INFO-F408: Computability & complexity

Rémy Detobel

23 Octobre, 2017

1 P vs NP

Définition

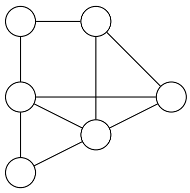
NP est un ensemble de langage vérifiable en temps polynomial.

Théorème

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

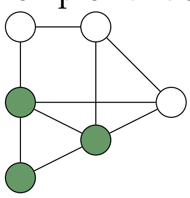
$$\text{SUBSET SUM} = \left\{ < S, t > | S = \{x_1, ..., x_k\} \text{ et pour certain } Y \subseteq \{1, ...k\}, \sum_{j \in Y} x_j = t \right\}$$

1.1 Problème du CLIQUE

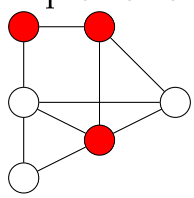


Le problème CLIQUE peut dont être définit comme étant $= \{ < G, K > |$ G est un graphe non dirigé avec k-clique$\}$

Exemple valide :



Exemple d'ensemble ne répondant pas au problème :



Définition

Une fonction $f : \Sigma^* \to \Sigma^*$ 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 polynimal 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

$$\begin{aligned}\phi =&(x_1 \vee x_1 \vee x_2)\wedge \\ &(\overline{x_1} \vee \overline{x_2} \vee \overline{x_2})\wedge \\ &(\overline{x_1} \vee x_2 \vee x_2)\end{aligned}$$

Tel que

$$x_1 \leftarrow F$$
$$x_2 \leftarrow T$$

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

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

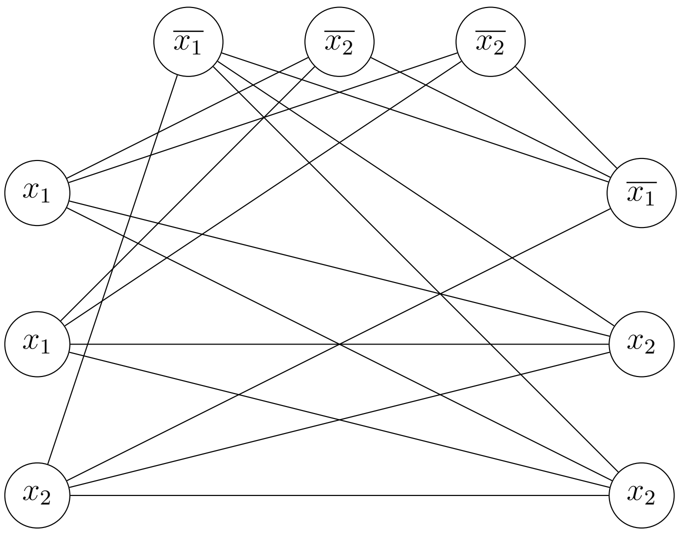
$3\ SAT \in NP$

Théorème

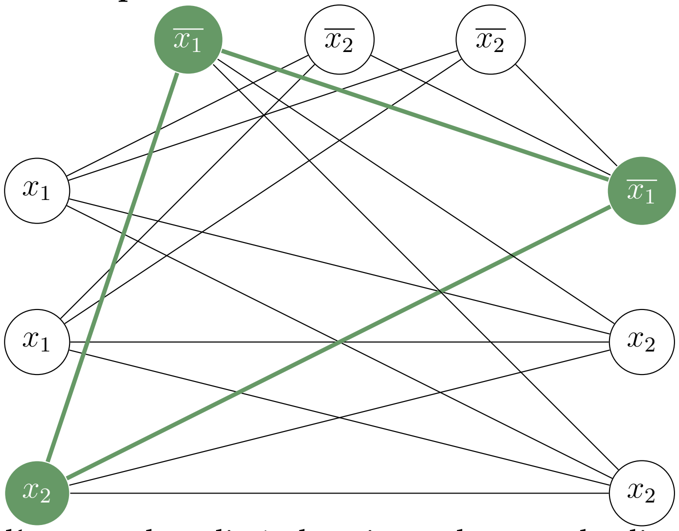
$3\ SAT \leq_p$ CLIQUE

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 $\overline{x_i}$).



$K = 3$, 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 :



$$\begin{aligned}\phi =&(x_1 \lor x_1 \lor \mathbf{x_2})\land \\ &(\overline{\mathbf{x_1}} \lor \overline{x_2} \lor \overline{x_2})\land \\ &(\overline{\mathbf{x_1}} \lor x_2 \lor x_2)\end{aligned}$$

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-Complet.

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-Complet.

On peut donc écrire par exemple :

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

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 $k = O(1)$

$$\#q_0 w_1 w_2 ... w_n \_\,\_ \cdots \_\,\_\#$$
$$\#a q_1 w_2 ... w_n \_\,\_ \cdots \_\,\_\#$$
$$...$$

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

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

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

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

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

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

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

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

↑ "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 :

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

	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