Contrôle

Exercice 1 (10 points)

1. Prouver que $\sum_{k=0}^{n} 3^k$ est en $O(3^n)$

Preuve par récurrence :

Pour n=0 on a $\sum_{k=0}^{0} 3^k = 3^0 = 1 \approx est\ en\ O(3^0) = O(1)\ vrai\ pour\ n = 0$

On suppose que la complexité est vraie pour n et on montre qu'elle est vraie pour n+1

On montre que $\sum_{k=0}^{n+1} 3^k\ est\ en\ O(3^{n+1})$

$$\sum_{k=0}^{n+1} 3^k = \sum_{k=0}^{n} 3^k + 3^{n+1}\ \ est\ en\ O(3^n + 3^{n+1}) = O(\max(3^n, 3^{n+1})) = O(3^{n+1})$$

Ou bien

$$0 \leq \sum_{k=0}^{n} 3^k \leq c.3^n$$

Cas de base : pour n=0 on a :

$0 \leq \sum_{k=0}^{0} 3^k \leq c.3^0$ vraie pour $c \geq 1$

On suppose l'inégalité est vraie pour l'ordre n et on montre qu'elle est vraie pour n+1

$$0 \leq \sum_{k=0}^{n+1} 3^k = \sum_{k=0}^{n} 3^k + 3^{n+1} \leq c.3^n + 3^{n+1}$$

Or $c.3^n + 3^{n+1} = c.3^{n+1}\left(\frac{1}{3} + \frac{1}{c}\right) \leq c.3^{n+1}\ si\ \frac{1}{3} + \frac{1}{c} \leq 1\ ce\ qui\ est\ vrai\ si\ c \geq \frac{3}{2}\ C.Q.F.D$

2. Classer les fonctions suivantes par ordre croissant de leur valeur asymptotique:

$\log(n^3)$, 2^{n^2}, $n^3 \log n$, $\sqrt{n!}$, $2\sqrt{n}$

Classement : $\log(n^3) \subseteq 2\sqrt{n} \subseteq n^3 \log n \subseteq \sqrt{n!} \subseteq 2^{n^2}$

3. Donnez la complexité des programmes suivants. Vous donnerez une borne supérieure avec un O dans un premier temps, puis vous affinerez votre calcul en utilisant la notation Θ.

Algorithme 1

```
1  pour i← 1 à n faire
2     pour j ← 1 à i faire
3        x←x+3
```

Algorithme 2

```
1  pour i ← 5 à n-5 faire
2     pour j ← i-5 à i+5 faire
3        x ←x+3
```

Algorithme 3

```
pour i ← 1 à n faire
    pour j ← 1 à i faire
        pour k ← 1 à j faire
                x ← x+a
```

Algorithme 4

```
for (i = n; i>1; i = i/2)
        for (j=0; j<i; j++)
                x = x+a
```

1. O(n^2) n est une borne sup du nombre de tours de boucles
$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} \backsim O(n^2)$$
2. O(n) la boucle interne a toujours un nombre constant d'opérations (11 itérations). Les variations de longueur sur la boucle externe étant constantes, on les ignore.

3. O(n^3) car les boucles internes font au plus n itérations

$$\sum_{i=1}^{n}\sum_{j=1}^{i}\sum_{k=1}^{j} = \sum_{i=1}^{n}\sum_{j=1}^{i} j = \sum_{i=1}^{n}\frac{i(i+1)}{2} = \frac{1}{2}(\sum_{i=1}^{i} i^2 + \sum_{i=1}^{n} i) = \frac{1}{2}(\frac{(2n+1)(n+1)n}{6} + \frac{n(n+1)}{2}) \backsim \Theta(n^3)$$

4. *n log n* une fois qu'on sait que diviser le travail à chaque itération introduit du log dans les coûts. Mais en fait, ce code est dans Θ(n) car la boucle interne s'arrête à i et pas à n. Donc, si on compte les itérations faites par j dedans on trouve :1+2+4+8+…+n.
Autrement dit si on pose $n=2^k$,

$$\sum_{p=1}^{k}\sum_{j=1}^{2^k} 1 = \sum_{p=1}^{k} 2^p = \sum_{p=0}^{k} 2^p - 2^0 = 2^{k+1} - 2 = 2n - 2 \ \epsilon\theta(n)$$