Complexité Turing, TD numéro 1
Laurent Jouhet et Irénée Briquel
17 - 18 septembre 2008

Un algorithme de Markov M sur l'alphabet $A_k = \{a_1, ..., a_k\}$ défini par la suite ordonnée finie de règles de production :

$$\begin{array}{ccc} p_1 & \rightarrow_{(t)} & q_1 \\ & \vdots & \\ p_n & \rightarrow_{(t)} & q_n \end{array}$$

où chaque p_i et q_i est un mot de A_k^* et (t) signifie qu'une flèche est ou non marquée par t selon que la règle est ou non terminale.

L'algorithme essaie toutes les règles à partir de la première, et l'applique à la première position possible.

Question 1

Donner un algorithme de Markov qui calcule la fonction successeur des entiers écrits en binaire.

Question 2

Donner un algorithme de Markov qui calcule la fonction prédécesseur des entiers écrits en binaire.

Question 3

Donner un algorithme de Markov qui à partir d'un mot écrit en binaire $x_1 ... x_n$ calcule $x_n ... x_1$.

Question 4

Donner un algorithme de Markov qui accepte les palindromes écrits en binaire.

Question 5

Que fait l'algorithme de Markov suivant sur un mot de $\{a, b\}^*$?

$$\begin{array}{ccc} \alpha\alpha & \rightarrow & \beta \\ \beta\alpha & \rightarrow & \beta \\ \beta a & \rightarrow & a\beta \\ \beta b & \rightarrow & b\beta \\ \beta & \rightarrow_t & \epsilon \\ \alpha aa & \rightarrow & a\alpha a \\ \alpha ab & \rightarrow & b\alpha a \\ \alpha ba & \rightarrow & a\alpha b \\ \alpha bb & \rightarrow & b\alpha b \\ \epsilon & \rightarrow & \alpha \end{array}$$

Rappel : pour une fonction, les quatre propriétés suivantes sont équivalentes :

1) être partiellement récursive (=semi-calculable)
2) être calculable par une RAM
3) être calculable par une machine de Turing

Nous allons montrer qu'être calculable par algorithme de Markov est encore équivalent à ces quatre propositions.

Question 6

Montrer qu'une fonction calculable par algorithme de Markov est calculable par machine de Turing.

Question 7

Montrer que l'on peut simuler le fonctionnement d'une machine de Turing par un algorithme de Markov.

On peut aussi définir un algorithme de Markov sur un alphabet Σ comme une suite de règles numérotées, de la forme :

$$i : x_i \rightarrow y_i : l_i$$

où i est le numéro de l'instruction, et l_i le numéro d'une autre instruction.

On applique d'abord la première règle possible; ensuite, lorsque l'on applique la règle numéro i, on essaie toutes les règles *à partir de la règle* l_i.

Question 8

Montrer que cette définition est équivalente à la première définition.