Fonctions calculables et récursives

Léo Gayral

2017-2018

ref : Wolper – Introduction à la calculabilité – p.135

Définition 1 (Fonction calculable)**.** On considère par la suite des machines de Turing déterministes, à un seul ruban semi-infini, sur l'alphabet $\Sigma \supset \{0,1\}$. On considère le codage binaire $\begin{array}{ccc} \mathbb{N} & \to & \{0,1\}^* \\ n & \mapsto & \overline{n}^2 \end{array}$.

Soit $f : I \to \mathbb{N}$ une fonction partielle sur $\mathbb{N}$. On dit que f est calculable si il existe une telle machine qui, sur toute entrée $\langle n \rangle$ $(n \in I)$, termine avec $\langle f(n) \rangle$ sur la gauche du ruban.

Par induction, on voit assez simplement que toute fonction récursive est calculable.

Théorème 1. Toute fonction calculable est récursive.

Démonstration.
Soit f calculable, et $M = (Q, \Sigma, \Gamma, q_0, F, \delta)$ une machine de Turing associée. On peut sans pertes de généralité supposer qu'il existe un unique état acceptant q_1. Par commodité, lorsque la machine atteint l'état acceptant, elle boucle sur q_1, de sorte que son exécution n'est jamais *bloquée* du point de vue des transitions.

Le point essentiel de la démonstration est de comprendre comment représenter et manipuler le ruban de la machine, du point de vue des fonctions récursives primitives.

On identifie Γ au segment $[\![0, |\Gamma| - 1]\!]$, en envoyant spécifiquement $\llcorner\!\lrcorner$ sur 0. La décomposition en base $|\Gamma|$ induit alors assez naturellement un codage bijectif $\sigma : \Gamma^* \llcorner\!\lrcorner^\infty \to \mathbb{N}$ sur les rubans de la machine.

On indexe les cases du ruban sur $\mathbb{N}$. L'opération de shift du ruban w d'une case vers la gauche correspond au calcul du quotient de $\sigma(w)$ dans la division euclidienne par $|\Gamma|$, qui est une fonction récursive primitive. Pour *lire* la valeur de la case n, on définit par récurrence $LIRE(0, m) = RESTE(m)$ et $LIRE(n+1, m) = LIRE(n, QUOTIENT(m))$.

On peut alors *effacer* la valeur inscrite sur cette case via :

$$EFFACER(n, m) = SOUSTRAIRE\left(m, |\Gamma|^n \times LIRE(n, m)\right) .$$

On peut donc *écrire* le symbole $x \in \Gamma$ codé par $k = \sigma(x)$ sur une case via :

$$ECRIRE(n, m, k) = EFFACER(n, m) + |\Gamma|^n \times k \, .$$

On identifie également Q au segment $[\![0, |Q| - 1]\!]$, avec 0 l'état initial et 1 l'état acceptant.

Ce travail préliminaire étant fait, on peut simuler un pas d'exécution de la machine. On représente alors un état de la machine par un triplet (n, m, q) où $m = \sigma(w)$ code le ruban, n est la position du curseur, et q l'état de la machine. On peut alors définir $NEXT(n, m, q)$ par une disjonction de cas finie, via δ :

$$(n+\epsilon, ECRIRE(n, m, k), q') \text{ avec } \delta(LIRE(n, m), q) = (k, q', \epsilon \in \{-1, 0, 1\}) \, .$$

On peut donc naturellement simuler n pas d'exécution sur l'entrée w via $EXE(0, m) = (0, m, 0)$ et $EXE(n+1, m) = NEXT(EXE(n, m))$. Quitte à utiliser un μ-schéma, on peut déterminer le plus petit rang n tel que $q = 1$, puis renvoyer la configuration du ruban à cette étape via $FINAL(m)$.

Notons enfin que la conversion $U : n \mapsto \sigma\left(\overline{n}^2\right)$ est une fonction récursive primitive, et qu'on peut construire un inverse à gauche V. On a enfin $f(n) = V \circ FINAL \circ U(n)$, récursive avec un seul μ-schéma. $\square$