Institut Galilée — Calculabilité
Année 2007-2008 — L3 INFO 1

TD 2

# Fonctions récursives primitives et récursives partielles

**Exercice 1.**
*Montrer que les fonctions suivantes sont récursives primites.*

1. *la fonction* zero $\begin{cases} \mathbb{N} \to \mathbb{N} \\ x \mapsto 0 \end{cases}$ ;
2. *les fonctions constantes égales à* 3 *d'arité* 0, 1 *et* 2 ;
3. *la fonction addition :* $(x, y) \mapsto x + y$ ;
4. *la fonction multiplication :* $(x, y) \mapsto x \times y$ ;
5. *la fonction exposant* $(x, y) \mapsto x^y = x \uparrow y$ ;

**Exercice 2.**
*Montrer que la fonction de division* $d : (x, y) \mapsto \lfloor x/y \rfloor$ *(arrondi par partie entière inférieure) est récursive. Pouvez-vous la rendre récursive primitive ?*

# Corrigé

## Correction de l'exercice 1.

1. On imagine bien que l'on a besoin d'utiliser la fonction constante $Z$ égale à 0 d'arité 0. Mais la réponse reste difficile. Cela tient au fait que les projections ne servent à rien pour passer de l'arité 1 à l'arité 0. On est obligé d'utiliser RecPri qui permet en particulier de mélanger une fonction d'arité 0 avec une fonction d'arité 2 pour obtenir une fonction d'arité 1.

   On pose zero = RecPri$(f, g)$. On résout alors en $f$ (d'arité 0) et $g$ (d'arité 2), le système obtenu à partir de la définition de RecPri :

   $$\begin{cases} \text{zero}(0) = f() \\ \text{zero}(n+1) = g(n, \text{zero}(n)) \end{cases}$$

   Comme zero doit être partout nulle, on en déduit que $f = Z$ et $g = p_2^2$ conviennent. Ainsi : zero = RecPri$(Z, p_2^2)$ ;
2. Une fois la fonction zero (d'arité 1) définie, ça va tout seul. La fonction trois d'arité 1 est simplement $\text{Comp}(\sigma, \text{Comp}(\sigma, \text{Comp}(\sigma, \text{zero})))$. À l'arité 0 il suffit d'utiliser $Z$ au lieu de zero dans l'expression précédente. Pour passer à l'arité 2 il suffit de précomposer avec une projection : $\text{Comp}(\sigma, \text{Comp}(\sigma, \text{Comp}(\sigma, \text{Comp}(\text{zero}, p_1^2))))$.
3. L'addition se définit récursivement en utilisisant le successeur $\sigma$. Notons $a$ la fonction telle que $a(x, y) = x + y$. On pose $a = \text{RecPri}(f, g)$ où $f$ d'arité 1 et $g$ d'arité 3 sont à trouver. On obtient par définition de RecPri le système :

   $$\begin{cases} a(x, 0) = f(x) \\ a(x, n+1) = g(x, n, h(x, n)) \end{cases}$$

   comme $a(x, 0)$ doit être égal à $x + 0 = x$ on prend $f = p_1^1$ (l'identité). Comme $a(x, n+1)$ doit être égal à $x + n + 1$ et que $a(x, n)$ devrait être égal à $x + n$ il suffit de prendre $g$ telle que $g(x, y, z) = \sigma(z)$. On pose donc $g = \text{Comp}(\sigma, p_3^3)$. Finalement, en posant $a = \text{RecPri}(p_1^1, \text{Comp}(\sigma, p_3^3))$ on peut montrer aisément (par ce qui prècède), par récurrence sur $y$, que $a(x, y) = x + y$.
4. Pour la mulitiplication on raisonne comme précedemment et on trouve que la multiplication peut être définie par la fonction $m = \text{RecPri}(\text{zero}, \text{Comp}(a, p_1^3, p_3^3))$.
5. De même pour l'exponentiation on trouve $e = \text{RecPri}(\text{Comp}(\sigma, \text{zero}), \text{Comp}(m, p_1^3, p_3^3))$.

## Correction de l'exercice 2.

L'idée est de simuler une boucle qui va parcourir les entiers jusqu'à trouver le plus grand $z$ tel que $z \times y \leq x$. Pour cela on va définir le prédicat récursif (primitif) $(z+1) \times y > x$, noté $P(x, y, z)$, puis utiliser le schéma $\mu$ non bornée pour définir $d$.

On doit donc définir une fonction récursive primitive $\chi_P$ telle que $\chi_P(x, y, z) = 1$ si $(z+1) \times y > x$ et sinon $\chi_P(x, y, z) = 0$. On a vu précedemment comment définir la différence tronquée $x \dot{-} y$ égale à zero lorsque $x \leq y$ et à $x - y$ (nécessairement supérieur à 0) sinon.

Comme on sait également que la multiplication et le successeur sont récursives primitives, on peut facilement calculer la différence tronquée

$$(z+1) \times y \dot{-} x$$

qui ne vaudra 0 que dans les cas où $(z+1) \times y \leq x$. Il est alors facile (on peut utiliser un *if*, vue précedemment) de définir $\chi_P$ comme fonction récursive primitive.

On pose alors $d = \mu P$, autrement dit : $d(x, y) = \mu_t P(x, y, t)$. La fonction obtenue est partielle car lorsque $y = 0$ elle diverge (et uniquement dans ce cas). Cela correspond à une division par zéro.

Si l'on voulait une fonction récursive primitive il suffirait de borner $t$ par $x$ dans la boucle pour n'utiliser que le schéma $\mu$ borné et de traiter à part le cas où $y$ vaut zéro. Pour ce cas à part il faut décider ce qu'on veut renvoyer comme valeur. La valeur 0 semble le suel choix possible. Ainsi notre définition récursive primitive peut être :

$$\begin{cases} d(x, 0) = 0 \\ d(x, y+1) = \mu_{t \leq x} P(x, y+1, t) \end{cases}$$

Finalement, $d = \text{RecPri}(\text{zero}, g)$ où $g(x, y, z) = \mu_{t \leq x} P(x, y, t)$ ($z$ inutilisé).