Complexité structurelle – TD 6
Classes probabilistes, 2

Si on ordonne les mots sur $\{0, 1\}$ par ordre lexicographique, on peut parler du i-ème mot binaire, que l'on notera x_i. À un langage L on associe la suite $u \in \{0, 1\}^{\mathbf{N}}$ définie par

$$u_i = \begin{cases} 1 & \text{si } x_i \in L \\ 0 & \text{sinon.} \end{cases}$$

Enfin, sur l'ensemble des suites de $\{0, 1\}$, on met la mesure de probabilité uniforme : elle est induite par la mesure d'équiprobabilité sur $\{0, 1\}$, ou en d'autres termes, pour tout i, la probabilité que $u_i = 1$ est $1/2$. On peut ainsi parler d'oracle aléatoire.

Exercice 1 On veut montrer que $\mathrm{P}^A = \mathrm{BPP}^A$ avec probabilité 1, c'est-à-dire que pour presque tout oracle A, $\mathrm{P}^A = \mathrm{BPP}^A$.

1. D'abord, quelques définitions. Le but du jeu est donc de montrer que pour tout $\delta > 0$, la probabilité que $\mathrm{P}^A \neq \mathrm{BPP}^A$ est $< \delta$. On considère pour cela une énumération (M_i) des machines de Turing probabilistes fonctionnant en temps polynomial. On dira que la machine M_i est de type BPP avec l'oracle A si pour tout x, dans le calcul de M_i^A sur x, soit la proportion de chemins acceptant est d'au moins $3/4$, soit la proportion de chemins rejetant est d'au moins $3/4$. On notera $p_i(n)$ le temps de calcul de la machine M_i sur des entrées de taille n.
2. L'idée est d'utiliser l'oracle A comme source de bits pseudo-aléatoires. Pour obtenir des bits « aléatoires », on interrogera A sur des mots que la machine M_i ne demandera pas. Par ailleurs, avec ces bits on souhaite simuler M_i plusieurs fois de façon à réduire suffisamment la probabilité d'erreur. Proposer une machine déterministe D_i avec oracle, fonctionnant en temps polynomial, qui calcule des bits pseudo-aléatoires grâce à son oracle, et qui s'en sert pour simuler M_i (avec le même oracle) un nombre de fois k à déterminer plus tard.
3. Si i est un entier et x un mot binaire, on note $E_{i,x}$ l'ensemble
$$E_{i,x} = \{A; M_i \text{ avec oracle } A \text{ est de type BPP, et } D_i^A(x) \neq M_i^A(x)\}.$$
Majorer $\mathrm{Prob}(A \in E_{i,x})$ en fonction du nombre k de simulations de M_i dans la question précédente.
4. Supposons que $\mathrm{P}^A \neq \mathrm{BPP}^A$ pour un certain ensemble A. A est-il dans un des $E_{i,x}$ avec $|x| > i$?
5. Majorer la probabilité que $\mathrm{P}^A \neq \mathrm{BPP}^A$ en fonction de k, choisir k correctement et conclure.

Un *circuit arithmétique* est un circuit (graphe orienté sans cycle) ayant pour entrées des indéterminées x_i et la constante 1, et ayant pour portes $+$, $-$ et $\times$.

Ainsi, un tel circuit calcule un polynôme dans $\mathbf{Z}[x_1, \ldots, x_k]$. La taille du circuit est le nombre de sommets du graphe.

Soit Polynôme nul le problème suivant. En entrée, on donne un circuit arithmétique C. Il calcule un polynôme $f(x_1, \ldots, x_k) \in \mathbf{Z}[x_1, \ldots, x_k]$. Il s'agit de décider si $f = 0$.

Exercice 2

1. Majorer la valeur absolue des nombres que l'on peut calculer avec un tel circuit en fonction de sa taille et du max de la valeur absolue des x_i.
2. Proposer un algorithme coRP pour Polynôme nul.
3. Soit le problème suivant, où C_1 et C_2 sont des circuits arithmétiques :

$$\{(C_1, C_2) : C_1 \in \text{Polynôme nul et } C_2 \notin \text{Polynôme nul}\}.$$

   Donnez la plus petite classe de complexité dans laquelle vous pouvez mettre ce problème.

On ne sait pas si RP est clos par complément, ou en d'autres termes si RP = coRP. Ainsi, on définit la classe ZPP *(zero error probabilistic polynomial time)* par ZPP = RP $\cap$ coRP.

Exercice 3

Montrer qu'un langage A est dans ZPP si et seulement s'il existe pour A un algorithme probabiliste dont le temps d'exécution *dans le pire cas* est polynomial, qui a la possibilité de répondre « je ne sais pas » mais avec probabilité $< 1/3$, et qui ne se trompe jamais dans les autres cas. *(Algorithmes de Las Vegas)*

Montrer une autre caractérisation : A est dans ZPP si et seulement s'il existe pour A un algorithme probabiliste ne se trompant jamais, et dont le temps *moyen* d'exécution est polynomial.

Exercice 4

Montrer que si NP $\subseteq$ BPP alors NP $\subseteq$ RP.

Exercice 5

Reprendre les exercices 2 et 3 du TD5 avec BPP à la place de RP.