

COMPLEXITÉ DES ALGORITHMES

SÉRIE D'EXERCICES N°2

EXERCICE 1

Exercice 1 :

- Soient A et B deux matrices et x un vecteur.
 1. Ecrire l'algorithme de Somme des éléments du vecteur x de taille n
 - Donner sa complexité, Est-il de coût minimal ?
 2. Ecrire l'algorithme de produit de la matrice A par le vecteur x.
 - Etudier le cas de matrices carrées de taille (n,n) et rectangulaires $A(n,m)$
 - Donner sa complexité, Est-il de coût minimal ?
 3. Ecrire l'algorithme de somme des deux matrices A et B.
 - Etudier le cas de matrices carrées de taille (n,n) et rectangulaires $A(n,m)$ et $B(n,m)$
 - Donner sa complexité, Est-il de coût minimal ?
 4. Ecrire l'algorithme de produit des deux matrices A et B.
 - Etudier le cas de matrices carrées de taille (n,n) et rectangulaires $A(n,m)$ et $B(m,p)$
 - Donner sa complexité, Est-il de coût minimal ?

Exercice 1 :

1. Ecrire l'algorithme de **Somme des éléments du vecteur** x de taille n
 - Donner sa complexité, Est-il de coût minimal ?

S = T[1]

pour i de 2 à n faire

S = S + T[i]

fin pour

$$\sum_{i=2}^n 1 = n - 2 + 1 = n - 1 = O(n)$$

$$|E(n)| = n$$

Coût minimal car $C(n) = O(|E(n)|) = O(n)$

Exercice 1 :

2. Ecrire l'algorithme **de produit de la matrice A par le vecteur x**.
- Etudier le cas de matrices carrées de taille (n,n) et rectangulaires $A(n,m)$
 - Donner sa complexité, Est-il de coût minimal ?

Matrice carrée (n,n)

pour i de 1 à n faire

y[i]=0

pour j de 1 à n faire

*y[i] = y[i] + A[i,j]*x[j]*

fin pour

fin pour

$$\sum_{i=1}^n \sum_{j=1}^n 2 = \sum_{i=1}^n 2n = 2n^2$$

Coût minimal

Matrice rectangulaire (n,m)

pour i de 1 à n faire

y[i]=0

pour j de 1 à m faire

*y[i] = y[i] + A[i,j]*x[j]*

fin pour

fin pour

$$\sum_{i=1}^n \sum_{j=1}^m 2 = \sum_{i=1}^n 2m = 2nm$$

Coût minimal

Exercice 1 :

3. Ecrire l'algorithme de **somme des deux matrices A et B**.
- Etudier le cas de matrices carrées de taille (n,n) et rectangulaires A(n,m) et B(n,m)
 - Donner sa complexité, Est-il de coût minimal ?

Matrices carrées (n,n)

pour i de 1 à n faire
 pour j de 1 à n faire
 $C[i,j] = A[i,j] + B[i,j]$
 fin pour
fin pour

$$\sum_{i=1}^n \sum_{j=1}^n 1 = \sum_{i=1}^n n = n^2$$

Coût minimal

Matrices rectangulaires (n,m)

pour i de 1 à n faire
 pour j de 1 à m faire
 $C[i,j] = A[i,j] + B[i,j]$
 fin pour
fin pour

$$\sum_{i=1}^n \sum_{j=1}^m 1 = \sum_{i=1}^n m = nm = O(nm)$$

Coût minimal

Exercice 1 :

4. Ecrire l'algorithme de **produit des deux matrice A et B**.

- Etudier le cas de matrices carrées de taille (n,n) et rectangulaires A(n,m) et B(m,p)
- Donner sa complexité, Est-il de coût minimal ?

Matrices carrées (n,n)

pour i de 1 à n faire

pour j de 1 à n faire

$C[i,j] = 0$

pour k de 1 à n faire

$C[i,j] = C[i,j] + A[i,k] * B[k,j]$

fin pour

fin pour

fin pour

$$\sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^n 2 = 2n^3 = O(n^3)$$

N'est pas de coût minimal

Matrices rectangulaires (n,m) et (m,p)

pour i de 1 à n faire

pour j de 1 à m faire

$C[i,j] = 0$

pour k de 1 à p faire

$C[i,j] = C[i,j] + A[i,k] * B[k,j]$

fin pour

fin pour

fin pour

$$\sum_{i=1}^n \sum_{j=1}^m \sum_{k=1}^p 2 = 2nmp = O(nmp)$$

N'est pas de coût minimal

EXERCICE 2

Exercice 2 :

Soient m et n deux entiers tels que $0 \leq m \leq n(n-1)/2$.

