M1 Informatique
X1II030

Université de Nantes
2017/2018

Examen
Complexité et Algorithmes

Janvier 2018

Durée : 1h30. Documents de CM/TD autorisés. Il est important de toujours justifier *vos réponses. Le barème est indicatif. De plus, dans tout l'énoncé, sauf mention contraire :*

— *Les complexités sont demandées sous la forme "notation de Landau"*
— *Les tableaux et matrices sont indicés à partir de 1*

Exercice 1 (Questions diverses – 5 points)

On se donne la procédure X suivante.

```
function X(int n): int
  Début
    int k:=1;
    int i:=0;

    Tant que k<=n faire
      i:=i+1;
      k:=3*k;
    FinTantQue

    return i;
  Fin
```

1. Quelle est la complexité au pire de X en fonction de n ?

On rappelle qu'une coloration *propre* des sommets d'un graphe est une coloration dans laquelle deux sommets voisins ne peuvent pas avoir la même couleur.

2. Quel est le nombre minimum de couleurs nécessaires pour colorier proprement un arbre ?

On suppose que l'on a étudié trois problèmes d'optimisation Pr_1, Pr_2 et Pr_3, qui ont chacun une taille des données égale à n. On suppose qu'on a trouvé pour chacun un algorithme qui approxime l'optimal avec un ratio $(1+\varepsilon)$ (pour n'importe quel $\varepsilon > 0$). Voici les complexités au pire trouvées pour ces trois algorithmes :

(a) $O(n^{\frac{2}{\varepsilon}})$ pour Pr_1 ;
(b) $O(1.5^{n+\frac{1}{\varepsilon}})$ pour Pr_2 ;
(c) $O(\frac{n^3}{\varepsilon^4})$ pour Pr_3.

3. Dans quelle classe de complexité se situe Pr_1 ? Même question pour Pr_2 puis pour Pr_3.
4. Un problème APX-dur peut-il avoir un ratio d'approximation de 1.01 ?
5. On a vu en CM que le problème KNAPSACK est dans FPTAS. Expliquer en quelques phrases les idées principales de la preuve de ce résultat.

Exercice 2 (Étude de MIN-VC3 – 8 points)
On rappelle ci-dessous la définition du problème d'optimisation MIN-VERTEX-COVER (ou MIN-VC).

MIN-VERTEX-COVER (MIN-VC)
Instance : Un graphe $G = (V, E)$
Solution : Un vertex cover $V' \subseteq V$ de G
Mesure : Le nombre de sommets de V'

On appelle MIN-VC3 (respectivement MIN-VC4) le problème MIN-VC restreint aux instances "graphes de degré maximum 3" (respectivement "graphes de degré maximum 4").

1. Montrer que dans un graphe G_Δ de degré maximum Δ et à n sommets, tout vertex cover de G_Δ est de taille au moins égale à $\frac{n}{\Delta+1}$.

Partant de n'importe quel graphe G_4 de degré maximum 4, on le transforme en un graphe G_3 de degré maximum 3 en transformant *tous les sommets de degré 4* comme dans la Figure 1 (les autres sommets restent inchangés).

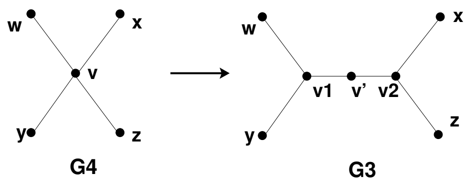


FIGURE 1 – Transformation d'un sommet v de degré 4 dans G_4, en v_1, v' et v_2 dans G_3

On appelle V_4 l'ensemble des sommets de G_4 de degré *exactement* 4, et $n_4 = |V_4|$ le nombre de ces sommets.

On part d'un vertex cover de G_4 qu'on appelle $\mathcal{V}_4$, et on souhaite construire à partir de lui un vertex cover $\mathcal{V}_3$ de G_3. Pour cela, on regarde chaque sommet v de G_4. Si v n'est pas dans V_4, on le met dans $\mathcal{V}_3$ uniquement s'il est dans $\mathcal{V}_4$. Si v est dans V_4, il y a deux cas possibles (voir Questions 2. et 3.) : v est dans $\mathcal{V}_4$, ou il n'y est pas.

2. Supposons que v est dans V_4 et aussi dans $\mathcal{V}_4$. Indiquer quels sont les deux sommets à ajouter à $\mathcal{V}_3$ pour obtenir un vertex-cover de G_3. Justifier.
3. Supposons que v est dans V_4 mais pas dans $\mathcal{V}_4$. Indiquer quel est le sommet à ajouter à $\mathcal{V}_3$ pour obtenir un vertex-cover de G_3. Justifier.
4. Montrer que l'ensemble $\mathcal{V}_3$ ainsi obtenu est un vertex cover de G_3.
5. Soit vc_3 la taille de $\mathcal{V}_3$ et vc_4 la taille de $\mathcal{V}_4$. Montrer que $vc_3 = vc_4 + n_4$.

On admettra que la réciproque est vraie, c'est-à-dire que s'il existe un vertex cover $\mathcal{V}'_3$ de taille vc'_3 dans G_3, alors on peut construire à partir de lui un vertex cover $\mathcal{V}'_4$ de taille $vc'_4 = vc'_3 - n_4$.

