The three dimensional knapsack problem with balancing constraints

The Three-Dimensional Knapsack Problem with Balancing Constraints

Mauro Maria Baldi*†, Guido Perboli†‡, and Roberto Tadei†
*Department of Control and Computer Engineering
Politecnico di Torino
Corso Duca degli Abruzzi 24, 10129 Torino (Italy)
Email: mauro.baldi, guido.perboli, roberto.tadei@polito.it
‡CIRRELT, Montreal, Canada
†Corresponding author

***Abstract*— In this paper we introduce a new packing problem, the Three-Dimensional Knapsack Problem with Balancing Constraints (3BKP), the extension of the standard Three-Dimensional Knapsack Problem (3KP) where additional constraints related to the center of mass of the three-dimensional packing are given. Given a set of box items $i = 1, \dots, n$ with sizes w_i, d_i and k_i, a profit p_i, and a mass m_i and a container called knapsack of fixed dimensions W, D and H, 3BKP consists in orthogonally packing a subset of the items into the knapsack in order to maximize the sum of the profits of the loaded items. The items must be accommodated into the knapsack such that they do not overlap. Moreover, the center of mass of the overall packing must lie into a predefined boxed domain within the knapsack. We assume that items can be rotated. We give a MIP formulation of the problem, used to derive bounds, as well as an efficient heuristic method able to solve, with a limited computational effort, the test instances. Moreover, new test instances are introduced and used to derive extensive computational results. The results show how the MIP model is able to find better bounds than other relaxations, and how the heuristic method is able to efficiently solve both instances explicitly designed for 3BKP, as well as to be competitive with methods explicitly designed to solve 3KP.**

***Keywords*— 3D Knapsack, load balancing, Heuristics, MIP Models**

I. Introduction

A major challenge in the loading problem is taking into account load balancing constraints. These kind of constraints arise in many practical applications as aircraft loading (Kaluzny and Shaw [23]), space cargo loading (Colaneri et al. [9], and Perboli Perboli2002) and maritime transportation (Bischoff and Ratcliff [6]). The issue is critical in some applications, as in the space cargo loading, while it is relevant from the safety and the economic points of view in air and maritime cargo applications. For example minor displacements from an ideal center of mass can result in increased fuel consumption for aircrafts and ships (Mongeau and Bés [26]). Despite its importance, the issue of the loading balancing has not deeply studied. This is mainly due to the difficulty of extending formulations, exact and heuristic methods for multidimensional packing to the balanced case. In fact, the majority of these methods use geometric properties in order to reduce the computational effort which cannot be extended to the balancing constraints case. The aim of this paper is twofold. First, we introduce a new packing problem, the Three-Dimensional Knapsack Problem with Balancing Constraints, an extension of the standard Three-Dimensional Knapsack Problem (3KP) where additional constraints related to the center of mass of the three-dimensional packing are given. Given a set of box items $i = 1, \dots, n$ with sizes w_i, d_i and k_i, a profit p_i, and a mass m_i and a container called knapsack of fixed dimensions W, D, and H, the Three-Dimensional Knapsack Problem with Balancing Constraints (3BKP) consists in orthogonally packing a subset of the items into the knapsack in order to maximize the sum of the profits of the loaded items. The items must be accommodated into the knapsack such that they do not overlap. Moreover, the center of mass of the overall packing must lie into a predefined boxed domain within the knapsack. We assume that items can be rotated. Second, we give a MIP formulation of the problem, used to derive bounds. Finally, we introduce 3BKP-U, a heuristic method generalizing the UniPack heuristic for multi-dimensional packing (Crainic et al. [28]) and the extreme point rule (Crainic et al. [10]) for the accommodation of the items in order to deal with the center of mass constraints. New tests instances are introduced and used to derive extensive computational results. The results show how the MIP model is able to find better bounds than other relaxations, as well as the heuristic method is able to efficiently solve both

instances explicitly designed for 3BKP, as well as to be competitive with methods explicitly designed for solving 3KP.

In more details, the paper is organized as follows. In Section II we formally introduce the problem. After showing conventions and rules involved in 3BKP (most of them are common to other packing problems), in II-A we give a formulation for the model. In section III a state of the art is given; first for multidimensional packing problems, second for multidimensional knapsack problems and, finally, for problems dealing with balancing constraints. Section IV introduces UniPack and its history. It is based on the concept of Extreme Points, which are discussed in detail. UniPack specialization leads to 3BKP-U, the heuristic employed to solve 3BKP. In section V results of our work are shown and we conclude in section VI.

II. Problem description

The 3BKP is defined as follows: given a container C with dimensions W, D, and H, volume $V = W \times D \times H$, and a set of items $J = \{1, ..., n\}$ with profit p_j, dimensions w_j, d_j, and h_j, and specific weight sw_j, we want to assign a subset of items $J' \subseteq J$ to the container C such that J' is a feasible loading for the container itself, and the total profit of the loaded items is maximum. Feasibility requires that loaded items do not overlap and limits the position of the overall center of mass inside a given three-dimensional domain. Figure 1 shows an example of a three-dimensional domain with its projections on the (X, Y), (X, Z), and (Y, Z) plans. Following the classification of Wäscher et al. [29], 3BKP is a Three-Dimensional Single Large Object Placement Problem (3D-SLOPP) with balancing constraints (3DB-SLOPP).

