)$ et $(\forall E)$.

4. [0,5 pt.] Soient S, T deux ensembles saturés. Montrer que si $u[x := v] \in T$ pour tout $v \in S$, alors $\lambda x \cdot u$ est dans $S \Rrightarrow T$. A titre d'indication, c'est dans le même esprit, mais beaucoup plus simple qu'un lemme similaire dans le cours. Il n'y a notamment pas à démontrer ou à utiliser de propriété de la forme (CR1), (CR2) ou (CR3).

   Il suffit de démontrer que pour tout $v \in S$, $(\lambda x \cdot u)v$ est dans T. Or $(\lambda x \cdot u)v$ se β-réduit en $u[x := v]$, qui par hypothèse est dans T. Il est donc lui-même dans T, puisque T est saturé.

5. [3 pts.] Montrer que pour toute dérivation de jugement $x_1 : F_1, \cdots, x_n : F_n \vdash u : F$ en système F, pour tout contexte saturé $\mathcal{C}$, pour tous $v_1 \in R^{\mathcal{C}}_{F_1}, \ldots, v_n \in R^{\mathcal{C}}_{F_n}$, $u[x_1 := v_1, \cdots, x_n := v_n]$ est dans $R^{\mathcal{C}}_F$.

   C'est la même démonstration que pour le théorème de normalisation forte du système F, en remplaçant les candidats par les ensembles saturés.

   Par récurrence sur la dérivation de typage. C'est évident si la dernière règle est (Ax), c'est la définition de $R^{\mathcal{C}}_{F_1 \Rightarrow F_2}$ si la dernière règle est $(\Rightarrow E)$, c'est la question précédente si la dernière règle est $(\Rightarrow I)$.

   Dans le cas de $(\forall E)$, on obtient par hypothèse de récurrence que $u[x_1 := v_1, \cdots, x_n := v_n]$ est dans $R^{\mathcal{C}}_{\forall \cdot F}$, donc dans $R^{\mathcal{C}[\alpha \mapsto S]}_F$ pour tout ensemble saturé S ; en choisissant $S = R^{\mathcal{C}}_G$ et en utilisant (v), on obtient le résultat désiré $u[x_1 := v_1, \cdots, x_n := v_n] \in R^{\mathcal{C}}_{F[\alpha := G]}$.

   Dans le cas de $(\forall I)$, on obtient $u[x_1 := v_1, \cdots, x_n := v_n] \in R^{\mathcal{C}}_F$, et ce pour tout contexte saturé, en particulier pour tout contexte saturé de la forme $\mathcal{C}[\alpha \mapsto S]$, où S est un ensemble saturé arbitraire. Il est dans donc $R^{\mathcal{C}}_{\forall \alpha \cdot F}$.

On en déduit notamment que :

$(viii)$ tout terme clos u, typable en système F d'un type clos F, est dans R_F.

3 Types de données en système F

On appellera *type de données* tout type F qui est :

– clos,

– et tel que tout terme $u \in R_F$ se β-réduit en un terme clos.

Attention ! On différenciera bien les *types* tout court (les formules) des types *de données*. Le but de la section 3 est d'exhiber certains types de données, et nous verrons un type qui n'est pas de données à la section 4.

6. [2 pts.] Montrer qu'un type clos F est un type de données si et seulement si tout terme $u \in R_F$ est $\beta\eta$-équivalent à un terme clos. (On notera bien : pas juste β-équivalent. Bien relire le tout début du sujet si nécessaire.)

   Si F est un type de données, pour tout $u \in R_F$, u se β-réduit en un terme clos u'. En particulier, u' est $\beta\eta$-équivalent à u.

   Réciproquement, supposons que pour tout $u \in R_F$, u soit $\beta\eta$-équivalent à un terme clos v. Comme $\beta\eta$ est confluente (ii), u et v ont un $\beta\eta$-réduit commun w. On a admis que (i) tout β-réduit d'un terme clos était clos, et la question 1 implique que tout η-réduit d'un terme clos est clos : donc w est clos.

