Calculabilité et Logique

TD2 – Modèles de calculs

24 septembre 2008

Exercice 1 – *Machines à plusieurs rubans*

Une machine de Turing à k rubans est un tuple $(Q, q_0, \Sigma, \delta, \{B, \$\})$ où Q, q_0, Σ, B et $\$$ ont leurs définitions inchangées par rapport aux machines de Turing classiques, mais où la fonction de transition δ est une fonction de $Q \times \Sigma^k$ dans $(Q \cup \{\textbf{accept}, \textbf{reject}\}) \times (\Sigma \times \{\leftarrow, \downarrow, \rightarrow\})^k$.

1. Montrer que tout langage accepté par une machine de Turing M à k rubans qui calcule en temps $f(n) \geqslant n$, est aussi accepté par une machine M' à un seul ruban qui calcule en temps $O(f(n)^2)$.
2. En reprenant une des machines de Turing construites au TD précédent, illustrer ce gain en complexité.

Exercice 2 – *Alphabet de ruban minimal*

Montrer que tout langage récursivement énumérable sur l'alphabet $\{0, 1\}$ est accepté par une machine de Turing à deux rubans dont l'alphabet de ruban ne contient que les symboles 0, 1, B et $\$$.

Exercice 3 – *Machines avec un ruban bi-infini*

On considère les machines de Turing munies d'un ruban infini dans les deux sens : pour ces machines, la tête de lecture peut toujours se déplacer vers la gauche. Au départ le ruban est complètement blanc, sauf la portion qui contient le mot d'entrée, et la tête de lecture est positionnée sur le premier symbole du mot d'entrée.

Montrer que tout langage accepté ou décidé par ce type de machine de Turing l'est aussi par une machine classique (dont le ruban est infini dans un seul sens).

Exercice 4 – *Machines de Turing non déterministes*

Une machine de Turing non déterministe est un tuple $(Q, q_0, \Sigma, \delta, \{B, \$\})$ où Q, q_0, Σ, B et $\$$ ont leurs définitions inchangées par rapport aux machines de Turing classiques, mais où δ est une relation de $(Q \times \Sigma) \times ((Q \cup \{\textbf{accept}, \textbf{reject}\}) \times \Sigma \times \{\leftarrow, \downarrow, \rightarrow\})$. Autrement dit, à partir d'une configuration donnée, plusieurs configurations peuvent être dérivées en une étape. Un mot est accepté par une machine de Turing non déterministe, s'il existe une exécution acceptante pour ce mot.

Montrer que tout langage accepté par une MT non déterministe est aussi accepté par une MT déterministe.

Exercice 5 – *Automates à compteurs*

Un automate à compteurs est un tuple (Q, q_0, F, n, δ) tel que :
- Q est un ensemble fini d'états,
- $q_0 \in Q$ est l'état initial,
- $F \subseteq Q$ est l'ensemble des états finaux,
- $n \in \mathbb{N}$ est le nombre de compteurs,
- $\delta : Q \times \{0,1\}^n \to Q \times \{-1,0,1\}^n$ est la fonction de transition, telle que, si $\delta(q, t_1, \ldots, t_n) = (q', x_1, \ldots, x_n)$, alors $\forall i \in \{1, \ldots, n\}, (t_i = 0 \Rightarrow x_i \neq -1)$.

Une configuration est un tuple $(q, c_1, \ldots, c_n) \in Q \times \mathbb{N}^n$ et les mouvements de la machine sont définis par $(q, c_1, \ldots, c_n) \vdash (q', c_1 + x_1, \ldots, c_n + x_n)$ si $\delta(q, \Delta(c_1), \ldots, \Delta(c_n)) = (q', x_1, \ldots, x_n)$ où $\Delta : \mathbb{N} \to \{0,1\}$ est définie par $\Delta(k) = \begin{cases} 0 & \text{si } k = 0, \\ 1 & \text{si } k \geqslant 1. \end{cases}$

On dira qu'un automate à n compteurs calcule la fonction $f : \mathbb{N} \to \mathbb{N}$ si, pour tout $x \in \mathbb{N}$, partant de la configuration $(q_0, x, 0, \ldots, 0)$, la première configuration pour laquelle l'état est dans F contient $f(x)$ sur le premier compteur (la fonction n'est pas définie pour les valeurs de x sur lesquelles la machine n'atteint jamais d'état final).

1. Montrer que les fonctions $x \mapsto 2x$, $x \mapsto 2x + 1$, $x \mapsto x \mod 2$, $x \mapsto \lfloor x/2 \rfloor$ sont calculables par des automates à deux compteurs.
2. Montrer que l'on peut simuler une machine de Turing à l'aide d'un automate à trois compteurs (on utilisera deux compteurs pour coder le ruban, et un compteur pour les calculs).
3. Montrer que l'on peut simuler le fonctionnement d'un automate à n compteurs à l'aide d'un automate à 2 compteurs (aussi appelé "machine de Minsky"). Conclure.

Exercice 6 – *Énumérons*

Un *énumérateur* est une machine de Turing munie d'un dispositif d' "impression". Chaque fois que la machine veut produire un mot, elle l'envoie vers ce dispositif. Le langage accepté par un énumérateur E est l'ensemble de tous les mots produits par E.

Montrer que, pour tout langage récursivement énumérable, il existe un énumérateur qui génère successivement tous les mots de ce langage (comme le suggère justement le terme "récursivement énumérable").

Cette énumération peut-elle suivre l'ordre lexicographique ?

Exercice 7 – *Composons*

1. Montrer que le complément d'un langage récursif est récursif.
2. Montrer que l'union de deux langages récursifs est récursive.
3. Montrer que l'union de deux langages r.e. est également r.e.
4. Que peut-on dire si un langage et son complémentaire sont tous les deux r.e. ?