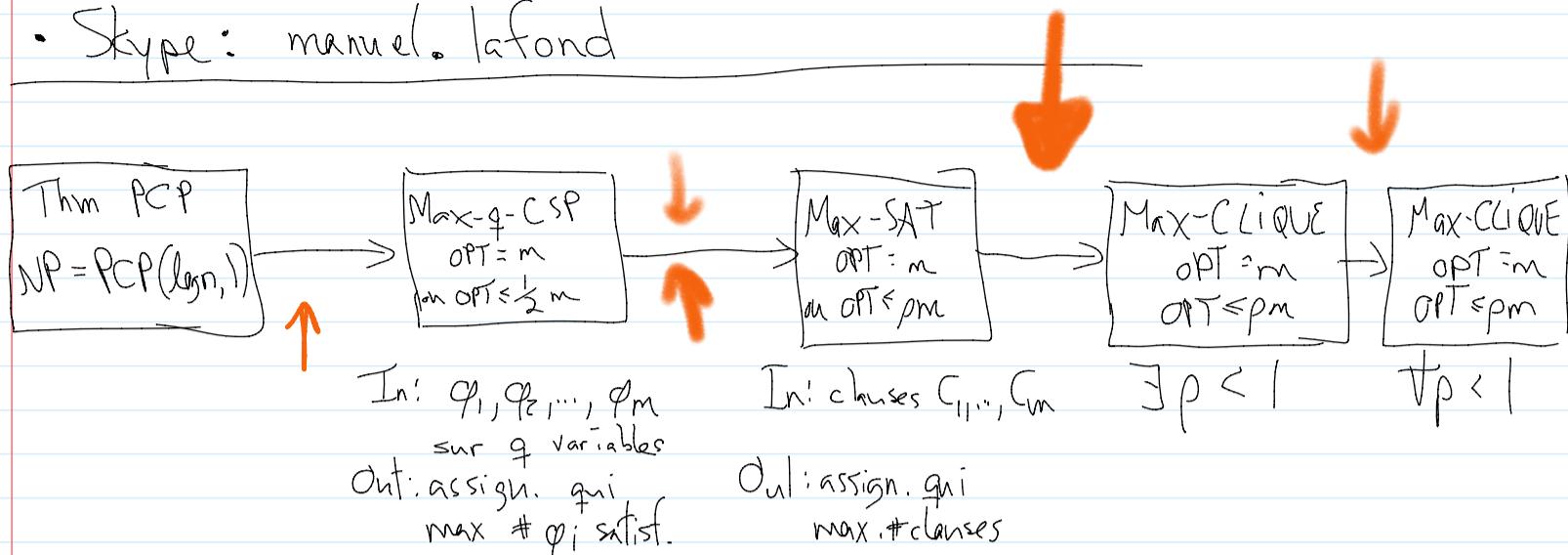


- Exercices - solutions
- Skype: manuel.lafond



Thm: Si $\boxed{NP = \text{PCP}(\log n, 1)}$, alors il est difficile de décider si une instance $\varphi = \{q_1, q_2, \dots, q_m\}$ de Max-q-CSP a :

$$\begin{aligned} \text{OPT}(\varphi) &= m \\ \text{ou } \text{OPT}(\varphi) &\leq \frac{1}{2} m \end{aligned}$$

Soit $L \in NP$. Donc puisque $NP \subseteq \text{PCP}(\log n, 1)$, alors $L \in \text{PCP}(\log n, 1)$ et il existe un vérificateur $(d \log n, q)$ -PCP V pour L .

$$\begin{aligned} w \in L &\Rightarrow \exists c \text{ t.q. } \Pr[V \text{ accepte } \langle w, c \rangle] = 1 \\ w \notin L &\Rightarrow \forall c \quad \Pr[V \text{ accepte } \langle w, c \rangle] \leq \frac{1}{2} \end{aligned}$$

Soit $w \in \{0,1\}^*$. Soit $r \in \{0,1\}^{d \log n}$. On fixe $w \in r$, soit

$$V_w^r(c) = \begin{cases} 1 & \text{si } V \text{ accepte } \langle w, c \rangle \text{ quand ses choix sont } r \\ 0 & \text{sinon} \end{cases}$$

Puisque V consulte $\leq q$ bits de c , $V_w^r(c)$ dépend de q bits d'entrée. Donc V_w^r est une fonction booléenne qui dépend de q bits (sur q variables).

Soient $r_1, r_2, \dots, r_{2^{\log n}} = \{0,1\}^{d \log n}$ et

$$Q = \{V_w^{r_1}, V_w^{r_2}, \dots, V_w^{r_{2^{\log n}}} \}$$

C'est une instance de Max-q-CSP.

— un vrai modèle pour un vrai système.



