Hifi, Mhand; Wu, Lei

An Equivalent Model for Exactly Solving the Multiple-choice Multidimensional Knapsack Problem


Available in: http://www.redalyc.org/articulo.oa?id=265224452004

International Journal of Combinatorial Optimization Problems and Informatics

México

Morelos, México
An Equivalent Model for Exactly Solving the Multiple-choice Multidimensional Knapsack Problem

Mhand Hifi* and Lei Wu
Université de Picardie Jules Verne, Unité de Recherche EPROAD
5 rue du Moulin Neuf, 80000 Amiens, France.

Abstract

The Multiple-choice Multi-dimensional Knapsack Problem (MMKP) is a problem which can be encountered in real-world applications, such as service level agreement, model of allocation resources, or as a dynamic adaptation of system of resources for multimedia multi-sessions. In this paper, we investigate the use of a new model-based Lagrangian relaxation for optimally solving the MMKP. In order to tackle large-scale problem instances, we curtail the search process for providing approximate solutions. We then apply the Cplex solver using both original and equivalent models. In this case, the Cplex solver becomes more efficient when the new model is used. Also, when the proposed method is considered as a heuristic, then it outperforms the Cplex solver using the original model: new solution values are obtained.

Keywords. Heuristic, Knapsack, Lagrangian relaxation, Optimality.

1 Introduction

In this paper we investigate the use of a tailored model based upon Lagrangian relaxation in order to solve the Multiple-choice Multi-dimensional Knapsack Problems (MMKP). The MMKP is a problem which may be encountered in real-world problems, like service level agreement, allocation resources, or as a dynamic adaptation of system of resources for multimedia multi-sessions (for more details the reader can refer to [1] and [20]).

In MMKP we are given a group of knapsack constraints $R = (R^1, R^2, \ldots, R^m)$ characterizing the constraints of available resources, a set $S = (S_1, \ldots, S_i, \ldots, S_n)$ of items partitioned into $n$ disjoint classes, where each class $i, i = \{1, \ldots, n\}$, is composed of $r_i$ items representing the cardinality of the $i$-th set $J_i$. Furthermore, each item $j, j \in \{1, \ldots, r_i\}$, of the $i$-th class is characterized by a profit $c_{ij} \geq 0$, and requires resources represented by a weight vector $w_{ij} = (w_{1ij}, w_{2ij}, \ldots, w_{mij})$, where $w_{ij} \geq 0, \forall(i, j)$. The objective of the problem is to fill all knapsacks with exactly one

*Corresponding author (M. Hifi: hifi@u-picardie.fr)
item from each class and to determine the subset whose objective value is maximum. Formally, the MMKP, an ILP, can be formulated as follows:

\[(\text{MMKP}) \quad \max \quad Z = \sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij} x_{ij} \]

s.t. \[\sum_{i=1}^{n} \sum_{j=1}^{r_i} w_{ij}^k x_{ij} \leq R^k, \quad \forall k = 1, \ldots, m, \quad (1)\]

\[\sum_{j=1}^{r_i} x_{ij} = 1, \quad \forall i = 1, \ldots, n, \quad (2)\]

\[x_{ij} \in \{0, 1\}, \quad \forall i = 1, \ldots, n, \quad j = 1, \ldots, r_i, \quad (3)\]

where constraints of type (1) represent the group of knapsack constraints, the constraints of type (2) represent the group of the choice constraints, and type (3) denotes the integrality constraints.

The remainder of the paper is organized as follows. Section 2 presents a brief reference of some solution procedures for the MMKP. The concept of Lagrangian relaxation applied for the MMKP is described in Section 3. Section 4 shows how an auxiliary problem can be obtained from a Lagrangian relaxation solution. The behavior of the Cplex solver when using the auxiliary model is presented in Section 5. It is evaluated on a set of problem instances varying from small to large sized ones. Finally, Section 6 summarizes the contributions of this paper.

2 Related works

The MMKP - an ILP- is an NP-hard combinatorial optimization problem, which in some cases, arises as a component of more complex optimization problems. Its induced structure in complex problems allows the computation of upper bounds and the design of heuristic and exact methods for these complex problems. For instance, such a problem may be encountered in real-world problems, like service level agreement, allocation resources, or as a dynamic adaptation of system of resources for multimedia multi-sessions (for more details the reader can refer to Khan [1] and Khan et al. [20]).

To our knowledge, very few papers addressing the problem of multiple-choice multi-dimensional knapsacks are available. The MMKP was first addressed by Moser et al. [23] who tackled the problem by proposing a heuristic, which is based on the concept of graceful degradation from the most valuable items as priced by Lagrange multipliers. Khan et al. [1] designed a heuristic using the principle of the aggregate resources introduced by Toyoda [25] in order to solve the multi-dimensional knapsack problem (MDKP). Hifi et al. [16] tailored a special guided local search in which the trajectories of the solutions are guided following an augmented cost function using some penalties parameters. A more general reactive local search has been designed by Hifi et al. [13]; it is based on a reactive local search combining the principle of degrading the solution at hand in order to escape local optima and...
to try a diversification search. An additional memory list, based on simple moves, was introduced to the search process in order to escape some visited regions. Akbar et al. [4] proposed a heuristic based on mapping the multidimensional resource consumption to the single one and tried to apply the provided convex hulls in order to reduce the search process. Hiremath and Hill [17] proposed a series of fast and greedy algorithms for approximately solving the MMKP; the aim of the proposed work is to design quick approaches providing moderate and stable solutions. Khan et al. [3] proposed a parallelization of Hifi et al.’s [13] algorithm in order to accelerate the search processes. Cherfi and Hifi [7, 8] designed a column generation-based heuristic in which a rounding solution procedure is combined with a truncated branch-and-bound. Finally, Cherfi and Hifi [7] tackled large-scale MMKP using an augmented method based on combining the column generation (developed in Cherfi and Hifi [6]) with a truncated branch-and-bound/cut. Hanafi et al. [14] proposed an iterative heuristic which is based on exploiting some information reached from a series of relaxations to the MMKP. In order to improve the quality of solutions, a local search strategy was applied. For the optimal approaches, regarding the NP-Hardness of the problem and also the difficulty in finding a feasible solution, Hifi et al. [15] proposed a method based on a tree-search using the surrogate technique in which a pseudo-sequential dynamic programming (see also also [24] for a summary) was considered. The last algorithm, as shown in [15], is able to solve only small size instances and uncorrelated ones (for other knapsack type problems, the reader can refer to Kellerer et al. [19]).

