Applied Mathematics and Computation 227 (2014) 518–530

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

New upper bounds and exact methods for the knapsack sharing problem

Mhand Hifi *, Lei Wu

Laboratoire de Recherche EPROAD – EA 4669, Université de Picardie Jules Verne, 7 rue du Moulin Neuf, 80000 Amiens, France

ARTICLE INFO

Keywords:
Combinatorial optimization
Decomposition
Discrete optimization
Knapsack
Optimality
Sharing
Upper bound

ABSTRACT

In this paper, we study the *knapsack sharing problem* (KSP), which is a variant of the well-known NP-Hard knapsack problem. We investigate the use of a dichotomous search-based exact method for solving the KSP. Such a method is based on decomposing the original problem into a series of minimizing and maximizing knapsack problems, where each of them is embedded into a dichotomous search bounded by both lower and upper bounds. Throughout the interval search, we introduce new upper bounds and incremental valid lower bounds. One of these upper bounds can be viewed as an extended of Dantzig upper bound whereas a series of valid lower bounds are obtained when solving a subset of knapsack problems. Finally, the proposed algorithm is evaluated on a set of problem instances of the literature and other new harder ones. The provided results are compared to those reached by the Cplex solver and a more recent exact algorithm of the literature. The computational part shows the dominance of the new exact method.

1. Introduction

In this paper, we investigate the use of an exact dichotomous search-based algorithm for solving the *knapsack sharing problem* (KSP). KSP is a variant of the well-known knapsack problem (KP), an NP-hard combinatorial optimization problem. An instance of KSP is characterized by a knapsack of fixed capacity R and a set of items $\mathcal{N}$, where the item set $\mathcal{N}$ denotes a collection of m disjoint classes of items: $\mathcal{N} = \cup_{i=1}^{m}\mathcal{N}_i$ and for $p \in \{1,\ldots,m\},\ q \in \{1,\ldots,m\}$ and $p \neq q, \mathcal{N}_p \cap \mathcal{N}_q = \emptyset$. Assume that, each item (i,j) $(\forall i \in I = \{1,\ldots,m\}$ and $\forall j \in \mathcal{N}_i)$ has a weight w_{ij} and a profit c_{ij}. The objective of the problem is to determine a subset of items that maximizes the minimal value of a set of linear functions under the ordinary capacity constraint. Therefore, the KSP can be formulated as follows:

$$(PKSP)\ \max \min_{1\leqslant i\leqslant m}\left\{\sum_{j\in\mathcal{N}_i} c_{ij}x_{ij}\right\}$$
$$\text{s.t.} \sum_{i}^{m}\sum_{j\in\mathcal{N}_i} w_{ij}x_{ij} \leqslant R,$$
$$x_{ij} \in \{0,1\}, \quad \forall i \in I, \quad \forall j \in \mathcal{N}_i,$$

where x_{ij} $(\forall j \in \mathcal{N}_i$ and $\forall i \in I)$ denotes the binary decision variable such that $x_{ij} = 1$ if the item (i,j) is in the solution set, $x_{ij} = 0$ otherwise. Without any loss of generality, we assume that $\sum_{i=1}^{m}\sum_{j\in\mathcal{N}_i} w_{ij}x_{ij} > R$, all the classes are indexed from $\mathcal{N}_1$ to $\mathcal{N}_m$

* Corresponding author.
E-mail addresses: mhand.hifi@u-picardie.fr, hifi@u-picardie.fr (M. Hifi), lei.wu@u-picardie.fr (L. Wu).

0096-3003/$ - see front matter © 2013 Published by Elsevier Inc.
http://dx.doi.org/10.1016/j.amc.2013.11.040

while the elements of the class $\mathcal{N}_i$ are indexed from 1 to $|\mathcal{N}_i|, \forall i \in I$. We also assume that $c_{ij},\ w_{ij}\ (\forall i \in I,\ \forall j \in \mathcal{N}_i)$ and R take the nonnegative integer values.

KSP has a wide range of commercial applications (see Brown [3] and Tang [16]) and, the binary version of the problem becomes NP-hard since it represents an intuitive generalization of the well-known knapsack (Bansal and Deep [1] and Kellerer et al. [10]). We note also that such a problem is classified as $KSP(Bn/m/1)$ (see Yamada and Futakawa [17], and Hifi and Sadfi [7,9]), which means that they are n items of binary (B) type, divided into m classes and, there is only one knapsack constraint. Other problems of the knapsack-family have been tackled by using several solution procedures, like optimal methods and heuristics (cf., Ghassemi-Tari and Jahangiri [5], Hifi and M'Hallah [6] and Rong et al. [15]).

