Master Informatique Fondamentale (1ère année)
Algorithmes d'approximation — **TD6**

Forces et faiblesses de la randomisation

novembre 2004

Exercice 1 : MAX-CUT. Soit A un algorithme randomisé qui constitue une α-approximation pour un problème de maximisation Π (*i.e.* pour toute instance I, $\mathbb{E}\big(A(I)\big) \geq \alpha OPT(I)$). Une famille d'instances critiques pour A est une famille infinie (I_n) d'instances telle que

$$\limsup_{n\to\infty} \frac{\mathbb{E}\big(A(I_n)\big)}{OPT(I_n)} = \alpha.$$

Question 1. *Donner une famille d'instances critiques pour l'algorithme randomisé du cours approximant le problème MAX-CUT.*

Réponse : L'algorithme sélectionne en moyenne la moitié des arêtes. Un graphe dans lequel la coupe maximum est constituée de toutes les arêtes est donc critique pour l'algorithme. La famille d'instances suivante répond à la question :

$$G_n = \big(\{u_1, \ldots, u_n, v_1, \ldots, v_n\}, \{(u_i, v_i), 1 \leq i \leq n\}\big).$$

□

Exercice 2 : MAX-SAT. Étant donné un ensemble de variables $x_1, \ldots, x_n$ et un ensemble de clauses $C_1, \ldots, C_k$ (disjonctions de variables et de négations de variables) définies sur ces variables, le problème MAX-SAT est de trouver une valuation des variables qui maximise le nombre de clauses satisfaites.

Question 2. *Montrer que le problème est NP-difficile. Montrer que l'algorithme randomisé qui tire uniformément sur $\{0, 1\}$ une valuation est une $\frac{1}{2}$-approximation. Donner une famille d'instance critiques.*

Réponse : Le problème de maximisation est NP-difficile car il contient SAT : en effet, demander si le nombre maximum de clauses satisfaisables dépasse le nombre total de clauses revient à demander si la formule est satifaisable.

On cherche à maximiser le nombre de clauses satisfaites, il est donc inutile de considérer les clauses qui ne sont pas satisfaisables (*i.e.* celles qui contiennent une variable et sa négation). Par ailleurs, on peut supposer sans perte de généralité que chaque clause contient au plus 1 occurrence de chaque variable. Considérons la clause (satisfaisable) C_i qui contient les variables $v_{i_1}, \ldots, v_{i_p}$. La probabilité qu'une exécution de l'algorithme satisfasse C_i est :

$$\Pr\big(A(C_i) = 1\big) = 1 - \frac{1}{2^{i_p}} \tag{1}$$

Car, le tirage de la valuation étant uniforme, les valeurs des variables sont des évènements indépendants. Si X_i est la variable aléatoire représentant la valeur (0 ou 1) de la clause C_i, on a alors

$$\mathbb{E}(X_i) \geq \frac{1}{2}.$$

Or, en notant A les nombre de clauses satisfaites par l'algorithme, et par linéarité de l'espérance, on a :

$$\mathbb{E}(A) = \sum_{i \in J} \mathbb{E}(X_i) \geq \frac{|J|}{2},$$

où J est l'ensemble des clauses satifaisables. Comme $|J|$ est un majorant de OPT, on a bien motré que l'algorithme est une 1/2-approximation.

D'après l'équation 1, l'algorithme satisfait en moyenne la moitié des clauses à une variable. Les instances où chaque clause contient exactement une variable distincte sont donc critiques. □

On se restreint, pour la question suivante seulement, aux instances dont les clauses sont toutes de taille au moins m et au plus M.

Question 3. *Quel est le facteur d'approximation de l'algorithme pour ces instances (faire une analyse et donner une famille d'instances critiques) ? Quelle est la "consommation en aléatoire" de l'algorithme (nombre de bits, type d'indépendance) ?*

Réponse : D'après l'équation 1 et si les clauses sont de taille au moins m, on a $\mathbb{E}(A) \geq |J|(1-2^{-m})$ et le facteur d'approximation de l'algorithme est alors $1-2^{-m}$. Par ailleurs, s'il y a au plus M variables par clauses, l'équation 1 reste vraie en supposant seulement la M-à-M indépendance des valeurs des variables. □

Retour au cas général.

Question 4. *Étudier l'algorithme suivant : soit τ une valuation quelconque (par exemple, celle qui met toutes les variables à faux) ; sur l'instance I évaluer le nombre de clauses satisfaites par τ et le nombre de clauses satisfaites par $\overline{\tau}$ (complémentaire de τ) puis renvoyer celle de τ ou $\overline{\tau}$ qui maximise ce nombre.*

Réponse : Nous allons montrer que cet algorithme A est une $1/2$-approximation :

Soit $I = \{C_1, .., C_k | x_1, .., x_n\}$ une instance du problème de $MAX-SAT(n,k)$ et $i \in 1..k$:
On pose $C_i = X_{i1} \vee X_{i2} \vee ... \vee X_{ip_i}$ où X_α est une variable ou une négation de variable.
Si $X_{i1} = x_j$ alors :
– Si $\tau(x_j) = 1$, on a $C_i(\tau) = 1$.
– Si $\tau(x_j) = 0$, on a $\neg\tau(x_j) = 1$ et donc $C_i(\neg\tau) = 1$.
C_i est donc satisfaite par τ ou par $\neg\tau$.
De même si $X_{i1} = \neg x_j$.

