Analyse d'Algorithmes et Programmation

Contrôle Continu 1

Christina Boura et Yann ROTELLA
{christina.boura, yann.rotella}@uvsq.fr

17 mars 2021

1 Complexité assymptotique

Exercice 1 Calculer en fonction de n la complexité asymptotique $\Theta(n)$ des trois programmes Python suivants dans le cas le plus défavorable. Expliquez comment vous avez obtenu votre réponse.

(a)
```
for i in range(10) :
    for j in range(n) :
        print("Bonjour")
```

(b)
```
for i in range(n*n) :
    for j in range(i) :
        print("Bonjour")
```

(c)
```
for i in range(n) :
    t = 1
    while t < n :
        print("Bonjour")
        t = t * 2
```

Exercice 2 Montrer que

$$\max(f(n), g(n)) = \Theta(f(n) + g(n)).$$

Exercice 3 On suppose que le temps d'exécution d'un algorithme de tri est donné par l'équation récursive suivante :

$$T(1) = c,$$
$$T(n) = 3T(n/3) + 2cn \text{ pour } n > 1,$$

où c est une constante positive et n est de la forme $n = 3^m$. Résolvez cette récurrence afin de trouver une formule explicite pour $T(n)$ et donnez la complexité $\mathcal{O}(n)$ de l'algorithme.

2 Tri rapide

Le tri rapide est un algorithme de tri qui utilise, comme le tri fusion, le principe de programmation dit "diviser pour régner". Pour cela, le tri rapide fait appel à une procédure PARTITIONER décrite ci-dessous.

Soit n la taille d'un tableau d'entiers T. Pour tout $0 \leq i \leq j \leq n$, on note $T[i : j]$ le sous-tableau composé des éléments à la position i à j inclus. $T[i]$ dénote l'élement du tableau T à la position i.

Algorithm 1 PARTITIONER(T, n, m, p)

```
ECHANGER(T, m, p) (mettre T[p] à la fin du sous-tableau)
j = p
for i = n ... m − 1 do
    if T[i] ≤ T[m] then
        ECHANGER(T, i, j)
        j = j + 1
    end if
end for
ECHANGER(T, j, m)
return j
```

Algorithm 2 TRIRAPIDE(T, n, m)

```
if n < m then
    p = n
    p = PARTITIONER(T, n, m, p)
    TRIRAPIDE(T, n, p − 1)
    TRIRAPIDE(T, p + 1, m)
end if
```

1. Montrez que l'algorithme TRIRAPIDE termine.
2. Montrez par récurrence que pour tout $n \leq i \leq m-1$, à la fin de la $i-n+1$-ième étape de la boucle de PARTITIONER, les éléments du sous-tableau $T[n : j-1]$ sont tous inférieurs à $T[m]$ et que les éléments du sous-tableau $T[j : i]$ sont tous supérieurs à $T[m]$.
3. En utilisant une récurrence forte et la question précédente, déduire que TRIRAPIDE(T, n, m) tri le tableau $T[n : m]$ où $t = m - n + 1$ est la tailles du tableau.
4. Donnez le nombre de comparaisons de l'algorithme PARTITIONER en fonction de la taille du tableau (on négligera par simplicité toutes les autres opérations dans cette question et dans les suivantes).
5. La complexité du tri rapide dépend principalement du pivot (p) choisi à chaque appel récursif. Quel est le pire cas dans l'algorithme TRIRAPIDE ?
6. On note $C_w(t)$ le coût de l'algorithme TRIRAPIDE dans le pire cas où t désigne la taille du tableau ($t = m - n + 1$). Donnez la relation de récurrence de $C_w(t)$ en fonction de $C_w(t)$.
7. On admet maintenant que le pivot p à chaque étape suit une loi uniforme. En d'autres termes, on admet ici que la valeur p en sortie de PARTITIONER est unifromément distribuée parmis toutes les valeurs de p possibles. On note $C_a(t)$ le coût moyen du tri rapide où t est la taille du tableau à trier. Montrez que pour tout $t > 0$,

$$C_a(t) = t - 1 + \frac{2}{t} \sum_{k=0}^{t-1} C_a(k) \,.$$

8. Montrez [1] que pour tout t, on a

$$\frac{C_a(t+1)}{t+2} - \frac{C_a(t)}{t+1} = \frac{2t}{(t+1)(t+2)} \,.$$

9. Montrez que pour tout t, $C_a(t) \leq 2(t+1) \sum_{k=0}^{t-1} \frac{1}{k+1}$.
10. Donnez alors les classes de complexité du pire cas et du cas moyen pour le tri rapide.

1. indice, calculez $(t+1)C_a(t+1) - tC_a(t)$