

INFO-F408: Computability & complexity

Rémy Detobel

23 Octobre, 2017

1 P vs NP

Définition

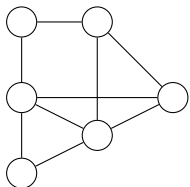
NP est un ensemble de langage vérifiable en temps polynomial.

Théorème

Un langage appartient à NP \Leftrightarrow il peut être décidé dans une machine de turing non déterministe à temps polynomial.

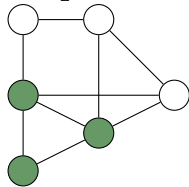
$$\text{SUBSET SUM} = \left\{ \langle S, t \rangle \mid S = \{x_1, \dots, x_k\} \text{ et pour certain } Y \subseteq \{1, \dots, k\}, \sum_{j \in Y} x_j = t \right\}$$

1.1 Problème du CLIQUE

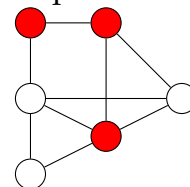


Le problème CLIQUE peut être défini comme étant $= \{ \langle G, K \rangle \mid G \text{ est un graphe non dirigé avec } k\text{-clique} \}$

Exemple valide :



Exemple d'ensemble ne répondant pas au problème :



Définition

Une fonction $f : \Sigma^* \rightarrow \Sigma^*$ est calculable dans un temps polynomial s'il existe une machine de turing tel qu'elle s'arrête avec seulement $f(w)$ sur son ruban avec une entrée w et s'exécute en un temps polynomial.

Définition

Un langage A est réductible en temps polynomial dans un langage B , écrivons :

$$A \leq_p B$$

S'il existe une fonction f s'exécutant en un temps polynomial tel que :

$$\forall w, w \in A \Leftrightarrow f(w) \in B$$

Important

Si $A \leq_p B$ et $B \in P$, alors $A \in P$

1.1.1 Exemple : Problème de 3 SAT

$$\begin{aligned} \phi = & (x_1 \vee x_1 \vee x_2) \wedge \\ & (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_2) \wedge \\ & (\bar{x}_1 \vee x_2 \vee x_2) \end{aligned}$$

Tel que

$$\begin{aligned} x_1 & \leftarrow F \\ x_2 & \leftarrow T \end{aligned}$$

3-CNF formule (il s'agit d'une forme normal conjonctive)

$$3 \text{ SAT} = \{ \phi \mid \phi \text{ est une formule boolean satisfaisant 3-CNF} \}$$

$3 \text{ SAT} \in NP$

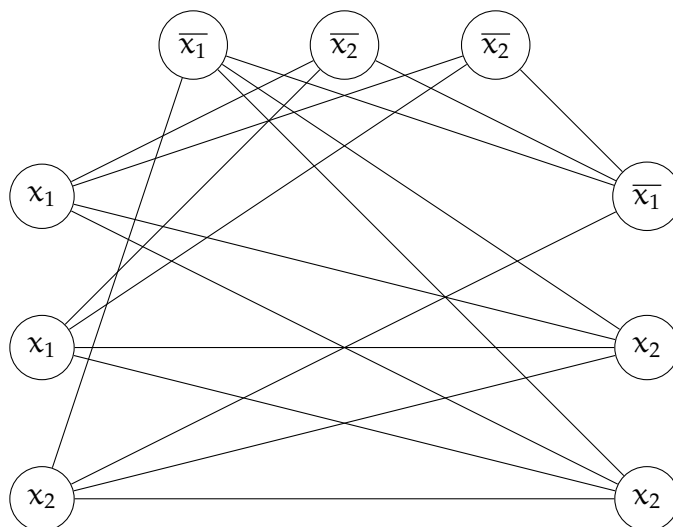
Théorème

$$3 \text{ SAT} \leq_p \text{ CLIQUE}$$

1.1.2 Passage de 3-SAT à CLIQUE

On va donc devoir trouver un algorithme pouvant transformer un problème "3 SAT" en un problème de type "K-CLIQUE" en un temps polynomial. Cela permettra donc de résoudre "3 SAT" avec seulement un algorithme de résolution de type "K-CLIQUE".