   Par le résultat admis (iii), u se β-réduit en un terme u' qui se η-réduit en w. Par la question 1 de nouveau, u' a les mêmes variables libres que w, et est donc clos. Donc u se β-réduit en un terme clos.

   Comme u est arbitraire, F est un type de données.

Lorsque F est un type de données, les classes d'équivalences, pour la $\beta\eta$-équivalence, des termes (clos) de R_F seront appelées les *valeurs* du type de données F.

On commence par le type `unit`, défini comme étant $\forall\alpha \cdot \alpha \Rightarrow \alpha$.

Pour tout ensemble A de λ-termes, notons $\uparrow A$ l'ensemble des termes qui se β-réduisent, en un nombre arbitraire d'étapes de β-réduction, en un terme de A. Clairement, $\uparrow A$ est un ensemble saturé.

Par abus de langage, si A est un singleton $\{t\}$, on notera $\uparrow t$ plutôt que la notation plus lourde $\uparrow\{t\}$.

7. [0,5 pt.] Montrer que, pour tout terme u de $R_{\texttt{unit}}$, u est dans $\uparrow x \Rrightarrow \uparrow x$, et en déduire que ux se β-réduit en x.

   Par définition, u est dans $R_{\alpha\Rightarrow\alpha}^{[\alpha\mapsto S]}$ où S est n'importe quel ensemble saturé… par exemple $\uparrow x$. Par définition, ceci signifie que u est dans $\uparrow x \Rrightarrow \uparrow x$.

   On expanse la définition de $\Rrightarrow$, et on obtient notamment que (puisque $x \in \uparrow x$) ux se β-réduit en x.

8. [0,5 pt.] Montrer que si u est un terme, où x n'apparaît pas libre, et tel que ux est $\beta\eta$-équivalent à x, alors u est $\beta\eta$-équivalent à $\lambda x \cdot x$.

Si ux est $\beta\eta$-équivalent à x, alors $\lambda x \cdot ux$ est $\beta\eta$-équivalent à $\lambda x \cdot x$ par application de contexte. Si l'on prend bien soin de choisir x non libre dans u, le premier terme est $\beta\eta$-équivalent à u, donc u est $\beta\eta$-équivalent à $\lambda x \cdot x$.

9. [0,5 pt.] En déduire que `unit` est un type de données, et déterminer ses valeurs.

   Bien sûr, `unit` *est un type clos.*

   Par la question précédente, tout terme de $R_{\texttt{unit}}$ est $\beta\eta$-équivalent au terme clos $\lambda x \cdot x$. Le type `unit` *est donc un type de données, et (la classe de $\beta\eta$-équivalence de) $\lambda x \cdot x$ est son unique valeur.*

10. [1 pt.] Passons au type $\texttt{bool} = \forall \alpha \cdot \alpha \Rightarrow \alpha \Rightarrow \alpha$. En considérant l'ensemble saturé $\uparrow\{x, y\}$, pour deux variables distinctes x et y, et en utilisant la même stratégie que plus haut, montrer que `bool` est un type de données, et déterminer ses valeurs.

    Il est facile de voir que `bool` *est un type clos.*

    Soit $u \in R_{\texttt{bool}}$. Par définition, $u \in \uparrow\{x, y\} \Rrightarrow \uparrow\{x, y\} \Rrightarrow \uparrow\{x, y\}$, donc uxy se $\beta\eta$-réduit en x ou en y.

