COMPLEX - Examen 2
2017-2018

Énoncé

Durée : 2 heures

Notes manuscrites autorisées à l'exclusion de toute autre document
L'utilisation de tout matériel électronique (en dehors d'une montre non connectée) est interdite

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

Exercice 1 : Matrices doublement monotone

Soit $n \geq 1$ un entier. Dans cet exercice, nous considérons des matrices carrées de taille n contenant des entiers telles que chaque ligne et chaque colonne de la matrice est triée par ordre croissant (*i.e.* nous considérons des matrices $A = (a_{i,j})_{1 \leq i,j \leq n}$ avec pour tout $i \in \{1, \ldots, n\}$ et tout $j \in \{1, \ldots, n-1\}$, $a_{i,j} \leq a_{i,j+1}$ et $a_{j,i} \leq a_{j+1,i}$). Une telle matrice sera dite *doublement monotone*.

1.a] Proposer un algorithme (déterministe) qui retourne le maximum et le minimum d'une telle matrice en temps $O(1)$.

1.b] Utiliser un algorithme probabiliste de type Las Vegas (vu en cours) pour retourner l'élément médian d'une matrice doublement monotone $A = (a_{i,j})_{1 \leq i,j \leq n}$ en temps espéré $O(n^2)$.

1.c] Décrire et analyser un algorithme (déterministe) pour résoudre le problème suivant en temps $O(n)$: étant donnés quatre indices i, j, i', j' dans $\{1, \ldots, n\}$ et une matrice doublement monotone $A = (a_{i,j})_{1 \leq i,j \leq n}$, calculer le nombre d'entiers de A plus petit que $a_{i,j}$ et plus grand que $a_{i',j'}$.

1.d] Décrire et analyser un algorithme probabiliste pour résoudre le problème suivant en temps $O(n)$: étant donnés des indices i, j, i', j' et une matrice doublement monotone $A = (a_{i,j})_{1 \leq i,j \leq n}$, retourner un élément de M tiré uniformément aléatoirement parmi les éléments plus petits que $a_{i,j}$ et plus grand que $a_{i',j'}$. On supposera que l'ensemble demandé est toujours non vide.

1.e] Décrire un algorithme probabiliste de type Las Vegas (utilisant les questions précédentes) pour calculer l'élément médian d'une telle matrice $A = (a_{i,j})_{1 \leq i,j \leq n}$.

1.f] Montrer que l'algorithme de la question précédente termine (dans le pire des cas sur l'entrée) en temps espéré $O(n \log n)$.

Exercice 2 : Coupe maximum

En théorie des graphes, une *coupe* d'un graphe est une partition des sommets en deux sous-ensembles (on appelle parfois aussi coupe l'ensemble des arâtes ayant une extrâmité dans chaque sous-ensemble de la partition).

Étant donné un graphe non orienté $G = (V, E)$, un sous-ensemble S de V définit la coupe $\{S, V \setminus S\}$ et le *poids* de cette coupe, noté $c(S)$ est le nombre d'arâtes de E ayant une extrâmité à l'intérieur de cet ensemble et l'autre à l'extérieur :

$$c(S) = \#\{\{i, j\} \in E \mid i \in S, j \in V \setminus S\}.$$

Le problème de la coupe maximum est le suivant : étant donné un graphe non orienté $G = (V, E)$, trouver un coupe (ou un sous-ensemble S) de poids maximum (parmi toutes les coupes de G). Il s'agit d'un problème NP-difficile. Nous allons étudier dans cet exercice un algorithme probabiliste de type Monte-Carlo efficace qui retourne une 4-approximation de la coupe maximum (*i.e.* dont le poids est au plus 4 fois plus petit que celui d'une coupe maximum).

2.a] Soit $e \in E$ une arrète du graphe. Montrer que pour un ensemble S aléatoire, l'arâte e a une extrâmité à l'intérieur de S et l'autre à l'extérieur avec probabilité $1/2$.

2.b] En déduire que, pour un ensemble S aléatoire, le poids moyen de la coupe définie par S est égale à $(\#E)/2$.

2.c] Considérons l'algorithme suivant : l'ensemble S est initialement vide et pour chaque nœud $v \in V$, le nœud v est ajouté à l'ensemble S avec probabilité $1/2$. Lorsque tous les nœuds ont été considérés, l'algorithme retourne l'ensemble S construit (qui définit la coupe $\{S, V \setminus S\}$).

Montrer que cet algorithme retourne une 4-approximation de la coupe maximum avec probabilité au moins $1/3$.

Indication : On pourra appliquer l'inégalité de Markov au nombre d'arâtes ne traversant pas la coupe construite.

