IFT 2125 H12 **Geña Hahn**

Introduction à l'algorithmique

EXAMEN FINAL

le 28 septembre 2012

Durée : 165 minutes

Valeur: 55% de la note finale (70% si prédoc)

Directives:

- Aucune documentation n'est permise.
- Répondez sur le questionnaire , dans l'espace libre qui suit chaque question. **L'espace alloué n'est aucune indication de la longuer de la réponse! Il est souvent beaucoup trop grand.**
- Sauf indication contraire, aucun point ne sera accordé pour une réponse, correcte ou pas, si elle n'est pas accompagnée d'une justification.
- Notez la différence entre *justifier* (argument rapide et court, peut-être intuitif) et *prouver* ou *démontrer* (argument détaillé).
- Vous pouvez vous servir de résultats vus en cours, en TP ou dans des livres à condition de les énoncer précisément. Vous pouvez aussi vous servir des résulats donnés dans l'annexe.
- Pour répondre à une question, vous pouvez également vous servir de résultats énoncés dans d'autres questions dans l'examen, même si vous ne les avez pas démontrés.
- Rappel: $\mathbb{N}$ est l'ensemble des entiers non négatifs, $\mathbb{R}$ l'ensemble des réels, $\mathbb{R}^{\geq 0}$ l'ensemble des réels non négatifs et $\mathbb{R}^{>0}$ l'ensemble des réels positifs. Aussi, $\lg n = \log_2 n$ et $\ln n = \log_e n$ (où e est la base du logarithme naturel). Finalement, $\log n$ est le logarithme "générique" : la base peut être n'importe quel $b \in \mathbb{R}^{>1}$.
- Les définitions utilisées sont celle du cours et du livre Brassard/Bratley et ce sont celles-ci qui s'appliquent dans cet examen.

1. ________ /20	6. ________ /20
2. ________ /20	7. ________ /20
3. ________ /20	8. ________ /20
4. ________ /20	9. ________ /20
5. ________ /20	

Total: ________ /180

Nom: ______________________ **Code permanent:** ______________

☐ **cocher si cet examen fait partie de votre examen prédoctoral.**

1. **(20 x 1 point)** Pour chacun des énoncés suivants indiquez s'il est vrai ou non en encerclant OUI ou NON. *AUCUNE JUSTIFICATION N'EST NECESSAIRE.*

(1)	$O(n^2) = O(2^{2\log_3 n})$	OUI NON
(2)	Un algorithme dont la complexité est dans $O(n \lg n)$ est toujours préférable à un autre qui résoud le même problème en $O(n^3)$.	OUI NON
(3)	L'algorithme de Prim peut être modifié pour qu'il trouve les composantes connexes d'un graphe.	OUI NON
(4)	Pour toute fonction $f : \mathbb{N} \longrightarrow \mathbb{R}^{>0}$, $O(f) = \Omega(f)$.	OUI NON
(5)	Le principe d'optimalité dit que *si S est une solution optimale à un problème de taille n alors une solution optimale pour ce problème de taille $2n$ peut être obtenue à partir de S.*	OUI NON
(6)	L'algorithme de Floyd ne marche que si les poids des arêtes sont positifs.	OUI NON
(7)	Si $f \in O(g)$ alors $g \in \Omega(f)$.	OUI NON
(8)	$\Theta(2 + \lvert\sin(n)\rvert) = \Theta(3)$	OUI NON
(9)	La structure de données "ensembles disjoints" avec ses algorithmes "trouver" et "fusioner" permet des exécutions quasi-linéaires des suites intercalées de "trouver" et "fusioner".	OUI NON
(10)	La programmation dynamique et la méthode "diviser-pour-régner" sont interchengeables.	OUI NON
(11)	Si $f = \Theta(g)$, alors $lim_{n\to\infty} \frac{f(n)}{g(n)} = c,\ c \in \mathbb{R}$.	OUI NON
(12)	Soit $T(n) = 5T(\frac{n}{3}) + \theta(n)$, $\theta(n) \in \Theta(n^2)$. Alors $T(n) \in \Theta(n^2)$	OUI NON
(13)	Un monceau (heap) de n clés peut être construit en temps dans $O(\lg n)$.	OUI NON
(14)	Il existe une fonction $f : \mathbb{N} \longrightarrow \mathbb{R}^{>0}$ telle que $O(f) = \Omega(f)$.	OUI NON
(15)	$P \subseteq NP$ (P et NP sont les classes de complexité décrits à la fin du cours).	OUI NON
(16)	Multiplier deux matrices de dimensions $n \times n$ chaque demande au moins n^3 multiplications.	OUI NON
(17)	Tout algorithme de tri fait au moins $n \lg n$ comparaisons sur un fichier de n clés.	OUI NON
(18)	Quel que soit le problème, on peut toujours trouver un algorithme glouton pour le résoudre.	OUI NON
(19)	Trouver l'ordre optimal de multiplications chaînées de matrices demande un temps exponentiel si toutes les matrices sont de la même taille.	OUI NON
(20)	Pour minimiser le temps total d'exécution de toutes les tâches dans une queue donnée, il faut accomplir les tâches dans l'ordre de temps d'execution individuel non-décroissant.	OUI NON

2. **(10 + 10 points)** Justifiez DEUX de vos réponses à la question 1. **Votre justification ne vaut rien si elle ne comprend pas l'énoncé de ce que vous justifiez!**

   - Question ....

   - Question ....

