TD 04 – Oracles

Exercice 1. *Mme de Pompadour* [1]

1. Montrer que TAUTOLOGIE $\in \mathsf{P}^{\mathrm{SAT}}$.
2. Montrer que pour tout $A \in \mathsf{P}$, $\mathsf{P}^A = \mathsf{P}$.
3. Soit $E = \{\langle \alpha, x, 1^n \rangle$: la machine M_α accepte x en au plus 2^n étapes$\}$. Montrer que $\mathsf{NP}^E \subseteq \mathsf{EXP}$ (ce qui conclut la preuve que $\mathsf{P}^E = \mathsf{NP}^E = \mathsf{EXP}$, esquissée en cours).

Exercice 2. *Saut de Turing*

Pour un langage X, on définit $X' = \{\alpha : L(M_\alpha^X) = \emptyset\}$.

1. Soit $A = \{\langle \alpha, w \rangle : M_\alpha(w) = 1\}$ et $B = \{\alpha : L(M_\alpha) = \emptyset\}$. Montrer que A est décidable par une machine de Turing avec l'oracle B, et inversement.
2. Montrer que X' n'est pas décidable par une machine de Turing avec l'oracle X.

Exercice 3. *Karp vs. Cook-Turing*

On définit la *réduction polynomiale à la Cook-Turing* par $A \leq_T^p B$ si $A \in \mathsf{P}^B$. Dans cet exercice, on notera $\leq_m^p$ la réduction polynomiale habituelle, appelée *réduction many-one*, ou *réduction Karp*.

1. Montrer que $A \leq_T^p B$ et $B \leq_T^p C$ implique $A \leq_T^p C$.
2. Montrer que pour tout A, $\bar{A} \leq_T^p A$.
3. Montrer que si TAUTOLOGIE $\leq_m^p$ SAT alors $\mathsf{NP} = \mathsf{coNP}$.
4. Montrer que $\mathsf{NP} = \mathsf{coNP}$ ssi NP est close pour $\leq_T^p$ ($A \leq_T^p B$ et $B \in \mathsf{NP} \implies A \in \mathsf{NP}$).
5. Quelles relations existent entre $\leq_m^p$ et $\leq_T^p$?

** **Exercice 4.** *Théorème de hiérarchie en temps non déterministe (Cook 1972)*

Théorème. *Soit f et g deux fonctions constructibles en temps telles que $f(n+1) = o(g(n))$. Alors*

$$\mathsf{NTIME}(f(n)) \subsetneq \mathsf{NTIME}(g(n)).$$

Dans la suite on suppose donc que $f(n+1) = o(g(n))$, et on veut montrer ce théorème.

1. Rappeler l'idée de la preuve du théorème de hiérarchie en temps déterministe et expliquer pourquoi cette preuve ne peut pas simplement être adaptée au cas présent.
2. Expliquer comment effectivement énumérer les MTND fonctionnant en temps $O(f(n))$.

On va utiliser une *diagonalisation paresseuse*. D'habitude pour diagonaliser, on cherche à « éliminer » la machine M_i sur l'entrée i. Dans la version paresseuse, on cherche à éliminer M_i non pas sur une entrée bien précise mais sur l'une des entrées d'un ensemble I_i d'entrées.
A chaque machine M_i de l'énumération précédente, on associe un ensemble unaire $I_i = \{1^k : \alpha_i \leq \beta_i\}$ où α_i et β_i seront définis plus tard. Soit N la MTND suivante : sur l'entrée x, N détermine i tel que $x \in I_i$, puis

1. Diable, mais pourquoi avoir choisi ce nom ?

1. si $x \in I_i \setminus \{1^{\beta_i}\}$, N simule $M_i(x \cdot 1)$ de manière non déterministe en s'arrêtant au bout d'au plus $g(|x|)$ étapes, et accepte ssi M_i s'est arrêtée en ce temps-là et a accepté ;
2. si $x = 1^{\beta_i}$, N simule $M_i(1^{\alpha_i})$ **de manière déterministe**, et répond le contraire de M_i.

3. Comment choisir α_i et β_i tels que $L(N) \in \mathsf{NTIME}(g(n))$? **Indication.** Trouver i tel que $x \in I_i$ doit être *suffisamment rapide* et l'étape **(b)** également.

4. Supposons que $L(N) \in \mathsf{NTIME}(f(n))$, grâce à une MTND M. Montrer qu'il existe i tel que $M = M_i$ et tel qu'à l'étape **(a)**, M_i est toujours simulée jusqu'au bout.

5. Conclure.