

$f \in O(g)$ ssi $\exists C > 0, f(n) \leq Cg(n)$

$f \in \Theta(g)$ ssi $\exists C_1, C_2, \exists n_0$
 $\forall n > n_0 \quad C_1 g(n) \leq f(n) \leq C_2 g(n)$

Algorithmique (AL5)

TD n° 1 : complexité, graphes

Complexité algorithmique

Exercice 1 : ordres de grandeur

Est-ce que les apparteness ci-dessous sont correctes ? Justifier vos réponses.

- | | | | |
|--------------------------------------|---|---|--|
| 1. $3n^2 + 4n - 6 \in O(n^2)$ | <input checked="" type="checkbox"/> | 9. $3 + 5 \cdot \sin(n) \in \Theta(1)$ | <input checked="" type="checkbox"/> |
| 2. $3n^2 + 4n - 6 \in O(n^5)$ | <input checked="" type="checkbox"/> | 10. $2n + 3 \in \Theta(n)$ | <input checked="" type="checkbox"/> |
| 3. $3n^2 + 4n - 6 \in \Theta(n^2)$ | <input checked="" type="checkbox"/> | 11. $3^n \in O(2^n)$ | <input checked="" type="checkbox"/> |
| 4. $3n^2 - 4n - 6 \in \Theta(n^4)$ | <input checked="" type="checkbox"/> | 12. $(n+1)! \in O(n!)$ | <input checked="" type="checkbox"/> Faux ?? |
| 5. $3n^3 - 4n^2 - 6 \in \Theta(n^3)$ | <input checked="" type="checkbox"/> | 13. $n! \in O(n^n)$ | <input checked="" type="checkbox"/> Vrai $n \times (n-1) \times (n-2) \dots 3 \times 2 \times 1 < n^n$ |
| 6. $3n^2 + 2^n \in \Theta(2^n)$ | <input checked="" type="checkbox"/> | 14. $n^n \in O(n!)$ | <input checked="" type="checkbox"/> |
| 7. $3n^2 + 2^{3n+2} \in \Theta(2^n)$ | <input checked="" type="checkbox"/> $3n+2 > n$ | 15. $n^n + 2^n + n^{10} + n! \in \Theta(n^n)$ | <input checked="" type="checkbox"/> |
| 8. $3n^2 + 2^{3n^2} \in O(2^{n^3})$ | <input checked="" type="checkbox"/> | | |

Exercice 2 : analyse de complexité

Quelle est la complexité des algorithmes suivants ?

- | | |
|--|--|
| 1. <pre>s = 0 for (int i = n; i > 1; i = i / 2) for (int j = 0; j < i; j = j + 1) s = s + 1;</pre> | 3. <pre>s = 0 for (int i = n; i > 0; i = i - 1) for (int j = i; j > 0; j = j - 1) s = s + 1;</pre> |
| 2. <pre>s = 0 for (int i = 1; i < n; i = i * 2) for (int j = 0; j < n; j = j + 1) s = s + 1;</pre> | 4. <pre>s = 0 for (int i = 1; i < n; i = i * 2) for (int j = 1; j < n; j = j * 2) s = s + 1;</pre> |

Algorithmes de graphes

Exercice 3 : complexité d'algorithmes

- Quelle est la complexité (en temps dans le pire cas) de l'algorithme Algo1 qui manipule un graphe orienté $G = (V, E)$? On distinguera le cas où G est représenté sous forme de liste d'adjacence ou sous forme matricielle. On donnera un ordre de grandeur de la complexité sous forme de Θ .

```
Algo1 (G) :
// G = (V, E)
For i in V:
    For (i,j) in E :
        print("(i -> j)")
```

2. Que fait l'algorithme Algo2 et quelle est sa complexité? On supposera que le graphe $G = (V, E, w)$ est un graphe orienté valué avec $w : E \rightarrow \mathbb{N}$ et tel que chaque sommet a au moins un voisin sortant.

```

Algo2 (G) :
// G = (V,E,w)
W est un tableau d'entiers de taille |V|
Trier les arcs par poids w croissant
For i in V:
    W[i] = undef
For (i,j) in E [pris dans l'ordre...]
    If W[i] == undef then W[i] = w(i,j)
Return W

```

3. Proposer un algorithme plus efficace pour remplacer Algo2.

Exercice 4 : voyageur de commerce

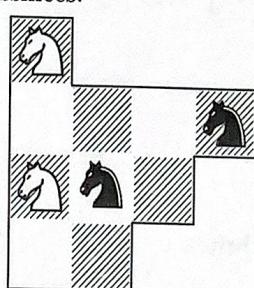
Mr. Smith veut visiter toutes les villes de France avec son jet privé depuis Paris. Mais il ne veut pas consommer trop d'essence, donc il veut trouver la route la plus courte.

