Les fonctions récursives primitives sont Lambda-définissables

Julien Devevey

2018-2019

Ref : Lassaigne, Rougemont - Logique et fondements de l'informatique p.193
Odifreddi - Classical Recursion Theory p.84

Définition 1. (Entiers de Church) On pose, et on appelle entier de Church, le λ-terme $\underline{k} = \lambda f x.f^k x$, f^k étant f concaténée k fois.

Définition 2. Soit f une fonction récursive primitive de $\mathbb{N}^n$ dans $\mathbb{N}$. Le terme F du λ-calcul représente la fonction f si, pour tout entier $k_1, \dots, k_n$: si $f(k_1, \dots, k_n) = k$ alors $F\underline{k_1} \dots \underline{k_n}$ est β-équivalent à $\underline{k}$. Une fonction f est alors dite λ-définissable , ou représentable, s'il existe un terme du λ-calcul qui la représente.

Remarque 3. Toute fonction λ-définissable est récursive, car si $f(k_1, \dots, k_n)$ est définie, il suffit de calculer la forme normale du terme qui la représente pour trouver le résultat de l'évaluation. Or on peut faire cette réduction par une réduction gauche, ce qui est effectif.

Théorème 4. Toute fonction récursive primitive de $\mathbb{N}^n$ dans $\mathbb{N}$ est λ-définissable .

Démonstration.

Pour démontrer ce théorème, il suffit de montrer, par exemple, que la classe des fonctions λ-définissables contient les fonctions de base et est close par composition et schéma de récurrence.

Commençons par les fonctions élémentaires :

— On représente la fonction constante 0 par le terme $\lambda x.\underline{0}$.

— La projection π_i^n est représentée par $\lambda x_1 \dots x_n.x_i$.

— La fonction successeur est représentée par le terme $suc = \lambda z f x.f(zfx)$.

En effet, on vérifie aisément que $suc\underline{n} >_\beta \underline{n+1}, \forall n \in \mathbb{N}$.

Pour la composition, soient $h, (g_i)_{i\in[\![1,m]\!]}$ récursives primitives et λ-définissables, qu'on représente par $H, (G_i)_{i\in[\![1,m]\!]}$. Alors, en posant $f(x_1, \dots, x_n) = h(g_1(x_1, \dots, x_n), \dots, g_m(x_1, \dots, x_n))$, on peut représenter f par le λ-terme :

$$F = \lambda x_1 \dots x_n.H(G_1 x_1 \dots x_n) \dots (G_m x_1 \dots x_n)$$

En effet, soient $(x_1, \ldots, x_n) \in \mathbb{N}^n$, alors on a :

$$\begin{aligned} Fx_1 \ldots x_n &=_\beta H\underline{g_1(x_1, \ldots, x_n)} \ldots \underline{g_m(x_1, \ldots, x_n)} && \text{par hypothèse sur les } G_i \\ &=_\beta \underline{h(g_1(x1, \ldots, x_n), \ldots, g_m(x_1, \ldots, x_n))} = \underline{f(x_1, \ldots, x_n)} \\ & && \text{par hypothèse sur } H \end{aligned}$$

Pour la récurrence, on a besoin de définir un opérateur de récursion dans le λ-calcul . On peut le faire en utilisant un combinateur de point fixe : c'est un terme clos M tel que pour tout terme F, on ait $MF =_\beta F(MF)$. Le terme $Y = \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$ convient : $YF =_\beta (\lambda x.F(xx))(\lambda x.F(xx)) =_\beta F((\lambda x.F(xx))(\lambda x.F(xx))) = F(YF)$.

On montre un résultat intéressant pour plus tard, qui est que, si Z est un λ-terme dont les variables libres sont $x, y_1, \ldots, y_n$, alors on pose $X = Y(\lambda x y_1 \ldots y_n.Z)$ et

$$\begin{aligned} Xy_1 \ldots y_n &=_\beta ((\lambda x y_1 \ldots y_n.Z)X)y_1 \ldots y_n && \text{Y combinateur de point fixe} \\ &=_\beta (\lambda y_1 \ldots y_n.Z[X/x])y_1 \ldots y_n \end{aligned}$$

On admet ensuite le lemme suivant :

Lemme 5. La fonction prédécesseur, définie par $p(0) = 0$ et $p(k+1) = k$ pour $k \in \mathbb{N}$ est λ-définissable. On la représentera par le λ-terme P.

On prend maintenant g primitive récursive de $\mathbb{N}^n$ dans $\mathbb{N}$, h une autre fonction récursive primitive de $\mathbb{N}^{n+2}$ dans $\mathbb{N}$ toutes deux λ-définissables, on les représente par les λ-termes G et H respectivement. On pose alors f la fonction définie par schéma de récurrence sur g et h.

On cherche un opérateur de récursion R tel que :

$$\begin{aligned} Ruv\underline{0} &=_\beta u \\ Ruv\underline{k+1} &=_\beta v\underline{k}(Ruv\underline{k}) \end{aligned}$$

S'il existe un tel terme, on peut représenter f par $F = \lambda \overrightarrow{x} y.R(G\overrightarrow{x})(\lambda ab.H\overrightarrow{x}ab)y$, avec $\overrightarrow{x} = x_1 \ldots x_n$. En effet, on peut vérifier que :

$$\begin{aligned} F\overrightarrow{x}\underline{0} &=_\beta G\overrightarrow{x} \\ &=_\beta \underline{g(\overrightarrow{x})} && \text{Par hypothèse sur G} \\ F\overrightarrow{x}\underline{k+1} &=_\beta (\lambda ab.H\overrightarrow{x}ab)\underline{k}(R(G\overrightarrow{x})(\lambda ab.H\overrightarrow{x}ab)\underline{k}) \\ &=_\beta (\lambda ab.H\overrightarrow{x}ab)\underline{k}(F\overrightarrow{x}\underline{k}) \\ &=_\beta H\overrightarrow{x}\underline{k}(F\overrightarrow{x}\underline{k}) \\ &=_\beta \underline{h(\overrightarrow{x}, k, f(\overrightarrow{x}, k))} && \text{Par hypothèse sur H} \end{aligned}$$

Il faut remarquer à un moment que $F\overrightarrow{x}\underline{k} =_\beta R(G\overrightarrow{x})(\lambda ab.H\overrightarrow{x}ab)\underline{k}$.

Il reste alors à construire le terme R. On passe par le λ-terme intermédiaire $U =$

$\lambda xyz.z(\lambda d.y)x$, qui va représenter la définition au cas par cas :

$$\begin{aligned} Uuv\underline{0} &=_\beta \underline{0}(\lambda d.v)u \\ &=_\beta u \\ Uuv\underline{k+1} &=_\beta \underline{k+1}(\lambda d.v)u \\ &=_\beta (\lambda d.v)^{k+1}u \\ &=_\beta v \end{aligned}$$

A partir de U et de P, on voit alors qu'il suffit que R vérifie : $Ruv\underline{k} =_\beta Uu(v(P\underline{k})(Ruv(P\underline{k})))\underline{k}$. Finalement, le combinateur de point fixe nous donne la solution $R = Y(\lambda cxyz.Ux(y(Pz)(cxy(Pz))z)$, ce qui conclut la preuve. En effet, vu ce qui précède le lemme 5, on a que $Ruv\underline{k} =_\beta$ $Uu(v(P\underline{k})(Ruv(P\underline{k}))\underline{k}$. □