Calculabilité et Logique

TD6 – Fonctions récursives

22 octobre 2008

Exercice 1 – *Existence de fonctions calculables non récursives primitives*

1. Démontrer que l'ensemble des fonctions recursive primitive est dénombrable de façon effective.

   On note $f_0, f_1, \ldots$ l'énumération des fonctions récursives primitives construite et on note $g_i(n) = f_i(n, \ldots, n)$. On définit alors $h(n) = g_n(n) + 1$.
2. Montrer que h n'est pas récursive primitive.
3. Montrer que h est néanmoins calculable.
4. Pourquoi ne peut-on pas appliquer la même technique aux fonctions récursives totales ?

Exercice 2 – *Récursion primitive comme itération*

Définissons l'ensemble des fonctions de codage ainsi :
- pour tout $n \geq 2$, une injection récursive primitive $<\cdot, \ldots, \cdot>_n$ codant les tuples de $\mathbb{N}^n$ sur $\mathbb{N}$,
- pour tout $n \geq 2$, et $1 \leqslant k \leqslant n$, une fonction récursive primitive π_k^n décodant un tuple de $\mathbb{N}^n$ codé sur $\mathbb{N}$ et retournant sa $k^{\text{ème}}$ composante.

Par ailleurs, étant donnée une fonction $h : \mathbb{N} \to \mathbb{N}$, définissons la fonction f_h suivante :

$$f_h(n, x) = h^{(n)}(x) = h(h(\ldots h(x) \ldots))$$

$f_h(n, x)$ est l'itérée $n^{\text{ème}}$ de h appliquée à x.

Montrer que l'ensemble $\mathcal{I}$ défini comme le plus petit ensemble de fonctions contenant les fonctions initiales (successeur, zéro, projection sur une composante), ainsi que les fonctions de codage, et qui de plus est clos par composition et itération, est exactement la classe des fonctions primitives récursives.

Indication : Si f est définie par récursion primitive, définir $s(m, \vec{n}) = <f(m, \vec{n}), m, \vec{n}>$, et montrer que s peut être définie comme une itération.

Exercice 3 – *La fonction d'Ackermann*

Se rappeler qu'il existe une fonction récursive primitive C qui code chaque suite d'entiers $\sigma = \sigma_1\sigma_2 \cdots \sigma_n$ (de taille non fixée) comme un entier $C(\sigma) \geqslant n$. Se rappeler également que $C(\sigma) \geqslant \sigma_i$ pour tout $i \leqslant n$ et qu'on peut définir :

– une fonction de décodage $\pi : \mathbb{N}^2 \to \mathbb{N}$, récursive primitive, telle que :

$$\pi(i, C(\sigma)) = \begin{cases} \sigma_i + 1 \text{ si } |\sigma| \geqslant i \\ 0 \text{ sinon} \end{cases}$$

– une relation primitive récursive *suite*, définie comme : $suite(v)$ est vrai ssi v est le codage d'une suite d'entiers.

On dira que la suite σ *contient* l'entier x si il existe $i \leqslant |\sigma|$ tel que $x = \sigma_i$.

1. Soit la relation in telle que $in(x, C(\sigma))$ est vrai ssi σ contient x. Démontrer que in est récursive primitive.

2. On dit que la suite d'entiers σ est *adéquate* quand les conditions suivantes sont vérifiées (où $< x, y, z >= 2^x 3^y 5^z$) :

a) si σ contient $< 0, m, z >$, alors $z = m + 1$;
b) si σ contient $< n + 1, 0, z >$, alors σ contient $< n, 1, z >$;
c) si σ contient $< n + 1, m + 1, z >$, alors il existe un entier u tel que σ contient $< n + 1, m, u >$ et σ contient $< n, u, z >$;

On dit que σ est *adéquate pour* (n, m) si σ est adéquate et il existe z tel que σ contient $< n, m, z >$.

Montrer que la relation Ad définie comme suit est récursive primitive :

$$Ad(v, n, m) \text{ est vrai ssi il existe une suite } \sigma \text{ adéquate pour } (n, m) \text{ telle que } v = C(\sigma)$$

3. Si σ est une suite *adéquate pour* (n, m), que peut on dire des entiers z tels que σ contient $< n, m, z >$?

4. Montrer que la fonction $f(n, m) = min_v \; Ad(v, n, m)$ est récursive totale.

5. Montrer qu'on peut exprimer la fonction de Ackermann A comme $A(n, m) = min_z \; R(n, m, z)$, où R est à déterminer. Conclure que $A(n, m)$ est récursive totale.