COMPLEX - Partie 2
2023-2024

Énoncé

Nom ou numéro d'anonymat :

Durée : 2 heures

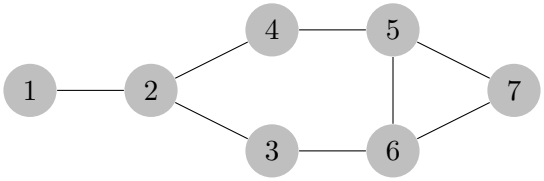
Notes manuscrites et documents de cours autorisés
L'utilisation de tout matériel électronique (en dehors d'une montre non connectée) est interdite

Les exercices sont indépendants.
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 : Couverture par sommets

Soit $G = (V, E)$ un graphe non-orienté. Une *couverture par sommets* de G est un sous-ensemble X de E tel que tout arête de V est adjacente à au moins un sommet de X.

1.a] Trouver une couverture par sommets de cardinal minimal pour le graphe $G = (V, E)$ défini par $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $E = \{\{1, 2\}, \{2, 3\}, \{2, 4\}, \{3, 6\}, \{4, 5\}, \{5, 6\}, \{5, 7\}, \{6, 7\}\}$



Le problème de décider, étant donné un graphe $G = (V, E)$ et un entier k, si il existe une couverture par sommets de cardinal au plus k est un problème $\mathcal{NP}$-complet. L'objectif de cet exercice est de proposer un algorithme probabiliste de type Monte-Carlo avec une probabilité d'erreur inférieure à $1/3$ et de complexité temporelle $O(2^k \cdot \mathsf{poly}(n, m, k))$ où n désigne le nombre de sommets de G, m désigne le nombre d'arêtes et poly une fonction polynomiale que nous ne chercherons pas à déterminer de façon exacte.

1.b] Proposer un algorithme déterministe simple qui, étant donné un graphe $G = (V, E)$ et un entier k, détermine si il existe une couverture par sommets de cardinal au plus k. Donner sa complexité temporelle.

Considérons l'algorithme probabiliste suivant :

Algorithme 1 : Algorithme probabiliste pour la couverture par sommets

Entrée : $G = (V, E)$, un graphe non-orienté k un entier
Sortie : $X \subseteq V$ ou ÉCHEC

$c \leftarrow 0$, $A \leftarrow E$, $X \leftarrow \emptyset$
tant que $A \neq \emptyset \wedge c \leq k$ **faire**
  $e = \{u, v\} \xleftarrow{\text{🎲}} A$ ▷ `tirage aléatoire d'une arête` e `dans` A
  $s \xleftarrow{\text{🎲}} \{u, v\}$ ▷ `tirage aléatoire d'un sommet` s `de l'arête` $e = \{u, v\}$
  $A \leftarrow A \setminus \{e \in A | s \in e\}$ ▷ `suppression des arêtes adjacentes à` s `dans` A
  $X \leftarrow X \cup \{s\}$
  $c \leftarrow c + 1$
si $c \leq k$ **alors**
  retourner X
sinon
  retourner ÉCHEC

1.c] Montrer que si cet algorithme retourne un ensemble X, alors il s'agit effectivement d'une couverture par sommets de cardinal inférieur ou égal à k de G.

Soit un graphe non-orienté $G = (V, E)$ et soit k un entier. Supposons que G possède une couverture par somme X de cardinal inférieur ou égal à k.

1.d] Soit $\{u, v\} \in E$ une arête de G. Considérons l'expérience aléatoire qui consiste à tirer uniformément aléatoirement l'un des deux sommets de cet arête (c'est-à-dire, le sommet u avec probabilité $1/2$ ou le sommet v avec probabilité $1/2$). Quelle est la probabilité que le sommet sélectionné appartienne à X ?

1.e] En déduire que l'algorithme 1 retourne X avec probabilité supérieure ou égale à 2^{-k}.