Discuter et simplifier les expressions suivantes en considérant les 3 cas :
 $m = O(1)$, $m = O(n)$ et $m = O(n^2)$

- $O(m + n)$;
- $O(m^2 + n)$;
- $O(m + n^2)$;
- $O(m + n \log n)$;
- $O(m \log m + n)$;

Exercice 2 :

Soient les 3 cas : $m = O(1)$, $m = O(n)$ et $m = O(n^2)$

- $O(m + n)$;
 - $m = O(1) \Rightarrow O(\textcolor{red}{m} + n) = O(\textcolor{red}{1} + n) = O(n)$
 - $m = O(n) \Rightarrow O(\textcolor{red}{m} + n) = O(\textcolor{red}{n} + n) = O(n)$
 - $m = O(n^2) \Rightarrow O(\textcolor{red}{m} + n) = O(\textcolor{red}{n}^2 + n) = O(n^2)$
- $O(m^2 + n)$;
 - $m = O(1) \Rightarrow O(\textcolor{red}{m}^2 + n) = O(\textcolor{red}{1}^2 + n) = O(n)$
 - $m = O(n) \Rightarrow O(\textcolor{red}{m}^2 + n) = O(\textcolor{red}{n}^2 + n) = O(n^2)$
 - $m = O(n^2) \Rightarrow O(\textcolor{red}{m}^2 + n) = O((\textcolor{red}{n}^2)^2 + n) = O(n^4)$

Exercice 2 :

Soient les 3 cas : $m = O(1)$, $m = O(n)$ et $m = O(n^2)$

- $O(m + n^2)$;
 - $m = O(1) \Rightarrow O(\textcolor{red}{m} + n^2) = O(\textcolor{red}{1} + n^2) = O(n^2)$
 - $m = O(n) \Rightarrow O(\textcolor{red}{m} + n^2) = O(\textcolor{red}{n} + n^2) = O(n^2)$
 - $m = O(n^2) \Rightarrow O(\textcolor{red}{m} + n^2) = O(\textcolor{red}{n^2} + n^2) = O(n^2)$
- $O(m + n \log n)$;
 - $m = O(1) \Rightarrow O(\textcolor{red}{m} + n \log n) = O(\textcolor{red}{1} + n \log n) = O(n \log n)$
 - $m = O(n) \Rightarrow O(\textcolor{red}{m} + n \log n) = O(\textcolor{red}{n} + n \log n) = O(n \log n)$
 - $m = O(n^2) \Rightarrow O(\textcolor{red}{m} + n \log n) = O(\textcolor{red}{n^2} + n \log n) = O(n^2)$

Exercice 2 :

Soient les 3 cas : $m = O(1)$, $m = O(n)$ et $m = O(n^2)$

- $O(m \log m + n)$;
 - $m = O(1) \Rightarrow O(m \log m + n) = O(1 \log 1 + n) = O(n)$
 - $m = O(n) \Rightarrow O(m \log m + n) = O(n \log n + n) = O(n)$
 - $m = O(n^2) \Rightarrow O(m \log m + n) = O(n^2 \log n^2 + n) = O(n^2 \log n)$

Exercice 2 :

Complexité	$m = O(1)$	$m = O(n)$	$m = O(n^2)$
$O(m + n)$	$O(n)$	$O(n)$	$O(n^2)$
$O(m^2 + n)$	$O(n)$	$O(n^2)$	$O(n^4)$
$O(m + n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$
$O(m + n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(n^2)$
$O(m \log m + n)$	$O(n)$	$O(n \log n)$	$O(n^2 \log n^2)$ $= O(n^2 \log n)$

EXERCICE 3

Exercice 3 :

Déterminer les complexités du nid suivant dans le meilleur et le pire des cas :

Pour* i de 1 à n **faire*

Pour* j de 1 à n **faire*

Si* $(A(i, j)) \neq 0$ **Alors*

Pour* k de 1 à n **faire*

Si* $(B(k, j)) \neq 0$ **Alors*

$C(j, k) = A(k, j) * B(i, k)$

Fin Si

Fin Pour

Fin Si

Fin Pour

Fin Pour

A et B sont deux tableaux de booléens, 'un test est une opération logique de coût 1, l'opération booléenne * est de coût 1.

