Computers & Industrial Engineering 99 (2016) 280–286

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

# A tree search based combination heuristic for the knapsack problem with setup

Mahdi Khemakhem [a,*], Khalil Chebil [b]

[a] College of Computer Engineering and Science, Prince Sattam Bin Abdulaziz University, Kingdom of Saudi Arabia
[b] LOGIQ, University of Sfax, Tunisia

ARTICLE INFO

*Article history:*
Received 19 July 2015
Received in revised form 29 March 2016
Accepted 26 July 2016
Available online 27 July 2016

*Keywords:*
Knapsack problems
Setup
Tree search
Combination
Filter-and-fan metaheuristic
Avoid duplication

ABSTRACT

Knapsack Problems with Setups (KPS) have received increasing attention in recent research for their potential use in the modeling of various concrete industrial and financial problems, such as order acceptance and production scheduling. The KPS problem consists in selecting appropriate items, from a set of disjoint families of items, to enter a knapsack while maximizing its value. An individual item can be selected only if a setup is incurred for the family to which it belongs. In this paper, we propose a tree search heuristic to the KPS that generates compound moves by a strategically truncated form of tree search. We adopt a new avoid duplication technique that consists in converting a KPS solution to an integer index. The efficiency of the proposed method is evaluated by computational experiments involving a set of randomly generated instances. The results demonstrate the impact of the avoiding duplication technique in terms of enhancing solution quality and computation time. The efficiency of the proposed method was confirmed by its ability to produce optimal and near optimal solutions in a short computation time.



## 1. Introduction

We will refer to the Knapsack Problem with Setup as KPS. It is described as a knapsack problem with additional fixed setup costs discounted both in the objective function and in the constraints. This problem is particularly prevalent in production planning applications where resources need to be set up before a production run.

Our interest in this model was originally motivated by practical problems at a production project with a leading manufacturer and supplier of agro-alimentary glass packing industry. This company produces several types of products, including bottles, flacons, and pots. The most important phase in the manufacturing process, is the phase of shaping. In fact, to change the production from one product family to another, the production machinery must be set up and molds must be changed in the molding machine. These changes in the manufacturing process require significant setup time and costs. Assume at time T, the company receive some orders (jobs), wich belong to $N$ product families. Each product family $i$, has $n_i$ jobs. Also assume that these jobs should be produced in the next planning period and the company's manufacturing capacity is fixed and can't be changed in the short term. Accordingly, the company needs to decide on how to choose orders so as to maximize the total profit. This represents a typical case involving a knapsack problem with setup model that can be used to solve this problem.

The knapsack problem with setup is defined by a knapsack capacity $b \in \mathbb{N}$ and a set of $N$ classes of items. Each class $i \in \{1,\dots,N\}$ consists of $n_i$ items and is characterized by a negative integer $f_i$ and a non-negative integer $d_i$ representing its setup cost and setup capacity consumption, respectively. Each item $j \in \{1,\dots,n_i\}$ of a class $i$ is labeled by a profit $c_{ij} \in \mathbb{N}$ and a capacity consumption $a_{ij} \in \mathbb{N}$. The objective is to maximize the total profit of the selected items minus the fixed costs incurred for setting-up the selected classes.

The KPS can be formulated by a $0-1$ linear program as follows:

$$\text{Max } z = \sum_{i=1}^{N}\sum_{j=1}^{n_i} c_{ij}x_{ij} + \sum_{i=1}^{N} f_i y_i \tag{1}$$

$$s.t. \quad \sum_{i=1}^{N}\sum_{j=1}^{n_i} a_{ij}x_{ij} + \sum_{i=1}^{N} d_i y_i \leqslant b \tag{2}$$

$$x_{ij} \leqslant y_i \qquad \forall i \in \{1,\dots,N\}, \forall j \in \{1,\dots,n_i\} \tag{3}$$

$$x_{ij}, y_i \in \{0,1\} \qquad \forall i \in \{1,\dots,N\}, \quad \forall j \in \{1,\dots,n_i\} \tag{4}$$

* Corresponding author.
*E-mail addresses:* m.khemakhem@psau.edu.sa (M. Khemakhem), khalil.chebil@issatm.rnu.tn (K. Chebil).

http://dx.doi.org/10.1016/j.cie.2016.07.021
0360-8352/

Eq. (1) represents the KPS objective function. Constraint (2) ensures that the weight of selected items in the knapsack, including their setup capacity consumption, does not exceed knapsack capacity $b$. Constraints (3) guarantee that each item is selected only if it belongs to a class that has been set up. Constraints (4) require the decision variables to be binary, where $x_{ij}$ refers to the item variables and $y_i$ to the setup variables. In fact, $y_i$ is equal to 1 if the knapsack is set up to accept items belonging to class $i$ and is equal to 0 otherwise. $x_{ij}$ is equal to 1 if the item $j$ of the class $i$ is placed in the knapsack and is equal to 0 otherwise.