For (P_{KSP}), we can observe that it can be rewritten as an Integer Linear Program (ILP) defined as follows:

$$\begin{aligned}
(\mathrm{ILP}_{KSP})\ &\max \gamma \\
&\text{s.t.} \sum_{i=1}^{m}\sum_{j\in\mathcal{N}_i} w_{ij}x_{ij} \leqslant R, \\
&\sum_{j\in\mathcal{N}_i} c_{ij}x_{ij} \geqslant \gamma, \quad \forall i \in I \\
&x_{ij} \in \{0,1\}, \quad \forall i \in I, \quad \forall j \in \mathcal{N}_i
\end{aligned} \tag{1}$$

Also, in order to compute an upper bound for (ILP_{KSP}), we can solve its linear relaxation problem, defined as follows:

$$\begin{aligned}
(\mathrm{LP}_{KSP})\ &\max \gamma \\
&\text{s.t.} \sum_{i=1}^{m}\sum_{j\in\mathcal{N}_i} w_{ij}x_{ij} \leqslant R, \\
&\sum\nolimits_{j\in\mathcal{N}_i} c_{ij}x_{ij} \geqslant \gamma, \quad \forall i \in I \\
&x_{ij} \in [0,1], \quad \forall i \in I, \quad \forall j \in \mathcal{N}_i
\end{aligned}$$

The remainder of the paper is organized as follows. First, Section 2 describes a brief literature survey of the knapsack sharing problem. Second, Section 3 exposes both new upper bound and the proposed exact algorithm. In fact, Section 3.1 discusses a special upper bound which can be viewed as an extended of Dantzig upper bound tailored for the KSP and, the proposed dichotomous search-based exact algorithm is detailed in Section 3.4. Third, Section 4 evaluates, on a set of instances taken from the literature and other harder ones, the performance of the proposed exact algorithm while the provided results are compared to those reached by both Cplex solver and the more recent exact algorithm of the literature due to Boyer et al. [2]. Finally, Section 5 summarizes the main results of the paper.

2. Background

The KSP, namely max–min allocation problem, has been firstly studied by Brown [3]. Different exact and approximate approaches have been designed especially for this problem (Brown [4], Kuno et al. [11], Luss [12], Pang and Yu [14] and Tang [16]). For the particular continuous KSP, Kuno et al. [11] have proposed a linear time solution algorithm. Based on the same principle, Yamada and Futakawa [17] proposed an improved algorithm for the continuous KSP. The last two algorithms are based upon a binary search procedure, where a series of continuous knapsack problems is solved.

The KSP has been addressed by Yamada and Futakawa [17] who extended the heuristic approach tailored to the $KSP(Cn/m/1)$ to $KSP(Bn/m/1)$. Yamada et al. [18] designed several exact algorithms which are based upon branch and bound procedures or binary search method. The authors indicated that the binary search approach outperformed the branch and bound algorithm.

Hifi et al. [8] proposed a metaheuristic which is based upon a reactive tabu search. The authors showed that their method performed well for correlated and uncorrelated problem instances.

Hifi and Sadfi [7] designed a dynamic programming algorithm in which the original problem is decomposed into a series of knapsack problems. These knapsacks are simultaneously solved using an incremental "fictive capacity" until reaching the optimal vector of sub-capacities characterizing an optimal solution of the problem. The performance of the algorithm was evaluated on benchmark instances of the literature and showed its good behavior especially for instances containing an important set of classes.

In order to enhance Hifi and Sadfi's [7] exact algorithm, Hifi et al. [9] proposed a new version of the dynamic programming algorithm. The new version of the algorithm is based on two conditions: (i) the *order condition* and (ii) the *optimality one*. Both conditions were introduced in order to accelerate the search process. The computational results showed that the new version was able to improve the performance of Hifi and Sadfi's [7] algorithm.

Finally, Boyer et al. [2] proposed an algorithm in order to enhance Hifi et al.'s [9] exact algorithm. The algorithm is mainly based on applying the notion of the dominance throughout the resolution of a series of knapsack problems.