3 Lagrangian relaxation for the MMKP

This section discusses the tailoring of Lagrangian relaxation for the MMKP. We develop the standard Lagrangian relaxation and its sub-gradient method used for determining a series of upper bounds for the MMKP.

One of the strategic choices in Lagrangian relaxation lies in the selection of the group of constraints to relax. In fact, two types of constraints can be considered: (1) complicating constraints and (2) others. The most important constraints to define, which allow to reduce the computational effort to solve the problem, are those representing the complicated constraints. Herein, we opt to dualize the group of constraints of type (1) that we select as the complicating constraints.

Let $\lambda \geq 0$ be the Lagrange multipliers vector for constraints of type 1. Then, introducing the slacks constraints of type (1) with weights $\lambda$ into the objective, we get the following Lagrangian of MMKP:

$$\text{(MMKP}_{LR}(\lambda)) \quad Z_D(\lambda) = \max \sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij}x_{ij} + \sum_{k=1}^{m} \lambda_k \left( R^k - \sum_{i=1}^{n} \sum_{j=1}^{r_i} w_{ij}^k x_{ij} \right)$$

s.t. (2) and (3).

Observe that $\text{MMKP}_{LR}(\lambda)$ is still defined in the primary space of the solution $x$ at hand. Then, given $\lambda \geq 0$, the associated upper bound $Z_D(\lambda)$ for MMKP can be
computed by solving the following problem:

$$\max \sum_{i=1}^{n} \sum_{j=1}^{r_i} \left( c_{ij} - \sum_{k=1}^{m} \bar{\lambda}_k w_{ij}^k \right) x_{ij} + \sum_{k=1}^{m} \bar{\lambda}_k R_k$$

s.t. (2) and (3).

The objective function value of MMKP$_{LR}(\lambda)$, noted $Z_D$, is minimized through the so-called standard sub-gradient method and the optimal Lagrange multipliers vector is then noted as $\lambda^*$ (for more details the reader can refer to Guignard [11]). Herein, we adapt the sub-gradient method by using the following notations.

Let

$$\mathcal{X} := \left\{ x^t \text{ subject to (2) and (3)}, \ t = 1, \ldots, T \right\},$$

be the finite set of feasible solutions associated to MMKP$_{LR}(\lambda)$, where $x^t = (x^t_{11}, \ldots, x^t_{ij}, \ldots, x^t_{nn})$ and $\lambda \geq 0$. The dual problem of MMKP$_{LR}(\lambda)$ can be written as follow:

$$Z_D = \min u$$

s.t. $u \geq \sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij} x^t_{ij} + \sum_{k=1}^{m} \lambda^t g_k^t, \ t = 1, \ldots, T,$

where $g^t = (g^t_1, \ldots, g^t_m)$ is a $m$-vector such that $g^t_k = R_k - \sum_{i=1}^{n} \sum_{j=1}^{r_i} w_{ij}^k x^t_{ij}$ ($k = 1, \ldots, m$).

We recall that the $m$-vector $v$ is called a sub-gradient of $Z_D(\lambda)$ at $\bar{\lambda}$ when the following condition is satisfied:

$$Z_D(\lambda) \leq Z_D(\bar{\lambda}) + v(\lambda - \bar{\lambda}), \ \forall \lambda \geq 0.$$ \hspace{1cm} (4)

In our case, the $m$-vector $g^t$ is a sub-gradient of $x^t$ on $\bar{\lambda}$, where $x^t$ is the optimal solution of MMKP$_{LR}(\bar{\lambda})$.

The optimal Lagrange multiplier $\lambda^*$ can be progressively approached by generating a sequence of $\lambda$ according to Inequality (4). Given an initial value $\lambda^1$, a sequence $\{\lambda^t\}$ can be iteratively determined by following formula:

$$\lambda^{t+1} = \lambda^t + p_t g^t, \ t = 1, \ldots, T.$$ \hspace{1cm} (5)

$p_t$ is a positive scalar step size and it can be computed as follow:

$$p_t = \frac{\varepsilon_t (Z_D(\lambda^t) - Z^*)}{||g^t||^2}$$

where $\varepsilon_t$ is a constant sequence such that $0 < \varepsilon_t \leq 2$ ($\forall t = 1, \ldots, T$) and $Z^*$ is the optimal value of MMKP. Note that, in practice, $Z^*$ is usually replaced by the best objective function value (i.e. the best lower bound) of MMKP.
4 An auxiliary problem