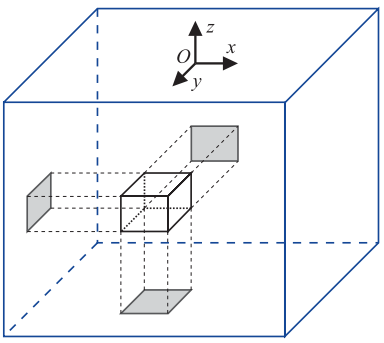

Fig. 1. A three dimensional domain

Furthermore the following assumptions are made:

- the items and the container have parallelepiped shape;
- the origin of the container and the origin of each item is located at their own left-back-down corners (see Figure 2);
- the container is located in the positive quadrant of the Cartesian coordinate system, with its origin placed in position $(0,\ 0,\ 0)$ (see Figure 2);
- items can rotate so that each item side is parallel to one axis;
- container and items walls have negligible thickness;
- container and items dimensions are assumed to be non-negative integers;
- the mass of each item is uniformly distributed over its volume.

Let $|J'| = k \leq n$ be the number of accommodated items; then the value of the overall profit P can be calculated as:

$$P = \sum_{j=1}^{k} p_j. \tag{1}$$

Note that high or low values of P do not necessarily correspond to a high or a low exploitation of the bin because, in principle, there is no correlation between volumes and profits of items.

Given a parallepiped item j, its position $\vec{x}_j$, and its mass m_j, then the position of its center of mass is:

$$\vec{x}_{CM_j} = (x_j + w_j/2,\ y_j + d_j/2,\ z_j + h_j/2)\,. \tag{2}$$

The position of the overall center of mass is then:

$$\vec{x}_{CM} = \frac{\sum_{j=1}^{k} \vec{x}_{CM_j} m_j}{\sum_{j=1}^{k} m_j}. \tag{3}$$

Let the solution unbalancing index U be the dispersion index of packed items center of mass $\vec{x}_{CM}$ with respect to a desired position $\vec{x}'_{CM}$. The standard deviation of a set of values x_j, with $j \in [1, k]$ and arithmetic average $\bar{x}$, is defined as follows:

$$\sigma_x = \sqrt{\frac{\sum_{j=1}^{k} (x_j - \bar{x})^2}{k}}. \tag{4}$$

Formula for the solution unbalancing index U is found by plugging in (4) centers of mass coordinates. Moreover, since positions are assumed to be vectors, the square power of the difference at the numerator must be applied to the modulus of the difference:

$$U = \sqrt{\frac{\sum_{j=1}^{k} \left|\vec{x}_{CM_j} - \vec{x}'_{CM}\right|^2}{k}}. \tag{5}$$

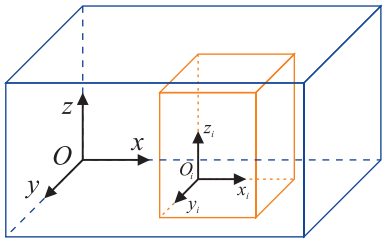

Fig. 2. Container and items placement

A. The model

Before showing the model, constants and variables need to be introduced. As mentioned above, J is the set of items and its cardinality is n. Associated indexes may be i or j. Δ is the set of dimensions. Its cardinality is 2 for two-dimensional problems and 3 for three-dimensional ones. In the latter case $\Delta = \{1, 2, 3\}$. Its associated index is δ (Greek d delta stands for *dimension*). R is the set of rotations. Its cardinality is 4 for two-dimensional problems and 6 for three-dimensional ones. These numbers represent the ways one can rotate items in a 2D or 3D domain. Index associated to set R is r (r stands for *rotation*). Due to reason of space in the representation of the model, we omit (but we do mean it) $i \in J$, $j \in J$ and $\delta \in \Delta$ when subscripts appear within the constraints. p_j, v_j and m_j are, respectively, the profit, the volume and the mass of item j. s^{δ}_{ir} is the dimension (s stands for *size*) of item i along direction δ when rotated with respect to index r (see Figure 3). S^{δ} is the bin *size* along direction δ; in particular $\left(S^1, S^2, S^3\right) = (W, D, H)$. V is the bin volume (note that $V = \Pi_{\delta \in \Delta} S^{\delta}$). L^{δ} and U^{δ} are the bounds along direction δ which limit the domain where the overall center of mass must lie within. γ^{δ}_{ir} is the position of item i center of mass along direction δ when it is rotated according to rotation r and when it is placed with its bottom-left point in the axis origin. For instance, according to Figure 3 if item i undergo a rotation with $r = 3$, then we have $\left(\gamma^1_{i3}, \gamma^2_{i3}, \gamma^3_{i3}\right) = (d_i/2, w_i/2, h_i/2)$. Greek g gamma stands for *gravity* because the center of mass is the point where gravity force acts on a body like if it were concentrated just on that point. UB is an upper bound on the overall profit. Its value may come from a pre-processing step or may be known a priori. Talking about variables, t_j is a binary variable which value is 1 if item j is loaded in the container, 0 otherwise (t stands for *taken*). χ^{δ}_i is the coordinate of the bottom-left point of item i along direction δ. For instance, if such a point is placed in position (x_i, y_i, z_i), then we have $(\chi^1_i, \chi^2_i, \chi^3_i) = (x_i, y_i, z_i)$. b^{δ}_{ij} is a binary variable which value is 1 if item i comes *before* item j along direction δ, 0 otherwise. Finally ρ_{ir} is a binary variable which is 1 if item i is rotated according to rotation r, 0 otherwise. The model for the Three-Dimensional Knapsack Problem with Balancing Constraints can then be formulated as follows:

$$\max \sum_{j \in J} p_j t_j \tag{6}$$

$$\text{s.t.} \quad \sum_{j \in J} v_j t_j \leq V \tag{7}$$

$$\sum_{\delta \in \Delta} (b^{\delta}_{ij} + b^{\delta}_{ji}) \geq t_i + t_j - 1, \quad i < j \tag{8}$$

