Optimisation combinatoire
le 15 juillet 2003, durée 3 heures
Maitrise MIAGE

Exercice 1 Stockage d'objets(7 points)

On considère un ensemble d'objets $\mathcal{F} = \{1, \ldots, n\}$ de tailles respectives (en octets) $p_1, \ldots, p_n$. On dispose de deux sacs de taille P. Chaque objet i, s'il est placé dans le contenant 1, rapporte a_i, alors qu'il rapporte b_i s'il est placé dans le sac 2. Les $a_1, \ldots, a_n$ et $b_1, \ldots, b_n$ sont des données du problème. On cherche à choisir des objets de sorte qu'ils rentrent dans les sacs et qu'ils rapportent le plus possible.

On remarquera qu'il est possible que la taille des sacs ne soit pas suffisante pour stocker tous les objets. Ainsi, chaque objet peut se trouver soit dans le sac 1, soit dans le sac 2, soit n'être pas emporté.

1 En notant x_i la variable de décision qui vaut 1 si l'objet est stocké dans le premier contentant, 0 sinon, et y_i la variable de décision qui vaut 1 si l'objet est stocké dans le sac 2, et 0 sinon, modéliser le problème à l'aide d'un programme linéaire à variables bivalentes.

2 Soient $m_1, m_2 \leq P$. Appelons $F_k(m_1, m_2)$ le profit maximum pouvant être espéré d'un choix d'objets parmi $\{k, \ldots, n\}$ stockés dans les deux sacs supposés de tailles respectives m_1, m_2. Exprimer la solution du problème initial par rapport à cette notation.

3 Calculez $F_n(m_1, m_2)$, et établir l'équation de récurrence permettant de calculer $F_k(m_1, m_2)$ pour tout $1 \leq k < n$ et toutes les valeurs admissibles de m_1, m_2.

4 Quelle est la complexité de l'algorithme de programmation dynamique qui en résulte ?

Exercice 2 Bin Packing (5 points)

Dans le problème de bin Packing, on dispose d'un ensemble de n objets de poids respectifs $p_1, \ldots, p_n$. Les objets doivent être placés dans un nombre minimum de boites de capacité P (poids maximum qu'elles peuvent contenir).

1 Rappelez le fonctionnement de l'algorithme First Fit vu en cours, et appliquez le à l'exemple suivant, pour $P = 5$:

i	1	2	3	4	5	6	7	8	9	10
p_i	1	1	1	2	3	3	2	4	4	4

2 L'algorithme appelé First Fit Decreasing fonctionne comme l'algorithme First Fit, à ceci près que les objets sont préalablement triés par ordre de poids décroissant. Executer cet algorithme pour l'exemple.

3 Montrer que le nombre de boites générées par First Fit Decreasing est strictement inférieur à deux fois le nombre de boites optimales (on pourra utiliser le résultat vu en cours).

Exercice 3 Problème d'ordonnancement sur une machine (8 points)

Dans la suite de l'exercice, on s'intéresse à la définition d'une méthode arborescente pour résoudre le problème d'ordonnancement suivant :

On a n tâches. Chaque tâche i possède deux paramètres : sa durée p_i et sa date d'échéance d_i. Ces tâches doivent être effectuées par une machine, sans temps d'arrêt. Etant donné un ordonnancement, si l'on note C_i la date de fin de la tâche i, le retard de i est noté $T_i = \max(C_i - d_i, 0)$. On cherche un ordonnancement (c'est à dire une permutation σ des tâches) qui minimise $\sum_{i=1}^{n} T_i^2$.

Dans cet exercice on définit une méthode arborescente pour ce problème.

Un noeud S sera associé à une sous-suite de tâches $J_S = \{i_1, \ldots, i_r\}$, et comprend l'ensemble des ordonnancements σ qui se terminent par la séquence J_S : $\sigma(n) = i_r, \ldots, \sigma(n - r + 1) = i_1$. Par exemple, si $J_S = 4, 3$ alors l'ensemble des ordonnancements associés au noeud S est l'ensemble des séquences qui se terminent par $4, 3$. Sur 5 tâches $2, 1, 5, 4, 3$ est l'une de ces séquences.

Notons $P_S = \sum_{i \notin J_S} p_i$.

1 Calculez α_S la somme des carrés des retards des tâches de la séquence

J_S dans tout ordonnancement associé a noeud S en fonction des données du problème (durées des tâches, échéances).

2 Montrer que dans tout ordonnancement associé au noeud S, une tâche qui n'appartient pas à J_S aura un retard au moins égal à

$$\beta_S = \min_{i \notin J_S}(\max(0, P_S - d_i))$$

3 En déduire que $h(S) = \alpha_S + \beta_S^2$ est une évaluation par défaut du noeud S.

4 Proposer un algorithme glouton permettant de construire une solution particulière associée au noeud S.

5 Un noeud S est séparé en autant de fils qu'il y a de tâches $j \notin J_S$, correspondant aux sous-séquences $j, i_1, \ldots, i_r$. Appliquez la méthode arborescente (en fonction du temps en développant l'arbre sur un ou deux niveaux) à l'exemple suivant en précisant, pour les noeuds que vous développerez, le choix du noeud à séparer, les évaluations de chacun des noeuds développés, les mises à jour et les troncatures effectuées.

i	1	2	3	4	5
p_i	4	3	7	2	5
d_i	5	6	8	8	12