PCP et inapproximabilité

PCP

Un retour sur NP...

- SAT :

  - n variables booléennes $(x_1,\ldots,x_n)$

  - m clauses : $(x_1 \text{ ou } \overline{x_2} \text{ ou } x_5),\ldots$

  Question : peut-on trouver une valeur de vérité satisfaisant la formule (toutes les clauses) ?

PCP

SAT est dans NP car : « étant donnée I, si on me donne une valeur de vérité, je peux vérifier en temps polynomial si cette valeur vérifie les clauses de I ».

- NP :
  - Instance I
  - Certificat : c(I) ($\rightarrow$ valeur de vérité) – de taille poly
  - Vérificateur : algo polynomial A(I,c) $\rightarrow$ {vrai,faux}

    I satisfiable : $\exists$ c : A(I,c)=vrai

    I non satisfiable : $\forall$ c : A(I,c)=faux

PCP

Certificat SAT : taille *n*

- Pourrais-je vérifier le certificat sans lire (dévoiler) toute la preuve ?
  - (Vérificateur) : « Mmmmh, peu de chance que je réponde tout le temps correctement si je n'ai pas toute la preuve »
  - OK : on lit aléatoirement certains bits de la preuve, et on veut un vérificateur qui marche 'souvent'

PCP

- Théorème PCP (Arora et al. 1992)

*Il existe un vérificateur presque parfait qui lit un nombre **constant de bits** de la preuve !*

Quel que soit $\varepsilon>0$: vérificateur A_ε :

- $O(\log(|I|))$ bits aléatoires
- lit un nombre **constant de bits** de la preuve
- I satisfiable : $\exists c : Pr(A(I,c)=vrai)=1$
- I non satisfiable : $\forall c : Pr(A(I,c)=faux)>=1-\varepsilon$

PCP et inapproximabilité

A partir d'une instance I de SAT

- Exécuter le vérificateur A pour tous les f=O(log n) bits aléatoires et les D bits de la preuve lus

  (→ temps polynomial)

- Construire une graphe G : sommets = cas où le vérificateur accepte
- I satisfiable → certificat où le vérif. accepte tout le temps → 'gros' stable dans G
- I non sat → le vérificateur refuse presque tout le temps → tous les stables de G sont 'petits'

  → stable n'est pas dans APX

PCP et inapproximabilité

Exécuter le vérificateur pour tous les f=O(log n) bits aléatoires et les D bits de la preuve lus,

I satisfiable : il existe c tel Pr(A(I,c)=vrai)=1

I non satisfiable : pour tout c, Pr(A(I,c)=vrai)<=ε

Bits de la preuve (2^D possibilités)

Bits aléatoires obtenus

- 2^f possibilités

	000	001	010	...	111
f_1					
f_2					
f_3					
...					
$f_{2\text{^}f}$					

PCP et inapproximabilité

Bits de la preuve (2^D possibilités)

Bits aléatoires obtenus

2^f possibilités

	000	001	010	...	111
f_1	●		●		●
f_2		●	●		
f_3					
...					
f_2^f	●				●

Sommet lorsque le vérificateur accepte (dans la case correspondante)

PCP et inapproximabilité

Bits de la preuve (2^D possibilités)

Bits aléatoires obtenus

2^f possibilités

	000	001	010	...	111
f_1	●		●		●
f_2		●	●		
f_3					
...					
f_2^f	●				●

			X		X				X		

Bits lus avec f_1

					X		X				X

Bits lus avec f_2

PCP et inapproximabilité

Bits de la preuve (2^D possibilités)