$$\chi^{\delta}_i + \sum_{r \in R} s^{\delta}_i \rho_{ir} \leq S^{\delta} \tag{9}$$

$$\chi^{\delta}_i + \sum_{r \in R} s^{\delta}_{ir} \rho_{ir} \leq \chi^{\delta}_j + M(1 - b^{\delta}_{ij}),\ i < j \tag{10}$$

$$\chi^{\delta}_j + \sum_{r \in R} s^{\delta}_{jr} \rho_{ir} \leq \chi^{\delta}_i + M(1 - b^{\delta}_{ji}),\ i < j \tag{11}$$

$$\chi^{\delta}_i \leq M t_i \tag{12}$$

$$b^{\delta}_{ij} \leq t_i \tag{13}$$

$$b^{\delta}_{ij} \leq t_j \tag{14}$$

$$\sum_{i \in J} m_i \chi^{\delta}_i + \sum_{i \in J} \sum_{r \in R} m_i \gamma^{\delta}_{ir} \rho_{ir} \geq L^{\delta} \sum_{i \in J} m_i t_i \tag{15}$$

$$\sum_{i \in J} m_i \chi^{\delta}_i + \sum_{i \in J} \sum_{r \in R} m_i \gamma^{\delta}_{ir} \rho_{ir} \leq U^{\delta} \sum_{i \in J} m_i t_i \tag{16}$$

$$\sum_{j \in J} p_j t_j \leq UB \tag{17}$$

$$t_i \in \{0, 1\} \tag{18}$$

$$b^{\delta}_{ij} \in \{0, 1\} \tag{19}$$

$$\chi^{\delta}_i \geq 0 \tag{20}$$

$$\rho_{ir} \in \{0, 1\}. \tag{21}$$

The objective function (6) is expressed as the sum of the profit of the items (including the selection variables t_j). Constraint (7) expresses capacity constraints, i.e. the sum of the volumes of the selected items must not exceed the volume of the container. Constraints (8) ensures that two packed items do not overlap. Constraints (9) state that items must lie within the container, i.e., for each direction δ, the sum of the coordinate of the bottom-left

$$\begin{aligned}
\left(s^1_{i1},\, s^2_{i1},\, s^3_{i1}\right) &= (w_i,\, d_i,\, h_i)\\
\left(s^1_{i2},\, s^2_{i2},\, s^3_{i2}\right) &= (w_i,\, h_i,\, d_i)\\
\left(s^1_{i3},\, s^2_{i3},\, s^3_{i3}\right) &= (d_i,\, w_i,\, h_i)\\
\left(s^1_{i4},\, s^2_{i4},\, s^3_{i4}\right) &= (d_i,\, h_i,\, w_i)\\
\left(s^1_{i5},\, s^2_{i5},\, s^3_{i5}\right) &= (h_i,\, w_i,\, d_i)\\
\left(s^1_{i6},\, s^2_{i6},\, s^3_{i6}\right) &= (h_i,\, d_i,\, w_i).
\end{aligned}$$

Fig. 3. List of all the possible rotations of an item in 3D

point with the dimension of the item must give a value less or equal than the size of the bin along dimension δ. Constraints (10) state that, if item i comes *before* item j, then the sum of the position of item i plus its size must be less or equal than the position value of item j along direction k. Constraints (11) have the same meaning, this time with item j coming before item i. Constraints (12) express that, if item i is not selected, then its placement coordinates must be zero. A similar meaning have constraints (13) and (14) that state that, if an item is not selected, then it cannot be placed before another one. Constraints (15) and (16) ensures balancing conditions and they can be derived from the center of mass definition (3). Constraint (17) is used if an upper bound UB on the overall profit is known. Finally follow the involved variables domains.

III. State of the art

3BKP can be classified as a problem beloning to the Cutting and Packing (C&P) family. Wäscher et al. [29] have recently published a classification for C&P problems which extendes an older one due to Dyckhoff [12]. The authors characterize a C&P problem with two sets of elements: a set of large objects (bins, containers, input, supply) and a set of small items (output, demand). Multidimensional packing literature is really widespread. As stated by Wäscher et al. [29], at the time of their publication, *the number of pubblications in the area of C&P has increased considerably over the last two decades*. This issue was also one of the motivations to create a new typology for C&P problems. A first attempt to model multidimensional packing was due to Gilmore and Gomory [19]. Their column generation approach has been reviseted by Baldacci and Boschetti [1]. Other contributions come from Beasley [3], Hadjiconstantinou and Christofides [21], Chung et al. [8], Berkey and Wang [5], George and Robinson [18], Fekete and Schepers [15], [17], and Perboli [27]. Martello et al. [24] introduced the concept of corner points. Extentions of their work can be found in den Boef et al. [11], and Martello et al. [25]. Crainic et al. [10] introduced the concept of extreme points, an extension of the corner points previously introduced by Martello et al. [24]. Extreme Points are the basis for UniPack, the heuristic introduced in this paper, able to efficiently solve several packing problems. In multidimensional knapsack problems the available bins reduce to one. Papers which tackle this problem are Beasley [2], Hadjiconstantinou and Christofides [21], Boschetti et al. [7], and Fekete and Schepers [15], [16], [17]. To the best of our knowledge the latest contribution comes from Egeblad and Pisinger [13], where the authors also propose an exact model. An exact MIP model for the 3BKP can be found in Fasano [14], where additional equations to meet the balancing conditions (i.e. the overall center of mass must lie within a given convex domain) are taken from Williams [30]. As stated in Fasano [14], *the MIP model [...] is hard to solve using standard techniques* and that justifies a heuristic way in the solution of 3BKP. Note that balancing conditions must not be meant as of stability ones. In Junqueira et al. [22] stability is defined as the capacity of the loaded boxes to withstand the gravity force acceleration (vertical stability) or the inertia of its own bodies (horizontal stability), whilst in 3BKP balancing conditions are requirements on the overall center of mass of the loaded items. The latter, in fact, must lie within a given convex domain or, when possible, at a certain position inside the container.

