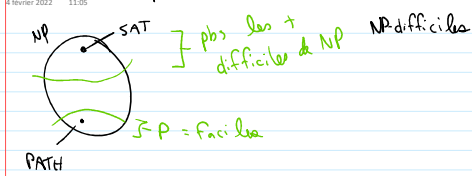
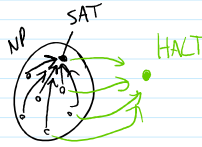


Problèmes NP-complets



- Pour définir NP-difficile, on utilise une réduction polynomiale $\leq_p$
- Soit A, B deux langages. On dit $A \leq_p B$ s'il existe une fct $f: \Sigma^* \rightarrow \Sigma^*$ telle que $w \in A \iff f(w) \in B$ et f est calculable en temps polynomial.

Def: un langage L est NP-difficile si:
 $\forall L' \in NP, \text{ on a } L' \leq_p L.$



De plus, si $L \in NP$, alors
 L est NP-complet.

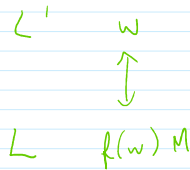
Thm: s'il existe une MT en temps poly pour un L NP-difficile,
 alors $\forall L' \in NP, \text{ on a } L' \in P.$

Si $L \in P$, alors tout $L' \in NP$ est dans P .

- Supposons qu'un L NP-difficile est dans P . Donc, $\exists$ MT M pour décider L en temps $O(n^k)$.
- Soit $L' \in NP$. On sait $L' \leq_p L$, et donc $\exists f$ calculable en $O(n^k)$ telle que
 $w \in L' \iff f(w) \in L.$

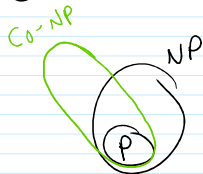
Pour décider L' , on construit la MT:

sur entrée w : ($w \in L' ?$)
 $O(n^k)$ calculer $f(w)$
 $O(n^k)$ Passer $f(w)$ à M
 $O(1)$ accepter si M a accepté



co-NP

Un langage L est dans co-NP si son complément $\bar{L}$ est dans NP.
 $(\bar{L} = \{w : w \notin L\})$

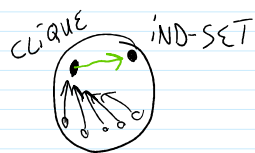


ex: UNSAT = $\{\varphi : \varphi \text{ est une formule bool. non-satisfaisable}\}$

$\overline{\text{UNSAT}} = \text{SAT}$ Puisque $\text{SAT} \in NP$, alors $\text{UNSAT} \in \text{co-NP}.$

$\text{IND-SET} = \{ \langle G, k \rangle : G \text{ a un ensemble indep. de taille } k \}$
est NP-complet.

Pour prouver que IND-SET est NP-difficile, on va utiliser un autre langage NP-difficile et le réduire à IND-SET.



Thm: CLIQUE est NP-difficile. (boîte noire)

On veut montrer que $\text{CLIQUE} \leq_p \text{IND-SET}$

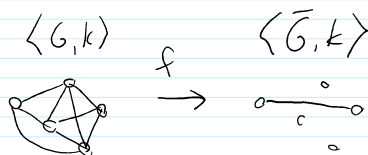
On veut trouver f t.q. $\langle G, k \rangle \in \text{CLIQUE} \Leftrightarrow f(\langle G, k \rangle) \in \text{IND-SET}$

Soit $\langle G, k \rangle$ une instance de CLIQUE.

On définit $f(\langle G, k \rangle)$ comme $\langle \bar{G}, k \rangle$, où

$\bar{G}$ est le complément de G , c'est-à-dire $V(\bar{G}) = V(G)$

$$E(\bar{G}) = \{uv : uv \notin E(G)\}$$



Il est facile de voir que
 f est calculable en temps poly.

Il faut montrer que G a une clique de taille k
 $\Leftrightarrow \bar{G}$ a un ens. indep. de taille k

$\Rightarrow$ Supposons que G a une clique de taille k . Soit C une telle clique. On voit que C est un ens. indep. de $\bar{G}$ car il n'y aura aucune arête entre les sommets de C dans $\bar{G}$.

$\Leftarrow$ Supposons que $\bar{G}$ a un ens. indep. de taille k . Soit C un tel ens. indep. Alors C est une clique de taille k dans G . $\square$

Recette générale pour montrer qu'un langage L est NP-complet.

① $L \in \text{NP}$. Souvent trivial avec certificat.

② L est NP-difficile. Trouver un langage NP-difficile A .

Réduire A à L (montrer que $A \leq_p L$)

2.1. Décrire la transformation f de A à L .

2.2. Montrer $w \in A \Rightarrow f(w) \in L$

2.3. Montrer $f(w) \in L \Rightarrow w \in A$

Thm: CLIQUE est NP-complet.

CLIQUE = $\{ \langle G, k \rangle : G \text{ a une clique de taille } k \}$

Pour le prouver, on va utiliser une réduction via 3-SAT.

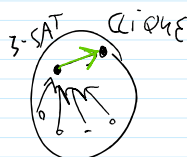
3-SAT = $\{ \varphi : \varphi \text{ est une formule booléenne en 3-CNF satisfaisable} \}$
 conjunctive normal form

CNF: φ est un ensemble de clauses liées par des $\wedge$
 clause: variables liées par des $\vee$

$$(x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_2 \vee x_4 \vee \bar{x}_5) \wedge (x_1 \vee \bar{x}_2 \vee \bar{x}_4)$$

$\underbrace{(x_1 \vee x_2 \vee \bar{x}_3)}_{\text{clause } C_1} \wedge \underbrace{(\bar{x}_2 \vee x_4 \vee \bar{x}_5)}_{\text{clause } C_2} \wedge \underbrace{(x_1 \vee \bar{x}_2 \vee \bar{x}_4)}_{\text{clause } C_3}$
 3-CNF: chaque clause a 3 variables.