Bits aléatoires obtenus

2^f possibilités

	000	001	010	...	111
f_1					
f_2					
f_3					
...					
f_2^f					

			X		X				X		

Bits lus avec f_1

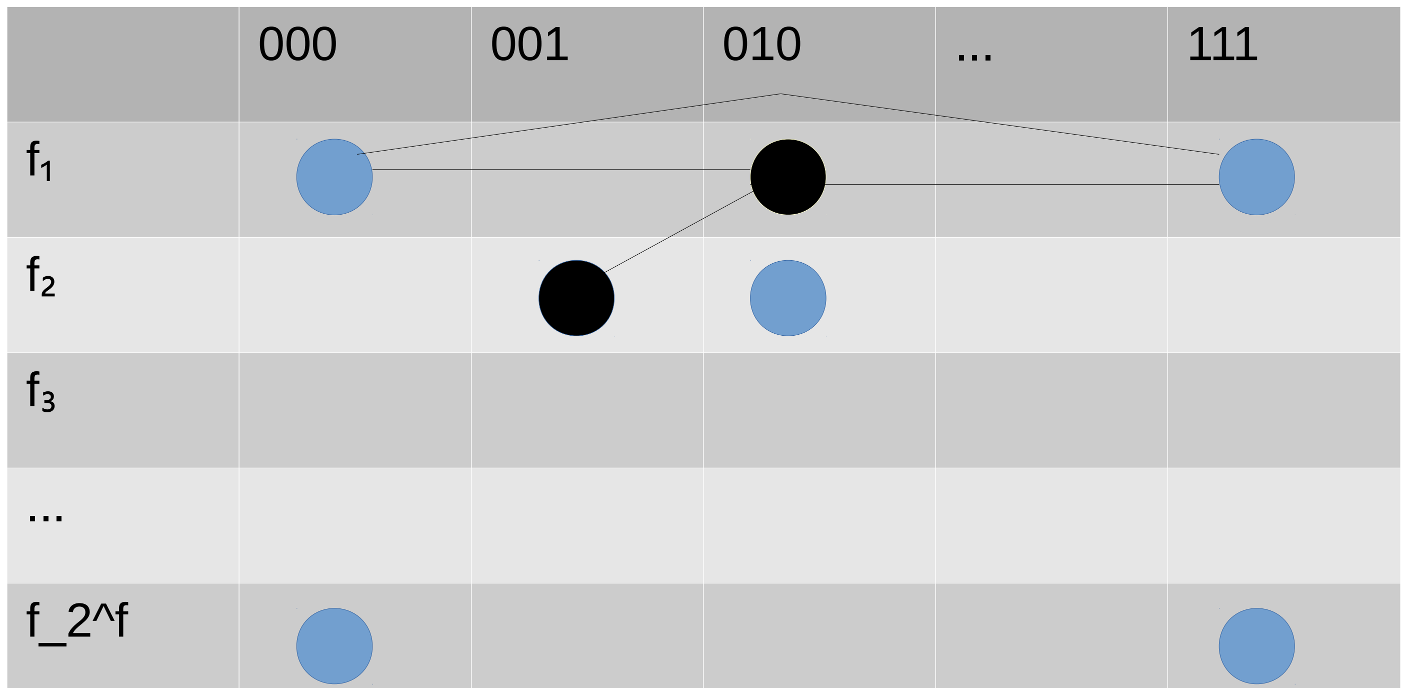
					X		X				X

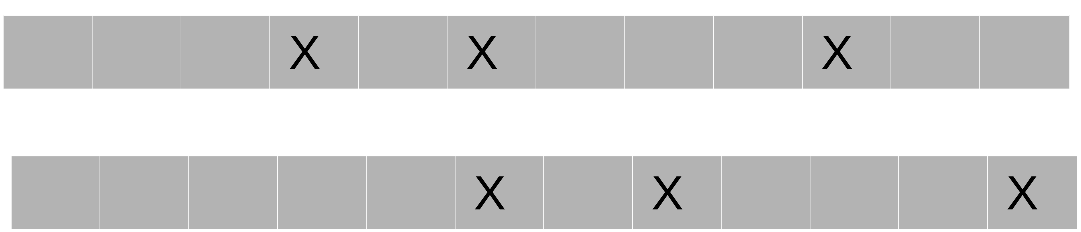
Bits lus avec f_2

PCP et inapproximabilité

Bits de la preuve (2^D possibilités)