IV. An efficient heuristic

UniPack is based on the fundamental idea to separate the feasibility of a solution, i.e. the accomodation of the items, from its optimality, which is related to the specific packing problem. UniPack is a heuristic able to solve many packing problems having different objective functions and constraints. It has been created following the ideas of Crainic et al. [10] where Extreme Points (*EP*s) have been introduced. These are a further extension of the Corner Points introduced by Martello et al. [24].

Corner points are the nondominated locations where an item can be placed into an existing packing. In two dimensions, corner points are defined where the envelope of the items in the bin changes from vertical to horizontal (the large dots in Figure 4).

Heuristics using Corner Points can be inefficient in terms of container utilization. Consider, for example, the packing depicted in Figure 4 and item 11. According to the definition of the Corner Points, one can add the item on any of the large black dots. It is clear, however, that item 11 could also be placed into one of the shaded regions, which the corner points do not allow us to exploit.

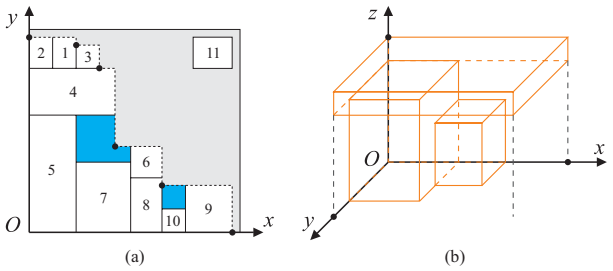

Fig. 4. Corner points in 3D and 2D packings

Extreme Points (*EP*s) introduced in Crainic et al. [10] provide the means to exploit the free space defined inside a packing by the shapes of the items already in the container. Figure 5 illustrates *EP*s in 3D and 2D packings.

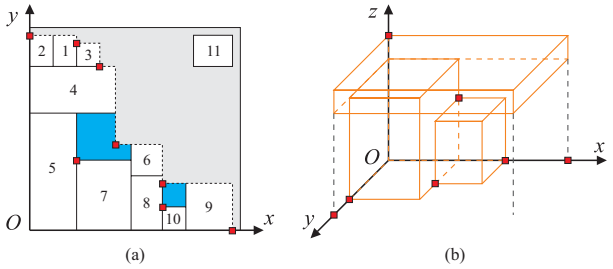

Fig. 5. Extreme Points in 3D and 2D packings

The basic idea of the *EP*s is that when an item j with sizes $(w_j,\, d_j,\, h_j)$ is added to a given packing and it is placed with its left-back-down corner in position $(x_j,\, y_j,\, z_j)$, it generates a series of new potential points, the *EP*s, where additional items can be accommodated. The new *EP*s are generated by projecting the points with coordinates $(x_j + w_j,\, y_j,\, z_j)$, $(x_j,\, y_j + d_j,\, z_j)$, and $(x_j,\, y_j,\, z_j + h_j)$ on the orthogonal axes of the container. Figure 6 illustrates the concept.

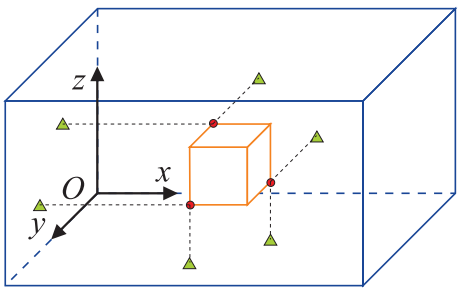

Fig. 6. Extreme Points defined by an items (the triangles

Beside the saving of space in applying extreme points rather than corner points, another advantage is the time complexity to find an extreme points set. As proved in Crainic et al. [10], the overall computational effort is $\mathcal{O}(n)$- where n is the number of items- whilst corner points require a $\mathcal{O}(n^2)$ complexity.

As stated before, UniPack exploits the concept of Extreme Points. Its general scheme is depicted in Figure 7.

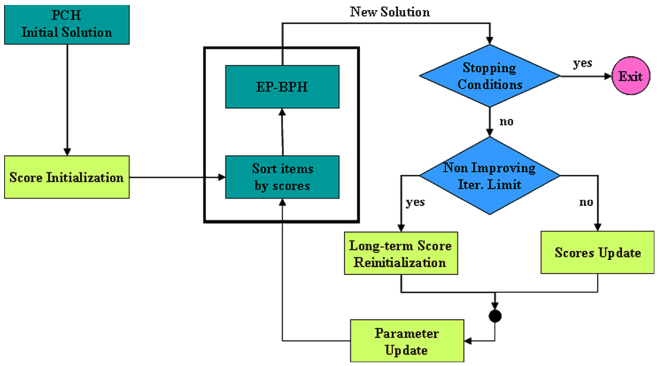

Fig. 7. General scheme of UniPack

The core of UniPack is an accomodation procedure, the *EP*-based constructive heuristic, EP-BPH.

When applied to the initial solution, the name of the heuristic becomes PCH because items undergo several sortings which lead to different initial solutions and only the best one is selected.

