Master RSD (2011-2012)

Corrigé du contrôle N°2 du module Complexité

Exercice1 :

a- $3n^3 + 2^{n-2}$

Nous devons trouver les constantes c, $n_0 > 0$ telles que :

$0 \leq 3n^3 + 2^{n-2} \leq c.\ 2^n$ *pour tout* $n \geq n_0$.

$$0 \leq \frac{3n^3 + 2^{n-2}}{2^n} \leq \frac{c.2^n}{2^n}$$

$$0 \leq \frac{3n^3}{2^n} + \frac{2^{n-2}}{2^n} \leq c$$

$$0 \leq 3\frac{n^3}{2^n} + \frac{1}{4} \leq c$$

$$\frac{n^3}{2^n} \leq 1 \quad \forall n \geq 10$$

$\Longrightarrow c = \frac{13}{4}$ *et* $n_0=10$ *donc la complexité=*$O(2^n)$

b- $4n^3 + 12;$

Nous devons trouver les constantes c, $n_0 > 0$ telles que :

$0 \leq 4n^3 + 12 \leq c.\ n^3$ *pour tout* $n \geq n_0$.

$\Longrightarrow c = 6$ *et* $n_0=2$ *donc la complexité=*$O(n^3)$

c- $n^2 log(5n^4);$

Nous devons trouver les constantes c, $n_0 > 0$ telles que :

$0 \leq n^2 log(5n^4) \leq c.\ g(n)$ *pour tout* $n \geq n_0$.

$n^2 log(5n^4) = n^2 x\ (log5 + log\ n^4)$

$\qquad = n^2 x\ log5 + n^2 x\ 4log\ n$

en appliquant la propriété de la somme on a

$0 \leq n^2 log(5n^4) \leq c.\ n^2 x\ log\ n$

avec $C=7$ *et* $n_0=2$ *donc la complexité=*$O(n^2\ log\ n)$

d- $\frac{1}{2}n^2 - 10n - 60;$

Nous devons trouver les constantes *c*, n_0 > 0 telles que :

$0 \leq \frac{1}{2}n^2 - 10n - 60 \leq c.\ n^2$ *pour tout* $n \geq n_0$.

$\Rightarrow$ *la complexité*=$O(n^2)$ *avec* $C=\frac{1}{2}$ *et* $n_0=1$

e- $\frac{1}{n}$

Nous devons trouver les constantes *c*, n_0 > 0 telles que :

f- $0 \leq \frac{1}{n} \leq c.\ \log n$ *pour tout* $n \geq n_0$.

$\Rightarrow$ *la complexité* = $O(\log n)$ *avec* $c=1$ *et* $n_0=2$

Classement : Dans l'ordre croissant

$\log n \rightarrow n^2 \rightarrow n^2 \log n \rightarrow n^3 \rightarrow 2^n$

Exercice 2 :

a- fonction CarreIter(A :tableau[1..n] de caractères ; n :entier) :entier ;

var i, j, m :entier ;

debut

si n mod 2$\neq$0 alors retourner 0

sinon

i$\leftarrow$ 1; m$\leftarrow\lfloor n/2 \rfloor$; j$\leftarrow$m+1 ;

tantque (i<=m) et (A[i]=A[j]) faire

i$\leftarrow$i+1; j$\leftarrow$j+1;

fait;

si (i>m) alors retourner 1

sinon retourner 0 ;

fsi ;

fsi ;

fin ;

b- ***Invariant de boucle*** : On pose ***m***=$\lfloor n/2 \rfloor$

« A la fin de la $k^{ième}$ itération de la boucle ***tantque*** tous les caractères ***A[1 ..i]*** sont égaux à ***A[m+1..j]*** puis on incrémente **i** et on incrémente **j.** La fonction s'arrête si $i_{k+1}>m$ et retourne **1** ou bien elle s'arrête si $A[i_{k+1}]\neq A[j_{k+1}]$ et retourne **0** »

Ceci à condition que la taille du mot soit paire, sinon il ne peut pas être carré.

c- On montre la validité de l'invariant par récurrence sur ***i.***

L'invariant est facilement vérifié pour la **1ère** itération on a i=1 et j=m si A[1]=A[m+1] alors on incrémente $i_2=i_1+1$ et on incrémente $j_2=j_1+1$, si i>m alors la fonction se termine et le mot est carré sinon si A[1] $\neq$ A[m+1] la fonction se termine et le mot n'est pas carré.

On suppose que l'invariant est vrai à la fin de l'itération **i** et on montre qu'il est vrai, s'il y en a, pour l'itération **i+1**, c'est que $\forall$ $k \in$ {1..i} et $l \in$ {m+1..j} A[k]=A[l]. Si $k_{i+1}< l_{i+1}$ et A[k_{i+1}]=A[l_{i+1}] on incrémente k et on incrémente l

sinon si $k_{i+1} > l_{i+1}$ alors la fonction se termine et le mot est carré sinon si A[k_{i+1}] ≠A[l_{i+1}] alors la fonction se termine et le mot n'est pas carré. L'invariant à la fin de l'itération i+1 est donc vérifié.

Au pire des cas le mot est carré et on aura fait ***m***=$\lfloor n/2 \rfloor$ comparaison, donc la complexité au pire des cas est de ***O(n/2)*** et dans le meilleur des cas la fonction s'arrête quand A[1] ≠ A[m+1] donc la complexité est en O(1).

d- fonction CarreRec(A :tableau[1..n] de caractères; n, i, j :entier) :entier ;

debut (1er appel i=1 et j=m+1)

si (i>m) alors retourner 1

sinon si (A[i]≠A[j]) alors retourner 0

sinon retourner(PalindromeRec(A, n, i+1, j+1);

fsi ;

fsi ;

fin ;

<u>Invariant :</u>

« A la fin de la $k^{ième}$ itération si i>m la fonction se termine et le mot est carré ou bien A[i] ≠ A[j] et la fonction se termine et le mot n'est pas carre sinon on fait un appel récursif en incrémentant i et en incrémentant j »