Chacune des n clauses est donc satisfaite par τ ou par $\neg\tau$ donc une de ces valuations satisfait au moins $\lceil \frac{n}{2} \rceil$ clauses.
On a donc prouvé que $A(I) \geqslant \lceil \frac{n}{2} \rceil$ or $OPT(I) \leqslant n$, On a ainsi :

$$A(I) \geqslant \frac{1}{2} OPT(I)$$

L'algorithme A est polynomial, c'est donc bien une $1/2$-approximation. □

Question 5. *On fixe une instance du problème $\wedge C_j$ sur les variables $x_1, \ldots, x_n$ et p valeurs booléennes $b_1, \ldots, b_p$. Calculer l'espérance conditionnelle du nombre de clauses satisfaites par une valuation aléatoire τ sachant que $\tau(x_i) = b_i$ pour $1 \leq i \leq p$.*

Réponse : On peut, pour une instance I du problème, calculer aisement l'espérance du nombre de clauses satisfaites par une valuation aléatoire :

$$E(I) = \sum_{i=1}^{k} (1 - (\frac{1}{2})^{p_i})$$

Pour calculer l'espérance conditionnelle sachant que $\tau(x_i) = b_i$ pour $i = 1..p$, on va transformer notre instance I en I' sur laquelle on pourra appliquée la formule ci-dessus.
$E = 0$
Pour $i = 1..k$
On note $C_i = X_{i1} \vee X_{i1} \vee .. \vee X_{ip_i}$
Pour $j = 1..p_i$
Si $\tau(X_{ij}) = 1$
Retirer X_{ij} de C_i

Si $\tau(X_{ij}) = 0$
Retirer C_i
$E = E + 1$
Calculer E' =espérance de I
Renvoyer $E + E'$

□

Exercice 3 : MAX-k-CUT. Étant donné un graphe muni d'une fonction de poids sur les arêtes, le problème est de trouver une partition des sommets en k ensembles telle que le poids total des arêtes dont les extrémités ne sont pas dans le même ensemble est maximum.

Question 6. *Donner un algorithme glouton qui soit une $(1 - \frac{1}{k})$-approximation. Trouver une famille d'instances critiques.*

Réponse : Idée de l'algorithme glouton :
Partir du multi-ensemble $\varnothing^k$
Pour $i = 1..n$
Mettre v_i dans l'ensemble qui maximise le poid des arrètes sortantes

Plus formellement :
INPUT : $G = (V, E), k \in N, w : E \to \Re$.
OUTPUT : $(S_i)_{i=1..k}$ k-partition de V.

$n = |V|$
Pour $i = 1..k$
$S_i = \varnothing$
Pour $j = 1..n$
$max = 0$
$i = 1$
Pour $l = 1..k$
$temp = \sum_{v\ voisin\ de\ j, v \notin S_l} w(jv)$
Si $temp > max$
$max = temp$
$i = l$
$S_i = S_i \cup \{j\}$
Renvoyer $(S_i)_{i=1..k}$.

Soit I une instance du problème de MAX-k-CUT, on note :
-$W(I)$ la somme des poids des arrètes de I.
-$A(I)$ le poid de la solution retournée par l'algorithme.
-$OPT(I)$ le poid de la solution optimale.

Montrons par récurrence sur n que $A(I) \geqslant (1 - \frac{1}{k})W(I)$:

Si le graphe ne contient qu'un sommet on a $A(I) = OPT(I) = 0$.

Si $A(I) \geqslant (1 - \frac{1}{k})W(I)$ pour tout I de taille $n - 1$:
Soit $I : G = (V, E), k, w$ une instance du problème de MAX-k-CUT de taille n, on note $I' : G' = (V', E'), k, w$ l'instance constituée des $n - 1$ premiers sommets de I.
Cet algorithme réalise lors du calcul de $A(I)$, le calcul de $A(I')$. On a donc, lors de la dernière itération :

$$A(I) = A(I') + \sum_{v\ voisin\ de\ j, v \notin S_i} w(jv)$$

Or, par hypothèse de récurrence :

$$A(I') \geqslant (1 - \frac{1}{k})W(I')$$

et d'après l'algorithme

$$\sum_{v\ voisin\ de\ j, v\notin S_i} w(jv) = max_{l=1..k}(\sum_{v\ voisin\ de\ j, v\notin S_l} w(jv))$$