We assign a score to each item, thus specifying the order in which items are to be considered by the accomodation heuristics. The score definition is problem specific.

Scores are thus first initialized through the *Score Initialization* procedure, and then are dynamically modified by means of the *Score Update* and *Long-term Score Reinitialization* procedures. *Score Update* proceeds through small changes, aiming to adjust the scores used to sort the items at iteration k of UniPack according to the quality of the solution built at iteration $k-1$. *Long-term Score Reinitialization* incorporates long-term decisions, as long-term memory structures, and proceeds through larger score modifications in the scores in order to avoid cycling on the same solutions and explore new regions of the solution space.

Score computation and updates depend upon a number of parameters. We aim to keep this number as low as possible to simplify their adjustment during computation. Indeed, no such adjustment is required for 2D Knapsack and Strip-Packing problems. For the other ones, UniPack provides a problem-specific, dynamically-adjusting parameter procedure denoted *Parameter Update* (see IV-C).

The main steps on UniPack are the following (refer to Figure 7 for a schema of the method):

- Build an initial solution of the packing problem and set the best-solution BS equal to the initial solution;

We use the PCH heuristic;
- Scoring Phase
  - Initialize the score of the items: the *Score Initialization* procedure;
  - While *Stopping Conditions* not encountered, repeat the following steps:
    - Sort the items according to their scores and apply a constructive heuristic to the sorted list, obtaining a new solution CS; We use the EP-BPH procedure;
    - If a given number of successive non-improving iterations is reached, reinitialize the scoring using the *Long-term Score Reinitialization* procedure; otherwise, update the scores using the *Score Update* procedure according to the CS solution;
    - If CS is better than BS, then set BS to CS;
    - The *Parameter Update* procedure then internally adjusts the parameters.

A. EP-*based Constructive Heuristics for Non-Guillotine Orthogonal Higher-Dimensional Packing Problems*

We now present the constructive heuristic PCH and the initial solution procedure EP-BPH we propose for Non-Guillotine Orthogonal Higher-Dimensional Packing problems. The procedures are based on the *Best Fit Decreasing* (BFD) idea and generalize the heuristic presented in Crainic et al. [10].

Following an initial sorting of the items by non-increasing order of their volumes, the BFD constructive heuristic for 1D Bin Packing problem tries to load each item into the best bin. The latter is defined as the bin which, after loading the item, has the maximum free volume, defined as the container volume minus the sum of the volumes of the items it contains. A new container is created whether the item cannot be accommodated into the existing bins. Despite its simplicity, the BFD heuristic offers good performances for 1D Bin Packing problems. Similar heuristics exist for other packing problems, e.g., Knapsack and Strip Packing. Unfortunately, extending these heuristics to a general constructive heuristics for Non-Guillotine Orthogonal Higher-Dimensional Packing problems is a non-trivial task. On the one hand, while in 1D cases the ordering is done considering a unique attribute characterizing both items and bins, i.e., their volume or profit, more choices exist in the multi-dimensional context. One may thus consider sorting items according to their width, height, or depth, as well as, derived from these attributes, according to their volume or the areas of their different faces. Consequently, the definition of the best bin in the BFD heuristic is not unique. On the other hand, while the item accommodation does not need to be considered in 1D problems, a 2D or 3D packing may vary significantly according to how items are placed inside the bin, even when the ordering of the items and the rule selecting the best bin are not changed. Moreover, according to the packing problem, the number of available bins may be unlimited or fixed and all the items or just a subset must be loaded.

We propose a new constructive heuristics based on BFD ideas, denoted *Extreme-Point Best Positioning Heuristic* (EP-BPH), which places the items into containers by using the Extreme Points concept of Crainic et al. [10]. As indicated earlier, the Extreme Points define the points where one may place an item that one wants to add to an existing packing.

The main steps of the algorithm are as follows:
- Order the items according to a sorting criterion;
- For each item in the resulting sequence, find the best *EP* of the best available bin where to load the item;
- If such a bin exists, load the item into it on the given *EP*;
- If the item cannot be loaded in any existing bin, a new bin is created if the total number of bins does not exceed the given maximum, otherwise the item is discarded.

Changing the maximum number of available bins adapts EP-BPH to different packing problems. For example, the number of bins is infinite for the Bin Packing problem, but it is equal to 1 for the 3BKP problem. The behavior of EP-BPH depends on how the best *EP* is selected and how the items are sorted. Computational experiments have shown that, from the *EP* selection point of view, the best trade off between solution quality and computational results is given by the *Residual Space* rule (see Crainic et al. [10]).

The *Residual Space* (*RS*) measures the free space available around an *EP*. Roughly speaking, the *RS* of an *EP* is the distance, along each axis, from the bin edge or the nearest item. The nearest item can be different on each axis. More precisely, when an *EP* is created, its *Residual Space* on each axis is set equal to the distance from its position to the side of the bin along that axis (Figure 8a). The algorithm puts an item on the *EP* that minimizes the difference between its *RS* and the item size:

$$f = [(RS_e^x - w_j) + (RS_e^y - d_j) + (RS_e^z - h_j)], \quad (22)$$

where RS_e^x, RS_e^y, and RS_e^z are the *RS*s of *EP* e on X, Y, and Z axes, respectively. Every time an item is added to

the packing, the *RS*s of all the *EP*s are updated. Figure 8b illustrates the concept. For "complex" packings, the *RS* gives only an estimate of the effective volume available around the *EP*s and, thus, potential overlaps with other items have to be verified when accommodating a new item on the chosen *EP*. See Crainic et al. [10] for further details.

