COMPLEX - Partie 2
2020-2021

Énoncé

Numéro d'anonymat :

Une rédaction claire et concise sera appréciée. Toute affirmation devra être justifiée.
Une question non résolue n'empêche pas de faire les suivantes
(dans ce cas indiquez clairement que vous admettez le(s) résultat(s) de la question non faite).

Exercice 1 : Classe de complexité probabiliste $\mathcal{PP}$

Soit Σ un alphabet arbitraire fini (avec $\#\Sigma > 1$). Nous considérons la classe de complexité $\mathcal{PP}$ définie comme étant l'ensemble des langages L définis sur Σ pour lesquels il existe une machine de Turing probabiliste $\mathcal{M}$ telle que :

(1) $\mathcal{M}$ s'arrête sur toute entrée et s'exécute en temps polynomial ;

(2) pour tout $x \in L$, $\Pr[\mathcal{M}(x) = 1] > 1/2$;

(3) pour tout $x \notin L$, $\Pr[\mathcal{M}(x) = 0] > 1/2$;

1.a] Nous considérons la classe de complexité $\mathcal{PP}^{\geq}$ définie comme étant l'ensemble des langages L définis sur Σ pour lesquels il existe une machine de Turing probabiliste $\mathcal{M}$ qui vérifie la propriété (1) précédente et

(2') pour tout $x \in L$, $\Pr[\mathcal{M}(x) = 1] \geq 1/2$;

(3') pour tout $x \notin L$, $\Pr[\mathcal{M}(x) = 0] \geq 1/2$;

Montrer que tout langage défini sur Σ appartient à $\mathcal{PP}^{\geq}$.

1.b] Nous considérons la classe de complexité $\mathcal{PP}_{1/4}$ définie comme étant l'ensemble des langages L définis sur Σ pour lesquels il existe une machine de Turing probabiliste $\mathcal{M}$ qui vérifie la propriété (1) précédente et

(2") pour tout $x \in L$, $\Pr[\mathcal{M}(x) = 1] > 1/4$;

(3") pour tout $x \notin L$, $\Pr[\mathcal{M}(x) = 0] > 3/4$.

Montrer que $\mathcal{PP} \subseteq \mathcal{PP}_{1/4}$

Indication : Pour un langage L de $\mathcal{PP}$, on pourra considérer une machine de Turing probabiliste $\mathcal{M}$ vérifiant les propriétés (1), (2) et (3') et considérer la machine de Turing probabiliste $\mathcal{M}'$ qui

- avec probabilité $1/2$, rejette son entrée x (sans faire de calcul) ;
- avec probabilité $1/2$, exécute $\mathcal{M}$ sur son son entrée x et retourne la réponse de $\mathcal{M}$.

1.c] Montrer que $\mathcal{PP}_{1/4} \subseteq \mathcal{PP}$ (et donc $\mathcal{PP}_{1/4} = \mathcal{PP}$).

Indication : Pour un langage L de $\mathcal{PP}_{1/4}$, on pourra considérer une machine de Turing probabiliste $\mathcal{M}$ vérifiant les propriétés (1), (2") et (3") et considérer la machine de Turing probabiliste $\mathcal{M}'$ qui

- avec probabilité $3/8$, accepte son entrée x (sans faire de calcul) ;
- avec probabilité $1/8$, rejette son entrée x (sans faire de calcul) ;
- avec probabilité $1/2$, exécute $\mathcal{M}$ sur son son entrée x et retourne la réponse de $\mathcal{M}$.