The main idea discussed in this paper is how to explore the information from an optimal Lagrangian solution. In what follows, we try to describe a way that serves to provide a feasible or an improved feasible solution within the neighborhood of a current optimal Lagrangian solution. Indeed, for any solution $\bar{x}$ of MMKP$_{LR}(\lambda)$, it is possible to establish an auxiliary problem characterizing the original MMKP problem:

$$\text{(MMKP}_{\text{AP}}(\bar{x})) \quad Z_{\text{AP}} = \max \sum_{i=1}^{n} \sum_{j=1}^{r_i} c'_{ij} x_{ij}$$

subject to

$$\sum_{i=1}^{n} \sum_{j=1}^{r_i} w^k_{ij} x_{ij} \leq R^k, \quad \forall k = 1, \ldots, m,$$  \hspace{1cm} (5)

$$\sum_{j=1}^{r_i} x_{ij} \leq 1, \quad \forall i = 1, \ldots, n,$$  \hspace{1cm} (6)

$$x_{ij} \in \{0, 1\} \quad \forall i = 1, \ldots, n, j = 1, \ldots, r_i.$$

On the one hand, we can observe that $c'_{ij}$ can be viewed as the cost characterizing the exchange between two items contained to the same class. On the other hand, the same argument can be used for the weights, i.e., $w^k_{ij}$. Formally, both values can be rewritten as follows:

$$c'_ {ij} = c_{ij} - \bar{c}_{ij}, \quad \forall i = 1, \ldots, n, j = 1, \ldots, r_i,$$

$$w^k_{ij} = w^k_{ij} - \bar{w}^k_{ij}, \quad \forall i = 1, \ldots, n, j = 1, \ldots, r_i,$$  \hspace{1cm} (7)

where $\bar{c}_{ij}$ (resp. $\bar{w}^k_{ij}$) denotes the profit (resp. weight) associated with the item whose value $\bar{x}_{ij}$ is equal to 1, in a current solution. Furthermore, $R^k$ is computed as follows:

$$R^k = R^k - \sum_{i=1}^{n} \sum_{j=1}^{r_i} \bar{w}^k_{ij}, \quad \forall k = 1, \ldots, m.$$  \hspace{1cm} (8)

Since both the cost and weight of the item $(\bar{i}, \bar{j})$ associated with $\bar{x}_{ij} = 1$ are reduced to zero in MMKP$_{AP}(\bar{x})$. As result, all these items can be omitted from the auxiliary problem. Hence, MMKP$_{AP}(\bar{x})$ contains fewer decision variables than the original MMKP.

Then, we can now express the sense of each type of considered constraints (5) and (6). Indeed, constraints (5) ensure that the total exchange weight is under the residual resource of the original MMKP, while constraints (6) guarantee that the exchange is performed in a one-to-one way. One can also remark that using constraints (6) in MMKP$_{AP}(\bar{x})$ implies that no more equality constraints need be considered as more complicated constraints.

We now prove that the solution of the auxiliary problem represents an optimal solution for the original MMKP. Indeed, let $\hat{x}$ be a feasible solution of MMKP$_{AP}(\bar{x})$. It is easy to see that a feasible solution for MMKP, namely $x$, can be obtained as follows:
Let \( \forall i \in \{1, \ldots, n\} \) and \( \forall j \in \{1, \ldots, r_i\} \), \( x_{ij} = 1 \) if one of the two following conditions is satisfied:

(i) \( \hat{x}_{ij} = 1 \) or

(ii) \( \tilde{x}_{ij} = 1 \) and \( \hat{x}_{ij} = 0 \), \( \forall j \in \{1, \ldots, r_i\} \).

Condition (i) indicates that the exchange is happened in the \( i^{th} \) class and condition (ii) indicates that no-exchange is realized in the \( i^{th} \) class. Denote the resulting vector as \( x^o := x(\tilde{x}, \hat{x}) \) which is a feasible solution for MMKP, where \( \tilde{x} \) denotes the solution vector of \( \text{MMKP}_{LR}(\lambda) \) and \( \hat{x} \) is a feasible solution of \( \text{MMKP}_{AP}(\tilde{x}) \).

**Proposition 4.1** Let \( \tilde{x} \) be a solution vector of \( \text{MMKP}_{LR}(\lambda) \) (\( \forall \lambda \in \mathbb{R}^+ \)). Then, \( x^o \) is a feasible solution for the original MMKP if \( \hat{x} \) is a feasible solution of \( \text{MMKP}_{AP}(\tilde{x}) \).

**Proof.** Let \( I = I_n \cup I_c \) be the index set of the items whose decision variables are equal to one in \( x^o \), where \( I_n \) characterizes the set of indexes selected in both \( \tilde{x} \) and \( x^o \), \( I_c \) is the set of indexes containing the items selected in both \( x^o \) and \( \hat{x} \). Differently stated, these sets of indexes can be defined as follows:

\[
I = \{(i,j) \mid x^o_{ij} = 1, i = 1, \ldots, n, j = 1, \ldots, r_i\},
\]

\[
I_n = \{(i,j) \mid x^o_{ij} = 1 \text{ and } \tilde{x}_{ij} = 1, i = 1, \ldots, n, j = 1, \ldots, r_i\} \quad \text{and}
\]

\[
I_c = \{(i,j) \mid x^o_{ij} = 1 \text{ and } \hat{x}_{ij} = 1, i = 1, \ldots, n, j = 1, \ldots, r_i\}.
\]

Thus, \( w^k x^o \) can be rewritten as follows:

\[
w^k x^o = \sum_{(i,j) \in I_n} w^k_{ij} x^o_{ij} + \sum_{(i,j) \in I_c} w^k_{ij} x^o_{ij}, \forall k \in \{1, \ldots, m\}.
\]