Ligne=clique





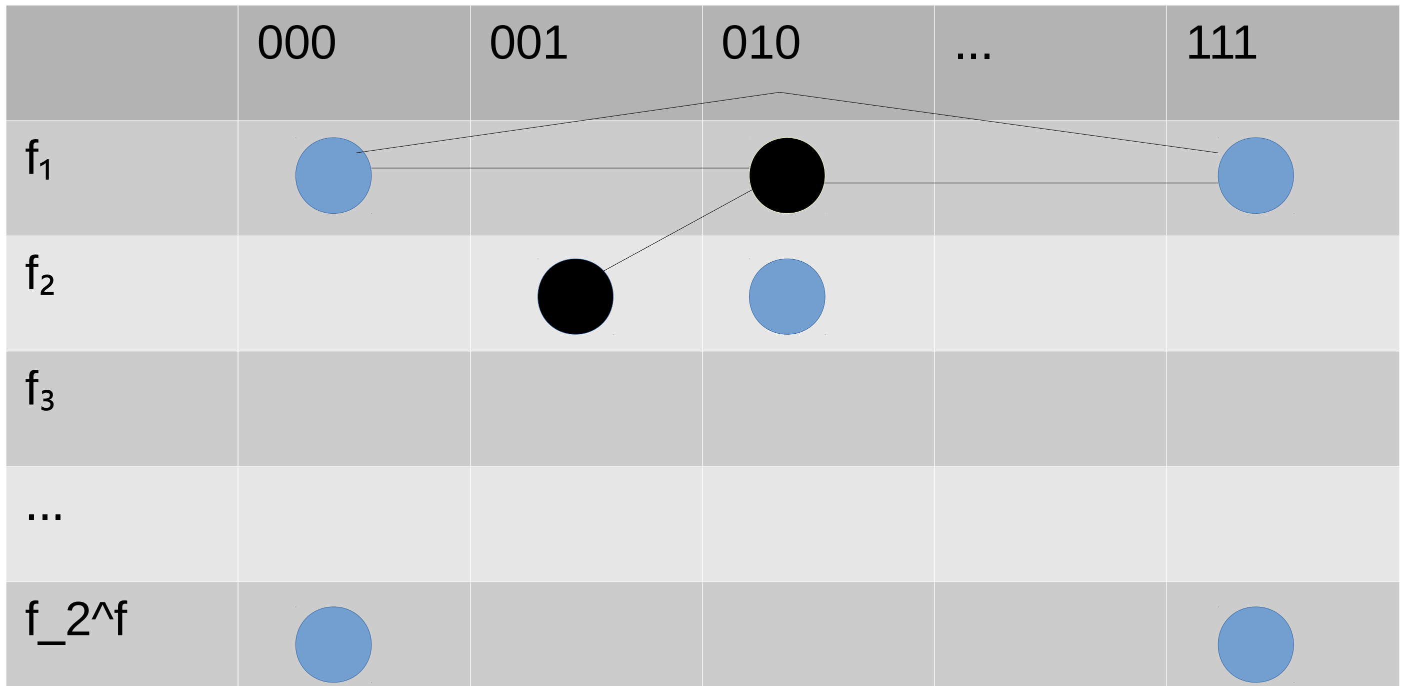
Bits lus avec f_1

Bits lus avec f_2

PCP et inapproximabilité

Bits de la preuve (2^D possibilités)

Ligne=clique



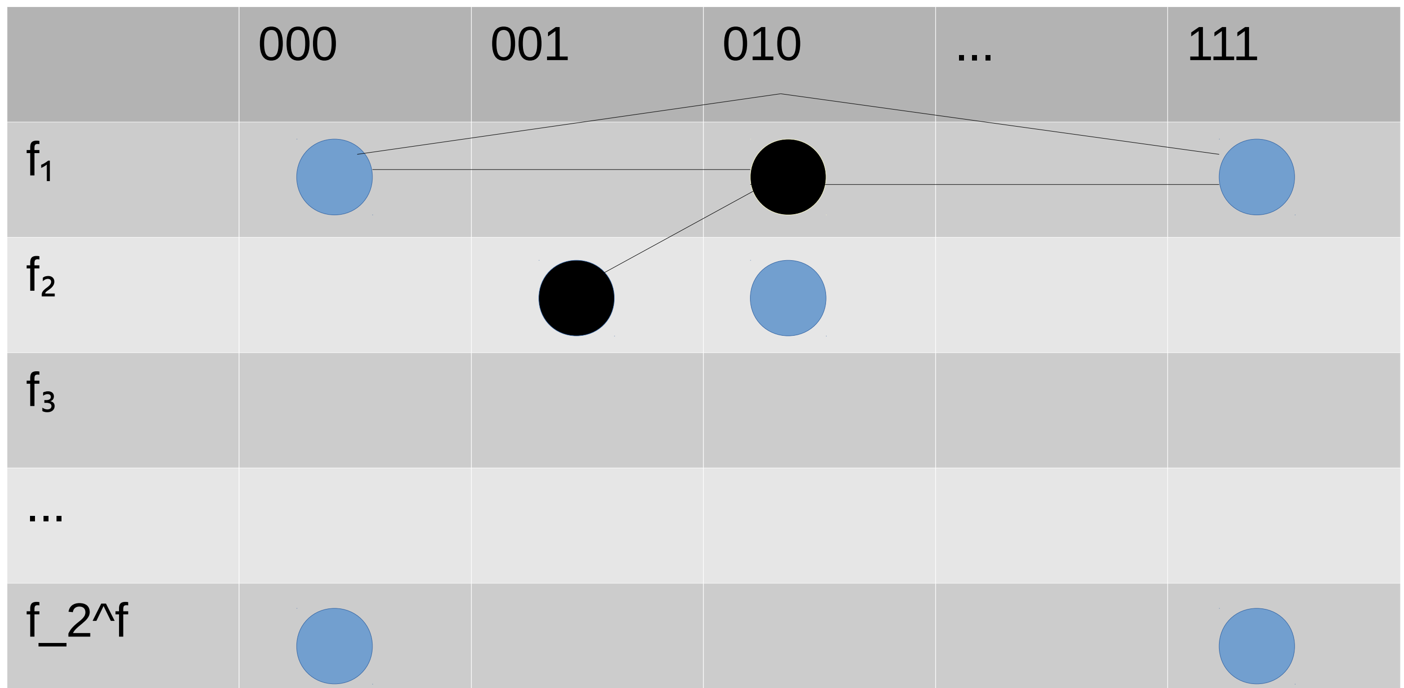
Si I SAT : certif c tq A(I,c)=vrai pour tout f_i

→ stable de taille 2^f (sommet correspondant à chaque ligne)

PCP et inapproximabilité

Bits de la preuve (2^D possibilités)

Ligne=clique



Si I non SAT : stable de taille > ε 2^f ?

→ correspond à un certificat c tq Pr(A(I,c)=vrai)> ε 2^f / 2^f=ε

→ impossible

PCP et inapproximabilité

Si I SAT : stable max $=2^f$

Si I non SAT : stable max $\leq \varepsilon\, 2^f$

→ Un algo mieux que ε-approché pour stable permet de résoudre SAT !