MIM2 - Algorithmes d'approximation- TD 11
Emmanuelle Lebhar
elebhar@ens-lyon.fr - Bureau 317

Schéma primal-dual et ordonnancement hétérogène
8 Décembre 2003

Exercice 1 (Schéma primal-dual et couverture par ensembles)

On utilise le schéma primal-dual pour obtenir une f-approximation pour la couverture par ensembles en prenant $\alpha = 1$ et $\beta = f$. On travaille sur la paire de programmes primal-dual suivante :

$$\begin{array}{lll} \text{minimiser} & \displaystyle\sum_{s\in\mathcal{S}} c_S x_S & \\ \text{sous les contraintes} & \displaystyle\sum_{S:e\in S} x_S \geq 1, & e \in U \\ & x_S \geq 0, & S \in \mathcal{S} \end{array}$$

$$\begin{array}{lll} \text{maximiser} & \displaystyle\sum_{e\in U} y_e & \\ \text{sous les contraintes} & \displaystyle\sum_{e:e\in S} y_e \leq c(S), & S \in \mathcal{S} \\ & y_e \geq 0, & e \in U \end{array}$$

1. Donnez les conditions de complémentarité primale et duale.
   On dit qu'un ensemble S est *plein* si $\sum_{e:e\in S} y_e = c(S)$. Interprétez simplement les conditions de complémentarité primales et la réalisabilité de la solution duale.
2. Comme $\mathbf{x}$ sera à valeurs dans $\{0, 1\}$, comment pouvez-vous reformuler les conditions duales simplement ? Qu'en concluez-vous au vu de la définition de f ?
3. Proposez un algorithme simple basé sur ces deux ensembles de conditions.
4. Montrez que c'est une f-approximation.
5. Proposez une instance critique.

Exercice 2 (Ordonnancement hétérogène)

Étant donné un ensemble de tâches J, un ensemble de machines M, et les temps d'éxécution $p_{i,j} \in \mathbb{Z}^+$ de chaque tâche $j \in J$ sur la machines $i \in M$, le problème de l'*ordonnancement hétérogène* (en anglais : scheduling or unrelated parallel machines) consiste à ordonnancer les tâches sur les machines de façon à minimiser le temps d'exécution total, i.e., la date de fin d'exécution de la dernière des tâches exécutées par les machines.
On note n le nombre de tâches et m le nombre de machines.

1. En notant $x_{i,j}$ la variable qui indique si la tâche j est ordonnancée sur la machine i, proposez un programme linéaire entier qui encode ce problème.
2. Montrez, à travers un exemple simple, que le saut intégral de ce programme en nombres entiers n'est pas borné. Où est le problème ?
3. *On va utiliser la méthode de l'élagage paramétré pour contourner cette difficulté en définissant une famille de programmes $LP(T)$ pour $T \in \mathbb{N}$, $LP(T)$ restreint les variables aux couples (i, j) qui satisfont $p_{i,j} \leq T$ et fait le pari que l'optimal du temps d'exécution est inférieur à T (donc remplace t par T).*
   Montrez que toute solution extrémale de $LP(T)$ admet au plus $n + m$ valeurs non nulles.
4. *Soit $\mathbf{x}$ une solution extrémale de $LP(T)$. On dit que la tâche j est entière dans $\mathbf{x}$ (en anglais : integrally set in $\mathbf{x}$) si elle est ordonnancée intégralement sur une unique machine par $\mathbf{x}$. Sinon, on dit qu'elle est fractionnaire dans $\mathbf{x}$ (fractionnally set in $\mathbf{x}$).*
   Montrez qu'au moins $n - m$ tâches sont entières dans toute solution extrémale de $LP(T)$.

   L'agorithme va combiner élagage paramétré et arrondi LP en recherchant le plus petit T^* tel que $LP(T^*)$ admette une solution réalisable (elle minore OPT), puis en calculant et arrondissant une solution extrémale de $LP(T^*)$ pour avoir un temps d'exécution total $\leq 2T^*$.
   Pour l'étape d'arrondi, on introduit le graphe biparti $G = (J \cup M, E)$ où $(i, j) \in E$ si $x_{i,j} \neq 0$. Soit $F \subseteq J$ l'ensemble des tâches fractionnaires dans $\mathbf{x}$ et H le sous-graphe induit par $M \cup F$, il s'agit de trouver un couplage parfait de ce graphe pour déterminer les machines où ordonnancer les tâches fractionnaires.
   On appelle pseudo-arbre tout graphe connexe de plus de $|V|$ arêtes. Un graphe est une pseudo-forêt si chacune de ses composantes connexes est un pseudo-arbre.

5. Montrez que G est une pseudo-forêt.
6. Montrez que H admet un couplage parfait.
7. Montrez que l'algorithme suivant est une 2-approximation pour le problème de l'ordonnancement hétérogène.
   La première étape détermine un intervalle de recherche pour la valeur de T. L'algorithme commence donc par construire un ordonnancement glouton, où les tâches sont

ordonnancées les unes après les autres sur la machine courante la moins chargée. Notons α le temps d'exécution total de cet ordonnancement. L'intervalle de recherche sera donc $[\alpha/m, \alpha]$.

Algorithme 1

(a) Rechercher par dichotomie la plus petite valeur T^* de $T \in \mathbb{N}$ dans l'intervalle $[\alpha/m, \alpha]$, telle que $LP(T)$ admette une solution réalisable.

(b) Calculer une solution extrémale $\mathbf{x}$ de $LP(T^*)$.

(c) Ordonnancer toutes les tâches entières de $\mathbf{x}$ sur les machines données par $\mathbf{x}$.

(d) Construire le graphe H et calculer un couplage parfait $\mathcal{M}$ de H.

(e) Ordonnancer les tâches fractionnaires sur les machines données par le couplage $\mathcal{M}$.

8. Proposez une famille d'instances critiques.