- Si $w \in L$, alors $\exists c$ t.q. $\Pr[V \text{ accepte } \langle w, c \rangle] = 1$
⇒ les bits de c correspondent à une assign.
qui satisfait chacun des V_w^i
⇒ $\text{OPT}(Q) = 2^{d \log n} := m$

- Si $w \notin L$, alors $\forall c \quad \Pr[V \text{ accepte } \langle w, c \rangle] \leq \frac{1}{2}$
⇒ pour tout c , le certificat correspond à une
assignation qui peut seulement satisfaire $\leq \frac{1}{2}$
des V_w^i
⇒ $\text{OPT}(Q) \leq \frac{1}{2} \cdot 2^{d \log n} = \frac{m}{2}$

Max-q-CSP \longrightarrow Max-SAT

$$\begin{matrix} m \\ \text{ou} \\ \leq \frac{1}{2}m \end{matrix}$$

$$\begin{matrix} m \\ \text{ou} \\ \leq pm \end{matrix}$$



$$Q = \{q_1, q_2, \dots, q_m\}$$
$$C = [2^q \text{ clauses}] \wedge [2^q \text{ clauses}] \wedge \dots \wedge [2^q \text{ clauses}]$$

$$\text{Si } \text{OPT}(Q) = m, \text{ alors } \text{OPT}(C) = m \cdot 2^q$$

$$\text{Si } \text{OPT}(Q) \leq \frac{1}{2}m, \text{ alors } \text{OPT}(C) \leq \frac{1}{2} \cdot m \cdot 2^q + \frac{1}{2}m(2^q - 1)$$

$$= \frac{m}{2} (2 \cdot 2^q - 1)$$

$$= \frac{m}{2} \cdot 2 \cdot 2^q \left(1 - \frac{1}{2 \cdot 2^q}\right)$$

$$= m \cdot 2^q \underbrace{\left(1 - \frac{1}{2^{q+1}}\right)}_{p}$$

$$= pm 2^q$$



Max-SAT

$$\text{OPT}(q) = m$$

$$\text{ou } \text{OPT}(q) \leq pm$$

Max-CLIQUE

$$\text{OPT}(G) = m$$

$$\text{ou } \text{OPT}(G) \leq pm$$

Max-CLIQUE

$$\text{OPT}(G) = m$$

$$\text{ou } \text{OPT}(G) \leq \alpha m$$

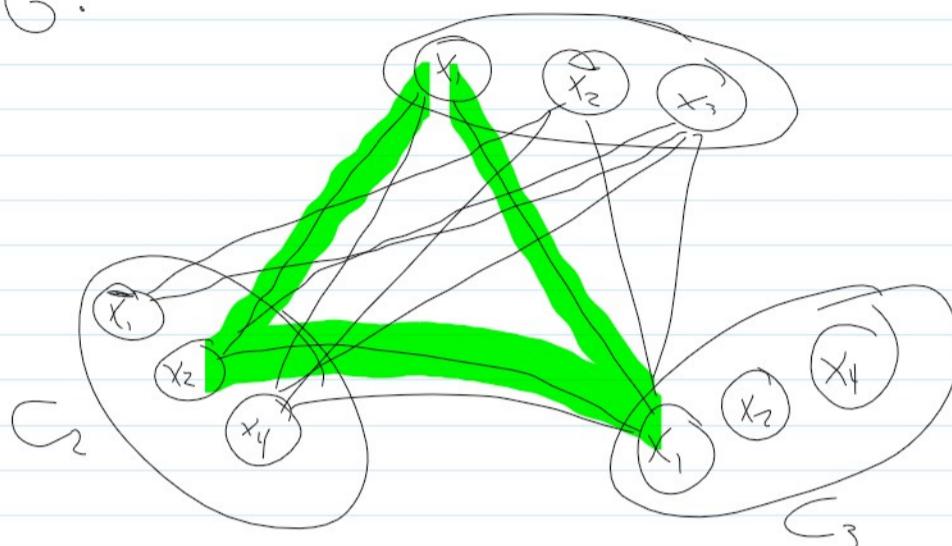
$\forall \alpha < 1$

Soit $\varphi = C_1 \wedge C_2 \wedge \dots \wedge C_m$ une instance de MAX-SAT

Soit $G = f(\varphi)$ obtenu par réduction classique de SAT vers CLIQUE.

$$\text{ex: } \varphi = (x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4) \wedge (x_1 \vee x_2 \vee x_4)$$

$G:$



- si $\text{OPT}(\varphi) = m$, alors G a une clique de taille m .

- si $\text{OPT}(\varphi) \leq pm$, on veut que $\text{OPT}(G) \leq pm$.

Soit C une clique de G . Alors C a $\leq |C|$ sommet par "grappe-clause". Ces sommets sont des littéraux non-contradictoires qui satisfont leur clause. Donc, C correspond à une assignation qui sat. $|C|$ clauses.