1. Modéliser ce problème avec un graphe.
2. Trouver un algorithme simple pour le résoudre (n'ayez pas peur s'il est très lent). On suppose qu'on a accès à une liste *ROUTES* qui contient toutes les permutations possibles de villes de France.
3. Évaluer sa complexité.

Pour aller plus loin

Exercice 5 : échange de cavaliers (Jean-Paul Davalan)

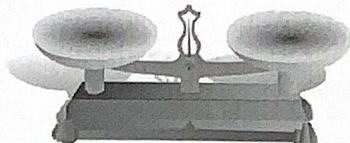
Pour résoudre ce casse-tête vous devez échanger les positions des deux cavaliers blancs avec celles des deux noirs, en respectant évidemment les règles du déplacement du cavalier sur un échiquier et en n'utilisant que les dix cases dessinées.



1. Modéliser le problème à l'aide d'un graphe.
2. Utiliser cette modélisation pour résoudre ce casse-tête.
3. Y-a-t-il des cases inutiles ?

Exercice 6 : des pièces et une balance

1. Vous avez 9 pièces, toutes de même poids exceptée une fausse, qui est légèrement plus légère que les autres. Vous disposez d'une balance Roberval à *deux plateaux*. Trouver la fausse pièce en 2 pesées seulement. Même problème avec 27 pièces.
2. Généraliser votre algorithme à tout n et estimer le nombre de pesées nécessaires. Justifier la correction de votre algorithme.
3. Deux extensions :
 - a. Que se passe-t-il si on ne sait pas si la fausse pièce est plus lourde ou plus légère ? Essayer avec 13 pièces.
 - b. Que se passe-t-il s'il on ne sait pas si toutes les pièces sont exactement égales ou s'il y a une pièce différente ? Essayer avec 12 pièces.



$$\mathcal{O} : \leftarrow \quad \Omega : \rightarrow \quad \Theta : \leftrightarrow$$

Ex 1 :

Exercice 1 : ordres de grandeur

Est-ce que les appartenances ci-dessous sont correctes ? Justifier vos réponses.

1. $3n^2 + 4n - 6 \in O(n^2)$ Vrai
2. $3n^2 + 4n - 6 \in O(n^5)$ Vrai
3. $3n^2 + 4n - 6 \in \Theta(n^2)$ Vrai
4. $3n^2 - 4n - 6 \in \Theta(n^4)$ Faux
5. $3n^3 - 4n^2 - 6 \in \Theta(n^3)$ Vrai
6. $3n^2 + 2^n \in \Theta(2^n)$ Vrai
7. $3n^2 + 2^{3n+2} \in \Theta(2^n)$ Faux
8. $3n^2 + 2^{3n^2} \in O(2^{n^3})$ Vrai

9. $3 + 5 \cdot |\sin(n)| \in \Theta(1)$ Vrai
10. $2n + 3 \in \Theta(n)$ Vrai
11. $3^n \in O(2^n)$ Faux
12. $(n+1)! \in O(n!)$ Faux
13. $n! \in O(n^n)$ Vrai
14. $n^n \in O(n!)$ Faux
15. $n^n + 2^n + n^{10} + n! \in \Theta(n^n)$ Vrai