1.f] En déduire qu'il existe un algorithme probabiliste qui, étant donné un graphe $G = (V, E)$ et un entier k, détermine si il existe une couverture par sommets de cardinal au plus k

— de type Monte-Carlo (avec une probabilité d'erreur inférieure à $1/3$)

— et de complexité temporelle $O(2^k \cdot \mathsf{poly}(n, m, k))$ où n désigner le nombre de sommets, m désigne le nombre d'arêtes (et poly une fonction polynomiale que nous ne chercherons pas à déterminer de façon exacte).

Exercice 2 : Égalité de multi-ensembles

Un *multi-ensemble* est un couple (S, m) où S est un ensemble appelé *support* et $m : S \to \mathbb{N} \setminus \{0\}$ est une fonction appelée *multiplicité*. Informellement un multi-ensemble est un ensemble dans lequel chaque élement peut apparaître plusieurs fois et dans le multi-ensemble (S, m), un élément x de S, apparaît alors $m(x)$ fois. Deux multi-ensembles (S_1, m_1) et (S_2, m_2) sont égaux si $S_1 = S_2$ et si pour tout $x \in S_1$, $m_1(x) = m_2(x)$.

Dans la suite, nous considérons des tableaux finis d'entiers positifs notés en utilisant des accolades doubles $\{\!\{\ldots\}\!\}$ et nous faisons correspondre à chaque tableau le *multi-ensemble* associé. Par exemple, pour le tableau $\{\!\{1, 2, 1, 2, 2, 3\}\!\}$, nous associons le multi-ensemble (S, m) avec $S = \{1, 2, 3\}$ avec $m(1) = 2$, $m(2) = 3$ et $m(3) = 1$. L'objectif de l'exercice est d'étudier deux approches pour décider si deux tableaux d'entiers sont associés au même multi-ensemble (c'est-à-dire si ils contiennent les mêmes éléments et chacun d'eux apparait le même nombre de fois dans chaque tableau). Ainsi, les tableaux $\{\!\{1, 2, 1, 2, 2, 3\}\!\}$ et $\{\!\{3, 2, 1, 2, 1, 2\}\!\}$ sont associés au même multi-ensemble mais ce n'est pas le cas du tableau $\{\!\{2, 3, 1, 2, 1, 1\}\!\}$.

2.a] Proposer un algorithme déterministe qui, étant donné deux tableaux de longeur $n \geq 1$, décide si ils sont associés au même multi-ensemble en $O(n \log n)$ comparaisons d'entiers.

2.b] Considérons (S_1, m_1) et (S_2, m_2) deux multi-ensembles d'entiers (ou de façon équivalente, deux tableaux d'entiers T_1 et T_2) et les deux polynômes

$$P_1(X) = \prod_{s \in S_1} (X - s)^{m_1(s)} \text{ et } P_2(X) = \prod_{s \in S_2} (X - s)^{m_2(s)}$$

Montrer que les deux multi-ensembles (S_1, m_1) et (S_2, m_2) sont égaux si et seulement si les polynômes P_1 et P_2 sont égaux.

2.c] Considérons deux tableaux d'entiers T_1 et T_2 de longueur $n \geq 1$ et les deux polynômes associés P_1 et P_2 de la question précédente. Montrer que si T_1 et T_2 ne sont pas associés au même multi-ensemble, alors la probabilité que $P_1(\alpha) = P_2(\alpha)$ pour un entier α tiré uniformément aléatoirement dans l'ensemble d'entiers $\{1, 2, 3, \ldots, 8n\}$ est inférieure ou égale à $1/8$.
Indication : On pourra utiliser le lemme de Schwartz-Zippel (en justifiant son usage).

2.d] Supposons que les deux tableaux contiennent des entiers inférieures à une borne $B \geq 0$. Montrer que pour un entier α de l'ensemble $\{1, 2, 3, \ldots, 8n\}$, nous avons $|P_1(\alpha)| \leq (8n + B)^n$ et $|P_2(\alpha)| \leq (8n + B)^n$.

