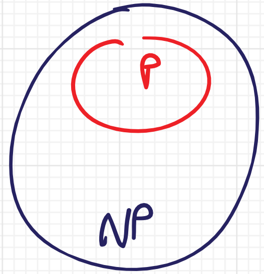
Plan de bataille pour prouver qu'un problème est dans P ou dans NP

P : $\exists$ machine M qui résout le pb en temps poly
Algo DETERMINISTE

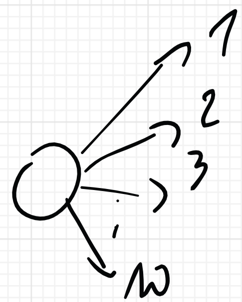


NP : $\exists$ machine M NON DET qui résout le pb en temps poly
Algo

Machine $\Longleftrightarrow$ Algo

Vous povuez tout à fait décrire des algos plutôt que des machines

```
int x;
  x ← un entier entre 1 et 10   (NON DET)
```
Choisi



Ceci est parexemple un algo non déterministe.
(à ne pas confondre avec un algo aléatoire)

TD 2

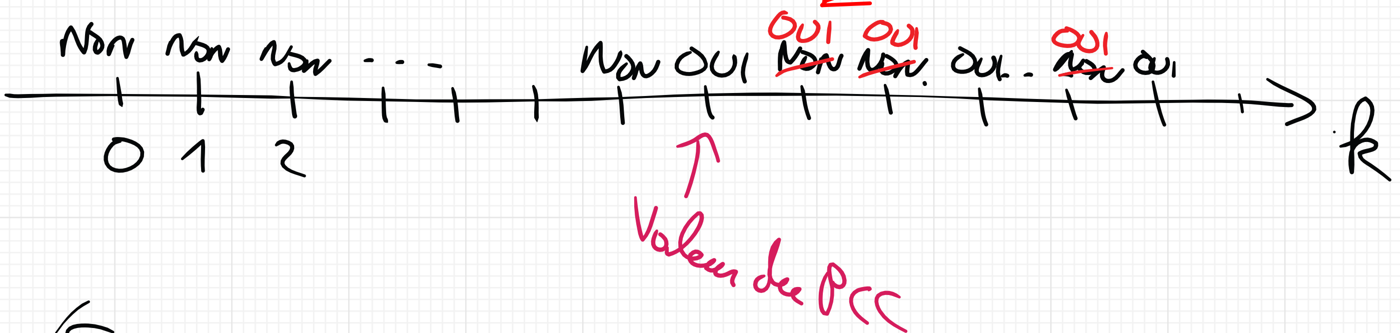
Exo 1

1) (SHP)

$G=(V,A)$ graphe orienté $\quad$ ω : ~~Poids~~ : $A \longrightarrow \mathbb{N}$ $\quad$ $s,t \in V$ $\quad$ $k \in \mathbb{N}$

$\exists$? un chemin P de s à t tq le poids du chemin $\displaystyle\sum_{a \in P} \omega(a) \leq k$?

Le <= permet de faire une dichotomie
derrière pour trouver l'optimal.



$(SHP) \in P$ car l'alg de Dijkstra le résout en temps $\underbrace{O(|A|+|V|\log(|V|))}$

$= O(|A|\lg(|A|))$

$= O(|A|^2)$

3 (3SAT) φ formule booléenne avec des variables $(x_1 x_2 \dots x_n)$

φ sous la forme $\varphi = C_1 \wedge C_2 \wedge \dots \wedge C_m$

avec $C_i = (l_i^1 \vee l_i^2 \vee l_i^3)$

avec $l_i^{1,2,3} \in \{ x_j, \overline{x_j} \}_{j \leq n}$

$$\varphi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_2} \vee \overline{x_3} \vee x_4) \wedge (x_1 \vee x_4 \vee x_5)$$

$\exists$? affectation de $x_1 x_2 \dots x_n$ qui rende φ vraie?

---

certificat — **Vérifieur**

Construire la solution — Vérifier la solution

NON DET

Pour chaque x_i
on choisit $x_i \leftarrow$ VRAI
ou
$x_i \leftarrow$ FAUX

$\Theta(n)$

$\longrightarrow$

DET

Dans φ, on remplace x_i
par sa valeur et on
vérifie si $\varphi =$ VRAI

$\Theta(3m + 3m) = \Theta(m)$

OUI Faible

NON Fort

$\exists$ une machine NON DET qui résout (3SAT) en temps $\Theta(n+m)$

donc en temps polynomial.