Ex 2

$\log n$

\uparrow

1) $s = 0$

for (int i=n; i > 1; i = i/2) \rightarrow nb opé

for (int j= 0; j < i; j = j + 1) \rightarrow nb opé

$s = s + 1;$

$\downarrow 1$

complexité

n

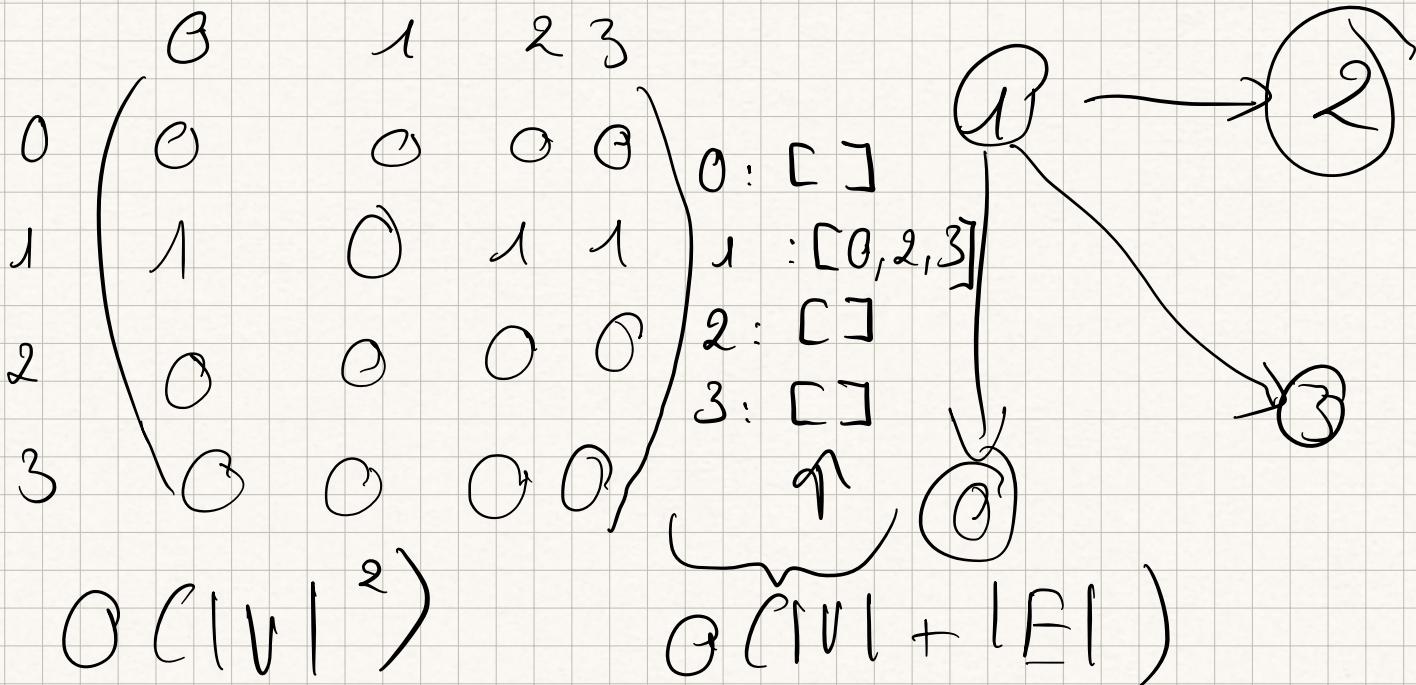
2) $\Theta(n \log n)$

3) $\Theta(n^2)$

4) $\Theta(\log(n)^2)$

Ex 3

Exemple



l'ensemble
des sommets

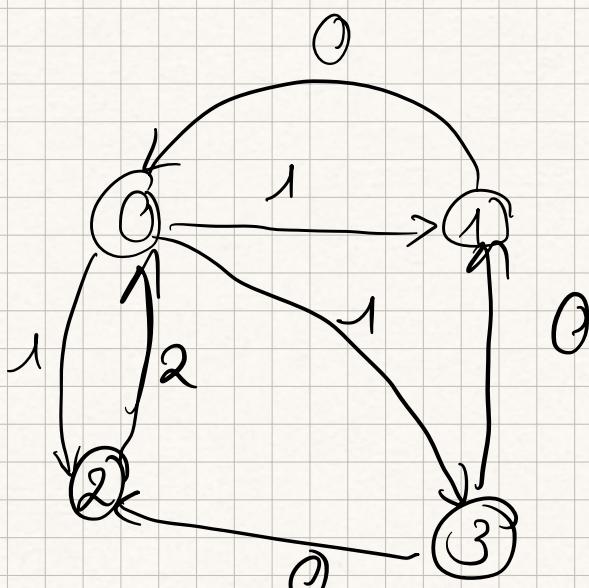
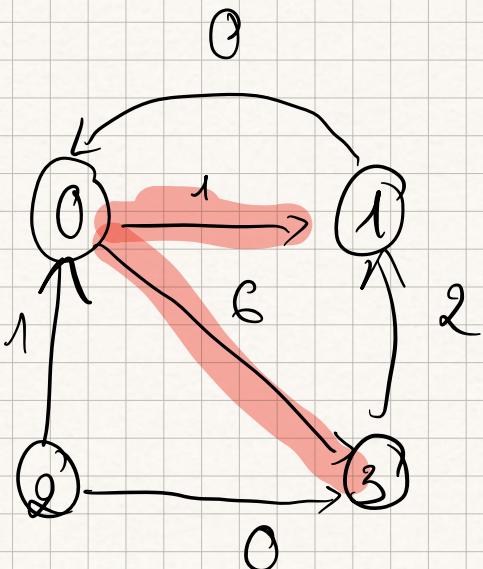
1) forme de matrice $O(|V|^2)$

liste d'adjacence $O(|V| + |E|)$

2x Algo 2 :