Exercice 2 : Algorithme de Hirsch pour MAX-3-SAT

Algorithme 1 : Algorithme de Hirsch pour MAX-3-SAT

Entrée : Formule Φ de m clauses $C_1, \ldots, C_m$ en n variables $x_1, \ldots, x_n$; $T \in \mathbb{N}$
Sortie : $(y, \mu) \in \{0,1\}^n \times \{0, \ldots, m\}$ tel que $\Phi(y)$ satisfait μ clauses
$\mu^\star \leftarrow -1$; $y^\star \leftarrow (0, \ldots, 0)$ ▷ `solution optimale`
pour j *de* 1 *à* T **faire**
    $y = (y_1, \ldots, y_n) \xleftarrow{} \{0,1\}^n$
    $\mu \leftarrow \#\{i \in \{1, \ldots, m\} | C_i(y) = 1\}$ ▷ `nombre de clauses satisfaites par` y
    si $\mu > \mu^\star$ **alors**
        $\mu^\star \leftarrow \mu$; $y^\star \leftarrow y$ ▷ `mise à jour de la solution optimale`
    pour i *de* 1 *à* n **faire**
        si $\Phi(y) = 0$ **alors**
            considérer une clause C_t (arbitraire) non satisfaite de Φ
            $j \xleftarrow{} \textsc{Variables}(C_t)$ ▷ `tirage d'une variable dans` C_t
            $y_j \leftarrow 1 - y_j$ ▷ `changement de la valeur de` y_j
            $\mu \leftarrow \#\{i \in \{1, \ldots, m\} | C_i(y) = 1\}$ ▷ `nombre de clauses satisfaites par` y
            si $\mu > \mu^\star$ **alors**
                $\mu^\star \leftarrow \mu$; $y^\star \leftarrow y$ ▷ `mise à jour de la solution optimale`
        sinon
            retourner $(y^\star, \mu^\star)$
retourner $(y^\star, \mu^\star)$

Dans cet exercice, nous considérons un algorithme probabiliste du à E. A. Hirsch pour résoudre le problème MAX-3-SAT de façon approchée. L'algorithme est inspiré de l'algorithme de Schöning pour le problème 3-SAT vu en cours.

Soit Φ une formule booléenne en forme normale conjonctive formée de m clauses contenant toutes au plus 3 littéraux. Soit $\ell \leq m$ le nombre maximal de clauses simultanément satisfiables pour Φ et soit $z = (z_1, \ldots, z_n)$ une assignation (optimale) des variables telle que $\Phi(z_1, \ldots, z_n)$ satisfait ℓ clauses.

2.a] Montrer que $\ell \geq m/2$.

Soit $\epsilon > 0$. Nous disons qu'une assignation des n variables de Φ est *admissible* si elle satisfait au moins $(1-\varepsilon)\ell$ clauses. Notons que si lors de l'exécution de l'algorithme, nous trouvons une assignation des n variables qui est admissible, alors l'algorithme retourne une assignation admissible.

2.b] Considérons une assignation qui ne satisfait pas strictement plus de $m-(1-\varepsilon)\ell$ clauses de Φ dans une itération de la boucle **pour** … **faire** des lignes 7 à 16. Notons probabilité p_ε que l'algorithme change la valeur d'une variable qui a une valeur différente dans cette assignation et dans z. Montrer que

$$p_\varepsilon \geq \frac{\varepsilon\ell}{3(m-(1-\varepsilon)\ell)}$$

2.c] En déduire que

$$p_\varepsilon \geq \frac{\varepsilon}{3(1+\varepsilon)}$$

2.d] Supposons que, lors de l'initialisation de la boucle **pour** … **faire** des lignes 2 à 16, l'assignation y diffère de celle de z en exactement t variables (ce qui se produit avec probabilité $\binom{n}{t}/2^{-n}$). Montrer que pour une telle initialisation, l'algorithme obtient une assignation admissible avec probabilité supérieure ou égale à p_ε^t.

2.e] En sommant pour toutes les valeurs possibles de t, montrer que lors de chaque itération de la boucle **pour** … **faire** des lignes 2 à 16, l'algorithme obtient une assignation admissible avec probabilité supérieure ou égale à $q = \left(\frac{1}{2}\left(1 + \frac{\varepsilon}{3(1+\varepsilon)}\right)\right)^n$.

2.f] Montrer que pour $T = 1/q$, l'algorithme obtient une assignation admissible avec probabilité supérieure ou égale à $\exp(-1) \simeq 0.368$ (et dans ce cas $T = c^n$ avec $c = (2 - 2\varepsilon/(3 + 4\varepsilon)) < 2$).