On appelle maintenant $opt(G_3)$ (resp. $opt(G_4)$) la taille du plus petit vertex cover de G_3 (resp. G_4).

6. Montrer que $opt(G_3) = opt(G_4) + n_4$.

Nous allons maintenant supposer que MIN-VC3 est dans `PTAS`, c'est-à-dire qu'il existe un algorithme $\mathcal{A}_3$ qui, pour tout graphe G_3 de degré maximum 3, détermine un vertex cover de taille $a_3(G_3) \leq (1+\varepsilon) \cdot opt(G_3)$.

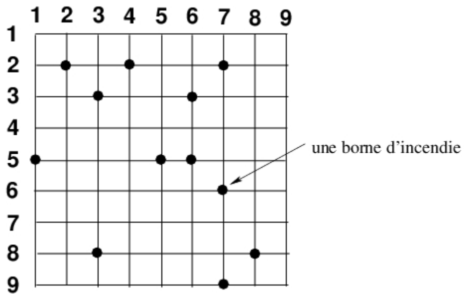
Partant d'un graphe G_4 de degré maximum 4, on propose l'algorithme suivant, que l'on appellera $\mathcal{A}_4$:
(a) transformer G_4 en G_3 comme décrit à la Figure 1
(b) appeler l'algorithme $\mathcal{A}_3$
(c) transformer le vertex cover obtenu par $\mathcal{A}_3$ pour G_3 en un vertex cover pour G_4, comme décrit ci-dessus

On appelle $a_3(G_3)$ (respectivement $a_4(G_4)$) la taille du vertex cover obtenu par $\mathcal{A}_3$ sur G_3 (resp. $\mathcal{A}_4$ sur G_4).

7. Montrer que $a_4(G_4) = a_3(G_3) - n_4$, et en déduire que $a_4(G_4) \leq (1+\varepsilon) \cdot opt(G_4) + \varepsilon \cdot n_4$.
8. En utilisant la Question 1., montrer que $a_4(G_4) \leq (1+6\varepsilon) \cdot opt(G_4)$.
9. Que signifie le résultat de la question précédente concernant le problème MIN-VC4 ?
10. Sachant que le problème MIN-VC4 est `APX`-dur, que peut-on en déduire pour MIN-VC3 ? (on attend ici une réponse précise et argumentée)

Exercice 3 (Bornes d'incendie à Manhattan – 7 points)

Dans un quartier de Manhattan, où les rues forment donc un quadrillage, la municipalité impose de mettre des bornes d'incendie à des carrefours déterminés. On propose de modéliser Manhattan par une grille à n lignes et n colonnes, où chaque borne d'incendie est représentée par un point de la grille : si une borne existe au carrefour situé à l'intersection de la ligne i et de la colonne j, alors le point qui la représente est déterminé par ses coordonnées (i, j). On suppose qu'il y a p bornes d'incendie.



Manhattan et ses bornes d'incendie - Ici, $n = 9$ et $p = 12$.

On veut déterminer la meilleure structure de données SD pour stocker les coordonnées des bornes d'incendie, sachant qu'on doit pouvoir effectuer les trois requêtes suivantes :

(a) ajouter une borne d'incendie sur un point (i, j) donné (Insertion d'un élément dans SD).
(b) retirer une borne d'incendie d'un point (i, j) donné (Suppression d'un élément dans SD).
(c) déterminer si une borne d'incendie existe sur un point (i, j) donné (Recherche d'un élément dans SD).

On vous donne le choix entre deux structures de données :

— Structure S_1 : une matrice M à n lignes et n colonnes, dont chaque case représente une borne d'incendie.
— Structure S_2 : une liste simplement chaînée L, dont chaque élément représente une borne d'incendie. Les éléments sont sans ordre particulier dans L.

1. Représenter le contenu de la structure S_1 dans le cas où les bornes d'incendie sont positionnées comme dans la figure ci-dessus.
2. Indiquer en français et de manière précise ce que contient, en général, la structure S_1.
3. Représenter le contenu de la structure S_2 dans le cas où les bornes d'incendie sont positionnées comme dans la figure ci-dessus.
4. Indiquer en français et de manière précise ce que contient, en général, la structure S_2.
5. Remplir le tableau ci-dessous. Dans chaque case, on demande la complexité en temps et au pire pour la requête indiquée, dans la structure indiquée. Pour chacune des 6 réponses, justifier en quelques phrases.

	Structure S_1	Structure S_2
Insertion		
Suppression		
Recherche		

6. Écrire en pseudo-code l'algorithme d'Insertion pour S_1.
7. Écrire en pseudo-code l'algorithme de Recherche pour S_2.
8. Quelle est la complexité en espace de la structure S_1 ? Même question pour la structure S_2.
9. En tenant compte de la complexité en espace de chacune des structures et de la complexité en temps de chacune des requêtes, quelle est la meilleure structure de données à utiliser si on sait que $p = O(1)$?
10. Même question avec $p = \Theta(n)$, puis avec $p = \Theta(n^2)$.