Une heuristique pour le problème du sac à dos multiple en variables 0-1

Mohamed Esseghir Lalami[1], Didier Elbaz[1], Moussa Elkihel[1], Vincent Boyer[1]

CNRS ; LAAS ; 7, avenue du Colonel Roche, F-31077 Toulouse, France.
Université de Toulouse ; UPS, INSA, INP, ISAE ; LAAS ; F-31077 Toulouse France
{mlalami,elbaz,elkihel,vboyer}@laas.fr

Mots-Clés : *sac à dos multiple, programmation dynamique, technique de dominance, heuristique.*

1 Introduction

Le problème du sac à dos multiple à variables bivalentes est une variante du problème du sac à dos (KP) dont la résolution est beaucoup plus difficile. Le fait qu'on rencontre ce problème dans des domaines d'application aussi différents que l'économie, l'industrie, les transports, le chargement de cargaisons et l'informatique répartie, lui confère un grand intérêt pratique [1]. Ce problème d'optimisation combinatoire sous contraintes est NP-complet, les méthodes exactes existantes sont limitées à de petites instances. Aujourd'hui, la résolution des instances difficiles s'effectue grâce à des approches heuristiques, l'aptitude de ces dernières à fournir des solutions de bonne qualité les rend indispensables dans le domaine pratique et elles s'avèrent aussi très utiles pour le développement de méthodes exactes fondées sur des techniques de séparation et d'évaluation.
Soit un ensemble $N = \{1, ..., n\}$ d'articles à charger dans m sacs à dos de capacité c_i $i \in \{1, ..., m\}$. Chaque article $j \in N$ est caractérisé par un poids unique w_j, un profit p_j et une variable de décision x_{ij} qui vaut 1 si l'article j est chargé dans le sac i et 0 dans le cas contraire. Il s'agit alors de trouver m sous-ensembles disjoints de N (où chaque sous-ensemble correspond au remplissage d'un sac) qui maximisent le profit total formé par la somme des articles sélectionnés [1]. Ce problème de sac à dos multiple noté MKP s'écrit de la manière suivante :

$$(\text{MKP}) \begin{cases} \max \displaystyle\sum_{i=1}^{m} \sum_{j=1}^{n} p_j x_{ij}, \\ \text{s.c} \displaystyle\sum_{j=1}^{n} w_j x_{ij} \leq c_i, \ i \in \{1, ..., m\}, \\ \quad \displaystyle\sum_{i=1}^{m} x_{ij} \leq 1, \ j \in \{1, ..., n\}, \\ x_{ij} \in \{0, 1\}, \ i \in \{1, ..., m\}, \ j \in \{1, ..., n\}. \end{cases} \tag{1}$$

avec p_j, c_i et w_j des entiers positifs. De plus, afin d'écarter les cas triviaux, nous nous plaçons dans le cas où : $\max\limits_{j \in N} w_j \leq \max\limits_{i \in \{1,...,m\}} c_i$, $\min\limits_{i \in \{1,...,m\}} c_i \geq \min\limits_{j \in N} w_j$ et $\displaystyle\sum_{j=1}^{n} w_j \geq \sum_{i=1}^{m} c_i$.

2 Présentation de l'heuristique

On considère dans ce qui suit, que les articles sont classés selon le ratio p_j/w_j decroissant. Soit le problème (KP_i) défini par : $(KP_i)\left\{\max \sum_{j\in N} p_j x_{ij} \mid \sum_{j\in N} w_j x_{ij} \leq c_i,\ x_{ij}\in\{0,1\},\ j\in N\right\}$, les capacités c_i étant classées par ordre décroissant. Si on note par s_i l'indice de base dans la relaxation continue du problème (KP_i) et par $2n_i$ la taille de son noyau [2], on construit alors une solution heuristique du MKP en résolvant successivement par programmation dynamique [3] les m sous-problèmes $(KP_i)(n_i)$ suivants :

$$(KP_i)(n_i)\begin{cases} z_i = \sum_{j=1}^{s_i-n_i} p_j + \max \sum_{j=s_i-n_i+1}^{s_i+n_i} p_j x_{ij} = \sum_{j=1}^{s_i+n_i} p_j x_{ij}^*, \\ \text{s.c : } \sum_{j=s_i-n_i+1}^{s_i+n_i} w_j x_{ij} \leq c_i - \sum_{j=1}^{s_i-n_i} w_j, \\ x_{ij}\in\{0,1\},\ j\in\{s_i-n_i+1, s_i+n_i\}. \end{cases} \tag{2}$$

Soit $N_i=\{j\in N : x_{ij}^*=1\}$, la valeur heuristique sera donnée par z en utilisant l'algorithme suivant.

$z = 0$;
$N_0 = \varnothing$;
Pour $i = 1$ à m faire
  $N = N - N_{i-1}$;
  Calculer s_i, n_i, N_i et z_i ;
  $z = z + z_i$;
Fin pour
Fig 1- Algorithme

m	type de problèmes	n	GAP %	temps (s)
2	Corrélés	1000	0.3200	0.006
	Non corrélés	10000	0.0062	0.003
	Non corrélés	100000	0.0003	0.004
5	Corrélés	1000	0.5500	0.000
	Non corrélés	10000	0.0100	0.016
	Non corrélés	100000	0.0004	0.013
10	Corrélés	1000	1.9200	0.096
	Non corrélés	10000	0.0128	0.016
	Non corrélés	100000	0.0005	0.018

TABLE 1 – Résultats obtenus pour des instances générées aléatoirement (Intel Core 2 Duo 2.2GHZ).

Les GAPs presentés dans la Table 1 ont été calculés par rapport à une borne supérieure du problème MKP. Ceux-ci montrent que notre heuristique permet d'avoir une bonne approximation de la valeur optimale (GAP $<$ 2%) avec des temps de calcul raisonnables. Il reste à valider notre approche en la comparant à d'autres heuristiques de la littérature.

Références

[1] D. Pisinger, H. Kellerer, U. Pferschy. Knapsack problems. *Springer*, 2004.

[2] M. Plateau, M. Elkihel. A hybrid method for the 0-1 knapsack problem. *Methods of Operations Research*, 49:277–293, 1985.

[3] V. Boyer, M. Elkihel, D. El baz. Heuristics for the 0-1 multidimensional knapsack problem. *European Journal of Operational Research*, 199(3):658–664, 2009.