$$\Rightarrow \text{OPT}(\varphi) \geq |C|.$$

Sachant que $\text{OPT}(\varphi) \leq pm$, on a que

$$|C| \leq \text{OPT}(\varphi) \leq pm. \quad \blacksquare$$

Max-CLIQUE

$$\text{OPT}(G) = m$$

$$\text{on } \text{OPT}(G) \leq pm$$

$(\exists \rho)$



Max-CLIQUE

$$\text{OPT}(G) = m$$

$$\text{on } \text{OPT}(G) \leq \alpha m$$

$\forall \epsilon$

Pas de

(n^ε) -approx

pour un $\varepsilon > 0$

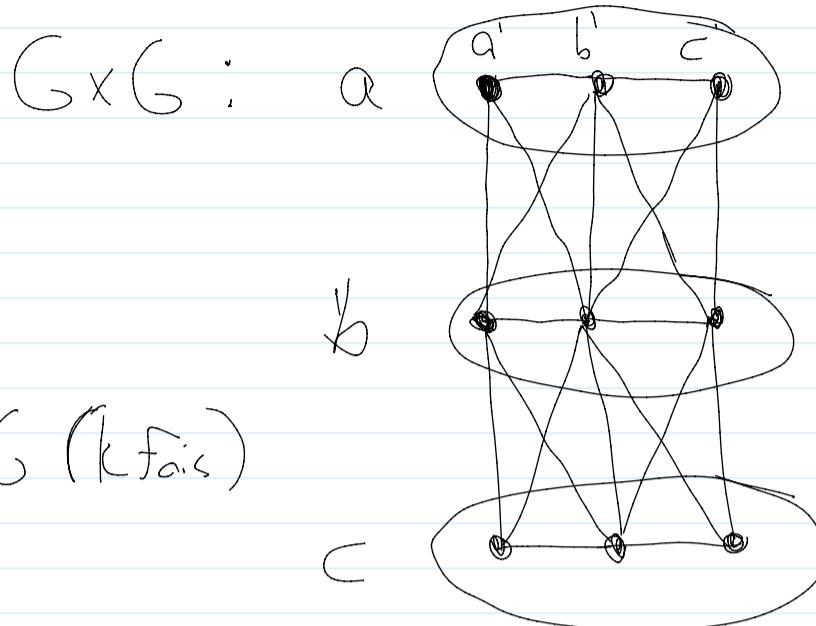
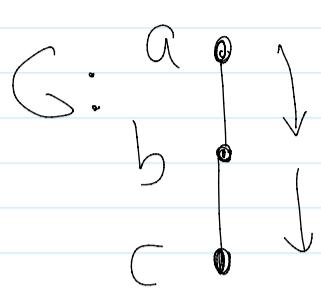
Soit G une instance de MAX-CLIQUE.

On transforme G en $G \times G$.

$$V(G \times G) = V(G) \times V(G)$$

$$V(6 \times 6) = V(6) \times V(6)$$

$$E(G \times G) = \left\{ \{(a, b), (x, y)\} : (a=x \vee ax \in E(G)) \wedge (b=y \vee by \in E(G)) \right\}$$



$$G^k = G \times G \times G \times \dots \times G \text{ (k fois)}$$

Thm: G a une clique de taille $m \iff G^k$ a une clique de taille m^k

Supposons que MAX-CLIQUE a une α -approx, $\alpha < 1$.

On pose k tel que $p^k < \alpha/2$. ($k = \text{constante}$).

$$\text{OPT}(G) = m \Rightarrow \text{OPT}(G^k) = m^k$$

$$\text{OPT}(G) \leq pm \Rightarrow \text{OPT}(G^k) \leq (pm)^k = p^k \cdot m$$

Dans, si on a une ϵ -approx pair MAX-CLIQUE, on

pour l'exécuter sur G^k .

- Si $\text{OPT}(G) = m$, alors l'approx retourne $\geq \alpha^k m^k$ sur G^k .
- Si $\text{OPT}(G) \leq pm$, alors l'approx retourne $\leq p^k m^k$ sur G^k

$$\text{Or, } p^k m^k < \alpha/2 m^k < \alpha m^k$$

\Rightarrow l'approx serait capable de distinguer si G a une digne
 $\geq m$ ou $< m$.

\Rightarrow cette approximation peut pas exister si $P \neq NP$,

ETH: Exponential-Time Hypothesis

Un algo est $2^{O(f(n))}$ poly(n) si il existe des constantes c, d, k telles que l'algo s'exécute en temps $\leq c \cdot 2^{d f(n)} \cdot n^k$.

ex: il existe un algo $2^{O(n)}$ pour SAT

pour chaque assignation C

si C satisfait φ , Accepter

Rejeter

$$O(2^n \cdot |\varphi|)$$

$$n = \# \text{variables} \quad \sim \text{poly}(|\varphi|)$$

• Un algo est $2^{o(n)}$ poly(n) si il existe une fonction $f(n) \in o(n)$ telle que l'algo est $2^{O(f(n))}$ poly(n).

• Une fct $f(n)$ est $o(n)$ si $f(n)$ grandit "+ lentement" que n . Ex: $f(n) = \sqrt{n}$, $f(n) = \log n$, $f(n) = n^{1/0.001}$
En général: $f(n)$ est $o(n)$ si $f(n) \in O(n)$ mais $n \notin O(f(n))$.

ETH: Pour tout $f(n) \in o(n)$, il n'existe pas d'algorithme pour décider 3-SAT en temps $2^{o(n)}$ poly(n), où $n = \# \text{de variables} + \# \text{de clauses}$.

(Hypothèse, conjecture)