2.d] Expliquer comment utiliser l'algorithme de la question précédente pour avoir un algorithme qui retourne une 4-approximation de la coupe maximum avec probabilité au moins $1 - (2/3)^k$ pour tout entier $k \in \mathbb{N}$. Donner sa complexité en temps et en espace.

Exercice 3 : Machines de Turing « je-ne-sais-pas » (Examen 2017-2018)

Dans cet exercice, nous considérons une variante des machines de Turing probabilistes. Soit Σ un alphabet fini. Une *?-machine de Turing probabiliste* sur l'alphabet Σ est une machine de Turing probabiliste [1] sur l'alphabet Σ qui s'arrâte sur toute entrée mais qui possède trois états finaux distincts (au lieu de deux) notés q_{acc}, q_{rej} et $q_?$. L'état q_{acc} est l'état d'acceptation, l'état q_{rej} est l'état de rejet et l'état $q_?$ est l'état d'indécision.

La machine retourne toujours une valeur dans l'ensemble $\{1, 0, ?\}$ correspondant aux états q_{acc}, q_{rej} et $q_?$ respectivement (avec l'interprétation suivante : 1 signifie que la machine accepte le mot en entrée, 0 signifie que la machine refuse le mot en entrée, et ? signifie que la machine ne sait pas répondre sur le mot en entrée). Pour une ?-machine de Turing probabiliste $\mathcal{M}$, nous

1. On pourra considérer au choix une machine de Turing à un ruban avec deux fonctions de transition (comme définie en cours) ou une machine à deux rubans et une fonction de transition o le deuxième ruban est un ruban d'aléa en lecture seul ou le curseur se déplace à chaque étape de calcul d'une case vers la droite.

notons $\mathcal{M}(x)$ la variable aléatoire correspondant à la valeur que retourne $\mathcal{M}$ à la fin de son exécution.

Nous définissons la classe de complexité (?)−PP comme l'ensemble des langages L de Σ^* pour lesquels il existe une ?-machine de Turing probabiliste $\mathcal{M}$ qui s'arrête en temps polynomial en la taille de son entrée et telle que

1. pour tout $x \in \Sigma^*$, $\Pr[\mathcal{M}(x) =?] \leq 1/2$;
2. pour tout $x \in L$, $\Pr[\mathcal{M}(x) = 0] = 0$;
3. pour tout $x \notin L$, $\Pr[\mathcal{M}(x) = 1] = 0$.

3.a] Montrer que $\mathsf{P} \subseteq (?)-\mathsf{PP}$

3.b] Montrer que $(?)-\mathsf{PP} \subseteq \mathsf{RP} \cap \text{co}-\mathsf{RP}$

3.c] Montrer que $\mathsf{RP} \cap \text{co}-\mathsf{RP} \subseteq (?)-\mathsf{PP}$ et donc que $\mathsf{RP} = \text{co}-\mathsf{RP} \subset (?)-\mathsf{PP}$.

3.d] Que peut-on en déduire sur $(?)-\mathsf{PP}$?
Donner un argument permettant de prouver directement ce résultat (les détails de la démonstration ne sont pas demandés).

Exercice 4 : Nombres pseudo-premiers forts en base 2

Nous avons vu en cours qu'il existe une infinité de nombres pseudo-premiers de Fermat dans une base donnée (et même qu'il existe une infinité d'entiers n qui sont pseudo-premiers de Fermat dans toute base première avec n, les *nombres de Carmichael*). Dans cet exercice, nous allons montrer qu'il existe une infinité de nombres pseudo-premiers forts en base 2.

4.a] Montrer qu'un nombre pseudo-premier de Fermat en base 2 est nécessairement impair.

4.b] Montrer que si un entier n est composé (*i.e.* $n = a \cdot b$ avec $a, b \geq 2$ des entiers), alors $2^n - 1$ est composé.
Indication : On pourra utiliser l'identité polynomiale $(X^a - 1) = (X - 1)(1 + X + \cdots + X^{a-1})$.

4.c] Soit n un nombre pseudo-premier de Fermat en base 2 et posons $N = 2^n - 1$.

1. Montrer que $(N - 1)/2$ est un nombre impair noté q.
2. Montrer qu'il existe un entier λ tel que $2^{n-1} - 1 = n \cdot \lambda$.
3. En remarquant que $2^n \equiv 1 \bmod (2^n - 1)$, en déduire que $2^q \equiv 1 \bmod N$.

4.d] Conclure

4.e] Existe-t'il une infinité d'entiers n qui sont pseudo-premiers forts dans toute base première avec n ?