    En particulier, u, qui est $\beta\eta$-équivalent à $\lambda x, y \cdot uxy$, est $\beta\eta$-équivalent à $\lambda x, y \cdot x$ ("vrai") ou à $\lambda x, y \cdot y$ ("faux"). Il suffit de choisir x et y distinctes et non libres dans u. Le type `bool` *est donc un type de données, et ses valeurs sont les deux booléens de Church (à $\beta\eta$-équivalence près)*

11. [3 pts.] On notera $\ulcorner n \urcorner$ l'entier de Church $\lambda f, x \cdot f^n(x)$. Montrer que le type $\texttt{nat} = \forall \alpha \cdot (\alpha \Rightarrow \alpha) \Rightarrow (\alpha \Rightarrow \alpha)$ est un type de données, et déterminer ses valeurs. Il est vivement recommandé, pour cette question, mais aussi pour la suite, de démontrer que pour tout terme $u \in R_{\texttt{nat}}$, pour toutes variables f et x, ufx se β-réduit en $f^n(x)$ pour un certain $n \in \mathbb{N}$; on choisira pour cela un ensemble saturé de la forme $\uparrow A$, où A est un ensemble de termes ayant une parenté forte avec les valeurs à trouver, comme dans les questions précédentes.

    Il est facile de voir que `nat` *est un type clos.*

    On fixe deux variables f et x distinctes, et l'on pose A l'ensemble des termes de la forme $f^n(x)$, $n \in \mathbb{N}$.

    Par définition, u est dans $(\uparrow A \Rrightarrow \uparrow A) \Rrightarrow (\uparrow A \Rrightarrow \uparrow A)$.

    On note que f est dans $\uparrow A \Rrightarrow \uparrow A$: pour tout terme $u \in \uparrow A$, donc qui se β-réduit en $f^n(x)$, fu se β-réduit en $f^{n+1}(x)$, et est donc dans $\uparrow A$.

    Aussi, $x \in \uparrow A$ (prendre $n = 0$).

    Donc ufx est dans $\uparrow A$. Il s'ensuit que ufx se β-réduit en un terme de la forme $f^n(x)$. En particulier, u, qui est $\beta\eta$-équivalent à $\lambda f, x \cdot ufx$, est $\beta\eta$-équivalent à l'entier de Church $\ulcorner n \urcorner = \lambda f, x \cdot f^n(x)$. (On prend f et x distinctes et non libres dans u.)

    Il s'ensuit que `nat` *est un type de données. Quant à ses valeurs, ce sont les (classes d'équivalence) des entiers de Church.*

12. [3 pts.] Le produit $F \times G$ de deux types F et G est $\forall \alpha \cdot (F \Rightarrow G \Rightarrow \alpha) \Rightarrow \alpha$, où α est une variable de type fraîche. Montrer que, si F est G sont tous les deux des types de données, alors $F \times G$ aussi. On déterminera aussi les valeurs de $F \times G$ en fonction de celles de F et de G.

    Il est facile de voir que $F \times G$ est un type clos dès que F et G sont clos.

    Fixons une variable z, et notons A l'ensemble des termes de la forme zst, où $s \in R_F$ et $t \in R_G$.

    Soit $u \in R_{F \times G}$. Par définition, $u \in (R_F \Rrightarrow R_G \Rrightarrow \uparrow A) \Rrightarrow \uparrow A$. La variable z est dans $R_F \Rrightarrow R_G \Rrightarrow \uparrow A$, car pour tout $s \in R_F$ et pour tout $t \in R_G$, zst est dans A (donc dans $\uparrow A$).

    Donc uz est dans $\uparrow A$, c'est-à-dire se β-réduit en un terme de la forme zst, où $s \in R_F$ et $t \in R_G$. On en déduit comme plus haut que u est $\beta\eta$-équivalent à $\lambda z \cdot zst$.

    Il y a un petit piège ici : z peut a priori être libre dans s ou t, ce qui empêche a priori d'identifier $\lambda z \cdot zst$ à la paire de Church $\langle s, t \rangle$. Ceci se résout en poursuivant le raisonnement ; rappelons que l'on cherche à montrer que u est $\beta\eta$-équivalent à un terme clos.

    Comme F et G sont des types de données (il faut bien utiliser l'hypothèse quelque part !) s et t sont $\beta\eta$-équivalents à des termes clos, disons s' et t'. Donc u est $\beta\eta$-équivalent au terme clos $\lambda z \cdot zs't'$; et ceci est maintenant bien la paire de Church $\langle s', t' \rangle$, puisque z n'est pas libre ni dans s' ni dans t' (qui sont clos !).