Since \( \hat{x} \) is feasible for \( \text{MMKP}_{AP}(\tilde{x}) \), the following inequality is satisfied:

\[
\sum_{(i,j) \in I_c} w^k_{ij} x^o_{ij} \leq R^k, \forall k \in \{1, \ldots, m\}.
\]

From Equalities (7) and (8), Inequality (9) can be re-expressed as:

\[
\sum_{(i,j) \in I_c} (w^k_{ij} - \bar{w}^k_{ij}) x^o_{ij} \leq R^k - \sum_{i=1}^{n} \sum_{j=1}^{r_i} \bar{w}^k_{ij}, \forall k \in \{1, \ldots, m\}.
\]

By adding \( \sum_{(i,j) \in I_n} w^k_{ij} x^o_{ij} \) to two sides, Inequality (10) can be rewritten as:

\[
\sum_{(i,j) \in I_n} w^k_{ij} x^o_{ij} + \sum_{(i,j) \in I_c} (w^k_{ij} - \bar{w}^k_{ij}) x^o_{ij} \leq R^k - \sum_{i=1}^{n} \sum_{j=1}^{r_i} \bar{w}^k_{ij} + \sum_{(i,j) \in I_n} \bar{w}^k_{ij} x^o_{ij}, \forall k.
\]

By re-expressing both sides of inequality (11), \( \forall k \in \{1, \ldots, m\} \) we obtain

\[
\sum_{(i,j) \in I_n} w^k_{ij} x^o_{ij} + \sum_{(i,j) \in I_c} \bar{w}^k_{ij} x^o_{ij} \leq R^k - \sum_{i=1}^{n} \sum_{j=1}^{r_i} \bar{w}^k_{ij} + \sum_{(i,j) \in I_n} w^k_{ij} x^o_{ij} + \sum_{(i,j) \in I_c} \bar{w}^k_{ij} x^o_{ij}.
\]
We recall that \( w_{ij}^k \) is a weight relative to the item selected in \( \bar{x} \). Therefore,

\[
\sum_{(i,j) \in I_n} w_{ij}^k x_{ij}^o = \sum_{(i,j) \in I_n} \bar{w}_{ij}^k \quad \text{and} \quad \sum_{(i,j) \in I_c} \bar{w}_{ij}^k x_{ij}^o = \sum_{(i,j) \in I_c} \bar{w}_{ij}^k.
\]

Recalling that \( I = I_n \cup I_c \), we then have

\[
\sum_{i=1}^{n} \sum_{j=1}^{r_i} w_{ij}^k = \sum_{(i,j) \in I_n} w_{ij}^k x_{ij}^o + \sum_{(i,j) \in I_c} \bar{w}_{ij}^k x_{ij}^o = R_k^k, \quad \forall k = 1, \ldots, m.
\]

Combining both inequalities (12) and (13), we deduce that

\[
\sum_{(i,j) \in I_n} w_{ij}^k x_{ij}^o + \sum_{(i,j) \in I_c} \bar{w}_{ij}^k x_{ij}^o \leq R_k^k, \quad \forall k = 1, \ldots, m.
\]

Hence, \( x^o \) is a feasible solution for the MMKP.

From Proposition (4.1), a lower bound \( LB_{AP}(\bar{x}) \) (resp. upper bound \( UB_{AP}(\bar{x}) \)) of \( MMKP_{AP}(\bar{x}) \) can be transformed to represent a lower bound \( LB \) (resp. upper bound \( UB \)) for MMKP as follow:

\[
LB = \sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij} \bar{x}_{ij} + LB_{AP}(\bar{x}) \quad \text{and} \quad UB = \sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij} \bar{x}_{ij} + UB_{AP}(\bar{x}).
\]

**Proposition 4.2** Let \( \bar{x} \) be a solution vector of the original \( MMKP_{LR}(\lambda) \) (\( \forall \lambda \in \mathbb{R}^+ \)). Then, \( x^o \) is an optimal solution for MMKP if \( \hat{x} \) is an optimal solution for \( MMKP_{AP}(\bar{x}) \).

**Proof.** Let \( \hat{x} \) be an optimal solution of \( MMKP_{AP}(\bar{x}) \). From Proposition (4.1), \( x^o \) is a feasible solution of MMKP. According to the formula (14), the corresponding value of \( c \cdot x^o \) can be computed as follow:

\[
c \cdot x^o = \sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij} \bar{x}_{ij}^o + \sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij} \hat{x}_{ij}.
\]

We proceed by contradiction on the optimality of the solution of the MMKP. Suppose that there exists a feasible solution \( x^* \) to MMKP such that \( c \cdot x^* > c \cdot x^o \). Let \( \hat{x}^* \) be a solution for \( MMKP_{AP}(\bar{x}) \) induced from \( x^* \) such that \( \forall i \in \{1, \ldots, n\}, j \in \{1, \ldots, r_i\} \):

\[
\hat{x}_{ij}^* = \begin{cases} 
1 & \text{if } x_{ij}^* = 1 \text{ and } \bar{x}_{ij} = 0 \\
0 & \text{otherwise}.
\end{cases}
\]