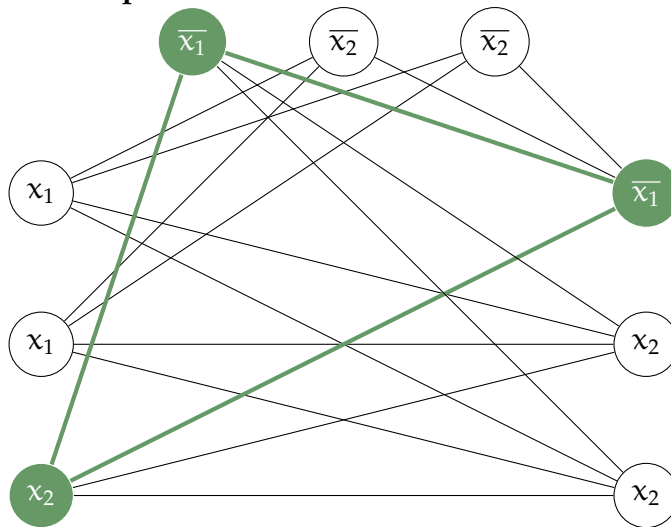
Un "litéral" est simplement une variable ou son inverse (ex : x_i ou \bar{x}_i).



$K = 3$, il s'agit donc du nombre de clause.

On peut donc écrire que si ϕ est satisfait, G as un k -clique.

Une solution pourrait être :



$$\phi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$$

Si l'on prend un littéral vrai sur chacune des lignes, on aura une solution au k -clique.

2 NP-Compleet

Définition

Un langage B est "NP-complet" si et seulement si :

1. $B \in NP$
2. Pour chaque $A \in NP, A \leq_p B$

2.1 Cook-Levin Théorème

Définissons SAT comme étant un problème retournant un boolean pour toutes les formules sous forme normal conjonctive.

SAT est NP-Compleet.

1. Si B est NP-complet et $B \in P$, alors $P = NP$
2. Si B est NP-complet et $B \leq_p C$ (pour $C \in NP$), alors C est NP-Compleet.

On peut donc écrire par exemple :

$$\begin{aligned} \text{Si } A &\leq_p B \\ \wedge B &\leq_p C \\ \Rightarrow A &\leq_p C \end{aligned}$$

Définition

Un langage B est NP-Hard (NP-Dur ou NP-Difficile) si et seulement si :
pour tout $A \in NP, A \leq_p B$

Considérons $A \in NP$. Il existe donc une machine de turing "N" non déterministe qui peut décider du langage A en un temps polynomial. Supposons en temps $\leq n^k$ pour un certain $k = O(1)$

#q₀w₁w₂...w_n _ _ ... _ #
 #a₁w₂...w_n _ _ ... _ #
 ...

Où on peut compter n^k lignes sur une longueur de n^k

$$C = Q \cup \Gamma \cup \{\#\}$$

Variables : $X_{i,j,s} | i, j = 1 \dots n^k, S \in C$

$$\phi = \phi_{\text{cell}} \wedge \phi_{\text{start}} \wedge \phi_{\text{move}} \wedge \phi_{\text{accept}}$$

$$\phi_{\text{start}} = X_{1,1,\#} \wedge X_{1,2,q_0} \wedge X_{1,3,w_1} \wedge \dots \wedge X_{1,n+2,w_n} \wedge X_{1,n+3,-} \wedge \dots \wedge X_{1,n^k-1,-} \wedge X_{1,n^k,\#}$$

$$\phi_{\text{cell}} = \bigwedge_{1 \leq i,j \leq n^k} \left[\left(\bigvee_{S \in C} X_{i,j,S} \right) \wedge \bigwedge_{s,t \in C | s \neq t} (\overline{X_{i,j,s}} \vee \overline{X_{i,j,t}}) \right]$$

$$\phi_{\text{accept}} = \bigvee_{1 \leq i,j \leq n^k} X_{i,j,q_{\text{accept}}}$$

$$\phi_{\text{move}} = \bigwedge_{i,j} \left[\text{où } (i,j) \text{ windows est légal} \right]$$

↑ "Use Legal Windows", il s'agit d'une fenêtre de 2 (ligne) par 3 (colonne)
 placé partout dans le tableau

Où une fenêtre légal est définit par :

$$\bigvee_{a_1, a_2, \dots, a_6 \text{ est une fenêtre légal}} (X_{i,j-1,a_1} \wedge X_{i,j,a_2} \wedge X_{i,j+1,a_3} \wedge \dots)$$

	j-1	j	j+1
i	a ₁	a ₂	a ₃
i+1	a ₄	a ₅	a ₆

#	q ₀	w ₀
#	a	q ₁

est légal.

$$(q_1, a, R) \in \delta(q_0, w_1)$$

a	b	c
a	d	e

$d \neq b$ et $e \neq c$ et $\in \Gamma$ n'est pas légal