$\Rightarrow (3SAT) \in NP$

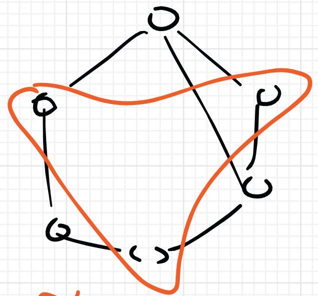
!!! On suivra toujours ce modèle : certificat + vérifieur. Seul le certificat est non déterministe.

Quand vous faites cette machine, demandez vous toujours si votre complexité du certificat et du vérifieur est correcte!

14 (WINS) Soit $G = (V, E)$ graphe non orienté, $k \in \mathbb{N}$ $\quad |V| = n \quad |E| = m$

$\omega : V \to \mathbb{N}$

$\exists ?\ W \subset V$ tq W stable $(\forall u,v,\ (u,v) \notin E)$ / $\sum_{v \in W} \omega(v) \geq k$?



Stable de taille 3

Certificat

NONDET

$\forall v \in V$ mettre v dans W ou non

$\Theta(n)$

$\longrightarrow$

Vérifieur

DET

$\forall u v \in W$ $(u,v) \notin E$? est ce que

$\sum_{v \in W} \omega(v) \geq k$?

→ OUI

→ NON

$\Theta(W^2 + n)$

$\Theta(n^2 + n) = \Theta(n^2)$

∃ machine non DET qui résout (WINS) en temps $\Theta(n^2)$

$\Downarrow$

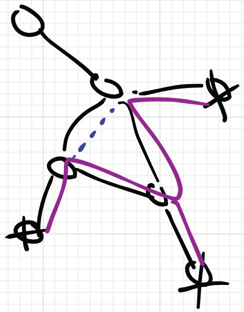
(WINS) $\in NP$

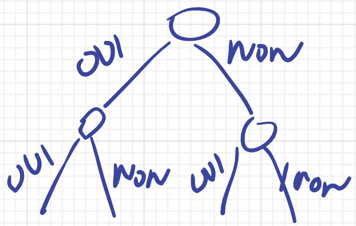
7. (QBF) Soit $m \in \mathbb{N}$ φ formule sur $x_1 x_2 \cdots x_{2m}$

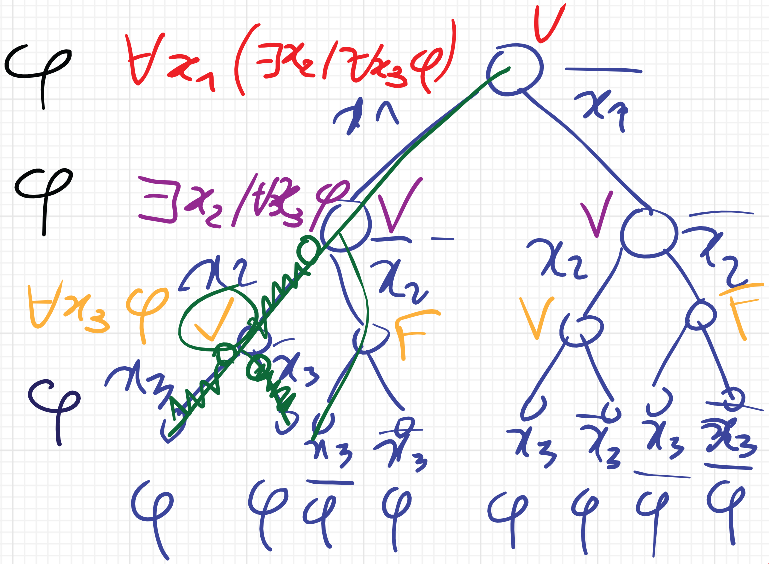
est-ce que $\forall x_1 \exists x_2 \forall x_3 \exists x_4 \cdots \forall x_{2m-1} \exists x_{2m} \varphi$?

$\forall x_1 \exists x_2 \forall x_3 \exists x_4 \varphi$

$\not\equiv \forall x_1 \forall x_3 \exists x_2 \exists x_4 \varphi$







$O(2^n) \Rightarrow$ EXPTIME

(la partie en vert était là pour montrer qu'on peut recoder cet algo en espace polynomial : on a besoin en mémoire que d'une branche à la fois il n'est pas nécessaire d'enregistrer en mémoire tout l'arbre ;
-> QBF appartient à PSPACE.)

par contre il est nécessaire de parcourir cet arbre en entier, donc l'algo est en temps exponentiel. Le temps est toujours au moins aussi grand que l'espace.