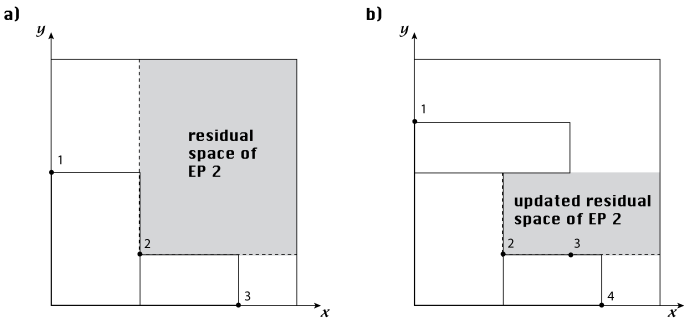

Fig. 8. Example of *Residual Space*

To build an initial solution, we apply EP-BPH using a number of sorting criteria. The resulting PCH heuristic builds an initial solution by iteratively applying the sorting criteria and selecting the best.

Items may have several attributes, but from the sorting algorithm perspective, the most important ones are:

1) *profit*: the worth or priority of an item;
2) *specific weight*;
3) *area*: for three-dimensional problems it must be meant as the item projection on the (X, Y) plane (see Figure 9).

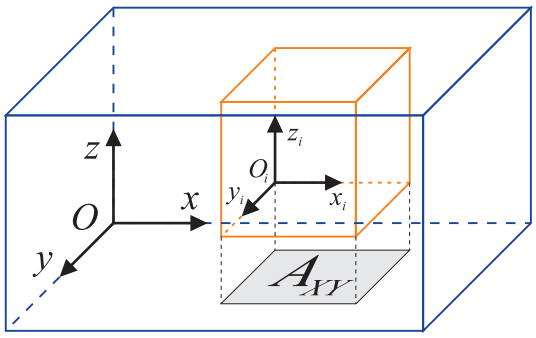

Fig. 9. Definition of the area of an item

Since items show more than one attribute, many ways to sort them are possible. Giving more importance to an attribute means to favor those items showing the highest values of that attribute or score. Often sorting procedure involves more than an attribute or more than a score. Sometimes items are sorted after they have been grouped into *clusters*. A cluster is a set of items showing "close" values of a particular attribute or score. By "close" we mean that the values are inside a given set. Suppose, for instance, to sort the items by clustered area (see Figure 9), let $A_{\min}$, $A_{\max}$ be the extreme values of the area interval that we want to cluster. Each cluster will have a length which is the length of the global interval $A_{\max} - A_{\min}$ times a given percentage $\theta/100$, with $\theta \in [1, 100]$. The number of clusters n_c is the ration of the overall interval length over the length of a single cluster. This ratio is $n_c = \lceil 100/\theta \rceil$. Each cluster $A_i(\theta)$ can then be expressed as:

$$A_i(\theta) = [A_{\min} + (i-1)(A_{\max} - A_{\min})\theta/100, \\ A_{\min} + i(A_{\max} - A_{\min})\theta/100], \quad (23)$$

with $i = 1, \ldots, n_c$. Note that, if we want to cluster the overall container (basis) area, then $A_{\min} = 0$ and $A_{\max} = W \times D$ and (23) becomes:

$$A_i(\theta) = [(i-1)WD\theta/100, \quad iWD\theta/100], \quad (24)$$

with $i = 1, \ldots, n_c$. By combining the three items attributes, six different sortings can be performed:

1) *a-sw*: clustered area, sorted specific weight;
2) *a-p*: clustered area, sorted profit;
3) *sw-a*: clustered specific weight, sorted area;
4) *sw-p*: clustered specific weight, sorted profit;
5) *p-sw*: clustered profit, sorted specific weight;
6) *p-a*: clustered profit, sorted area.

When a solution has been calculated, its corresponding objective function value is given by the following merit function:

$$F = P - \alpha U. \quad (25)$$

Note that (25) is a Lagrangean relaxation of the sum of the selected items profits. This means that, according to α, attention is also devoted to the balancing constraints, even before the center of mass optimization procedure. For values of α see Section IV-C.

B. Center of mass optimization

Given a three-dimensional convex domain inside the container, the balancing procedure tries to adjust packed items positions so that the global center of mass lies inside the domain. The heuristic just moves already packed items, therefore no items are added or removed by the container, nor the overall profit is modified by the procedure. The center of mass optimization heuristic works as follows: it first calculates the position $\vec{x}_{CM}$ of packed items center of mass as reported in equations (2) and (3), then it moves one item after another so that $\vec{x}_{CM}$

moves towards the desired position. Two issues arise: where to move an item and how to avoid it overlapping other items and the edges of the bin.

The first problem may be tackled starting from equations (2) and (3). Assuming k is the number of accommodated items, $M = \sum_{j=1}^{k} m_j$ is the overall mass of the items, and we want to move item i from its actual position $\vec{x}_i = (x_i, y_i, z_i)$ to an unknown new position $\vec{x}'_i = (x'_i, y'_i, z'_i)$ so that the overall center of mass moves from the actual position $\vec{x}_{CM}$ to the new desired position $\vec{x}'_{CM}$ in order to meet balancing conditions. By (3) the actual center of mass can be written as:

$$\vec{x}_{CM} = \sum_{j \neq i} m_j \vec{x}_{CM_i}/M + m_i \vec{x}_{CM_i}/M \qquad (26)$$

