Calculabilité et Logique

TD7 – Fonctions récursives, révisions

5 novembre 2008

Exercice 1 – *Quelques fonctions*

On note $\mathcal{C}$ le plus petit ensemble de prédicats et fonctions partielles sur les entiers qui contient les fonctions initiales, l'addition, la multiplication, l'égalité, et qui est clos par composition et minimisation (la minimisation est naturellement étendue aux prédicats : $f(\vec{x}) = min_z\{P(\vec{x}, z) \wedge \forall k < z\ P(\vec{x}, z)$ est défini$\}$).

1. Montrer *directement* que les fonctions ou prédicats suivants sont dans $\mathcal{C}$:

1. conjonction, disjonction, négation de prédicats dans $\mathcal{C}$;
2. si $P, f_1, f_2 \in \mathcal{C}$ sont totales, $f(n) =$ if $P(n)$ then $f_1(n)$ else $f_2(n)$;
3. dist$(x, y) = |x - y|$;
4. inégalité $x \geqslant y$;
5. soustraction entière : $moins(x, y) = \max(0, x - y)$;
6. $mod(n, m)$ qui calcule le reste de la division de n par m lorsque $m \neq 0$ (indéfini sinon) ;
7. $\exists x \leqslant f(\vec{y}).P(x, \vec{y})$ et $\forall x \leqslant f(\vec{y}).P(x, \vec{y})$ où $P, f \in \mathcal{C}$ sont totales ;
8. $D(n, m)$ qui est satisfait si $m \neq 0$ et m divise n (faux sinon) ;
9. *Prime*(n) qui est satisfait par les nombres premiers ;
10. *PP*(n), vrai ssi n est une puissance d'un nombre premier.

2. Supposer qu'il existe une fonction totale $\beta \in \mathcal{C}$ ayant la propriété suivante : pour toute suite d'entiers $a_0, \ldots, a_n$, il existe d tel que $\forall i = 0..n\ \beta(i, d) = a_i$. Montrer que l'ensemble des fonctions de $\mathcal{C}$ coïncide avec l'ensemble des fonctions récursives partielles.

(Suggestion : montrer avant que les fonctions récursives primitives sont dans $\mathcal{C}$, puis utiliser le théorème de Kleene)

Exercice 2 – *Un problème*

Le problème suivant est-il décidable ?

Donnée : $<M>$, le code d'une machine de Turing calculant une fonction *récursive primitive* $f : \mathbb{N} \to \mathbb{N}$.

Question : f s'annule-t-elle sur au moins une donnée ?

Exercice 3 – *Grammaires algébriques*

Une *grammaire* est un quadruplet $G = (A, V, S, R)$ où
– A est un alphabet terminal ;
– V est un ensemble fini de non terminaux ou variables ;
– $S \in V$ est le symbole initial ;
– R est un sous-ensemble fini de $B^*VB^* \times B^*$ avec $B = A \cup V$.
Le *langage engendré* par G est l'ensemble des mots de A^* que l'on peut atteindre en partant du symbole initial S et en utilisant des règles de réécriture de R.

Une grammaire $G = (A, V, S, R)$ est *algébrique* si toutes ses règles de production de R sont de la forme : $X \to \alpha$ où $X \in V$ et $\alpha \in (A \cup V)^*$.

Supposons que l'alphabet A comprend au moins trois lettres.

1. Soient G et G' des grammaires algébriques sur A. Montrer que le problème de savoir si $L(G) \cap L(G') = \emptyset$ est un problème indécidable.
2. Soit G une grammaire algébrique. Montrer que le problème de savoir si $L(G) = A^*$ est un problème indécidable.

On pourra utiliser l'indécidablité de (PCP) et considérer le langage :
$L(x_1, \ldots, x_n) = \{ba^{i_k}b \ldots ba^{i_1}cx_{i_1} \ldots x_{i_k} \mid 1 \leqslant i_1, \ldots, i_k \leqslant n\}$ pour des mots $x_1, \ldots, x_n$.

Exercice 4 – *Grammaires*

Montrer que l'ensemble des langages récursivement énumérables correspond à l'ensemble des langages engendrés par une grammaire.

– Pour montrer qu'un langage généré par une grammaire est accepté par une machine de Turing, on construira une machine non-déterministe.
– Dans l'autre sens, on pourra supposer, sans perte de généralité, que la machine de Turing qui accepte le langage s'arrête toujours (si elle s'arrête) dans une configuration $h\$B$ (où h est **accept** ou **reject**).

Exercice 5 – *Polynomialitude*

Rappel (ou pas) : on dit que $L \in$ PTIME s'il existe une machine de Turing M qui décide L en temps polynomial (autrement dit : il existe un polynôme P tel que, pour tout mot w, M s'arrête sur w après au maximum $P(|w|)$ transitions, en acceptant si $w \in L$, en rejetant sinon).

1. Si $L_1, L_2 \in$ PTIME, montrer que $L_1 \cap L_2 \in$ PTIME.
2. Le problème :

   Donnée : une machine de Turing M qui calcule en temps polynomial (donc $L(M) \in$ PTIME).

   Question : $L(M) = \emptyset$?

   est-il décidable ?