TD3 février 2022

# $\lambda$-calcul et logique informatique

leroux@lsv.fr

**Exercice 1 — Encodages**

1. Définir un codage des couples, c'est à dire des termes **couple**, $\pi_1$ et $\pi_2$ tels que pour tout $N_1$ et $N_2$ on a :

$$\pi_i\ (\textbf{couple}\ N_1\ N_2) \to^* N_i$$

   A-t-on $M =_\beta \textbf{couple}\ (\pi_1\ M)\ (\pi_2\ M)$ pour tout $M$ ?

2. Pour les entiers, on utilise l'encodage de Church :

$$\overline{n} = \lambda f.\lambda x.\ f^n\ x = \lambda f.\lambda x.\ f\ (f\ (\dots\ x))$$

   (a) Définir $f^n x$ de manière rigoureuse. S'il existe plusieurs définitions possibles, dire comment vous prouveriez qu'elles sont équivalentes.

   (b) Rappeler l'encodage de zéro et du successeur.

   (c) Trouver un terme qui encode l'application $n \mapsto n + 2$. Combien y a-t-il de termes naturels pour cela ?

   (d) Rappeler deux encodages de l'addition. Montrer que l'un d'entre eux est correct.

   (e) Rappeler l'encodage de la multiplication vu en cours. En proposer un autre. Montrer qu'ils sont corrects.

3. Définir/rappeler le test à zéro pour les entiers de Church et montrer qu'il est correct.

4. Définir le prédecesseur pour les entiers de Church, avec $\textbf{pred}\ \overline{0} = \overline{0}$.

5. Définir (et justifier le cas échéant)

   (a) un codage des listes. Quel est la liste vide ?

   (b) l'ajout d'un élement en tête d'une liste.

(c) le test du vide.

(d) le calcul du premier élément d'une liste.

(e) la concaténation.

(f) la longueur d'une liste, le renversement.

**Exercice 2 — Chemins sur $\Lambda^*$**

On définit l'ensemble des chemins d'un terme $u \in \Lambda$, noté $\mathcal{C}(u)$, comme l'ensemble des objets $p$ tels que $p \prec u$ est dérivable dans le système suivant :

$$\frac{}{x \prec x} \quad \frac{P \prec M}{\mathbf{L}\ P \prec M\ N} \quad \frac{P \prec N}{\mathbf{R}\ P \prec M\ N} \quad \frac{P \prec M}{\lambda x.P \prec \lambda x.M}$$

On travaille modulo $\alpha$, en renommant implicitement les variables liées pour pouvoir appliquer la règle de passage sous l'abstraction. Par conséquent, deux termes $\alpha$-équivalents ont les mêmes chemins, et l'ensemble des chemins d'un terme est clos par $\alpha$-équivalence.

1. Donner les chemins des termes $\lambda x.\ x\ y$, $\lambda x.\ x\ x$, et $\lambda x.\ (\lambda y.\ y)\ x$.
2. Que peut-on dire de deux termes qui ont les mêmes chemins ?
3. On étend notre définition de chemins à $\Lambda^*$ en rajoutant la règle suivante :
$$\frac{P \prec M \quad Q \prec N}{P[x := Q] \prec (\lambda^* x.\ M)\ N}$$
Montrer que $\mathcal{C}(A[x := B]) = \mathcal{C}(A)[x := \mathcal{C}(B)]$ pour tous termes $A$ et $B$ dans $\Lambda^*$.
4. Montrer que $M \to N$ implique $\mathcal{C}(M) = \mathcal{C}(N)$, où $\to$ est la réduction de $\Lambda^*$ qui ne touche qu'aux rédexes annotés d'une étoile.
5. Montrer la réciproque : $\mathcal{C}(M) = \mathcal{C}(N)$ implique $M \leftrightarrow^* N$, c'est à dire la convertibilité dans $\Lambda^*$.