Thm: 3-SAT est NP-complet. (boîte noire)



① CLIQUE $\in$ NP, car une clique C de taille k peut servir de certificat.

② Réduction via 3-SAT.

$$\varphi \xrightarrow{f} \langle G, k \rangle \quad \text{t.q.} \quad \varphi \in 3\text{-SAT} \Leftrightarrow G \text{ a une clique } |C| = k$$

Soit φ une instance de 3-SAT.

Soient $x_1, \dots, x_n$ les variables de φ .

Soient $C_1, \dots, C_m$ les clauses de φ .

2.1 • On pose $k = m$. (# de clauses)

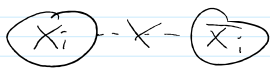
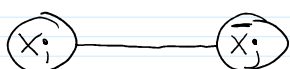
Pour construire G , pour chaque clause $C_i = (a_i \vee b_i \vee d_i)$

on ajoute à G les 3 sommets a_i, b_i, d_i .

Donc G a $3m$ sommets.

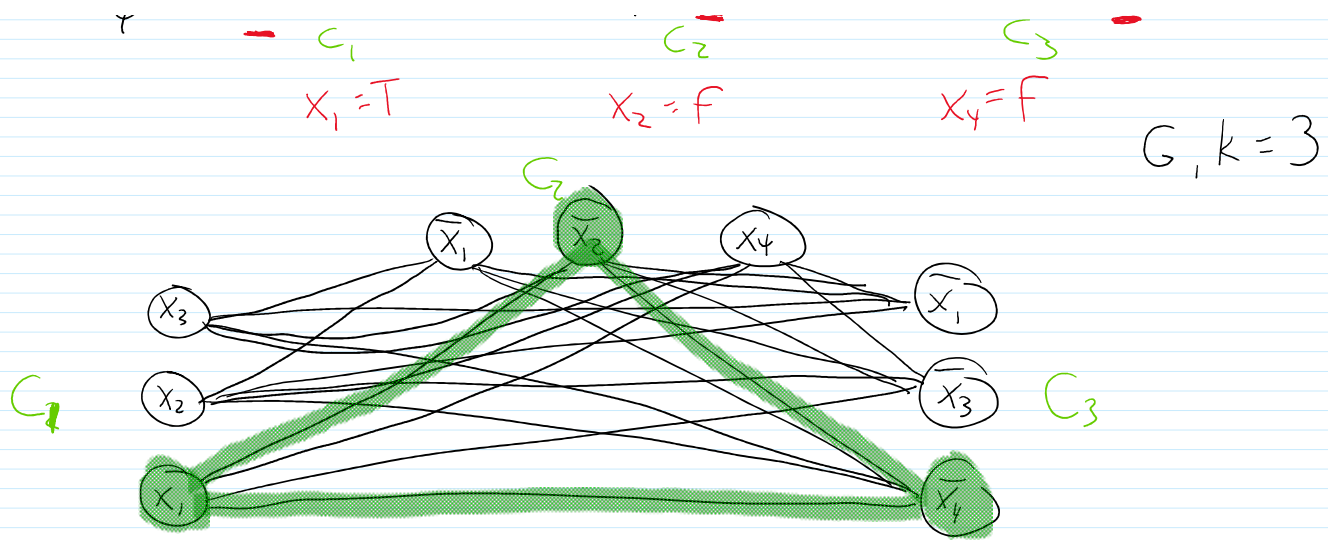
Pour les arêtes, on ajoute une arête entre y_i et z_j si:

y_i et z_j proviennent de clauses différentes et y_i n'est pas la négation de z_j .



$$\varphi = (x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_4) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee \bar{x}_4)$$

$\underbrace{\quad}_{C_1} \quad \underbrace{\quad}_{C_2} \quad \underbrace{\quad}_{C_3}$
 $x = T \quad \quad x = F \quad \quad x = F$



Il est facile de construire $\langle G, k \rangle$ à partir de φ en temps $O(m^2)$

On montre $\varphi \in 3\text{-SAT} \Leftrightarrow \langle G, k \rangle \in \text{CLIQUE}$

$\Rightarrow$ Supposons que $\varphi \in 3\text{-SAT}$.

Donc $\exists$ assignation des x_i qui satisfait chaque clause.

Pour chaque C_i , soit y_i une variable de l'assignation qui sat. C_i .

Notre clique est $K = \{ y_i : C_i \text{ est une clause de } \varphi \}$.

- Puisqu'il y a m clauses, $|K| = m = k$.

- Puisque l'assignation ne met pas une variable à T et F en même temps, les sommets de K ne sont pas la négation l'un de l'autre et donc K est une clique.

$\Leftarrow$ Supposons $\langle G, k \rangle \in \text{CLIQUE}$. Soit K une clique de G , $|K| = k = m$.

Alors K doit contenir exactement 1 sommet par triplet représentant une clause.

De plus, les sommets de K correspondent à des valeurs des x_i qui satisfont chaque clause. Ces valeurs correspondent à une assignation possible de φ sans contradiction ($x_i = T, \bar{x}_i = F$) et donc φ est satisfaisable. $\square$

$k\text{-CLIQUE} = \{ G : G \text{ a une clique de taille } k \}$

$k\text{-CLIQUE} \in P.$