Ce terme maximum est supérieur à la moyenne des termes :

$$M = moy_{l=1..k}(\sum_{v\ voisin\ de\ j, v\notin S_l} w(jv))$$

$$M = \frac{\sum_{l=1..k}\sum_{v\ voisin\ de\ j, v\notin S_l} w(jv)}{k}$$

$$M = \frac{\sum_{l=1..k}(\sum_{v\ voisin\ de\ j} w(jv) - \sum_{v\ voisin\ de\ j, v\in S_l} w(jv))}{k}$$

$$M = \frac{k\sum_{v\ voisin\ de\ j} w(jv) - \sum_{v\ voisin\ de\ j} w(jv)}{k}$$

$$M = (1-\frac{1}{k})\sum_{v\ voisin\ de\ j} w(jv)$$

On a donc :

$$A(I) \geqslant (1-\frac{1}{k})W(I') + (1-\frac{1}{k})\sum_{v\ voisin\ de\ j} w(jv)$$

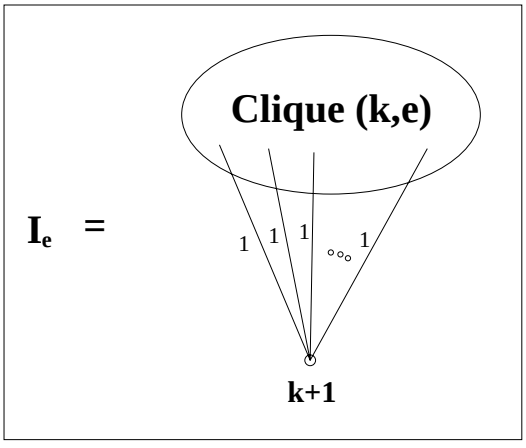
et comme $\{v\ voisin\ de\ j\} = E - E'$:

$$A(I) \geqslant (1-\frac{1}{k})W(I)$$

Ce qui prouve l'héréditée de la proposition et achève la récurrence :
$\forall I \in$ MAX-k-CUT, $A(I) \geqslant (1-\frac{1}{k})W(I)$

De plus $W(I)$ est une borne supérieure triviale de l'optimal :
$OPT(I) \leqslant W(I)$ donc $A(I) \geqslant (1-\frac{1}{k})OPT(I)$. Cet algorithme polynomial est donc bien une $(1-\frac{1}{k})$-approximation.

Famille d'instances critiques :
On pose I_ϵ l'instance composé d'une clique de taille $k+1$, où toutes les arrêtes ont un poid ϵ sauf celles issus du dernier sommet qui ont un poid 1.



L'algorithme mettra les k premiers sommets dans des parties distinctes et le dernier sera placé arbitrairement. On a donc :

$$A(I_\epsilon) = \frac{k(k-1)}{2}\epsilon + k - 1$$

La solution optimale, pourvu que ϵ soit suffisamment petit, consiste à isoler le dernier sommet et répartir les autres sommets dans les autres parties. On a alors :

$$OPT(I_\epsilon) = k + (\frac{k(k-1)}{2} - 1)\epsilon$$

On a ainsi :

$$\frac{A(I_\epsilon)}{OPT(I_\epsilon)} = 1 - \frac{1-\epsilon}{k + (\frac{k(k-1)}{2} - 1)\epsilon}$$

et donc

$$limite_{\epsilon \to 0}(\frac{A(I_\epsilon)}{OPT(I_\epsilon)}) = 1 - \frac{1}{k}$$

□

Question 7. *Étudier l'algorithme consistant à placer chaque sommet dans un ensemble tiré aléatoirement uniformément.*

Réponse : Montrons que cet algorithme est une $(1 - \frac{1}{k})$-approximation.
Nous allons utiliser la linéarité de l'espérance :

$$\mathbb{E}(A(I)) = \sum_{uv \in E} \mathbb{E}(w'(uv))$$

avec $w'(uv) = w(uv)$ si u et v sont dans des ensembles différants, $w'(uv) = 0$ sinon.
$\mathbb{E}(w'(uv)) = \mathbb{P}_r(\text{"}u \text{ et } v \text{ sont dans des ensembles différants"})w(uv)$
$\mathbb{E}(w'(uv)) = (1 - \mathbb{P}_r(\text{"}u \text{ et } v \text{ sont dans le même ensembles"})w(uv)$
Or, les événements "u et v sont dans i" sont mutuellements exclusifs donc la probabilité de l'union est la somme des probabilités :
$\mathbb{E}(w'(uv)) = (1 - \sum_{i=1..k} \mathbb{P}_r(\text{"}u \text{ et } v \text{ sont dans } i\text{"}))w(uv)$
Comme les tirages sont supposés indépandants :
$\mathbb{E}(w'(uv)) = (1 - \sum_{i=1..k} \mathbb{P}_r(u \in i)\mathbb{P}_r(v \in i))w(uv)$
Comme les tirages sont supposés uniformes :
$\mathbb{E}(w'(uv)) = (1 - \sum_{i=1..k} \frac{1}{k^2})w(uv)$
$\mathbb{E}(w'(uv)) = (1 - \frac{1}{k})w(uv)$
Ainsi :
$\mathbb{E}(A(I)) = (1 - \frac{1}{k})W(I)$
Et comme $W(I)$ est un majorant de $OPT(I)$, on a bien :
$\mathbb{E}(A(I)) \geqslant (1 - \frac{1}{k})OPT(I)$
De plus, l'algorithme est polynomial, c'est une $(1 - \frac{1}{k})$-approximation.

□

★