Exercice 3 :

Pire des cas :

Pour i de 1 à n **faire**

Pour j de 1 à n **faire**

Si $(A(i, j)) \neq 0$ **Alors**

Pour k de 1 à n **faire**

Si $(B(k, j)) \neq 0$ **Alors**

$C(j, k) = A(k, j) * B(i, k)$

Fin Si

Fin Pour

Fin Si

Fin Pour

Fin Pour

$A(i, j)$ et $B(k, j)$ sont toujours $\neq 0$

$$\begin{aligned} \sum_{i=1}^n \left(\sum_{j=1}^n \left(1 + \sum_{k=1}^n (1 + 1) \right) \right) &= \sum_{i=1}^n \sum_{j=1}^n (1 + 2n) = \sum_{i=1}^n n (1 + 2n) = n^2 (1 + 2n) \\ &= n^2 + 2n^3 = 2n^3 + O(n^2) = O(n^3) \end{aligned}$$

Exercice 3 :

Meilleur des cas :

Pour i de 1 à n **faire**

Pour j de 1 à n **faire**

Si $(A(i, j)) \neq 0$ **Alors**

Pour k de 1 à n **faire**

Si $(B(k, j)) \neq 0$ **Alors**

$C(j, k) = A(k, j) * B(i, k)$

Fin Si

Fin Pour

Fin Si

Fin Pour

Fin Pour

$A(i, j)$ est toujours = 0

$$\sum_{i=1}^n \sum_{j=1}^n (1) = \sum_{i=1}^n n = n^2 = O(n^2)$$

EXERCICE 4

Exercice 4 :

- Ecrire la fonction qui recherche un élément dans un tableau Recherche d'un élément x dans tableau Tab de taille n , dans le cas où :
 - Le tableau est non trié
 - Le tableau est trié
- Calculer le nombre d'opérations élémentaires dans les deux algorithmes

Exercice 4 :

- Tableau non trié

Solution 1

```
Trouve = 0
Pour i de 1 à n faire
    si (T[i] = x) alors
        Trouve = 1
    Finsi
fin pour
```

$C(n) = O(n)$

Solution 2

```
Trouve = 0; i = 1
Tantque (Trouve = 0 et i <= n)
    si (T[i] = x) alors
        Trouve = 1
    finsi
    i++
Fintantque
```

Pire des cas : $C(n) = O(n)$

Meilleur des cas : $C(n) = O(1)$

Exercice 4 :

- Tableau trié

On veut chercher un élément dans un tableau trié dans le sens croissant.

Le but de cette recherche est de diviser l'intervalle de recherche par 2 à chaque itération. Pour cela, on procède de la façon suivante:

- Soient deb et fin les extrémités gauche et droite de l'intervalle dans lequel on cherche la valeur x, on calcule mil, l'indice de l'élément médian:

$$\text{mil} = (\text{deb} + \text{fin}) \div 2$$

Exercice 4 :

- Tableau trié : Exemple

1	2	3	4	5	6	7	8	9	10
↑				↑					↑

X= 3

Mil = $(1+10) \text{ div } 2 = 5$

comparer le 3 et mil =5 =>> $3 < 5 \rightarrow \text{deb} = 1, \text{fin} = \text{mil}$

1	2	3	4	5	6	7	8	9	10
↑		↑		↑					

Mil = $(1+5) \text{ div } 2 = 3$

comparer le 3 et mil =3 =>> $3 = 3 \rightarrow \text{Trouve} = 1$

Exercice 4 :

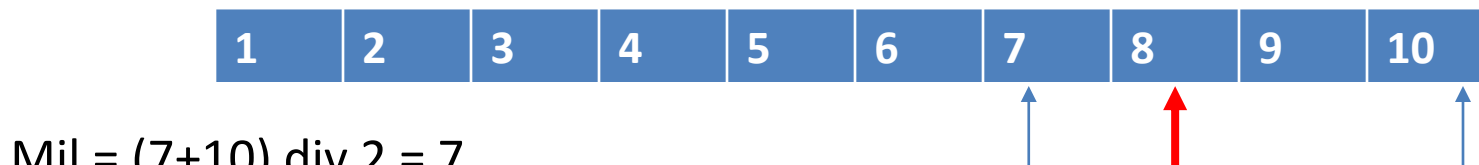
- Tableau trié : Exemple



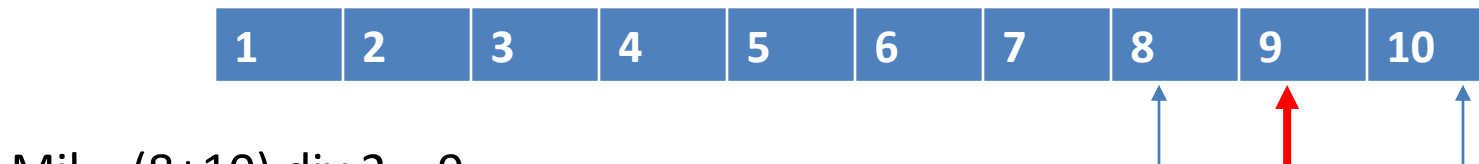
comparer le 9 et mil =5 $\Rightarrow 9 > 5 \rightarrow \text{deb} = \text{mil}, \text{fin} = \text{n}$