$w \leftarrow \text{tab}[v]$

$\Theta(|E| \log |E|)$



(

0	1	2	3
1	0	0	2

)

(

0	1	2	3
1	0	2	0

)

pas b
car algo
minimal

3x tél \Rightarrow inutile

Parcours liste adj

\rightarrow poids actuel minimal

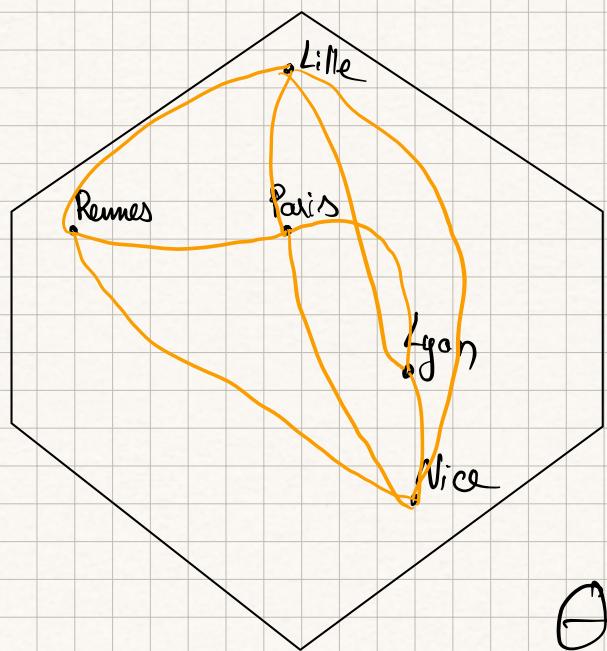
\hookrightarrow nouvelle arête

Si poids nouvelle arête

$<$ poids - actuel - minimal

$\Theta(|V| + |E|)$

Ex 4



Sommet $s = \text{ville } (n)$
 Arretes = $\text{ville}_1 - \text{ville}_2$
 (poids)

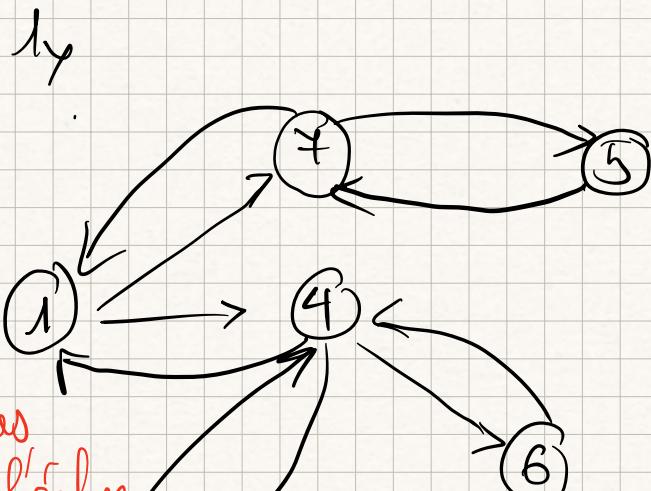
non orienté

$\Theta(n!)$ → tester toutes les villes

Ex 5

										C1b
2		3	4		5					
										C1n
	6	7	8							
9										C2b
										C2n

On respecte pas
les règles de l'échec



C1b	7	1	(4	10	2	8)	10
C2n	5	1	(1	4	10	2)	
C1n	6	5	(7	1	4	10)	
C2b	5	6	(4	1	7	5)	

gagné oublie

$$\begin{array}{ccccccc}
 & 1 & & & & & \\
 & \diagdown & & & & & \\
 & 4 & - & 6 & & & \\
 & \diagup & & & & & \\
 & 10 & - & 2 & - & 8 &
 \end{array}$$

C15	8	2	10	1
C2n	2	10	4	8
C1n	10	4	1	7

②

2	3	4	5
Cen			
6	7	8	
			C1b

③

C2n	6	4	10	2
-----	---	---	----	---

C15	1	4	10
-----	---	---	----

C1N	1	4	6	1
-----	---	---	---	---

$$7 - 5$$

$$4 - 6$$

$$10 - 2 - 8$$

C1n

3

9

1

④

C1b	10	4	1	7	W
C2n	2	10	4	1	W

Ex 6

1	0	0	0
0	0	0	
0	0	0	

Pf. y a 1 fausse

Trouver la fausse en 2

2x Cas de base : Si on a plus qu'une pièce
 \hookrightarrow C'est la bonne

Causer le far par 3 : 2 premier sont égaux (A, B, C)

On pèse A + lourd B

[On garde le B et on recommence l'algo

$A = B$
On prend C

$$\Theta(\log_3(n))$$

3a : 1 pesée de plus

b :