TD3

Grammaires et fonctions numériques

1 Grammaires

a- On considère le langage $L = \{a^i b^j c^k \mid i \leq j \leq k \wedge i, j, k \in \mathbb{N}\}$. Définissez une grammaire G telle que $L(G) = L$. Prouvez que $L(G) = L$.

b- On rappelle qu'une grammaire G calcule une fonction f si et seulement si pour tout mot $w \in \Sigma_0^*$, on a $SwS \Rightarrow^* v$ si et seulement si $f(w) = v$. Définissez un grammaire G_i calculant la fonction f_i dans chacun des cas suivants :

- $f_1(I^n) = I^{n+1}$
- $f_2(n) = n + 1$ (les entiers seront représentés en binaires)
- $f_3(w) = ww$ avec $\Sigma = \{a, b\}$

2 Fonctions récursives

a- Montrez que la fonction factorielle est une fonction primitive récursive.

b- En admettant les fonctions primitives vues en cours (addition, soustraction non négative, opérateurs booléens) définissez le prédicat :

$$ppt(x, y) = \begin{cases} 1 & \text{si } x < y \\ 0 & \text{sinon} \end{cases}$$

Définissez ensuite la fonction :

$$f(x, y) = \begin{cases} f_1(x, y) & \text{si } x < y \\ f_2(x, y) & \text{sinon} \end{cases}$$

c- Soit $f : \mathbb{N} \to \mathbb{N}$ une fonction primitive récursive, on note F la fonction définie à partir de f par : $F : \mathbb{N} \to \mathbb{N}$ et $F(n) = f(f(...f(n)...))$ où il y a n compositions de la fonction f. Montrez que la fonction F est aussi une fonction primitive récursive.

d- Utilisez l'opérateur de minimisation pour définir les fonctions :

$$quot(m, n) = \begin{cases} \lfloor \frac{m}{n} \rfloor & \text{si } n > 0 \\ 0 & \text{sinon} \end{cases}$$

$$mod(m, n) = \begin{cases} m - \lfloor \frac{m}{n} \rfloor . n & \text{si } n > 0 \\ 0 & \text{sinon} \end{cases}$$