Le calcul des entiers $P_1(\alpha)$ et $P_2(\alpha)$ peut donc être trop coûteux pour obtenir un algorithme qui rivalise avec celui de la question **2.a**. Pour terminer l'exercice, nous allons montrer qu'il est possible d'éviter ce problème en calculant $(P_1(\alpha) \bmod p)$ et $(P_2(\alpha) \bmod p)$ où p sera un nombre premier relativement petit et tiré aléatoirement. Nous supposons dans la suite que les deux tableaux T_1 et T_2 contiennent des entiers inférieures à une borne B et que $P_1(\alpha) \neq P_2(\alpha)$.

2.e] Montrer le nombre d'entiers premiers qui divise $(P_1(\alpha) - P_2(\alpha))$ est inférieur ou égal à un entier T avec $T = O(n \log(n + B))$ (On ne demande pas d'expliciter la constante du $\ll O(\cdot) \gg$).

2.f] Notons $\pi(y) = \#\{p \in \mathbb{N}, p \leq y \text{ et } p \text{ est premier}\}$. Montrer que si l'on choisit un nombre premier p uniformément aléatoirement dans l'ensemble d'entiers $\{2, 3, \ldots, y\}$, alors p divise $(P_1(\alpha) - P_2(\alpha))$ avec probabilité au plus $(T/\pi(y))$.

2.g] Nous admettons que $\pi(y) = \Omega(y/\log(y))$ (une version faible du théorème des nombres premiers). Montrer qu'il existe un entier y avec $y = O((n + B)^2)$ tel que la probabilité qu'un nombre premier p tiré uniformément aléatoirement dans l'ensemble d'entiers $\{2, 3, \ldots, y\}$, divise $(P_1(\alpha) - P_2(\alpha))$ avec probabilité est inférieure ou égale à $1/8$. (On ne demande pas d'expliciter la constante du « $O(\cdot)$ »).

2.h] En déduire un algorithme probabiliste de complexité $O(n \log^2(n))$ qui, étant donné deux tableaux de longueur n qui contiennent des entiers de taille polynomial en n, retourne un bit $b \in \{0, 1\}$ de sorte que :

- si les deux tableaux sont associés au même multi-ensemble, l'algorithme retourne toujours $b = 1$;
- si les deux tableaux ne sont pas associés au même multi-ensemble, la probabilité que la valeur retournée soit $b = 1$, est inférieure ou égale à $1/4$.

Remarque : Cet algorithme a le même complexité que celui de la question **2.a** mais il a plusieurs avantages intéressants (par exemple, il peut être utilisé lorsque les tableaux sont donnés sous forme d'un flot continu et que l'on dispose d'une mémoire limitée ne permettant pas de les stocker en entier ou lorsque les tableaux sont partagés par deux noeuds différents sur un réseau et que l'on souhaite limiter la communication).

Exercice 3 : Classe de complexité probabiliste $\mathcal{RP}$

Soit Σ un alphabet arbitraire fini (avec $\#\Sigma > 1$). Soit $\alpha \in [0, 1]$. Nous considérons la classe de complexité $\mathcal{RP}_\alpha$ définie comme étant l'ensemble des langages $L \subseteq \Sigma^*$ pour lesquels il existe une machine de Turing probabiliste $\mathcal{M}$ telle que :

(1) $\mathcal{M}$ s'arrête sur toute entrée $x \in \Sigma^*$ et s'exécute en temps polynomial $p(|x|)$ où $|x|$ désigne la longueur de x ;

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

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

La classe $\mathcal{RP}$ vue en cours correspond donc à la classe $\mathcal{RP}_{2/3}$.

3.a] Montrer que $\mathcal{RP}_1 = \mathcal{P}$.

3.b] Décrire l'ensemble des langages $\mathcal{RP}_0$.

3.c] Soient $\alpha, \beta \in]0,1[$. Montrer que si $\alpha \geq \beta$, alors $\mathcal{RP}_\alpha \subseteq \mathcal{RP}_\beta$.

3.d] Soient $\alpha, \beta \in]0,1[$. Montrer que si $\alpha \geq \beta$, alors $\mathcal{RP}_\alpha \supseteq \mathcal{RP}_\beta$.

Indication : On pourra utiliser les techniques d'amplification vues en cours.

3.e] Conclure