TD 1 : λ-calcul et codage des types de données

simon.castellan@ens-lyon.fr

Exercice 1 *λ-calcul*

Le λ-calcul est défini par la syntaxe

$$M, N \;:=\; x \quad | \quad \lambda x.\, M \quad | \quad M\, N \qquad\qquad \text{où } x \text{ est une variable}$$

et la règle de réécriture (la β-réduction)

$$(\lambda x.\, M)\, N \to M[N/x]\ .$$

La clôture symétrique réflexive transitive de $\to$ est notée $=_\beta$. Voici quelques termes classiques :

$$I := \lambda x.\, x \qquad K := \lambda x \lambda y.\, x \qquad S := \lambda x \lambda y \lambda z.\, x\, z\, (y\, z) \qquad \Delta := \lambda x.\, x\, x \qquad \Omega := \Delta\, \Delta$$

1. Réduire les λ-termes $\Delta\, I\, I$ et Ω.
2. Donner les graphes de réduction des λ-termes $S\, K\, K$, $\Delta\, (I\, I)$ et $K\, I\, \Omega$.

Exercice 2 *Couples et types somme*

Définissez des λ-termes pour

$\langle \_, \_ \rangle$	le constructeur de couples	ι_1	la première injection
π_1	la première projection	ι_2	la seconde injection
π_2	la seconde projection	case	le filtrage

tels que : $\pi_1\, \langle x, y \rangle =_\beta x$, $\pi_2\, \langle x, y \rangle =_\beta y$, $\text{case}\, (\iota_1\, x)\, f\, g =_\beta f\, x$, $\text{case}\, (\iota_2\, x)\, f\, g =_\beta g\, x$.

Exercice 3 *Entiers et booléens de Church*

La représentation de Church d'un entier n est le λ-terme $\bar{n} := \lambda f x.\, f^n x$, c'est à dire n itérations de la fonction f en x.

1. Écrire $\bar{0}$ et $\bar{3}$.
2. Écrire une fonction successeur : $\mathrm{S}\, \bar{n} =_\beta \overline{n+1}$.
3. Écrire un itérateur, c'est-à-dire un terme Iter tel que pour tous termes M, N, on ait

$$\text{Iter}\ \ M\ N\ \bar{0} =_\beta M \qquad \text{et} \qquad \text{Iter}\ \ M\ N\ (\mathrm{S}\ \ \bar{n}) =_\beta N\ (\text{Iter}\ \ M\ N\ \bar{n})\ .$$

4. Écrire des λ-termes représentant l'addition et la multiplication.
5. Quelle fonction représente le terme $\bar{n}\, \bar{m}$?

On représente les booléens par $\mathrm{T} := \lambda x y.x$ et $\mathrm{F} := \lambda x y.y$.

6. Donner une représentation de `if then else`.

7. Comment représenter les couples ?
8. En utilisant la représentation des couples vue à l'exercice 2, proposer un λ-terme représentant le prédécesseur.

Exercice 4 *λ-calcul (bis)*

1. Caractériser les λ-termes en forme β-normale.
2. Restreindre la β-réduction pour implémenter l'appel par nom et l'appel par valeur. Trouver un λ-terme qui distingue ces deux stratégies de réduction.

Exercice 5 *Entiers de Barendregt*

Les entiers de Barendregt $\ulcorner n \urcorner$ sont définis par

$$\ulcorner 0 \urcorner := I \qquad \ulcorner n+1 \urcorner := \lambda k.k \ \ \mathrm{F} \ \ulcorner n \urcorner$$

1. Proposer une implémentation du successeur, du prédécesseur et de la conditionnelle.
2. Proposer une implémentation de l'addition.

Exercice 6 *Listes et arbres*

On s'intéresse ici à l'encodage des listes en λ-calcul. On va écrire les listes sous la forme $\lambda c.\lambda n.M[c,n]$. Moralement, une liste est une fonction qui attend un constructeur, une liste vide et qui les manipule. Par exemple, la liste `["Salade"; "Tomate"; "Oignon"]` sera représentée par

$$\lambda c.\lambda n.(c \ \texttt{"Salade"} \ (c \ \texttt{"Tomate"} \ (c \ \texttt{"Oignon"} \ n))) \ .$$

1. Écrire les opérateurs nil et cons.
2. Écrire un *itérateur* fold tel que

$$\mathrm{fold} \ f \ u \ \mathrm{nil} =_\beta u \qquad \text{et} \qquad \mathrm{fold} \ f \ u \ (\mathrm{cons} \ a \ l) =_\beta f \ a \ (\mathrm{fold} \ f \ u \ l) \ .$$

3. Écrire la concaténation et le miroir.
4. Proposer un encodage pour les arbres binaires.