When item i moves from $\vec{x}_i$ to $\vec{x}'_i$ then its new center of mass becomes $\vec{x}'_{CM_i}$, while the overall center of mass is:

$$\vec{x}'_{CM} = \sum_{j \neq i} m_j \vec{x}_{CM_i}/M + m_i \vec{x}'_{CM_i}/M. \qquad (27)$$

Subtracting (26) from (27) we have:

$$\vec{x}'_{CM} - \vec{x}_{CM} = m_i(\vec{x}'_{CM_i}/M - m_i \vec{x}_{CM_i})/M, \qquad (28)$$

which lead to the new coordinates of item i center of mass:

$$\vec{x}'_{CM_i} = \vec{x}_{CM_i} + (\vec{x}'_{CM} - \vec{x}_{CM})M/m_i. \qquad (29)$$

Finally, the new coordinates of item i can be found by inverting equation (2); in particular:

$$\vec{x}'_i = (x'_{CM_i} - w_i/2,\ y'_{CM_i} - d_i/2,\ z'_{CM_i} - h_i/2) \qquad (30)$$

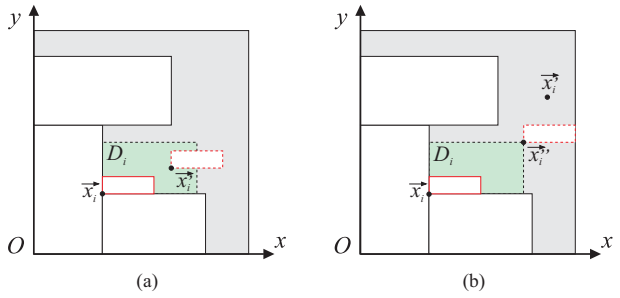

Fig. 10. Example of Permitted Movements of an Item in 2D

Unfortunately, due to overlapping issues, it is not always possible to move item i to $\vec{x}'_i$. Here comes the second problem: the definition of a *three-dimensional convex connected domain* where item i can freely move without having to overlap other items nor the bin. Actually, we define such a domain D_i as the set of allowed positions for the the origin of item i. To do so, an algorithm similar to the one used to calculate the *RS* of an *EP* is used. Once D_i has been defined, three possible scenarios may take place:

- $D_i = \{\varnothing\}$: item i cannot move;
- $\vec{x}'_i \in D_i$: item i moves to $\vec{x}'_i$ thus letting the balancing to be achieved (see Figure 10a for a two-dimensional example);
- $\vec{x}'_i \notin D_i$ and $D_i \neq \{\varnothing\}$: item i moves to an intermediate position $\vec{x}''_i$ defined as the point which better approximates $\vec{x}'_i$ on each axis (see Figure 10b for a two-dimensional example).

Items movements may lead to a state that does not take gravity effects into account. That would result in faulty solutions for many real-life applications, so the algorithm simulates the force of gravity by compacting all items along the Z axis towards the (X, Y) plan.

The heuristic stops when the packing is balanced, when no items can move, or after a given number of iterations.

C. Unipack specialization

In this Subsection we show how to adapt UniPack parameters and scores in order to deal with the 3BKP.

1) Score Initialization: The idea is to use the score as a measure of the willingness to accommodate an item in the knapsack. Consequently, we start from the initial solution decisions, and prioritize the items selected by the accommodation procedure by assigning them a higher score than to the non loaded ones. Two criteria are used to define such initial scores. First, the score should reflect the profit associated to each item. Second, the gap between a loaded and a non loaded item should be small enough to guarantee the possibility of changes in the ordered list. The initial score of an item is then set to $s_i = kp_i$ if the item has been loaded in the initial solution, and to $s_i = p_i$ otherwise. The value of k has been experimentally set to 3.

2) Score Update: Previous experience has shown that the various sorting criteria used by the procedure building the initial solution load into the knapsack a significant subset of the items making up the optimal solution. "Mistakes" usually are caused when selecting among items with similar profits, but with peculiar sizes, resulting in an underutilization of the knapsack. The knowledge given by the sorting criteria and the profits being already taken into account by the Score Initialization procedure, the update of the scores thus focuses on a special subset of items: the less profitable

items already loaded and the most profitable non loaded ones. The goal is to force at each iteration swaps between less profitable loaded and profitable non loaded items by changing the scores as follows:

- Find the item k loaded during the last iteration, minimizing $\mu_i = (1 + f_i^l)p_i/(w_i d_i)$, where f_i^l represents the number of iterations item i has been loaded into the container;
- Update the score of item k to $s_k = (1 - \alpha)s_k$;
- Find the item l non loaded during the last iteration, maximizing $\mu_i = p_i/(w_i d_i(1 + f_i^u))$, where f_i^u represents the number of iterations item i has not been loaded into the container;
- Update the score of item l to $s_l = (1 + \beta)s_l$;
- Swap the scores of items k and l;
- Keep the score unchanged for all items $i, i \neq k$ and $i \neq l$;

where, μ_i measures the willingness to accommodate an item into the knapsack, f_i^l and f_i^u maintain a long-term memory of the selected items to avoid always selecting from the same subset of items, and α and β represent the percentage score decrease and increase, respectively, and are experimentally set to 0.1. This procedure ensures that at least two items are swapped at each iteration.

3) Long-term Score Reinitialization: Given the sorted list of items which built the best solution found so far, we first give a score to each item according to the same rule used in Score Initialization. A fixed number of item pairs are then randomly selected and their scores are swapped.