**Solutions exercice 1**

1. **pair** $:= \lambda xyz.zxy$, $\pi_1 := \lambda xy.x$ et $\pi_2 := \lambda xy.y$.
2. (a) Par récurrence $f^0x := x$ et $f^{n+1}x := f(f^nx)$. Alternative $i(0,f,x) := x$ et $i(n+1,f,x) := i(n,f,fx)$. On pourrait montrer par récurrence sur $n$ que $f^nx = i(n,f,x)$.
   (b) $\overline{0} = \lambda fx.x$. C'est $\pi_2$, ou "faux" d'un TD précédent.
   Par exemple, $\overline{S} := \lambda yfx.f(yfx)$.
   (c) $D_1 := \lambda yfx.f(f(yfx))$ ou $D_2 := \lambda yfx.yf(f(fx))$ ou $D_3 := \lambda yfx.f(yf(fx))$.
   (d) Par exemple $\overline{+}_1 := \lambda yzfx.yf(zfx)$ ou $\overline{+}_2 := \lambda y.y\overline{S}$.
   — On calcule que $\overline{+}_1\overline{n}\,\overline{p} \to^* \lambda fx.f^n(f^px)$ et on prouve par récurrence sur $n$ ou $p$ (en fonction de la définition de $f^nx$) que $f^n(f^px) = f^{n+p}x$.
   — On calcule que $\overline{+}_2\overline{n}\,\overline{p} \to^* \overline{S}^n\overline{p}$. Une récurrence sur $n$ permet de conclure.
   (e) $\overline{\times}_1 := \lambda npf.n(pf)$ ou $\overline{\times}_2 := \lambda npfx.n(\overline{+}p)x$.
   — On calcule que $\overline{\times}_1\overline{np} \to^* \lambda x.\overline{p}^nx$ puis on conclut par une récurrence sur $n$.
   — On calcule que $\overline{\times}_2\overline{np} \to^*$
3. $\overline{isZero} := \lambda x.x(\lambda\_.\overline{\bot})\overline{\top}$. On a $\overline{isZero0} \to^* (\lambda fx.x)\bot\overline{\top} \to^* \overline{\top}$. On note que $(\lambda\_.\overline{\bot})^{n+1}x \to^* \overline{\bot}$.
4. Soit $R$ tel que $R\langle\overline{p},\overline{q}\rangle \to^* \langle\overline{p+1},\overline{p}\rangle$. Par exemple $R := \lambda x.\langle\overline{S}(\pi_1x),\pi_1x\rangle$. On note que $R\langle\overline{0},\overline{0}\rangle \to^* \langle\overline{1},\overline{0}\rangle$, et par récurrence sur $n$, $R^{n+1}\langle\overline{0},\overline{0}\rangle \to^* \langle\overline{n+1},\overline{n}\rangle$. Soit $P := \lambda y.\pi_2(yR\langle\overline{0},\overline{0}\rangle)$. On a donc $P\overline{0} \to^* \overline{0}$ et $P\overline{n+1} \to^* \overline{n}$ pour tout entier $n$.
5. (a) Les listes d'éléments d'un ensemble $A$ sont définies comme suit :
      i. $nil \in L(A)$.
      ii. Pour toute liste $l \in L(A)$ et $a \in A$, one pose $(a :: l) \in L(A)$.

   Soit $f,x$ deux variables. On définit
   — $[nil]^- := x$,
   — $[a :: l]^- := fal$ pour tout Lambda-terme $a$ et liste de Lambda-termes $l$.

   L'encodage d'une liste de Lambda-termes $l$ est défini par $[l] := \lambda fx.[l]^-$.

(b) Soit $[cons] := \lambda alfx.fa(lfx)$. Ainsi $[cons]a[l] \to^* \lambda fx.fa([l]fx) \to^* \lambda fx.fa([l]fx) = [a :: l]$.

(c) Soit $F := \lambda xy.\overline{\bot}$. Soit $[vide] := \lambda l.lF\overline{\top}$. Alors $[vide][nil] \to^* \overline{\top}$ et $[vide][a :: l] \to^* F\_\_ \to^* \overline{\bot}$.

(d) Soit $[hd] := \lambda l.l\pi_1[nil]$. Alors $[hd][nil] \to^* [nil]$ et $[hd][a :: l] \to \pi_1 a\_ \to a$.

(e) On montre que pour toutes listes de Lambda-termes $l$ et $m$, on a $[l][cons][m] \to^* [l + +m]$, où $l + +m$ est la concaténation de $l$ et $m$. On le prouve par récurrence sur $l$.
— Pour toute liste de Lambda-termes $m$, on a $[nil][cons][m] \to^* [m]$.
— Pour tout Lambda-terme $a$ et listes de Lambda-termes $l$ et $m$, on a $[cons]a[l][cons][m] \to^* [cons]a([l][cons][m]) \to^* [cons]a[l + +m] \to^* [(a :: l) + +m]$ par HR, le dernier term étant $\beta$-normal. Or $[cons]a[l][cons][m] \to^* [a :: l][cons][m]$, on peut donc conclure par confluence que $[a :: l][cons][m] \to^* [(a :: l) + +m]$.
Soit $[app] := \lambda lm.l[cons]m$. Par le résultat ci-dessus, on a $[app][l][m] \to^* [l + +m]$.

