Fondements de la programmation : TD 4
Fonctions (primitives) récursives (correction)

Paulin de Naurois et Virgile Mogbil et Lê Thành Dũng Nguyễn
Institut Galilée – Master 1 Informatique 10/10/2018

Exercice 1. Montrer que les fonctions suivantes sont primitives récursives :

(j) $\mathsf{mod}_2 : \ x \in \mathbb{N} \mapsto [x]_2 \in \mathbb{N}$ *(modulo 2, i.e. reste de la division par 2)*

Correction. On corrige ici la fonction mod_2 qui n'a pas été traitée en séance de TD.
Remarquons que si n est un entier strictement positif, n est impair si $n-1$ est pair et n est pair si $n-1$ est impair. D'où la définition récursive suivante, où neg est une négation booléenne :

$$\mathsf{mod}_2(0) = 0 \qquad \mathsf{mod}_2(s(x)) = \mathsf{neg}(\mathsf{mod}_2(x))$$

On vérifie qu'il s'agit bien d'une récursion *primitive*, dans le format autorisé par la définition des fonctions primitives récursives :

$$\mathsf{mod}_2(s(x)) = h(x, \mathsf{mod}_2(x)) \qquad \text{où} \qquad h(a,b) = b \quad \text{c'est-à-dire} \quad h = \pi_2^2$$

Il ne reste donc plus qu'à implémenter neg sous forme d'une fonction primitive récursive. Pour cela on va utiliser une définition "récursive" primitive, qui n'utilise pas son appel récursif.

$$\mathsf{neg}(0) = 1 \qquad \mathsf{neg}(s(x)) = 0 = h'(x, s(x)) \quad \text{où} \quad h' = \underline{0}$$

neg se comporte ainsi comme l'opérateur `!` de nombreux langages de programmation sur les entiers : on a `!x == (x != 0)`.

Exercice 3. Prouver que si $f : \mathbb{N} \to \mathbb{N}$ et $g : \mathbb{N} \to \mathbb{N}$ sont primitives récursives alors la fonction $f + g : \mathbb{N} \to \mathbb{N}$, définie par $(f+g)(x) := f(x) + g(x)$, est primitive récursive.

Correction. Il suffit d'utiliser la composition. On a vu plus haut que add est primitive récursive. Par conséquent $x \mapsto \mathsf{add}(f(x), g(x))$ est primitive récursive.