

Examen du 9 janvier 2023

Document autorisé : une feuille A4 manuscrite.

*Le niveau de détail de la rédaction attendu est de l'ordre d'un paragraphe pour toutes les questions non étiquetées * (y compris les questions de cours), en identifiant bien les arguments essentiels. Les questions * seront rédigées en détail. De plus, l'algorithme en temps polynomial des questions * (s'il existe) devra être déterministe et de complexité la meilleure possible (parmi les choix de paradigmes proposés).*

- Exercice 1 - Points Bonus. Parmi les cinq questions * de cet examen, l'une est un problème NP-difficile (pas forcément *NP-complet*, il n'est pas demandé de montrer l'éventuelle appartenance à NP) et les quatre autres utilisent les paradigmes : diviser pour régner, programmation dynamique, recherche locale, et algorithme glouton. Pour prouver la NP-difficulté on pourra utiliser une réduction via un des quatre problèmes : 3-SAT, CLIQUE, SOMME, SET COVER.

1. Trouver la bijection entre les questions * et les cinq cas possibles.

- Exercice 2 - Equations polynomiales. Dans le problème *NULL*, l'entrée est un ensemble de n variables réelles $x_1, \dots, x_n$ ainsi qu'un ensemble de polynômes $P_1, \dots, P_m$ à coefficients entiers en ces variables. La sortie est VRAI s'il existe une affectation réelle des variables qui annule tous les P_i . C'est à dire s'il existe des réels $r_1, \dots, r_n$ tels que pour tout $i = 1, \dots, m$ on a $P_i(r_1, \dots, r_n) = 0$.

1. Existe-t-il un algorithme polynomial pour NULL lorsque les polynômes ont degré au plus 1 (i.e. les monômes sont de la forme x_i ou bien constant) ?
2. * Existe-t-il un algorithme polynomial pour NULL lorsque les polynômes ont degré au plus 2 (i.e. les monômes sont de la forme $x_i, x_i^2, x_i x_j$ ou bien constant) ?

- Exercice 3 - Gestion de conflits. Vous gérez une équipe A de n agents $a_1, \dots, a_n$ que vous voulez affecter à trois projets. Malheureusement vous avez identifié un ensemble E de paires d'agents $\{a_i, a_j\}$ en situation de conflit personnel. Afin de minimiser les conflits lors des projets, vous voulez trouver une partition de A en trois équipes A_1, A_2, A_3 de telle sorte que le nombre total de conflits $\{a_i, a_j\} \in E$ tels que a_i et a_j sont dans des équipes différentes soit le plus grand possible (notons OPT ce nombre).

1. * Existe-t-il un algorithme polynomial en n qui retourne une solution de valeur au moins $2\text{OPT}/3$?

- Exercice 4 - Petit triangle. On se donne un ensemble de $n \geq 3$ points deux à deux distincts $P = \{p_1, \dots, p_n\}$ dans le plan. Un *P-triangle* est un triangle formé par trois points différents p_i, p_j, p_k de P . Sa *longueur* est son périmètre (i.e. $d(p_i, p_j) + d(p_i, p_k) + d(p_j, p_k)$). On autorise les triangles dégénérés (plats).

1. Montrer que si on se donne 51 points dans le carré unité $[0, 1]^2$, alors il existe un triangle de longueur inférieure à 1.
2. * Proposer un algorithme de la plus petite complexité en temps possible qui retourne un P -triangle de longueur minimale.

- Exercice 5 - Triangulation minimale. On se donne un ensemble de $n \geq 3$ points $P = \{p_1, \dots, p_n\}$ dans le plan qui forment un polygone convexe (i.e. tout P -triangle p_i, p_j, p_k ne contient aucun autre point p_ℓ). On suppose de plus que les points $p_1, \dots, p_n$ sont énumérés dans le sens des aiguilles d'un montre. Une *triangulation* de P est un ensemble de $n - 2$ P -triangles $T_1, \dots, T_{n-2}$ d'intérieurs disjoints et qui couvrent le polygone P . Par exemple, lorsque $n = 4$ il y a deux triangulations possibles, et lorsque $n = 5$ il y en a cinq. La *longueur* d'une triangulation est la somme des longueurs de ses $n - 2$ triangles.

1. Montrer que le nombre total de triangulations de P est au moins exponentiel en n (i.e. $2^{\Omega(n)}$, on ne cherchera pas à l'explicitier exactement, une borne inférieure suffira).
2. * Existe-t-il un algorithme polynomial qui calcule une triangulation de longueur minimale ?

- Exercice 6 - Code correcteur. Soit un code $C \subseteq \{0, 1\}^n$ qui corrige deux erreurs (i.e. si l'on change au plus deux bits dans $M \in C$, on peut retrouver M). On note $N = 2^n$.

1. Montrer que $|C| \leq 2^n / (1 + n + \binom{n}{2})$.
2. * Existe-t-il un algorithme polynomial en N qui retourne un code C corrigeant deux erreurs dont la taille est au moins $2^n / (1 + n + \binom{n}{2} + \binom{n}{3} + \binom{n}{4})$?

- Exercice 7 - Questions de cours.

1. Rappeler l'algorithme de Karatsouba pour la multiplication de polynômes ainsi que sa complexité.
2. Pourquoi le problème SAC-A-DOS est-il présenté parfois comme polynomial et parfois comme NP-complet ?