(f) Indice (mais il y a des solutions plus directes) : on pourra définir une fonction $[tl]$ qui renvoie la queue d'une liste, par une approche similaire à la définition du prédécesseur dans les entiers.

**Solutions exercice 2**

1. $\mathcal{C}(\lambda x.xy) = \{\lambda x.\mathbf{L}x, \lambda x.\mathbf{R}y\}, \mathcal{C}(\lambda x.xx) = \{\lambda x.\mathbf{L}x, \lambda x.\mathbf{R}x\}$ et $\mathcal{C}(\lambda x.(\lambda y.y)x) = \{\lambda x.\mathbf{R}x, \lambda x.\mathbf{L}\lambda y.y\}$
2. On remarque que
   — $\mathcal{C}(x) = \{x\}$, car $u \prec x$ ssi $u = x$.
   — $\mathcal{C}(\lambda x.A) = \lambda x.\mathcal{C}(A)$, car $u \prec \lambda x.A$ ssi $u = \lambda x.B$ où $B \prec A$.
   — $\mathcal{C}(AB) = (\mathbf{L}\ \mathcal{C}(A)) \cup (\mathbf{R}\ \mathcal{C}(B))$, pour des raisons similaires.

   La remarque ci-dessus nous permet de montrer par récurrence sur $A$ que $\mathcal{C}(A) = \mathcal{C}(B)$ implique $A = B$, à $\alpha$-équivalence près :
   — Si $A = x$, alors $\mathcal{C}(A) = \{x\}$, d'où $\mathcal{C}(B) = \{x\}$, donc $B = x$.
   — Si $A = \lambda x.A'$ alors $\mathcal{C}(A) = \lambda x.\mathcal{C}(A')$, d'où $\mathcal{C}(B) = \lambda x.\mathcal{C}(A')$, donc $B = \lambda x.A' = A$.
   — Si $A = A_1A_2$ alors $\mathcal{C}(A) = (\mathbf{L}\ \mathcal{C}(A_1)) \cup (\mathbf{R}\ \mathcal{C}(A_2))$, d'où $\mathcal{C}(B) = (\mathbf{L}\ \mathcal{C}(A_1)) \cup (\mathbf{R}\ \mathcal{C}(A_2))$, donc $B = A_1A_2 = A$.

3. On remarque que
   — $\mathcal{C}(x) = \{x\}$.
   — $\mathcal{C}(\lambda x.A) = \lambda x.\mathcal{C}(A)$.
   — $\mathcal{C}(AB) = (\mathbf{L}\ \mathcal{C}(A)) \cup (\mathbf{R}\ \mathcal{C}(B))$. (Application, i.e. $A$ n'est pas une "$\lambda^*$-abstraction".)
   — $\mathcal{C}((\lambda^* x.A)B) = \mathcal{C}(A)[x := \mathcal{C}(B)]$.

   On remarque également que les variables libres de $A$ sont exactement les variables libres de $\mathcal{C}(A)$, pour une définition naturelle du concept. On omet ici définition et preuve. La notion de substitution peut aussi se généraliser aux chemins, avec des propriétés naturelles à la clef. On omet définition et preuve dans ce cas également.

