IFT2105-A19 : Devoir 1

À remettre le mardi 24 septembre à 9h30, soit sur StudiUM, soit en main propre.

Problème 1 Programmer en RÉPÉTER

(*20 points*) Écrivez un programme RÉPÉTER qui implante la fonction RECHERCHE(r_1, r_2, r_3) : étant donné un tableau r_1 de taille r_2 encodé avec le codage de Gödel, retourner l'adresse de l'élément r_3, ou bien $r_2 + 1$ si r_3 ne figure pas dans le tableau.

Problème 2 Un peu de maths

Définissons les fonctions $\tilde{A} : \mathbb{N} \to \mathbb{N}$ et $\tilde{B}_i : \mathbb{N} \to \mathbb{N}$ pour $i \geqslant 1$ de la façon suivante :

$$\tilde{A}(x) = A(x, x)$$
$$\tilde{B}_i(x) = \begin{cases} \tilde{A}(x) & \text{si } i = 1 \\ \tilde{B}_{i-1}^{\langle x \rangle}(1) & \text{si } i \geqslant 2. \end{cases}$$

où $A(\cdot, \cdot)$ est, bien sûr, la fonction d'Ackermann. Prouvez les énoncés suivants sur ces fonctions :

(a) (*5 point*) Pour tout $i \geqslant 1$ et tout $x \geqslant 0$, $\tilde{B}_i(x) \geqslant x + 1$.

(b) (*5 point*) $\tilde{B}_i(x + 1) > \tilde{B}_i(x)$ pour tout $i \geqslant 1$ et $x \geqslant 0$.

(c) (*5 point*) $\tilde{B}_{i+1}(x) \geqslant \tilde{B}_i(x)$ pour tout $i \geqslant 1$ et $x \geqslant 0$.

(d) (*15 points*) Pour tout $i \geqslant 1$, $s \geqslant 1$ et $x \geqslant 0$: $sx \leqslant \tilde{B}_i^{\langle s \rangle}(x)$. (*Conseil :* Prouvez d'abord que $\tilde{A}(x) \geqslant 2x$.)

(e) (*5 points*) Pour tout $i \geqslant 1$, tout $x \geqslant 0$, $x < \tilde{B}_i^{\langle x \rangle}(1)$.

Vous pouvez supposer qu'Ackermann est une fonction strictement croissante dans ses deux paramètres. Vous pouvez également utiliser les énoncés précédents, mais pas les suivants.

Problème 3 Les programmes RÉPÉTACKERMANN

Existe-t-il un modèle de calcul intermédiaire entre les programmes RÉPÉTER et les programmes TANTQUE ? Considérons les programmes « RÉPÉTACKERMANN », qui sont des programmes RÉPÉTER avec un type d'instruction supplémentaire :

$$r_i \leftarrow \texttt{ack}(r_j).$$

Cette instruction assigne la valeur $A(r_j, r_j)$ au registre r_i.

(a) (*5 point*) Démontrez que les programmes RÉPÉTACKERMANN sont strictement plus puissants que les programmes RÉPÉTER en donnant une fonction calculable par un programme RÉPÉTACKERMANN qui n'est pas calculable par un programme RÉPÉTER.

(b) (*10 points*) En cours, nous avons défini $\mathcal{M}(\texttt{P}, r_1, \ldots, r_k)$ comme la valeur maximale de tous les registres après l'exécution de P sur un input $r_1, \ldots, r_k$. Considérons maintenant un programme RÉPÉTACKERMANN P qui ne contient aucune boucle. Montrez que

$$\mathcal{M}(\texttt{P}, r_1, \ldots, r_k) \leqslant \tilde{B}_1^{\langle s \rangle}(\max\{r_1, \ldots, r_k\}))$$

où $\tilde{B}_1$ est définie au Problème 2 et où s est une constante ne dépendant que du programme P (et pas des inputs).

(c) (*20 points*) Supposons maintenant que pour tout programme RÉPÉTACKERMANN ayant une profondeur de boucle au plus $i-1$,

$$\mathcal{M}(\texttt{P}, r_1, \ldots, r_k) \leqslant \tilde{B}_i^{\langle s \rangle}(\max\{r_1, \ldots, r_k\})$$

pour un s ne dépendant que du programme. Considérons maintenant un programme RÉPÉTACKERMANN P de la forme suivante :

P_1
répéter r_j fois [
    Q_1
]

où P_1 et Q_1 sont des blocs d'instruction RÉPÉTACKERMANN ayant une profondeur de boucle au plus $i-1$. Montrez que

$$\mathcal{M}(\texttt{P}, r_1, \ldots, r_k) \leqslant \tilde{B}_{i+1}^{\langle s \rangle}(\max\{r_1, \ldots, r_k\}),$$

où s est une constante ne dépendant que du programme. N'hésitez pas à utiliser les propriétés des $\tilde{B}_i$ prouvées au Problème 2 !

(d) (*10 point*) Montrez qu'il existe une constante s tel qu'aucun programme RÉPÉTACKERMANN ayant une profondeur de boucle au plus i ne peut calculer la fonction $\tilde{B}_{i+2}(s+x)$. Encore une fois, n'hésitez pas à utiliser les propriétés prouvées dans le Problème 2.

Note : Nous avons donc (presque) montré que la puissance de calcul des programmes `RÉPÉTACKERMANN` se situe strictement entre celle des programmes `RÉPÉTER` et celle des programmes `TANTQUE`—il ne nous manque que la preuve qu'il existe un programme `TANTQUE` pouvant calculer $\tilde{B}_i(x)$ sur l'input (i, x), ainsi que l'extension de la partie (c) au cas où on a plusieurs blocs avec P_1, Q_1, P_2, Q_2, etc.

Autre note : L'idée de prendre un modèle de calcul existant et d'ajouter une instruction « magique » qui lui donne plus de puissance est très fréquente. On appelle l'instruction magique un « oracle » et ce concept s'avère fort utile dans plusieurs contextes, par exemple pour modéliser une fonction de hachage idéale par une fonction aléatoire sans devoir l'instancier par une fonction de hachage concrète.

Problème 4 Le λ-calcul et les programmes `TANTQUE` (BONUS)

(*20 points*) Démontrez que le λ-calcul permet de simuler les programmes `TANTQUE`. Plus précisément, montrez comment transformer un programme `TANTQUE` arbitraire qui calcule une fonction $f : \mathbb{N}^k \to \mathbb{N} \cup \{\uparrow\}$ en un λ-terme qui calcule la même fonction sur des entiers codés avec l'encodage de Church vu en cours (ou qui mènerait à une suite infinie de β-réductions dans le cas de $\uparrow$).