École polytechnique
Promotion 2001

Majeure informatique

Cours de conception et analyse d'algorithmes

PC 5 – algorithmes d'approximation

1. Voyageur de commerce avec inégalité triangulaire

Ce problème consiste en la donnée d'un graphe complet non-orienté à n sommets numérotés de 1 à n (et donc $n(n-1)/2$ arêtes) ; à chaque arête est associée une distance. Le voyageur de commerce part du sommet 1 du graphe, et passe une fois et une seule par chacun des autres sommets du graphe avant de finalement revenir en 1, et ce en parcourant la distance minimale. La version "décision" de ce problème est un problème NP-complet.

On suppose que la distance vérifie l'inégalité triangulaire : pour tous sommets u, v et w, $d(u,v) \leqslant d(u,w) + d(w,v)$. On considère l'algorithme suivant :
- calculer un arbre couvrant minimal T enraciné en 1.
- renvoyer le trajet obtenu en regardant l'ordre dans lequel les sommets apparaissent lors d'un parcours en profondeur de l'arbre T.

Soit H le tour ainsi obtenu.

i. Quelle est la complexité de cet algorithme ?

ii. Soit H^* une tournée optimale. Montrer que le poids $c(T)$ de T est inférieur ou égal à la longueur totale de H^*.

iii. En utilisant l'inégalité triangulaire, montrer que la longueur $c(H)$ de H est inférieure ou égale à $2c(T)$.

iv. En déduire un algorithme d'approximation à un facteur 2.

v. Montrer qu'il existe des cas où l'algorithme construit un tour de coût supérieur à $(2-\epsilon)c(H^*)$, et donc que l'analyse ne peut pas être améliorée.

2. Couverture par ensembles.

On considère une famille F de m sous-ensembles de $\{x_1, \ldots, x_n\}$, où chaque élément appartient à au plus f ensembles. On souhaite choisir des ensembles de la famille qui "couvrent" les éléments, c'est-à-dire tels que chaque x_i appartienne à au moins un des ensembles choisis. Le but est de minimiser le nombre d'ensemble choisis. Soit OPT la valeur (nombre d'ensembles) de la solution optimale.

i. Montrer que le version "décision" est un problème NP-complet.

ii. Modéliser ce problème par un programme linéaire en entiers. On utilisera une variable x_S pour chaque ensemble S de F.

iii. On considère l'algorithme qui résoud le programme linéaire en réels, obtenant la solution fractionnaire $(x^*_S)_S$, puis arrondit chaque x^*_S à 1 s'il est $\geq 1/f$ et à 0 sinon. Montrer que cet algorithme construit une couverture d'ensembles.

iv. Montrer que le résultat du programme linéaire a valeur inférieure ou égale à OPT. Montrer que l'algorithme construit une solution de valeur au plus $f \cdot OPT$. En déduire que nous avons un algorithme d'approximation à un facteur f.

v. Montrer par un exemple dans le cas $f = 2$ que l'analyse ne peut pas être améliorée, c'est-à-dire que l'algorithme donne dans certains cas un résultat arbitrairement proche de $f \cdot OPT$.

3. Stable maximum

STABLE est le problème consistant à trouver un ensemble stable (sous-ensemble Y des sommets tel qu'aucune arête n'ait ses deux extrémités dans Y) de cardinal maximum dans un graphe non-orienté.

i. À partir d'un graphe $G = (X, E)$, on construit le graphe $G^{(2)}$ dont l'ensemble des sommets est $X \times X$ et dont les arêtes sont les paires $\{(x, x'), (y, y')\}$ telles que soit $x = y$ et $(x', y') \in E$, soit $(x, y) \in E$. Montrer que, si G admet un ensemble stable de taille k, alors $G^{(2)}$ admet un ensemble stable de taille k^2.

ii. Inversement, montrer que, si $G^{(2)}$ admet un ensemble stable de taille m, alors G admet un ensemble stable de taille $\sqrt{m}$.

ii. Soit A un algorithme d'approximation pour STABLE construisant un stable de taille au moins $c \cdot \mathrm{OPT}$. Construire algorithme A' d'approximation pour STABLE construisant un stable de taille au moins $c' \cdot \mathrm{OPT}$, où

$$c' = 1 - \sqrt{1 - c}.$$

iii. En déduire que, s'il existe un algorithme d'approximation à un facteur constant pour STABLE, alors il existe un schéma d'approximation.

4. Le problème MAX2SAT.

i. Proposer un algorithme d'approximation à un facteur 1/2 pour MAX2SAT.

ii. On suppose que toutes les clauses ont taille exactement égale à 2. On considère l'algorithme aléatoire qui prend les variables $x_1, \ldots, x_n$ une à une et fixe leurs valeurs de vérité au hasard. Montrer que le nombre moyen de clauses satisfaites est $(3/4)m$ pour une instance à m clauses et n variables au total.

iii. On considère l'algorithme glouton qui prend les variables $x_1, \ldots, x_n$ une à une et fixe leurs valeurs de vérité comme suit : soit F la formule courante au début de la i-ième étape. Soit a le nombre de clauses de taille 2 contenant le littéral x_i, a' le nombre de clauses de taille 1 contenant le littéral x_i, b le nombre de clauses de taille 2 contenant le littéral $\overline{x_i}$, et b' le nombre de clauses de taille 1 contenant le littéral $\overline{x_i}$. Si $a + 2a' > b + 2b'$ alors mettre x à vrai, sinon mettre x à faux.

On suppose qu'initialement toutes les clauses ont taille exactement égale à 2. Montrer par récurrence que l'algorithme mixte qui fait une choix glouton durant les i premières étapes, et un choix au hasard durant les $n - i$ dernières étapes, donne une approximation à un facteur 3/4 de l'optimal.

En déduire que l'algorithme glouton donne une approximation à un facteur 3/4 de l'optimal si initialement toutes les clauses ont taille exactement égale à 2.

iv. On ne suppose plus désormais que les clauses ont taille exactement égale à 2. Proposer un algorithme d'approximation à un facteur 3/4.