INFO-F408: Computability & complexity

Rémy Detobel

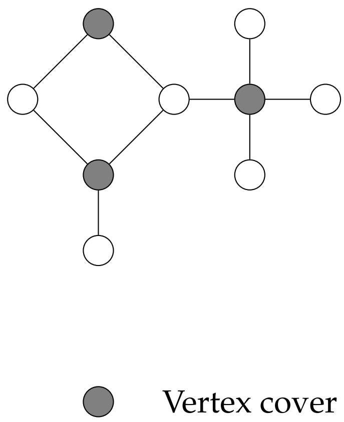
27 novembre, 2017

1 NP-Complete problems

- 3-SAT
- CLIQUE
- INDEPENDENT SET (complément de CLIQUE) $\equiv$ CLIQUE in $\overline{G}$
- VERTEX COVER
- HAMILTONIAN PATH
- SUBSET SUM

1.1 K-Independant Set in G

$\equiv$ CLIQUE in $\overline{G}$
$\equiv (n - K)$ VERTEX COVER in G



> Lemme :
> Pour $G = (V, E)$
> Si $S \leqslant V$ est un vertex cover, alors V_S est un INDEPENDANT SET

1.2 SUBSET SUM

$$
\begin{aligned}
=\Big\{ &< S, t > | S = \{s_1, s_2, ...s_k\} \\
&\exists \{y_1, ...y_l\} \subseteq S \\
&\text{tel que} \sum_{s \in S} s = t \Big\}
\end{aligned}
$$

> **Théorème**
>
> SUBSET-SUM est NP-Complet

Preuve :
Réduction depuis 3 SAT où $x_1...x_v$ sont des variables et où $c_1..c_n$ sont des clauses (conditions).

		1	2	3	...	v		1	2	3	...	m
x_1	y_1	1	0	0	0	0			1	1		
	z_1	1	0	0	0	0						
x_2	y_2	0	1	0	0	0			1	0		
	z_2	0	1	0	0	0				1		
x_3	y_3	0	0	1	0	0			1			
	z_3	0	0	1	0	0						
*	g_1	0	0	0	0	0		0	0	0	*	0
	h_i	0	0	0	0	0		0	0	0	*	0
t		1	1	1	1	1		3	3	3	3	3

$$C_2 = (x_1 \vee x_2 \vee x_3)$$
$$C_3 = (x_1 \vee \overline{x_2} \vee x_4)$$

C_3 contient $\overline{X_2}$
C_2 contient $\overline{X_1}$

*Pour chaque clause/colonne c_i, on inclut deux fois le nombre : 0...010...0 (où le 1 est placé en index i).

1. Supposons qu'il existe une instance de SAT :
   $\rightarrow$ construisons s' comme ceci : $\forall i = 1...V$ si $x_i = T$ prenons y_i dans s', z_i dans les autres cas.
   Jusqu'a ce que cela satisfasse chaque clause où chaque 1, 2 ou 3 littéraux sont égaux à “True”.
   $\forall i = 1...m$:
   — Si c_i a 1 litéral à vrai qui inclut g_i et h_i dans S'
   — Si c_i a 2 litéraux à vrai qui inclut g_i seulement.
2. Supposons qu'il existe $S' \subseteq S$ tel que $\sum S' = t$. L'assignement est construit tel que :
   $X_i = T \Leftrightarrow y_i \in S'(= F \text{ dans les autres cas}) \forall i = 1...V$
   Et ceci **n'est pas un assignement valide** pour SAT

2 Programmation dynamique

Algorithme pour SUBSET-SUM
$x[i]$ est un nombre dans S et t est la somme a calculer.

Table T
$T[i][j] = \mathrm{True} \Leftrightarrow$ il existe un sous-ensemble du premier nombre i où la somme est j
$i = 1...|S|$
$j = 0... \sum S$

Algorithme
Initialisons T, appliqué pour $i = 1 ... |S|, j = 0 ... t$.
$T[i][j] \leftarrow T[i-1][j] \vee T[i-1][j - x[i]]$
$Return T[|S|][t]$
n a la taille de l'entrée (par exemple : le nombre de bits).
La complexité de l'algorithme DP pour le problème SUBSET SUM : $|S| \times t$

$$n \simeq \Big(\sum_{x_i \in S} \log_2 x_i \Big) + \log_2 t$$

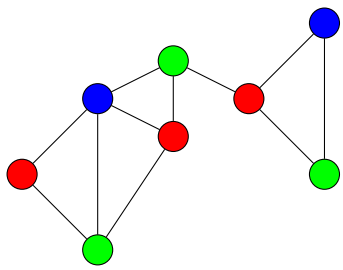
2.1 Exemple

$$\begin{aligned} S &= \{5\,000\,000\,000, 939\,000\,000, 333\,212\} \\ t &= 5\,939\,333\,212 \end{aligned}$$

Complexité : $n \leqslant 4 \times 10 \times 4 = 160$
$|S| \times t \geqslant 4 \times 5 \times 10^9$

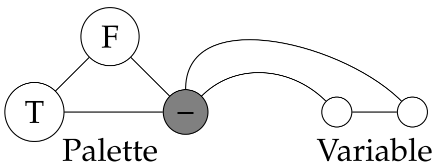
3 3-COLORING

Prenons un simple graphe non dirigé G, peut-on colorer ses points en 3 couleurs tel que deux point adjacent n'ont pas la même couleur.