Let \( I = I_n \cup I_d \) be the index set of the items whose decision variables are fixed to one in \( x^* \), where \( I_n \) is the set of indexes containing the items selected in both \( \bar{x} \) and
$x^*$ and $I_d$ is the set of indexes containing the items selected in both $x^*$ and $\hat{x}^*$, such that

$$I = \{(i, j) \mid x^*_{ij} = 1, \ i = 1, \ldots, n, \ j = 1, \ldots, r_i\},$$

$$I_n = \{(i, j) \mid x^*_{ij} = 1 \text{ and } \bar{x}_{ij} = 1, \ i = 1, \ldots, n, \ j = 1, \ldots, r_i\} \quad \text{and}$$

$$I_d = \{(i, j) \mid x^*_{ij} = 1 \text{ and } \bar{x}^*_{ij} = 1, \ i = 1, \ldots, n, \ j = 1, \ldots, r_i\}.$$ 

Since $x^*$ is feasible for MMKP, the following inequalities hold:

$$\sum_{(i,j) \in I} w_{ij}^k x^*_{ij} \leq R^k \iff \sum_{(i,j) \in I_n} w_{ij}^k x^*_{ij} + \sum_{(i,j) \in I_d} w_{ij}^k \hat{x}^*_{ij} \leq R^k, \ \forall k \in \{1, \ldots, m\}$$ (16)

By adding $\sum_{j=1}^{r_i} \bar{w}_{ij}^k$ in two sides of Inequality (16), we have By partitioning the second term, using the three sets $I$, $I_n$ and $I_d$, inequality (16) becomes

$$\sum_{(i,j) \in I_n} w_{ij}^k x^*_{ij} + \sum_{(i,j) \in I_d} w_{ij}^k \hat{x}^*_{ij} - \sum_{(i,j) \in I} \bar{w}_{ij}^k \leq R^k - \sum_{(i,j) \in I} \bar{w}_{ij}^k, \ \forall k \in \{1, \ldots, m\}$$

Re-expressing,

$$\sum_{(i,j) \in I_n} (w_{ij}^k - \bar{w}_{ij}^k)x^*_{ij} + \sum_{(i,j) \in I_d} (w_{ij}^k - \bar{w}_{ij}^k)\hat{x}^*_{ij} \leq R^k, \ \forall k \in \{1, \ldots, m\}$$ (17)

From inequality (17), we deduce that

$$\sum_{(i,j) \in I_d} w_{ij}^k \hat{x}^*_{ij} \leq R^k, \ \forall k \in \{1, \ldots, m\}. \quad (18)$$

From inequality (18) and the formula (14), we then deduce that $\hat{x}^*$ is a feasible solution for MMKP$_{AP}(\bar{x})$. Since $c \cdot x^* > c \cdot x^0$, from Equality (15) we then deduce that

$$\sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij} \bar{x}_{ij} + \sum_{(i,j) \in I_d} c_{ij} \hat{x}^*_{ij} > \sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij} \bar{x}_{ij} + \sum_{(i,j) \in I_d} c_{ij} \hat{x}_{ij}, \ \forall k \in \{1, \ldots, m\},$$

which yields

$$\sum_{(i,j) \in I_d} c_{ij} \hat{x}^*_{ij} > \sum_{(i,j) \in I_d} c_{ij} \hat{x}_{ij} \iff \sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij} x^*_{ij} > \sum_{i=1}^{n} \sum_{j=1}^{r_i} c_{ij} \hat{x}_{ij}. \quad (19)$$

Hence, Inequality (19) contradicts the fact that $\hat{x}$ is optimal for MMKP$_{AP}(\bar{x})$. □

Algorithm 1 describes the main steps of the Lagrangian neighborhood search (LNS) used for the MMKP. First, Step 1 of LNS is the initialization phase in which Lagrangian relaxation is applied to MMKP for computing an upper bound $UB$ and the optimal Lagrange Multipliers $\lambda^*$. We note $x(\lambda^*)$ as the optimal solution of MMKP$_{LR}(\lambda^*)$. Steps 2-4 are then applied for computing an optimal solution of the original MMKP by solving the auxiliary problem: MMKP$_{LR}(x(\lambda^*))$.  

50
Algorithm 1: Lagrangian Neighborhood Search: LNS

Require: An instance $I$ of MMKP.
Ensure: An optimal solution of MMKP.

1: Lagrangian relaxation is applied to MMKP for computing $(UB, x(\lambda^*))$
2: Build the auxiliary problem MMKP$_{LR}$ $(x(\lambda^*))$ on $I$
3: Solve MMKP$_{LR}$ $(x(\lambda^*))$
4: if MMKP$_{LR}$ $(x(\lambda^*))$ is infeasible then
5: Conclude that the given instance is infeasible.
6: else
7: Transform the optimal solution of MMKP$_{LR}$ $(x(\lambda^*))$ into an optimal solution of MMKP according to formula (14).
8: end if

5 Computational results

In this section, we investigate the effectiveness of the Lagrangian Neighborhood Search (LNS) on two sets of problem instances. The first set of instances contains moderate sized instances and the second one represents large-sized instances. First, we compare the performance of the Cplex solver with and without using the equivalent model. Second and last, we compare the quality of the provided results, when using the auxiliary model, with the best known solutions of the literature when the resolution is curtailed, i.e., the method is stopped after a limited runtime. The comparative study is made on a total of 48 instances, where the first group contains 15 instances (noted from InstHL1 to InstHL15) characterizing small- and middle-sized instances and the second group of instances contains 33 instances representing thirteen ones extracted from Khan et al. [1] (noted from I1 to I13, in fact, only the instances from I7 are considered since the other ones are easier to solve) and twenty instances taken from Hifi et al. [13] (noted from Inst1 to Inst20) varying from medium to large-scale ones. The Cplex solver using both the original and the equivalent models were tested on a Pentium Intel Core 2 Duo (2.93 Ghz and with 4 Gb of RAM).

