Turing calculable implique μ-récursive

Antoine Dequay

21 septembre 2022

Notes

— Prof : Lilian Besson.

— Leçon : 912, 913.

— Références :

  — Barbenchon.

Théorème 1 Toute fonction TURING calculable est μ-récursive.

Preuve. On impose que :

- Une fonction est dite calculable pas une machine de TURING M si pour $n \in \mathbb{N}$ écrit en unaire sur son ruban d'entrée, M termine et le ruban de sortie affiche $f(n)$ (en unaire).
- M est déterministe à ruban bi-infini, qu'elle possède exactement un état initial, un état final et qu'elle boucle sur son état final.
- Avec $M = (Q, \Sigma, \Gamma, E, q_0, F, \sharp)$, $Q = [\![0, m]\!]$, $q_0 = 0$, $F = \{m\}$, $\Sigma = \{0, 1\}$, $\Gamma = [\![0, k-1]\!]$, $\sharp = 0$ et on note $\delta : [\![0, m]\!] \times [\![0, k-1]\!] \to [\![0, m]\!] \times [\![0, k-1]\!] \times \{0, 1\}$ la fonction de transition.

On représente la configuration :

							$\downarrow$						
$\cdots$	0	0	g_j	$\cdots$	g_1	g_0	d_0	d_1	$\cdots$	d_i	0	0	$\cdots$

par le triplet (g, q, d) avec q l'état et g, d les mots (en base k) respectivement à gauche et droite du pointeur, avec comme bit de poids faible celui étant le plus proche du pointeur :

$$g = \sum_{p=0}^{j} g_p k^p \text{ et } d = \sum_{p=0}^{i} d_p k^p$$

Soit f une fonction calculable et M une machine de TURING associée. Montrons qu'on peut décrire son fonctionnement par une fonction μ-récursive.

Pour cela, on doit trouver :

- Une fonction `init` $:\mathbb{N} \to \mathbb{N}^3$ permettant d'écrire la configuration initiale de M sur l'entrée n,
- Une fonction `config_suivante` $:\mathbb{N}^3 \to \mathbb{N}^3$ qui permet de passer d'une configuration à la suivante en respectant δ,
- Une fonction `config` $:\mathbb{N}^3 \times \mathbb{N} \to \mathbb{N}$ qui simule n transitions successives à partir d'une configuration donnée,
- Une fonction `temps_arret` $:\mathbb{N}^3 \to \mathbb{N}$ qui donne le nombre d'étape pour arriver à un état final de la machine à partir d'une configuration donnée,
- Une fonction `trad` $:\mathbb{N} \to \mathbb{N}$ permettant de traduire la réponse du ruban de sortie (un nombre en unaire en base k) en un entier.

On a :

- `init` $: \begin{pmatrix} \mathbb{N} & \longrightarrow & \mathbb{N}^3 \\ n & \longmapsto & \left(0, 0, \sum_{p=0}^{n-1} k^p\right) \end{pmatrix}$, qui est bien primitive récursive, car l'exponentiation et la somme le sont,

— `config_suivante` : $\begin{pmatrix} \mathbb{N}^3 & \longrightarrow & \mathbb{N}^3 \\ (g,q,d) & \longmapsto & \begin{cases} (gk+\alpha, q', \lfloor d/k \rfloor) \text{ si } dir = 1, \\ (\lfloor g/k \rfloor, q', g_0 + k(\alpha + \lfloor d/k \rfloor k)) \text{ si } dir = 0. \end{cases} \end{pmatrix}$,
pour $\delta(q, d_0) = (q', \alpha, dir)$ qui est bien μ-récursive, car :

- $g_0 \equiv g \ (mod\ k)$, $d_0 \equiv d \ (mod\ k)$,
- δ est primitive récursive car définie sur un ensemble fini, donc somme finie d'indicatrices,
- $\lfloor \cdot / \cdot \rfloor$ correspond au quotient de la division euclidienne,
- Le reste, le quotient de la division euclidienne, l'addition, la multiplication et la conditionnelle pour la disjonction de cas sont bien primitives récursives.

— `config` : $\begin{pmatrix} \mathbb{N}^3 \times \mathbb{N} & \longrightarrow & \mathbb{N}^3 \\ (\overrightarrow{x}, k) & \longmapsto & \begin{cases} \overrightarrow{x} \text{ si } k = 0, \\ \texttt{config\_suivante}(\texttt{config}(\overrightarrow{x}, k-1)) \text{ sinon.} \end{cases} \end{pmatrix}$, qui est bien primitive récursive car défini par disjonction de cas et récursion primitive à partir de `config_suivante`,

— `temps_arret` : $\begin{pmatrix} \mathbb{N}^3 & \longrightarrow & \mathbb{N} \\ \overrightarrow{x} & \longmapsto & \mu_i \left(\pi_2^3(\texttt{config}(\overrightarrow{x}, i)) = m \right) \end{pmatrix}$, qui est bien μ-récursive car définie par minimisation non-bornée d'un prédicat primitif récursif,

— `trad` : $\begin{pmatrix} \mathbb{N} & \longrightarrow & \mathbb{N} \\ p & \longmapsto & \mu_{i \leq p}(k^i > p) \end{pmatrix}$, qui est primitive récursive par minimisation bornée,

Ainsi,

$$\begin{aligned} f(n) = \texttt{trad} \left(\pi_1^3 \left(\texttt{config} \left(\texttt{init}(n), \texttt{temps\_arret} \left(\texttt{init}(n) \right) \right) \right) \right) \\ + \texttt{trad} \left(\pi_3^3 \left(\texttt{config} \left(\texttt{init}(n), \texttt{temps\_arret} \left(\texttt{init}(n) \right) \right) \right) \right) \end{aligned}$$

D'où le résultat !

□