TD 01 – Machinations

Exercice 1. *Échauffements*
Construire (explicitement) les machines de Turing effectuant les opérations suivantes, et donner leur temps de calcul :

1. étant donnés deux entiers écrits en binaire et séparés par un blanc, calculer leur somme ;
2. étant donnés deux entiers écrits en unaire et séparés par un blanc, calculer leur produit.

Exercice 2. *Football*
Expliquer comment et en combien de temps peut-on simuler une machine de Turing

1. à k rubans par une machine à un ruban ;
2. à k rubans par une machine à deux rubans ;
3. travaillant sur un alphabet Γ par une machine travaillant sur l'alphabet $\{0, 1, \triangleright, \square\}$;
4. à ruban bi-infini par une machine à ruban mono-infini.

Définition. Une machine de Turing est dite *oublieuse*[1] si sur une entrée x, la position de la tête de lecture à l'instant i ne dépend que de i et de $|x|$.

5. Comment peut-on simuler une machine de Turing fonctionnant en temps (constructible) T par une machine oublieuse fonctionnant en temps $O(T^2)$? Et en $O(T \log T)$?

Exercice 3. *Algorithme Cocke Younger Kasami (CYK)*
Rappels :
– Une grammaire hors-contexte est un quadruplet $G = (V, T, P, S)$ où
  – V est l'alphabet fini de variables ;
  – T est l'alphabet terminal ;
  – S est l'axiome (variable de départ) ;
  – et P est l'ensemble de productions $A \to \omega$ où $A \in V$ et $\omega \in (V \cup T)^*$
– Une grammaire hors-contexte est sous forme normale de Chomsky si toutes ses productions sont de la forme $A \to BC$ et $A \to a$ où $B, C \in V$ et $a \in T$.

Théorème. *Tout langage hors-contexte sans le mot vide peut être engendré par une grammaire normale de Chomsky.*

✎ Montrer qu'un langage hors-contexte (engendré par une grammaire hors-contexte) peut être reconnu en temps polynomial. *Indication.* Par programmation dynamique, remplir un tableau qui en case (i, j) contient l'ensemble des variables de la grammaire qui génèrent le sous-mot $x_i x_{i+1} \dots x_j$ pour $i \leq j$, et est vide si $i > j$.

Exercice 4. *Eh ! ça va la vache ?*
Soit Σ un alphabet. Pour $x \in \Sigma^\star$, on note $\bar{x}$ le miroir de x ($\overline{saper} = repas$). Le langage des palindromes sur Σ est

$$\textsc{Pal} = \{x \in \Sigma^* : x = \bar{x}\}.$$

1. *Oblivious*, en anglais.

1. Décrire une machine de Turing à deux rubans qui reconnaît PAL en temps linéaire.

2. Décrire une machine de Turing à un seul ruban qui reconnaît PAL en temps quadratique.

On suppose maintenant que toutes les machines considérées ont un seul ruban, infini vers la droite mais pas vers la gauche, dont les cases sont numérotées à partir de 0. La *suite des franchissements* (en anglais, *crossing sequence*) de M sur l'entrée x, à la frontière entre les cases i et $(i+1)$ du ruban, est définie comme la suite des états de M à chaque passage de la tête de lecture de la case i à la case $(i+1)$ et inversement. On la note $C(x,i)$.
Fixons une machine de Turing M à un ruban reconnaissant PAL. Soit PAL_n le sous-ensemble de PAL défini par

$$\text{PAL}_n = \{x0^{2n}\bar{x} : x \in \Sigma^n\},$$

en supposant que $0 \in \Sigma$. Définissons

$$C(x) = \{C(x,i) : n \leq i \leq 3n\}.$$

3. Montrer que pour tout $x, y \in \text{PAL}_n$ tels que $x \neq y$, $C(x) \cap C(y) = \emptyset$.

4. Pour $x \in \text{PAL}_n$, soit m_x la plus courte suite de franchissements de $C(x)$, et notons $m = \max\{|m_x| : x \in \text{PAL}_n\}$. Soit Q l'ensemble des états de M. Montrer que

$$\frac{|Q|^{m+1} - 1}{|Q| - 1} \geq 2^n.$$

5. En déduire que la complexité de M est $\Omega(n^2)$.

Exercice 5. *Bonus :* NP*-dur, par bdur*

1. Soit MTND-TEMPS le problème de décider, étant données une machine de Turing M et une borne de temps t inférieure au nombre d'états de M, si M s'arrête en moins de t étapes lorsque son ruban d'entrée est initialement vide. Montrer que MTND-TEMPS est NP-complet.

Définition. Soit T un ensemble de tuiles carrées (de côté 1) dont chaque bord se voit attribuer une couleur choisie dans un ensemble C. Un **pavage valide du plan** consiste à recouvrir le plan avec ces tuiles de telle sorte que deux tuiles adjacentes aient leurs bords communs de même couleur.

2. Soit PFP le problème de décider, étant donnés un ensemble fini T de tuiles à couleurs dans un ensemble fini C contenant une tuile blanche et un entier n, si on peut paver un carré de taille $n \times n$ de manière valide et non triviale[2]. On demande de plus que les bords du carrés soient blancs. Montrer que PFP est NP-complet.

Commentaires. La simulation d'une machine de Turing à k rubans par une machine à un ruban en temps quadratique est due à Juris Hartmanis et Richard E. Stearns (*On the Computational Complexity of Algorithms*, 1965). Dans ce même papier, ils introduisent la classe $\text{DTIME}(T(n))$ quelque soit T et prouvent un certain nombre de résultats utiles. Ils utilisent aussi pour la première fois l'appellation *Computational Complexity*. Ce papier leur a valu le prix Turing en 1993. Le fait que la simulation nécessite un temps quadratique dans le pire

2. Au moins l'une des tuiles doit être non entièrement blanche.

des cas (exercice 4) a été démontré par Frederick C. Hennie (*One-Tape, Off-Line Turing Machine Computations*, 1965). Enfin, la simulation d'une machine à k ruban par une machine à deux rubans en temps $O(T \log T)$ est due à Hennie et Stearns (*Two-Tape Simulation of Multitape Turing Machines*, 1966).