The KPS is a generalization of the standard 0–1 Knapsack Problem (KP), which is known to be an NP-hard problem (Kong, Gao, Ouyang, & Li, 2015; Martello & Toth, 1990). In fact, it is not difficult to verify that a special case of the KPS, where $N = 1$, is equivalent to the KP. The works of Martello and Toth (1990) and Kellerer, Pferschy, and Pisinger (2004) provide a thorough overview of the research so far performed on the KP and its variations which reflect its ability to closely represent and respond to real world problems. Chajakis and Guignard (1994) consider a similar problem of KPS, where the setup cost $f_i$ and profit $c_{ij}$ of an item $j$ of a class $i$ can be negative or non-negative. An extra constraint is added to make sure that if the knapsack is set up for a class $i$, at least one item of this class must be selected. Chajakis and Guignard (1994) propose a dynamic programming algorithm and two versions of a two-phase enumerative scheme. The experiments show that the dynamic programming approach is most efficient for correlated instances with small knapsack capacity. Akinc (2004) presents a set of algorithms based on several properties of the linear programming relaxation of a special case of KPS with no setup capacity consumption, called the Fixed Charge Knapsack Problem (FCKP). He showed that if the FCKP is solved as an LP and if all the $y_i$ variables obtained are integers, then the optimal solution is obtained by solving the KP to optimality allocate the remainder capacity to all $x_{ij}$ for which $y_i = 1$. Yang (2006) has, however, demonstrated the inadequacy of the last proposal by presenting a counter-example. Mclay and Jacobson (2007) provide three dynamic programming algorithms to solve the Bounded Setup Knapsack Problem (BSKP) in a pseudo-polynomial time. The latter were not, however, practical for solving large problem instances. Yang (2006) developed an effective branch-and-bound algorithm and proposed a heuristic method for the KPS. A thorough survey of the literature on the KSP has recently been presented in the work of Michel, Perrot, and Vanderbeck (2009) who provided an extension of the branch-and-bound algorithm proposed by Horowitz and Sahni (1974).

In this paper, we present a tree search based combination (TSC) heuristic to solve the KPS. The results from experimental assays on a randomly generated set of benchmark problems demonstrate the effectiveness of our approach. The rest of this paper is organized as follows: Section 2 presents the proposed algorithm. Section 3 evaluates the efficiency of the TSC algorithm using a set of randomly generated instances and compares its performance to CPLEX 12.5. Section 4 concludes by providing a summary and perspectives for future research.

## 2. Tree search based combination for the KPS

In this section, we start by presenting some preliminary considerations in the knapsack problem with setup that are relevant to the design of the proposed algorithm. We then move to describe the general TSC procedure.

### 2.1. Preliminary considerations

As mentioned earlier, the knapsack problem is a special case of the KPS (where $n = 1$). The special structure of KPS allows us to fix the setup variables ($y_i = 0$ or $y_i = 1$) needed to transform the KPS into a KP. Let's consider a set of item classes $Y = \{i \in \{1,\dots,N\}/y_i = 1\}$, where the knapsack is set up to accept items belonging to each class in $Y$. We define the $KPS[Y]$ problem as a sub-problem of KPS. $KPS[Y]$ is a knapsack problem with a capacity $b - \sigma$ and an objective function minimized by a negative integer setup cost $\delta$. Then, the $KPS[Y]$ can be formulated by a KP 0–1 linear program as follows:

$$\text{Max } z = \sum_{i\in Y}\sum_{j=1}^{n_i} c_{ij}x_{ij} + \delta \tag{5}$$

$$s.t. \quad \sum_{i\in Y}\sum_{j=1}^{n_i} a_{ij}x_{ij} \leqslant b - \sigma \tag{6}$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in Y, \quad \forall j \in \{1,\dots,n_i\} \tag{7}$$

where

$$\delta = \sum_{i\in Y} f_i \text{ and } \sigma = \sum_{i\in Y} d_i$$

With these considerations in mind, we only fix the setup variables $y_i$ to 1 or to 0 and employ CPLEX to determine the best values for $x_{ij}$, which yields a feasible solution to the KPS. Thus, our neighborhood strategy focuses on finding the optimal combination of the setup variables $Y^*$.

A KPS solution can be represented by two sets: a set of item variables $X = \{x_{ij}, i = 1,\dots,N; j = 1,\dots,n_i\}$ and a set of selected setup variables $Y = \{i \in \{1,\dots,N\}/y_i = 1\}$. Consider an example of KPS instance defined by:

$$N = 3, b = 90, [n_i, i = 1,\dots,3] = [4,3,3],$$

$$[f_i, i = 1,\dots,3] = [-10,-13,-8], [d_i, i = 1,\dots,3] = [6,5,7],$$

$$[c_{ij}, i = 1,\dots,3; j = 1,\dots,n_i] = \begin{bmatrix} 20 & 24 & 19 & 23 \\ 26 & 22 & 26 & \\ 25 & 24 & 29 & \end{bmatrix}$$

and

$$[a_{ij}, i = 1,\dots,3; j = 1,\dots,n_i] = \begin{bmatrix} 15 & 19 & 14 & 18 \\ 17 & 17 & 21 & \\ 20 & 19 & 24 & \end{bmatrix}.$$

The optimal solution of this example is $X^* = \{\overbrace{0,0,0,0}^{class1}, \overbrace{1,1,0}^{class2}, \overbrace{1,0,1}^{class3}\}, Y^* = \{2,3\}$, with an objective value $z = 81$. It can be noted that the knapsack is set up to accept only items from class 2 and 3. Thus, all item variables belonging to class 1 are equal to 0. To obtain the set $X^*$, we just use CPLEX to optimally solve KPS[$Y^*$], which is, in this example, a knapsack problem with just 6 items (items belonging to class 2 and 3), with a capacity $b\prime = b - \sigma = 90 - 5 - 7 = 78$ and an initial value $z = \delta = -13 - 8 = -21$. In the rest of this paper, we consider only the set $Y$ to represent a KPS solution.

### 2.2. The TSC approach

The TSC approach is closely similar to the Filter-and-Fan (F&F) method which was initially proposed in Glover (1998) as a method for refining solutions obtained by scatter search. The F&F approach consists in the integration of the filtration and sequential disper-