Examen – Complexité (SINA09B)

Durée : 2 heures. *(Barème indicatif)*
Documents : non autorisés.
Important : on vous demande de présenter proprement et clairement vos réponses.
Indications : pour toute base $b > 1$ on a $\log_b(bn) = \log_b(n) + 1$ et $\log_b(1) = 0$.

Exercice 1. *Complexité algorithme itératif (4 points)*

Voici un algorithme `algo1` prenant en entrée un entier naturel $\texttt{n} \in \mathbb{N}$.

```
algo1(n):
    r = 0
    pour i de 1 à n faire:
        pour j de i à i+1 faire:
            si j est pair alors:
                r = r + 10
            sinon:
                pour k de 1 à n faire:
                    r = r + 2
    retourner r
```

1. Donner une majoration asymptotique aussi serrée que possible de la complexité dans le pire cas de l'algorithme `algo1` ci-dessus (notation grand-O), en justifiant brièvement votre analyse.
2. Cette complexité est-elle polynomiale ou exponentielle ? Justifier.

Exercice 2. *Complexité algorithme récursif (3 points)*

Voici un algorithme `algo2` prenant en entrée un entier naturel $\texttt{n} \in \mathbb{N}$.

```
algo2(n):
    si (n == 0) ou (n == 1) alors:
        retourner vrai
    si n est impair alors:
        retourner faux
    retourner algo2(n/2)
```

1. Pour quelles valeurs de `n` l'algorithme `algo2` retourne-t-il `vrai` ? Il s'agit du pire cas.
2. Soit $C(\texttt{n})$ la complexité dans le pire cas de `algo2`. Donner une équation de récurrence pour $C(\texttt{n})$, et sa condition initiale pour $\texttt{n} = 1$.
3. Donner une solution à votre équation de récurrence, qui vérifie également sa condition initiale pour $\texttt{n} = 1$. Justifier votre solution en montrant qu'à partir de votre équation de récurrence, en remplaçant $C(\texttt{n}/2)$ par votre solution pour $\texttt{n}/2$, on obtient votre solution pour $\texttt{n}$.

Exercice 3. *Algorithmes (4 points)*

> **Clique**
> *Entrée :* Un graphe non orienté $G = (V, E)$ à n sommets, et un entier k.
> *Question :* G possède-t-il une clique de taille au moins k ?

Une *clique* est un sous-ensemble de sommets $V' \subseteq V$ tous reliés entre eux par des arêtes.
Soit `clique(G,k)` un algorithme de résolution du problème de décision **Clique**.

1. Étant donné un graphe $G = (V, E)$ à n sommets, expliquer comment utiliser au plus $\lceil \log_2(n) \rceil$ appels à l'algorithme `clique` pour calculer la taille maximum d'une clique dans G.
2. Étant donné un graphe $G = (V, E)$ à n sommets, expliquer comment utiliser au plus $\lceil \log_2(n) \rceil + n^2$ appels à l'algorithme `clique` pour trouver une clique de taille maximum dans G.

Exercice 4. *Classe* NP *(6 points)*

Hamiltonien
Entrée : Un graphe non orienté $G = (V, E)$ à n sommets.
Question : G possède-t-il un cycle hamiltonien ?

Un *cycle hamiltonien* est un cycle de longueur n parcourant une et une seule fois chaque sommet de G.

Complétion-Hamiltonienne
Entrée : Un graphe non orienté $G = (V, E)$ à n sommets, et un entier k.
Question : Est-il possible d'ajouter au plus k arêtes à G pour qu'il possède un cycle hamiltonien ?

1. Démontrer que le problème **Complétion-Hamiltonienne** appartient à la classe NP, en décrivant le certificat, l'algorithme du vérificateur (déterministe), et le temps de calcul de ce dernier.
2. Donner la définition d'un problème NP-difficile, et celle d'un problème NP-complet.
3. Démontrer que **Hamiltonien** $\leq_T^P$ **Complétion-Hamiltonienne**.
4. Sachant que **Hamiltonien** est NP-difficile, expliquer pourquoi l'on peut déduire de la question 3 que **Complétion-Hamiltonienne** est lui aussi NP-difficile.

Maintenant, on fixe k à la valeur 1 dans l'énoncé du problème (k n'est plus donné en entrée).

Complétion-par-1-Hamiltonienne
Entrée : Un graphe non orienté $G = (V, E)$ à n sommets.
Question : Est-il possible d'ajouter au plus une arête à G pour qu'il possède un cycle hamiltonien ?

5. Justifier la réduction **Complétion-par-1-Hamiltonienne** $\leq_T^P$ **Hamiltonien**.
   Attention, on vous demande de ne pas emprunter de détour, c'est-à-dire de donner et analyser un algorithme reliant directement les deux problèmes de cette réduction.
   Indication : dans une réduction Turing (aussi appelée réduction de Cook) en temps polynomial, on peut appeler plusieurs fois un algorithme de résolution de l'autre problème de décision.

Exercice 5. *Modélisation (3 points)*

2-Coloration
Entrée : Un graphe non orienté $G = (V, E)$ à n sommets.
Question : G admet-il une coloration valide de ses sommets avec seulement 2 couleurs ?

2-SAT
Entrée : Une formule propositionnelle φ sur n variables, avec m clauses qui sont toutes de taille 2.
Question : φ est-elle satisfaisable ?

1. Démontrer que le problème **2-Coloration** est dans P, en utilisant le fait que **2-SAT** est dans P.
   Indication : votre algorithme (décrit en français ou en pseudo-code) devra donc, étant donné un graphe G, construire une formule φ_G telle que la réponse à **2-SAT** pour l'instance φ_G donne la réponse à **2-Coloration** pour l'instance G. N'oubliez pas de conclure en répondant à la question.