comparer le 9 et mil =7 $\Rightarrow 9 > 7 \rightarrow \text{deb} = \text{mil}, \text{fin} = \text{n}$



comparer le 9 et mil =8 $\Rightarrow 9 > 8 \rightarrow \text{deb} = \text{mil}, \text{fin} = \text{n}$



comparer le 9 et mil =9 $\Rightarrow 9 = 9 \rightarrow \text{Trouve} = 1$

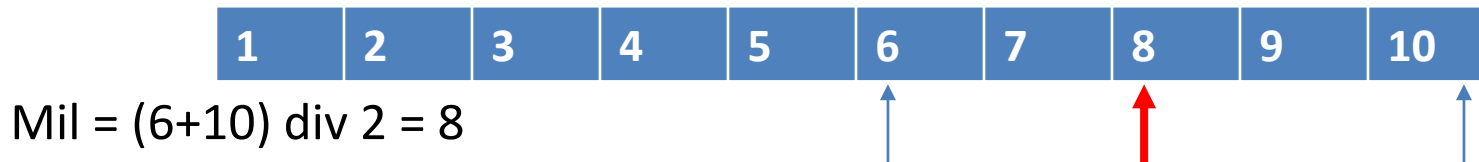
Exercice 4 :

- Tableau trié : Exemple - Amélioration



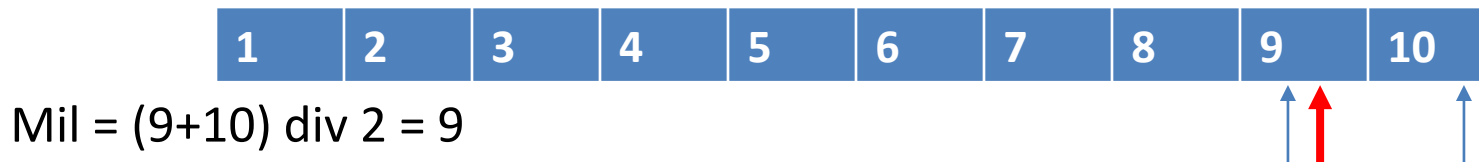
$$\text{Mil} = (1+10) \text{ div } 2 = 5$$

comparer le 9 et mil =5 ==> 9> 5 → **deb =mil+1**, fin =n



$$\text{Mil} = (6+10) \text{ div } 2 = 8$$

comparer le 9 et mil =8 ==> 9>8 → **deb =mil+1**, fin =n



$$\text{Mil} = (9+10) \text{ div } 2 = 9$$

comparer le 9 et mil =9 ==> 9=9 → **Trouve =1**

Exercice 4 :

- Tableau trié – Version itérative

deb = 1 ; fin = n

Trouve = 0

Tantque (Trouve = 0 et deb <= fin)

mil = (deb+fin) div 2

si (x = T[mil]) alors

Trouve = 1

sinon si (x < T[mil]) alors

fin = mil - 1

sinon

deb = mil + 1

fin si

Fin Tantque

Return (Trouve)

Exercice 4 :

- Tableau trié - Version récursive

deb = 1; fin = n

Fonction Recherche_T (Tab, x, deb, fin)

Début

Si (deb <= fin) alors

mil = (deb+fin) div 2

Si (T[mil] = x) alors

return 1

Sinon si T[mil] < x alors

return (Recherche_T(Tab, x, deb, mil-1))

Sinon

return (Recherche_T(Tab, x, mil+1,fin))

Fin si

Sinon

return 0

Finsi

Fin

Exercice 4 :

- Tableau trié

Supposant que le tableau est de taille n une puissance de 2, $n = 2^q$.

Le pire des cas pour la recherche d'un élément est de continuer à diviser jusqu'à obtenir un tableau de taille 1.

q est le nombre d'itérations nécessaires pour aboutir à un tableau de taille 1.

Itération 1 : $\rightarrow \frac{n}{2} = \frac{n}{2^1}$

Itération 2 : $\rightarrow \frac{n}{4} = \frac{n}{2^2}$

Itération 3 : $\rightarrow \frac{n}{8} = \frac{n}{2^3}$

Itération 4 : $\rightarrow \frac{n}{16} = \frac{n}{2^4}$

...

Itération $q-1$: $\rightarrow \frac{n}{2^{q-1}}$

Itération q : $\rightarrow \frac{n}{2^q}$

Dernière itération \rightarrow taille = 1

$$\frac{n}{2^q} = 1$$

$$\rightarrow 2^q = n$$

$$\rightarrow \log_2(2^q) = \log_2(n)$$

$$\rightarrow q = \log_2(n)$$

\rightarrow La complexité au pire des cas = $O(\log_2(n))$

\rightarrow La complexité au meilleur des cas = $O(1)$