# TD01 - Algorithmes de Markov

Un algorithme de Markov $M$ sur l'alphabet $A_k = \{a_1, ..., a_k\}$ est 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 $p_i$, $q_i$ 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. A chaque étape, l'algorithme essaie toutes les règles à partir de la première, et applique la première règle possible à la première position possible.

**Exercice 1.**

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

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

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

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

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

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

**Exercice 2.**

Rappel : pour une fonction, les propriétés suivantes sont équivalentes :
– être partiellement récursive (=semi-calculable)
– être calculable par une RAM
– être calculable par une machine de Turing

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

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

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

**Exercice 3.**
On peut aussi définir un algorithme de Markov sur un alphabet $\Sigma$ comme une suite de règles numérotées, de la forme :

$$i : x_i \to y_i : l_i$$

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* $l_i$.

**1.** Montrer que cette définition est équivalente à la première définition.