We recall that transforming the original MMKP to an auxiliary problem MMKP$_{PA}$ permits to reduce $n$ decision variables. Moreover, the feasibility constraints — the equality constraints, i.e., constraints of type (2) — have been removed from the auxiliary problem. Such constraints are replaced with inequalities constraints, i.e., constraints of type (6).

5.1 Behavior of LNS: an exact resolution

In this part, we analyze the behavior of the exact LNS, namely LNS$_{opt}$, on the first set of problem instances which contains small-sized instances. The group is composed of 15 instances, noted from InstHL1 to InstHL15, where each of them was generated following Khan et al. [1]’s generator. Each instance is characterized by $m$, the number of classes, taken from 10 to 150 with an incrementation of 10 and,
each class is composed of 10 items. Finally, for each item $i$, $i = 1, \ldots, n$, both $r_i$ and $m$ are fixed to 10.

The results provided by LNS$^{opt}$, on the first set of instances, is summarized in Table 1 and it tallies the results reached by the Cplex solver applied to the original MMKP. In fact, three versions of LNS$^{opt}$ are investigated which depend on the starting solution used for rewriting the auxiliary problem MMKP$_{PA}$. The aim is to show the impact of the Lagrangian relaxation when applying the Cplex solver with the auxiliary problem. In order to justify such a choice, we consider the following these versions:

1. LNS$^{1opt}$: In this version, we neglect the Lagrangian relaxation and we shall use a greedy procedure (or a quick non-feasible solution) which allows to determine an upper bound for the original problem.

2. LNS$^{2opt}$: we build a quick feasible solution by calling the Cplex solver on the original model. In this case, the solver is stopped when the first feasible solution is reached; then its structure will be used to determine the auxiliary model.

3. LNS$^{opt}$: It represents the version in which the Lagrangian relaxation is applied to the original model. The structure of the solution obtained is used in order to construct the auxiliary model.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Cplex</th>
<th>LNS$^{1opt}$</th>
<th>LNS$^{2opt}$</th>
<th>LNS$^{opt}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$t_{cpu}$</td>
<td>$Nb_n$</td>
<td>$t_{cpu}$</td>
<td>$Nb_n$</td>
</tr>
<tr>
<td>InstHL1</td>
<td>0.14</td>
<td>658</td>
<td>0.49</td>
<td>1033</td>
</tr>
<tr>
<td>InstHL2</td>
<td>0.54</td>
<td>1943</td>
<td>0.53</td>
<td>1493</td>
</tr>
<tr>
<td>InstHL3</td>
<td>27.8</td>
<td>49447</td>
<td>33.81</td>
<td>48070</td>
</tr>
<tr>
<td>InstHL4</td>
<td>20.03</td>
<td>33852</td>
<td>30.93</td>
<td>48320</td>
</tr>
<tr>
<td>InstHL5</td>
<td>49.89</td>
<td>83745</td>
<td>37.83</td>
<td>89842</td>
</tr>
<tr>
<td>InstHL6</td>
<td>2.32</td>
<td>6227</td>
<td>1.84</td>
<td>7099</td>
</tr>
<tr>
<td>InstHL7</td>
<td>27.36</td>
<td>27448</td>
<td>22.75</td>
<td>20685</td>
</tr>
<tr>
<td>InstHL8</td>
<td>0.66</td>
<td>758</td>
<td>0.72</td>
<td>913</td>
</tr>
<tr>
<td>InstHL9</td>
<td>84.54</td>
<td>88955</td>
<td>58.77</td>
<td>98117</td>
</tr>
<tr>
<td>InstHL10</td>
<td>242.2</td>
<td>302627</td>
<td>125</td>
<td>33273</td>
</tr>
<tr>
<td>InstHL11</td>
<td>23.9</td>
<td>21405</td>
<td>32.93</td>
<td>22615</td>
</tr>
<tr>
<td>InstHL12</td>
<td>579.8</td>
<td>653771</td>
<td>183.7</td>
<td>483921</td>
</tr>
<tr>
<td>InstHL13</td>
<td>2112</td>
<td>2150805</td>
<td>769.44</td>
<td>2469655</td>
</tr>
<tr>
<td>InstHL14</td>
<td>70.41</td>
<td>89492</td>
<td>42.43</td>
<td>79513</td>
</tr>
<tr>
<td>InstHL15</td>
<td>472.6</td>
<td>486087</td>
<td>162.84</td>
<td>251408</td>
</tr>
<tr>
<td>Average</td>
<td>247.6</td>
<td>266481</td>
<td>100.3</td>
<td>317693</td>
</tr>
</tbody>
</table>

Table 1: Impact of the equivalent model versus the original model MMKP when using the Cplex solver: the first group of instances.

Let us note that the Cplex solver is applied on the inferred problem, i.e., the models provided when considering the three solutions constructed following the three points described above.
Table 1 summarizes the results realized by the three versions LNS\textsubscript{opt}\textsuperscript{1}, LNS\textsubscript{opt}\textsuperscript{2} and LNS\textsubscript{opt} as well as the results obtained by the Cplex solver v.12.1. For each version, the table reports the runtime (noted $t_{cpu}$) which requires to reach the optimal solution and, the number of generated nodes (noted $Nb_n$) realized for reaching the optimal solution.