   Montrons par récurrence sur $A$ que $\mathcal{C}(A[x := B]) = \mathcal{C}(A)[x := \mathcal{C}(B)]$.
   — Cas (de base) : si $A = x$, alors $\mathcal{C}(A[x := B]) = \mathcal{C}(B) = \mathcal{C}(A)[x := \mathcal{C}(B)]$. Si $A = y$, alors $\mathcal{C}(A[x := B]) = \mathcal{C}(A) = \mathcal{C}(A)[x := \mathcal{C}(B)]$.
   — Cas $A = \lambda y.A'$. Alors $\mathcal{C}(A[x := B]) = \mathcal{C}((\lambda y.A')[x := B]) = \mathcal{C}(\lambda y.A'[x := B]) = \lambda y.\mathcal{C}(A'[x := B]) = \lambda y.\mathcal{C}(A')[x := \mathcal{C}(B)]$ par HR. Donc $\mathcal{C}(A[x := B]) = (\lambda y.\mathcal{C}(A'))[x := \mathcal{C}(B)] = \mathcal{C}(A)[x := \mathcal{C}(B)]$.
   — Cas $A = MN$. Alors $\mathcal{C}(A[x := B]) = \mathcal{C}((MN)[x := B]) = \mathcal{C}((M[x := B])(N[x := B])) = \mathbf{L}\mathcal{C}(M[x := B]) \cup \mathbf{R}\mathcal{C}(N[x := B]) = \mathbf{L}(\mathcal{C}(M)[x := \mathcal{C}(B)]) \cup \mathbf{R}(\mathcal{C}(N)[x := \mathcal{C}(B)])$ par HR. Donc $\mathcal{C}(A[x := B]) = (\mathbf{L}\mathcal{C}(M))[x := \mathcal{C}(B)] \cup (\mathbf{R}\mathcal{C}(N))[x := \mathcal{C}(B)] = (\mathbf{L}\mathcal{C}(M) \cup \mathbf{R}\mathcal{C}(N))[x := \mathcal{C}(B)] = \mathcal{C}(MN)[x := \mathcal{C}(B)]$.
   — Cas $A = (\lambda^* y.M)N$. Alors $\mathcal{C}(A[x := B]) = \mathcal{C}((\lambda^* y.M)N[x := B]) = \mathcal{C}((\lambda^* y.M[x := B])(N[x := B])) = \mathcal{C}(M[x := B])[y := \mathcal{C}(N[x := B])]$ d'après une remarque ci-dessus. Donc $\mathcal{C}(A[x := B]) = \mathcal{C}(M)[x := \mathcal{C}(B)][y := \mathcal{C}(N)[x := \mathcal{C}(B)]]$ par HR (deux fois). Ainsi $\mathcal{C}(A[x := B]) = (\mathcal{C}(M)[y := \mathcal{C}(N)])[x := \mathcal{C}(B)] = \mathcal{C}(A)[x := \mathcal{C}(B)]$, par un lemme de substitution, non prouvé ici, similaire à un TD précédent.

4. Par récurrence sur $M \to N$.
   — Cas (de base) $M = (\lambda^* x.A)B$ et $N = A[x := B]$. Ainsi $\mathcal{C}(M) = \mathcal{C}(A)[x := \mathcal{C}(B)]$ par une remarque ci-dessus, et $\mathcal{C}(N) = \mathcal{C}(A[x := B]) = \mathcal{C}(A)[x := \mathcal{C}(B)]$, par la question précédente.
   — Cas $M = \lambda x.A$ et $N = \lambda x.B$, avec $A \to B$. Donc $\mathcal{C}(A) = \mathcal{C}(B)$ par HR, d'où $\mathcal{C}(\lambda x.A) = \lambda x.\mathcal{C}(A) = \lambda x.\mathcal{C}(B) = \mathcal{C}(\lambda x.B)$.
   — Cas $M = AB$ et $N = A'B$, avec $A \to A'$. Donc $\mathcal{C}(A) = \mathcal{C}(A')$ par

HR, d'où $\mathcal{C}(M) = (\mathbf{L}\,\mathcal{C}(A)) \cup (\mathbf{R}\,\mathcal{C}(B)) = (\mathbf{L}\,\mathcal{C}(A')) \cup (\mathbf{R}\,\mathcal{C}(B)) = \mathcal{C}(N)$.

- Cas $M = AB$ et $N = AB'$, avec $B \to B'$. Similaire au cas $M = AB$ et $N = A'B$, avec $A \to A'$ ci-dessus.
- Cas $M = (\lambda^* x.A)B$ et $N = (\lambda^* x.A')B$ avec $A \to A'$. Ainsi $C(M) = C(A)[x := C(B)] = C(A')[x := C(B)] = C(N)$ par HR et une question ci-dessus.
- Cas $M = (\lambda^* x.A)B$ et $N = (\lambda^* x.A)B'$ avec $B \to B'$. Ainsi $C(M) = C(A)[x := C(B)] = C(A)[x := C(B')] = C(N)$ par HR et une question ci-dessus.

5. Soit $M'$ (resp. $N'$) la forme normale (dans le $\Lambda^*$-calcul) de $M$ (resp. $N$). Par la question précédente, on a $\mathcal{C}(M) = \mathcal{C}(M')$ et $\mathcal{C}(N) = \mathcal{C}(N')$, donc $\mathcal{C}(M') = \mathcal{C}(N')$. Les termes $M'$ et $N'$ étant normaux, ce sont des Lambda-termes, donc $\mathcal{C}(M') = \mathcal{C}(N')$ implique $M' = N'$. Ainsi $M \leftrightarrow^* N$.