Tutorial 01 – Machinations

Exercise 1. *Warm-ups*

(Explicitly) construct Turing Machines doing the following tasks, and give their running time:

1. given two binary integers separated by a blank, compute their sum;
2. given two unary integers separated by a blank, compute their product.

Exercise 2. *Football*

Show how and in how much time it is possible to simulate a Turing Machine

1. with k tapes by a one-tape TM;
2. with k tapes by a two-tape TM;
3. using an alphabet Γ by a TM using the alphabet $\{0, 1, \triangleright, \square\}$;
4. with a bi-infinite tape by a standard TM.

Definition. A TM is said *oblivious* if on an input x, the head position at time step i only depends on i and $|x|$.

5. How can one simulate a TM working in (constructible) time T by an oblivious one working in time $O(T^2)$? And $O(T \log T)$?

Exercise 3. *Cocke Younger Kasami Algorithm (CYK)*

Remainder :

- A context-free grammar is a tuple $G = (V, T, P, S)$ where
  - V is the finite alphabet of variables;
  - T is the alphabet of terminals;
  - S is the axiom (start variable);
  - and P is the set of productions $A \to \omega$ where $A \in V$ and $\omega \in (V \cup T)^*$
- A context-free grammar is in Chomsky's normal form if all its productions either $A \to BC$ or $A \to a$ where $B, C \in V$ and $a \in T$.

Theorem. *Every context-free language without the empty word can be generated by a grammar in Chomsky's normal form.*

✎ Show that a context-free language can be recognized in polynomial time. *Hint.* Use dynamic programing, fill up a table which contains in cell (i, j) the set of variables that generate the sub-word $x_i w_{i+1} \dots x_j$ for $i \leq j$, and is empty otherwise.

Exercise 4. *Eh ! ça va la vache ?*

Let Σ be an alphabet. For $x \in \Sigma^\star$, $\bar{x}$ denotes the mirror of x ($\overline{\textit{rennid}} = \textit{dinner}$). The language of palindromes over Σ is

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

1. Describe a two-tape TM that recognizes Pal in linear time.

2. Describe a one-tape TM that recognizes Pal in quadratic time.

Consider now only one-tape TMs, and suppose the cells of the tape are given a number starting at zero for the leftmost cell. The *crossing sequence* of M on input x, at the border between cells i and $(i+1)$, is defined as the sequence of states that M has when its head crosses from cell i to $(i+1)$ and conversely. It is denoted by $C(x,i)$.
Fix a one-tape TM M recognizing Pal. Let $\textsc{Pal}_n$ be the subset of Pal defined by

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

where $0 \in \Sigma$. Define

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

3. Prove that for all $x, y \in \textsc{Pal}_n$ such that $x \neq y$, $C(x) \cap C(y) = \emptyset$.

4. For $x \in \textsc{Pal}_n$, let m_x be the shortest crossing sequence of $C(x)$, and let $m = \max\{|m_x| : x \in \textsc{Pal}_n\}$. Let Q be the set of states of M. Prove that

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

5. Deduce that M runs in time $\Omega(n^2)$.

Exercise 5. *Bonus : NP-completeness*

1. Let MTND-Time be the problem of deciding, given a TM M and a time bound t not greater that the number of states in M, whether M halts in less than t steps when its input tape initially is empty. Prove that MTND-Time is NP-complete.

Definition. Let T be a set of square tiles (of unit length), each border of which has an attributed color from a set C. A **valid tiling of the plane** consists in covering the plane with those tiles so that the common border of two adjacent tiles has a unique color.

2. Let FTP be the problem of deciding, given a finite set T of tiles with color in a finite set C with a white tile and a integer n, whether it is possible to cover a square of size $n \times n$ in a valid and nontrivial[1] way. The borders of the squares are moreover required to be white. Prove that FTP is NP-complete.

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 $\mathsf{DTIME}(T(n))$ 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 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).

[1] At least one of the tiles is not white.