Complexité

INF303 *TD10*

Exercice 1. Pour chacune des fonctions ci-dessous, donner une borne supérieure asymptotique la plus simple possible en utilisant la notation $\mathcal{O}(\ldots)$.

1. $f(n) = 2n^2 + 3n + 3$
2. $f(n) = -n + 3n \log n$
3. $f(n) = -1 + 4n - 1/n$
4. $f(n) = 45$
5. $f(n) = 5n^{1.02} + 2^n$
6. $f(n) = 3n(n^3 + 5n^6)$
7. $f(n) = n + 3n^2(n^{0.6} \log^2 n + 6n^{0.7})$
8. $f(n) = n + 2^{\log n}$
9. $f(n) = 10n \log(10n)$
10. $f(n) = 2^{\log(10n)}$
11. $f(n) = 2^{\log^2(n)}$

Exercice 2.

Question 1. Écrire en pseudo-code un algorithme avec les spécifications suivantes :

Entrée: un entier n, et un graphe G à n sommets, donné sous la forme de sa matrice d'adjacence A.

Sortie: le degré moyen de G.

Question 2. Quelle sa complexité en fonction de n ? On considérera les opérations suivantes comme élémentaires :

- Accès à une entrée d'un tableau
- Écriture d'une valeur dans une entrée d'un tableau
- Addition, soustraction, multiplication ou division entre deux entiers
- Affectation d'une valeur à une variable
- Comparaison de deux entiers.

Exercice 3.

Question 1. Écrire en pseudo-code un algorithme avec les spécifications suivantes :

Entrée: deux entiers n et m, et un graphe G à n sommets, donné sous la forme de sa matrice d'incidence I de taille $n \times m$.

Sortie: le couple $(i, d(i))$ où i est le sommet de degré maximum.

Question 2. Quelle est sa complexité en fonction de n et m ? On considérera les mêmes opérations élémentaires que dans l'exercice précédent.

Il existe une troisième façon de représenter un graphe en mémoire (en plus de la matrice d'adjacence et la matrice d'incidence) : il s'agit de la représentation par listes d'adjacence. Dans ce cas, on n'a pas de matrice mais un ensemble L de n listes, une pour chaque sommet : la liste $L(i)$ contient la liste des voisins du sommet i. Par exemple, la représentation par

listes d'adjacence du graphe complet à 3 sommets est : $L(1) = \{2, 3\}$; $L(2) = \{1, 3\}$; $L(3) = \{1, 2\}$.

Question 3. On suppose maintenant que le graphe est donné en entrée sous forme de *listes d'adjacence*. Écrivez un algorithme en pseudo-code qui calcule la même chose que précédemment, à savoir le sommet de degré maximum et son degré.

Question 4. Prouver que la complexité de cet algorithme est de $\mathcal{O}(n+m)$. On considérera comme opérations élémentaires toutes celles évoquées ci-dessus, plus celles spécifiques aux listes, à savoir :

- Supprimer la tête d'une liste
- Connaître la valeur de la tête d'une liste
- Ajouter un élément à la tête d'une liste
- Tester si une liste est vide

Question 5. Comparer la complexité de cet algorithme avec celle de l'algorithme précédent, dans les trois cas suivant :

- $m = \mathcal{O}(1)$
- $m = \mathcal{O}(n)$
- $m = \mathcal{O}(n^2)$