Table 1 shows the behavior of the three versions of the method on the first group of instances. First, column 1 displays the instance’s name, columns 2 and 3 display the runtime needed by the Cplex solver (when using the original MMKP model) and $Nb_n$ corresponding to the number of nodes generated in order to reach the optimal solution. Second, the following three blocks tally the same information for the three versions of LNS, i.e., LNS\textsubscript{opt}\textsuperscript{1}, LNS\textsubscript{opt}\textsuperscript{2} and LNS\textsubscript{opt}, respectively. Third and last, the last line of the table reports the average runtime and the average number of nodes generated for all treated instances.

The study of Table 1 reveals the following:

1. Generally, the Cplex solver using the auxiliary problem has a better behavior, compared to the version using the original MMKP model. Indeed, for all instances, Cplex with the original model requires 247.6 seconds on average and evaluates 266481 nodes on average for reaching the optimum whereas the slowest version among the last three ones, where the equivalent model MMKP\textsubscript{AP} is considered, consumes only 110.5 seconds on average for reaching the optimal solutions.

2. Note that LNS\textsubscript{opt}, using the Lagrangian relaxation, has a better behavior than the two other versions (i.e., LNS\textsubscript{opt}\textsuperscript{1} and LNS\textsubscript{opt}\textsuperscript{2}). Indeed, such a version requires 58.7 seconds on average, compared to the 100.3 seconds on average of LNS\textsubscript{opt}\textsuperscript{1} and, it generates only 225831 nodes, on average.

Based on the above analysis, we can observe that introducing the auxiliary model induces a favorable average acceleration for solving all instances. In the same way, we can notice that the version using the Lagrangian relaxation has a better behavior compared to those two other versions.

5.2 Using LNS as a heuristic approach

Solving exactly a complex MMKP’s instance is generally not evident, i.e., an optimal solution of the instance cannot be even reached after a several hours. Hence, we considered LNS\textsubscript{opt} as a heuristic, namely LNS\textsubscript{heur}, especially when tackling complex medium and large-scale instances. In order to do it, we introduced a simple criterion, which is based on a limited runtime, a criterion often used in several experimental heuristics. We thus considered two versions of LNS\textsubscript{heur}: a first version limited by a runtime $t_1 = 3600$ seconds (noted LNS\textsubscript{heur}\textsuperscript{1}) and a second one in which the runtime limit is doubled: $t_2 = 7200$ seconds (noted LNS\textsubscript{heur}\textsuperscript{2}).

In this section, we study the behavior of both versions of the heuristic on the instances of the literature (cf., Cherfi and Hifi [7]), noted I7 to I13 and, Inst01 to Inst20. For such a group of instances, the provided results by both versions of the
heuristic are compared to the results of the literature extracted from Cherfi and Hifi [7], that we shall note CH. We also run the Cplex solver with both runtime limits $t_1$ and $t_2$, respectively.

Table 2 reports the obtained results: the first line shows the name of the tested instance, the second column displays the best solution values obtained by all the considered heuristics and, column 3 tallies the best solution values extracted from Cherfi and Hifi [7]. Column 4 (resp. column 5) reports the results reached by the Cplex solver (on the original model of the MMKP) with the runtime limit $t_1$ (resp. $t_2$) and column 6 (resp. column 7) displays the solutions obtained by LNS with $t_1$ (resp. $t_2$).