    Donc $F \times G$ est un type de données, et ses valeurs sont (les classes d'équivalence) de couples $\langle s', t', \rangle$, où s' est une valeur de F et t' une valeur de G.

Il y a de nombreux autres types de données : les sommes de deux types de données, les listes et les arbres dont les nœuds sont dans un type de données notamment.

4 Types de fonctions

Mais tous les types ne sont pas des types de données, comme on va le voir.

Par $(viii)$, tout entier de Church, qui est de type `nat`, est dans $R_{\texttt{nat}}$. Il y a cependant d'autres termes clos β-normaux dans $R_{\texttt{nat}}$:

13. [0,5 pt.] Montrer que $\lambda x \cdot x$ est dans $R_{\texttt{nat}}$.

    On peut exhiber une preuve de $\vdash \lambda x \cdot x : \forall \alpha \cdot (\alpha \Rightarrow \alpha) \Rightarrow (\alpha \Rightarrow \alpha)$, et utiliser $(viii)$, ou le montrer directement.

    On montre que $\lambda x \cdot x$ est dans $R_{\texttt{nat}}$ en montrant que pour tout ensemble saturé S, pour tout $v \in R_{\alpha \Rightarrow \alpha}^{[\alpha \mapsto S]}$, $(\lambda x \cdot x)v$ est dans $R_{\alpha \Rightarrow \alpha}^{[\alpha \mapsto S]}$. C'est évident car ce dernier ensemble est saturé et $(\lambda x \cdot x)v$ se β-réduit en v.

14. [0,5 pt.] En déduire que tout terme qui se β-réduit en $\lambda x \cdot x$ est dans $R_{\texttt{nat}}$.

    Par la question précédente, et en utilisant que $R_{\texttt{nat}}$ est saturé.

15. [2 pts.] Soit u_0 le terme $\lambda x \cdot x(\lambda y \cdot \ulcorner 0 \urcorner)\ulcorner 0 \urcorner z$, où z est une variable libre. En utilisant la question 11, montrer que pour tout $v \in R_{\texttt{nat}}$, $u_0 v$ se β-réduit en $\lambda x \cdot x$.

   Soit $v \in R_{\texttt{nat}}$. Comme `nat` *est un type de données, vfx se β-réduit en $f^n(x)$ pour un certain $n \in \mathbb{N}$, où f et x sont deux variables.*

   En remplaçant f et x par deux termes quelconques s et t, vst se β-réduit en $s^n(t)$.

   En particulier, $u_0 v$, qui se β-réduit en $v(\lambda y \cdot \ulcorner 0 \urcorner)\ulcorner 0 \urcorner z$, se β-réduit aussi en $(\lambda y \cdot \ulcorner 0 \urcorner)^n(\ulcorner 0 \urcorner)z$.

   Par analyse de cas, que n vaille 0 ou un nombre supérieur ou égal à 1, ceci se β-réduit en $\ulcorner 0 \urcorner z$, donc en $\lambda x \cdot x$.

16. [1 pt.] En déduire que le terme u_0 est dans $R_{\texttt{nat} \Rightarrow \texttt{nat}}$.

   Par la question 15, pour tout $v \in R_{\texttt{nat}}$, $u_0 v$ se β-réduit en le terme $\lambda x \cdot x$. Donc $u_0 v$ est dans $R_{\texttt{nat}}$ par la question 14. Comme v est arbitraire, par définition, u_0 est donc dans $R_{\texttt{nat} \Rightarrow \texttt{nat}}$.

17. [0,5 pt.] Pourquoi `nat` $\Rightarrow$ `nat` n'est-il pas un type de données ?

   Comme u_0 est β-normal et pas clos, il n'a pas de β-réduit clos. On en conclut que `nat` $\Rightarrow$ `nat` *n'est pas un type de données.*