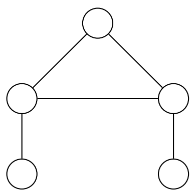


3.1 Exercice 7.27

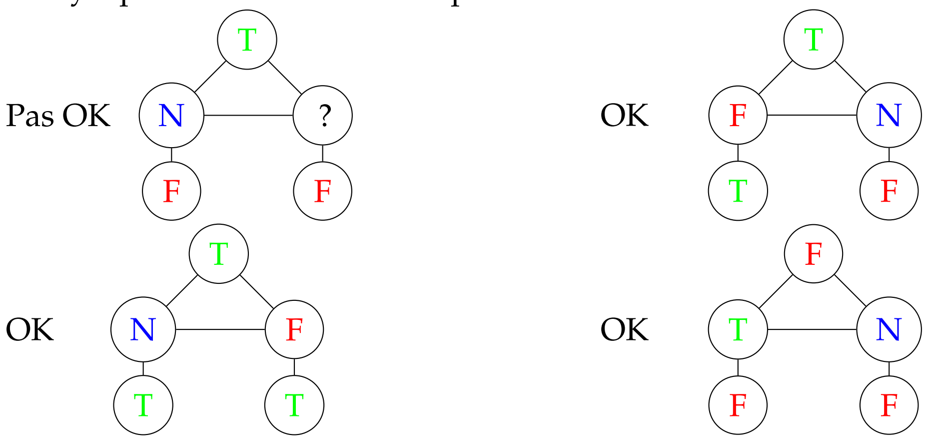
Prouver que 3-COLORING est NP-Complet



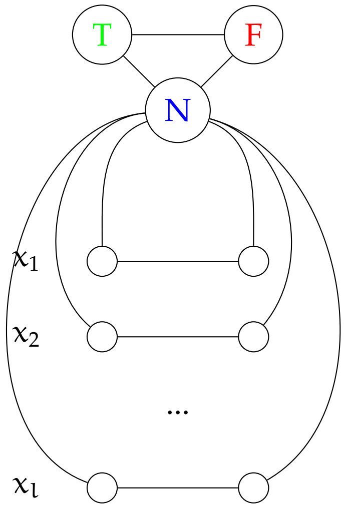
OR-Gadget



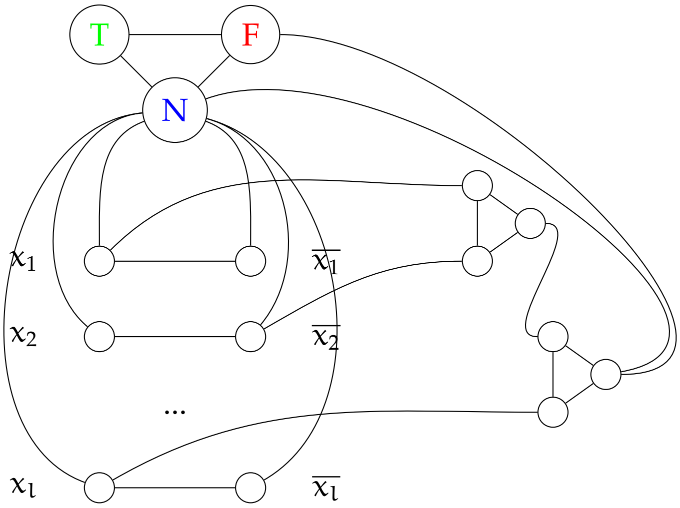
Il y a plusieurs manière de représenter le “OR-GADGET” :



Réduction depuis 3 STA : 3 CNF formule ϕ, variables : $x_1, ...x_l$:
Preuve :

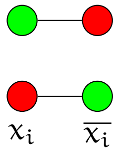


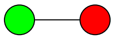
Par exemple : $(x_1 \vee \overline{x_2} \vee x_l$



$\exists 3 - \text{coloring} \Leftrightarrow y$ est satisfaisable.

1. Supposons que ϕ est satisfaisable :
   $\exists$ un SAT associé a $f : \{x_1, ...x_n\} \to \{T, F\}$. Colorer la palette en rouge vert bleu comme indiqué.

   $\forall$ varaible x_i : colorier la variable gadget et : Si $f(x_i) = T$ dans les autres cas
   $\forall$ "OR-GADGET" : utiliser la couleur A ou B pour le "second" ou gadgets et colorer A, B ou C pour le "premier".
2. Supposons $\exists$ un 3-COLORING colorer en rouge (F) vert (T) bleu (N).
   Mettre x_i a vrai si :  et a faux si