<table>
<thead>
<tr>
<th>Instance</th>
<th>Best solution</th>
<th>Cplex</th>
<th>LNS$_{heur}$</th>
<th>LNS$_{heur}^1$</th>
<th>LNS$_{heur}^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$t_1$</td>
<td>$t_2$</td>
<td>$t_1$</td>
<td>$t_2$</td>
</tr>
<tr>
<td>I7</td>
<td>24592</td>
<td>24584</td>
<td>24585</td>
<td>24592</td>
<td>24592</td>
</tr>
<tr>
<td>I8</td>
<td>36894</td>
<td>36879</td>
<td>36879</td>
<td>36886</td>
<td>36886</td>
</tr>
<tr>
<td>I9</td>
<td>49180</td>
<td>49178</td>
<td>49178</td>
<td>49180</td>
<td>49180</td>
</tr>
<tr>
<td>I10</td>
<td>61480</td>
<td>61480</td>
<td>61480</td>
<td>61468</td>
<td>61468</td>
</tr>
<tr>
<td>I11</td>
<td>73783</td>
<td>73777</td>
<td>73777</td>
<td>73783</td>
<td>73783</td>
</tr>
<tr>
<td>I12</td>
<td>86097</td>
<td>86077</td>
<td>86077</td>
<td>86084</td>
<td>86097</td>
</tr>
<tr>
<td>I13</td>
<td>98438</td>
<td>98431</td>
<td>98432</td>
<td>98436</td>
<td>98436</td>
</tr>
<tr>
<td>Inst 01</td>
<td>10738</td>
<td>10727</td>
<td>10738</td>
<td>10738</td>
<td>10738</td>
</tr>
<tr>
<td>Inst 02</td>
<td>13598</td>
<td>13598</td>
<td>13598</td>
<td>13598</td>
<td>13598</td>
</tr>
<tr>
<td>Inst 03</td>
<td>10955</td>
<td>10945</td>
<td>10945</td>
<td>10955</td>
<td>10955</td>
</tr>
<tr>
<td>Inst 04</td>
<td>14445</td>
<td>14441</td>
<td>14442</td>
<td>14442</td>
<td>14445</td>
</tr>
<tr>
<td>Inst 05</td>
<td>17059</td>
<td>17055</td>
<td>17059</td>
<td>17046</td>
<td>17055</td>
</tr>
<tr>
<td>Inst 06</td>
<td>16835</td>
<td>16826</td>
<td>16834</td>
<td>16835</td>
<td>16835</td>
</tr>
<tr>
<td>Inst 07</td>
<td>16440</td>
<td>16440</td>
<td>16440</td>
<td>16440</td>
<td>16440</td>
</tr>
<tr>
<td>Inst 08</td>
<td>17514</td>
<td>17507</td>
<td>17511</td>
<td>17507</td>
<td>17514</td>
</tr>
<tr>
<td>Inst 09</td>
<td>17761</td>
<td>17761</td>
<td>17743</td>
<td>17757</td>
<td>17757</td>
</tr>
<tr>
<td>Inst 10</td>
<td>19316</td>
<td>19297</td>
<td>19297</td>
<td>19305</td>
<td>19305</td>
</tr>
<tr>
<td>Inst 11</td>
<td>19446</td>
<td>19446</td>
<td>19446</td>
<td>19441</td>
<td>19441</td>
</tr>
<tr>
<td>Inst 12</td>
<td>21739</td>
<td>21739</td>
<td>21739</td>
<td>21739</td>
<td>21739</td>
</tr>
<tr>
<td>Inst 13</td>
<td>21577</td>
<td>21572</td>
<td>21572</td>
<td>21577</td>
<td>21577</td>
</tr>
<tr>
<td>Inst 14</td>
<td>32874</td>
<td>32869</td>
<td>32869</td>
<td>32873</td>
<td>32873</td>
</tr>
<tr>
<td>Inst 15</td>
<td>39160</td>
<td>39157</td>
<td>39157</td>
<td>39157</td>
<td>39159</td>
</tr>
<tr>
<td>Inst 16</td>
<td>43364</td>
<td>43361</td>
<td>43362</td>
<td>43362</td>
<td>43364</td>
</tr>
<tr>
<td>Inst 17</td>
<td>54360</td>
<td>54360</td>
<td>54360</td>
<td>54352</td>
<td>54360</td>
</tr>
<tr>
<td>Inst 18</td>
<td>60464</td>
<td>60461</td>
<td>60461</td>
<td>60463</td>
<td>60463</td>
</tr>
<tr>
<td>Inst 19</td>
<td>64930</td>
<td>64929</td>
<td>64930</td>
<td>64922</td>
<td>64928</td>
</tr>
<tr>
<td>Inst 20</td>
<td>75615</td>
<td>75613</td>
<td>75613</td>
<td>75609</td>
<td>75615</td>
</tr>
</tbody>
</table>

Table 2: Performance of LNS as a heuristic on the second group of instances of the literature.

From Table 2, we observe what follows:
1. For the first runtime limit $t_1$:

(a) $\text{LNS}^1_{\text{heur}}$ is able to improve 7 solutions out of 27, according to the solutions of CH. Also, on the one hand, CH outperforms the Cplex solver since 17 solution values reached by the Cplex remain less than those provided by CH. On the other hand, only 12 better solutions are matched by $\text{LNS}^1_{\text{heur}}$.

(b) Furthermore, $\text{LNS}^1_{\text{heur}}$ dominates the Cplex solver because it realizes 16 improvements and matches 5 other ones. Let us note that such observations allow, once again, to justify the importance of the use of the auxiliary model.

(c) Regarding the solutions produced by CH, $\text{LNS}^1_{\text{heur}}$ remains unable to improve in 12 cases the best solutions: (i) one less good solution on the first group and 11 less good solutions on the second one.

2. For the second runtime limit $t_2$:

(a) According to the results displayed, we can observe that the Cplex solver is able to improve 3 other solutions when doubling the runtime. In this case, ten new solutions are obtained with regard the solutions given by CH, six better solutions are matched and nine solutions are less good than those realized by CH.

(b) Besides, $\text{LNS}^2_{\text{heur}}$ gives 13 new solutions than those produced by CH, matches 7 solutions and fails in 9 occasions. When comparing both $\text{LNS}^2_{\text{heur}}$ and Cplex, we can observe that $\text{LNS}^2_{\text{heur}}$ realizes 18 better solutions than those given by the Cplex, it matches 5 solutions and fails in 4 occasions.

(c) From column 2, and by comparing all reported solutions produced by each solution procedure, we can remark that in 16 occasions $\text{LNS}^2_{\text{heur}}$ realizes the best solutions, CH matches 14 better solutions whereas Cplex matches only 9 solutions out of 27. Hence, such analysis confirms the interest of the auxiliary model $\text{MMKP}_{PA}$ instead of the original model.

Regarding the analysis made on the obtained results of Tables 1 and 2, we can conclude that the auxiliary problem is more interesting to use than the original one.

6 Conclusion

We solved the Multiple-choice Multi-dimensional Knapsack Problem (MMKP) using an algorithm based on Lagrangian neighborhood search. First, we proposed an equivalent model which is based on the Lagrangian relaxation for exactly solving the MMKP. Second, in order to tackle complex and large-scale instances, we adapted the method for providing a heuristic: a tree-search curtailed with a limited runtime. Third, we investigated the performance of both versions of the proposed method (exact and heuristic) experimentally using a set of benchmark instances. The use of the equivalent model, by application the Cplex solver, showed that it is more interesting
to consider the equivalent model contrary to the original one. Finally, we showed that the second version of the method, considered as a heuristic, dominates all the heuristic approaches yielding higher quality solutions within the same runtime limits.

Acknowledgments. The authors thank the referees for their helpful comments and suggestions which contributed to the improvement of the presentation of this paper.

References


