Benchmark et heuristiques pour le problème généralisé du multi sac-à-dos multipériode

Xiaokang CAO, Antoine JOUGLET, Dritan NACE

UMR CNRS/UTC 6599 Heudiasyc
Université de Technologie de Compiègne
xiaokang.cao@hds.utc.fr, antoine.jouglet@hds.utc.fr, dritan.nace@hds.utc.fr

Mots-Clés : *sac-à-dos multipériode, complexité, heuristiques.*

1 Introduction

Dans cet article, nous nous intéressons au problème généralisé du multi sac-à-dos multipériode (GMPMKP). Il s'agit d'une variante du problème de sac-à-dos qu'on rencontre lorsqu'on veut déterminer les dates de renouvellement d'un ensemble d'articles sur un ensemble de périodes consécutives. La formulation mathématique de ce problème est donnée ci-dessous :

Notation :
- N est un ensemble d'articles $\{1, \ldots, n\}$;
- M est un ensemble de périodes $\{1, \ldots, m\}$;
- $x_{i,j}$ sont des variables de décision : $x_{i,j} = 1$ lorsque l'article i est remplacé à la période j ;
- $p_{i,j}$ est le profit de l'article i lorsqu'il est remplacé à la période j ;
- w_i est le coût de remplacement de l'article i ;
- B_j est la capacité cumulative associée à la période j ce qui représente le coût total maximum de tous les articles remplacés entre la période 1 et la période j.

$$(GMPMKP) \qquad \max \sum_{i \in N, j \in M} p_{i,j} x_{i,j} \tag{1}$$

$$\sum_{t=1}^{j} \sum_{i \in N} w_i x_{i,t} \leq B_j, \qquad \forall j \in M, \tag{2}$$

$$\sum_{j \in M} x_{i,j} \leq 1, \qquad \forall i \in N, \tag{3}$$

$$x_{i,j} \in \{0,1\}, \quad \forall i \in N, j \in M, \tag{4}$$

où $B_1 \leq B_2, ..., \leq B_m$.

2 Résultats de complexité

Présentons d'abord sans preuve un résultat fondamental sur la complexité :

Théorème 1 *Le problème GMPMKP est NP-difficile au sens fort.*

La preuve est basée sur une réduction du problème de 3-partition. De plus, nous montrons que chaque instance de ce problème peut être transformée à une instance équivalente où les profits $p_{i,j}$ sont décoissants pour tous les articles i.

3 Benchmarks

Nous proposons 4 générateurs d'instances pour ce problème. Pour le générateur de type 1, les valeurs w_i et $p_{i,1}$ ne sont pas corrélés et $p_{i,j}$ décroissent linéairement pour chaque article i. Pour celui de type 2, les valeurs w_i et $p_{i,1}$ ne sont pas corrélés et $p_{i,j}$ décroissent de façon aléatoire pour chaque article i. Pour celui de type 3, les valeurs w_i et $p_{i,1}$ sont corrélés et $p_{i,j}$ décroissent de façon aléatoire pour chaque article i. Pour celui de type 4, les valeurs $p_{i,j}$ sont générés aléatoirement pour chaque article i.

4 Heuristiques et résultats numériques

Nous avons adapté l'heuristique de Dantzig (DZ) [2] et celle de Martello&Toth (MT) [1] pour notre problème. Nous avons aussi proposé deux autres heuristiques :

Heuristique Back&Forth. Cette heuristique est une modification de l'heuristique de Dantzig. Elle est réalisée en deux étapes :

- Trouver l'article $i = \text{argmax}\,(p_{i,j}/w_i)$ et lui associer une période dans $[j, j+p]$, p dépend de i. Répéter cette opération jusqu'au moment où le remplacement n'est plus possible ;
- Faire avancer les articles de certaine manière itérativement afin d'avoir un profit total plus grand.

Heuristique CombMT&D. Cette heuristique est une combinaison directe de Martello&Toth et Dantzig. L'article i avec $i = \text{argmax}\,((p_{i,j}/w_i) * (p_{i,j} - p_{i,j+1})/w_i)$ est itérativement remplacé jusqu'au moment où le remplacement n'est plus possible.

Nous avons généré des instances avec $n = [50, 100, 500, 1000]$ et $m = [10, 30, 50]$. Pour chaque taille, 10 instances sont générées. Nous avons comparé surtout les résultats des heuristiques avec la borne supérieure donnée par la relaxation continue. Sur les instances de type 1, les résultats de tous les quatre heuristiques sont très proches. Dantzig et CombMT&D sont légèrement meilleurs que les deux autres. Pour le type 2, Back&Forth et CombMT&D donnent toujours de meilleurs résultats que les deux autres. Néanmoins, CombMT&D paraît plus performante que Back&Forth. Pour le type 3, Back&Forth donne toujous de meilleurs résultats que les autres. Pour le type 4, Dantzig et Back&Forth sont les plus performantes.

Références

[1] Silvano Martello and Paolo Toth. *Knapsack problems : algorithms and computer implementations.* John Wiley & Sons, Inc., New York, NY, USA, 1990.

[2] David Pisinger. Where are the hard knapsack problems ? *Computers & Operations Research*, 32(9) :2271–2284, September 2005.