3. **(20 points)** Prouvez que VC est NP – complet (voir l'annexe).

4. (**20 points**) Soit $f, g : \mathbb{N} \longrightarrow \mathbb{R}^{\geq 0}$ et soit $c \in \mathbb{R}^{>0}$ une constante. Prouvez rigoureusement à partir de la définition que $\Omega(cf(n)) = \Omega(f(n))$ en précisant les conditions nécessaires s'il y a lieu (les conventions de l'annexe s'appliquent et il n'est pas nécessaire de les répéter). Est-ce que $\Omega(f(cn)) = \Omega(f(n)$ également? Prouvez-le ou donner un contre-exemple.

5. (**20 points**) Donnez un algorithme qui fait *moins que* $\frac{3}{2}n$ comparaisons pour trouver la plus grande et la plus petite clé dans un fichier de n clés. Prouvez votre réponse.

6. **(20 points)** Rappelez-vous du problème des billes: vous en avez n, toutes pareilles d'aspect et de poids, sauf une qui est plus lourde ou plus légère que les autres. Vous avez également une balance qui permet de comparer les poids de deux tas de billes. Combien de pesées faut-il au minimum pour trouver la bille différente et sa différence (i.e. si elle est plus lourde ou plus légère que les autres). Prouvez votre réponse.

7. **(20 points)** Soit $G = (V, A)$ un graphe orienté. Ecriver un algorithme – en pseudo code et français, pas du code – basé sur celui de Floyd qui trouvera la femeture transitive de G (c'est l'agorithme de Warshall que vous allez écrire).

8. (**20 points**) Soit

$$T(n) = 4T(\lfloor \frac{n}{2} \rfloor) - 4T(\lfloor \frac{n}{4} \rfloor) + 1$$

quand $n > 2$ et $T(0) = T(1) = 0$, $T(2) = 3$. Trouvez l'ordre exact de $T(n)$ lorsque n est une puissance de 2, $n > 2$. Que peut on conclure de l'ordre exact de $T(n)$ pour $n \in \mathbb{N}$? N'oubliez pas de justifier.

9. (**20 points**) Qu'est ce qui ne va pas dans la preuve suivante?

Le problème $NP-$ complet $VC(G, k)$ (voir l'annexe) peut être résolu en temps polynomial par l'algorithme suivant :

- générer successivement les parties a k éléments de V;
- pour chacun des ensembles généré, tester si au moins une extrémité de chaque arête est dans l'ensemble;
- si oui, arrêter en disant OUI, sinon générer l'ensemble suivant;
- si toutes les parties à k éléments ont été générées sans que l'on dise OUI, dire NON.

Au pire, on génère tous les sous-ensembles à k éléments. Il y en a $\binom{n}{k} \in O(n^k)$. Les tests de la deuxième étape prennent au pire $ck|E| \leq ckn^2$ pour une constante $c \in \mathbb{R}^{>0}$. Donc en tout, on reste dans $O(n^k)$.

ANNEXE

Rappels: $\mathbb{N}$ est l'ensemble des entiers non négatifs, $\mathbb{R}$ l'ensemble des réels, $\mathbb{R}^{\geq 0}$ l'ensemble des réels non négatifs et $\mathbb{R}^{>0}$ l'ensemble des réels positifs. Aussi, $\lg n = \log_2 n$ et $\ln n = \log_e n$ (où e est la base du logarithme naturel). Finalement, $\log n$ est le logarithme "générique" : la base peut être n'importe quel $b \in \mathbb{R}^{>1}$. Les fonctions utilisées sont de $\mathbb{N}$ dans $\mathbb{R}^{>0}$ sauf indication contraire.

Théorème. Soit $T : \mathbb{N} \longrightarrow \mathbb{R}^{>0}$ telle que

$$T(n) = lT(\frac{n}{b}) + f(n)$$

pour $n \geq n_0$, des constantes $l, b \in \mathbb{N}$, $l \geq 1, b \geq 2$, et $f(n) \in \Theta(n^k)$. Alors

$$T(n) \in \begin{cases} \Theta(n^k) & \text{si } l < b^k \\ \Theta(n^k \log n) & \text{si } l = b^k \\ \Theta(n^{\log_b l}) & \text{si } l > b^k \end{cases}$$

à condition que T soit événtuellement non-décroissante.

Vous pouvez supposer les problèmes suivants *NP*-complets (sauf si c'est à prouver!).

$STABLE(G, k)$ est le problème :
Données : un graphe $G = (V, E)$, un naturel $k > 0$.
Question : Est-ce que G contient un ensemble S, $|S| \geq k$, tel que $uv \notin E$ pour tout $u, v \in S$?

$CLIQUE(G, k)$ est le problème :
Données : un graphe $G = (V, E)$, un naturel $k > 0$.
Question : Est-ce que G contient un ensemble S, $|S| \geq k$, tel que $uv \in E$ pour tout $u, v \in S$?

$k - COL(G)$ est le problème :
Données : un graphe $G = (V, E)$.
Question : Est-ce que V peut être colorié avec k couleurs par une fonction $c : V \longrightarrow \{0, 1, \ldots, k-1\}$ de façon à ce que si $uv \in E$, alors $c(u) \neq c(v)$?

$VC(G, k)$ est le problème :
Données : un graphe $G = (V, E)$, un naturel $k > 0$.
Question : Est-ce que G contient un ensemble S, $|S| \leq k$, tel que si $uv \in E$ alors $\{u, v\} \cap S \neq \emptyset$?