TD Complexité – Calcul de complexité

K. Bertet – J.L. Guillaume – A. Huchet

1. Calcul de complexités

Exercice 1. On considère les nombres d'opérations élémentaires suivants de plusieurs algorithmes. En déduire leur complexité sachant que la taille de l'entrée est n et que k est une constante.

a. $6n^3 + 10n^2 + 5n + 2$
b. $3\log n + 5$
c. $6n^3 + 10\log n + 7$
d. $100k + 2$
e. $5n + 20\log n$
f. $2^{3n} + 10n^2 + 5\log n$

Exercice 2. Déterminer la complexité f(n) des deux algorithmes suivants, en spécifiant des valeurs de c n_0 et *f(n)* (valeurs qui ne sont pas uniques) qui permettent de vérifier que, pour tout $n \geq n_0$, on a bien $T(n) \leq c\, f(n)$:

a. $T_1(n) = 9n^2$
b. $T_2(n) = 100\, n + 96$

2. Fibonacci

« Le problème de Fibonacci est à l'origine de la suite dont le n-ième terme correspond au nombre de paires de lapins au n-ième mois. Dans cette population (idéale), on suppose que :

- au (début du) premier mois, il y a juste une paire de lapereaux ;
- les lapereaux ne procréent qu'à partir du (début du) troisième mois ;
- chaque (début de) mois, toute paire susceptible de procréer engendre effectivement une nouvelle paire de lapereaux ;
- les lapins ne meurent jamais (donc la suite de Fibonacci est croissante). »

La suite de Fibonacci se définit récursivement par :

$$F(1) = 1\,; F(2) = 1\,; F(n) = F(n-1) + F(n-2)$$

Exercice 3. Les trois algorithmes suivants prennent en entrée un entier n, et calculent et renvoient le n-ième nombre de Fibonacci.

a. Appliquer ces algorithmes pour calculer le 10ème nombre de Fibonacci.
b. Calculer la complexité de ces trois algorithmes dans l'ordre Fibo2, Fibo3, Fibo1.
c. En déduire quel est le meilleur algorithme

Fibo1 (n)
Entrée : un entier n
Résultat : le n-ème nombre de Fibonacci

```
si n ≤ 2 retourner 1
retourner Fibo1(n-1)+Fibo1(n-2)
```

Fibo2 (n)
Entrée : un entier n
Résultat : le n-ème nombre de Fibonacci

```
a=1
b=1
pour i allant de 3 à n
    b = a+b
    a = b-a
retourner b
```

Fibo3 (n)
Entrée : un entier n
Résultat : le n-ème nombre de Fibonacci
Calculer m = Puissance ($\begin{pmatrix}1 & 1\\ 1 & 0\end{pmatrix}$, n-1)
Retourner m[1,1]

Puissance(M,n)
Entrée : une matrice M, un entier n
Entrée : M à la puissance n

Si n=1 alors retourner matrice
Si n est pair alors
    Retourner Puissance(M,n/2) x Puissance(M,n/2)
Si n est impair alors
    Retourner Puissance(M,n/2) x Puissance(M,n/2) x M

3. Equations récursives

Exercice 4. Résoudre les équations récursives suivantes et en déduire la complexité de T(n) :

a. $T(n) = 2T\left(\frac{n}{2}\right) + n^2$; $T(1) = 1$
b. $T(n) = 8T\left(\frac{n}{2}\right) + n^2$; $T(1) = 1$

Exercice 5. Donner l'équation récursive du calcul logarithmique du nème nombre de Fibonnacci selon que :

- Puissance (M, n/2) est appelé 2 fois, implémentation directe de l'algo
- Puissance (M, n/2) est appelé une seule fois, le résultat stocké dans une variable

Exercice 6. Donner l'équation récursive pour la recherche dichotomique dans un tableau trié, ainsi que sa résolution.

4. Choix d'un algorithme et d'une structure de donnée

Exercice 7. On trouve deux algorithmes pour résoudre le même problème pour un graphe de n noeuds et m arcs. Lequel choisir ?

- L'algorithme avec une complexité de $O(n^2+m)$
- L'algorithme avec une complexité de $O(n^2.\log m)$

Exercice 8. Indiquer dans chacun des cas suivant quel algorithme choisir pour traiter un graphe de n nœuds et m arcs, selon que le graphe soit dense ou non :

- Le premier algorithme a une complexité en O(n + m) et le second en $O(n^2)$?
- Le premier algorithme a une complexité en $O(2^n)$ et le second en $O(2^m)$?
- Le premier algorithme a une complexité en O(n + log m) et le second en O(m) ?

Exercice 9. L'algorithme de Prim calcule un arbre couvrant minimum d'un graphe non orienté et valué. Cet algorithme trie les arêtes par ordre croissant de leur poids, puis les arêtes sont parcourues selon l'ordre du tri. Une arête sera sélectionnée, et donc ajoutée à l'arbre, à condition qu'elle ne crée pas de cycle avec l'ensemble des arêtes déjà ajoutées. Les arêtes à traiter sont stockées dans une structure de données.

- Quels sont les traitements à réaliser sur cette structure de données ?
- Quelle structure de données est la plus appropriée pour ces traitements ? Justifier votre choix.

Exercice 10. L'algorithme de Dijkstra calcule les plus courts chemins dans un graphe orienté et valué. Cet algorithme utilise une structure de données pour stocker une valeur associée à chaque nœud du graphe qu'il reste à traiter. A chaque itération, cette structure de données est utilisée pour récupérer le noeud de valeur minimale, puis les valeurs de certaines autres noeuds seront modifiées.

- Quels sont les traitements à réaliser sur cette structure de données à chaque itération ?
- Quelle structure de données est la plus appropriée pour ces traitements ? Justifier votre choix.