4) Initialization of Parameters and Stopping Criteria:

- $\alpha = \beta = 0.1$;
- Long-term Score Reinitialization every 1000 iterations;
- number of item-pairs: 5% of the items.
- α. If the best solution is unfeasible, $\alpha = 2 \times \alpha$. If the best solution is feasible, $\alpha = \alpha/2$ remains unchanged if the center of mass lies in the central half of its feasibility domain, while is unchanged otherwise.

The overall process stops after 5 seconds.

V. Computational results

In this section, we analyze the behaviour of the model and the heuristics in term of solution quality and computational efficiency. While the standard 3KP is known in the literature, the 3BKP is introduced in this paper for the first time. Thus, in Subsection V-A we define some benchmark instances. The first two sets, namely Set1 and Set2, are obtained by extending the instances for 3KP literature, while the third one, Set3, extends the rules used in the previous sets in order to diversify the instances. All the tests have been performed on a Intel I7 2.8 GhZ Workstation with 4 Gb of Ram. The model has been solved by means of Gurobi 4.0 solver limited to 1 core [20]. Subsection V-B is devoted to compare the computational results of the MIP model and the heuristic, while subsection V-C shows the behaviour of the developed model and heuristic compared with state-of-the-art algorithms. Being 3BKP firstly introduced in this paper, we compare the model and the heuristic with the results of heuristics developed specifically for the problem which is more similar to 3BKP, the Three-Dimensional Knapsack Problem.

A. Test Instances

In this section we introduce different instance sets for 3BKP. Following the test for the 3KP, the instances cover up to items and different types of item, knapsack and weight distributions. The sets, namely Set1 and Set2, are obtained by extending the instances by Egeblad and Pisinger [13]. All instance sets can be downloaded from the web site of OR-Library [4]. In the following we give a detailed description of the different set parameters. In these two sets of instances, the size of the knapsack, as well as the sizes of the items are the same that the ones in [13], while weights are added as additional items' attribute. Thus, the two sets differ for the weight generation, i.e., the weights in Set1 are generated in a smaller interval than in Set2. In order to give a better of the instances, in the following we report the full list of the parameters used to generate the instances.

- *number of items*: $n \in \{20, 40, 60\}$;
- *items' generation strategy*: $t \in \{C, R\}$, where:
  - C alias *clustered*, because instance consists of only 20 items which are duplicated appropriately;
  - R alias *random*, because instance consists of independently generated items;
- *bin size*: $p \in \{50, 90\}$, expressed in percentage of the total volume of the items.
- the items' attributes; assuming $i = 1, ..., n$ is the i^{th} item, then i is identified by:
  - *size*: $s_i = (w_i, d_i, h_i)$, which must belong to one among the following *geometric classes*:
    - Cubes (C). The items are cubic and their sizes are defined as $w_i \in [1, 100]$, $d_i = w_i$, $h_i = w_i$;
    - Diverse (D). The sizes of the items are randomly chosen in the following ranges $w_i \in [1, 50]$, $d_i \in [1, 50]$, $h_i \in [1, 50]$;

  - ∗ Long (L). The sizes of the items are randomly chosen in the following ranges $w_i \in [1,\ 200/3]$, $d_i \in [50,\ 100]$, $h_i \in [1,\ 200/3]$;
  - ∗ Uniform (U). The sizes of the items are randomly chosen in the following ranges $w_i \in [50,\ 100]$, $d_i \in [50,\ 100]$, $h_i \in [50,\ 100]$.
- *profit*: $p_i = 200 + w_i\, d_i\, h_i$;
- Center of mass position: the center of mass of each item is placed int he geometrical center of the item itself, i.e. $CM_i = \{w_i/2, d_i/2, h_i/2\}$
- *specific weight*: sw_i, uniformly distributed on the interval I_{sw}, where the limits of the interval depend on the set:
  - ∗ Set1: $I_{sw} = [70,\ 100]$;
  - ∗ Set2: $I_{sw} = [10,\ 1000]$;
- *CoM domain*: the domain constraints are fixed as $L^\delta = \{W/4, D/4, 0\}$ and $U^\delta = \{3W/4, 3D/4, H/2\}$. These limits are given by practical issues in maritime and air cargo applications. In particular, for the limits on Z, for stability reasons the requirement is usually to be as near a s possible to 0, i.e., the bottom of the container [23].

The combination of all the values give 120 instances, 60 for each set.

B. Model and Heuristic results

C. State-of-the-Art results

As stated in Section III, 3BKP is introduced in this paper for the first time. Thus, no other method than our model and heuristic is present in the literature. Moreover, computing specific upper bounds for 3BKP is quite difficult. In fact, upper bounds obtained by model 3BKP-M are quite poor and have mainly the same quality a trivial bound obtained by computing the optimal solution of the mono-dimensional Knapsack Problem where the size of the knapsack is the volume of the 3D one and the size of the items is the volume of the 3D items, in the following referred as 1DB, [14], [13]. Moreover, additional upper bounds that can be obtained by means of conservative scales in the 3D packing without rotation are not valid for the problems where the rotations are allowed [13]. On the other hand, 3BKP is an extension of the standard 3KP and thus the solutions obtained by 3BKP-M and 3BKP-U as valid for 3KP. Thus, in Table I we compare 3BKP-M and 3BKP-U with the results obtained by H_{EP}, heuristic by Egeblad and Pisinger on their instances for 3KP. The computational times have been fixed to 120 seconds for H_{EP}, 200 seconds for 3BKP-M and 5 seconds for 3BKP-U. 3BKP-M is solved by means of Gurobi 4.0 [20], while 3BKP-U is implemented in C++. For H_{EP} the results have been given by Egeblad and Pisinger.

The meaning of the columns is the following:

- Columns 1-4. The columns give the instance name defined in [13], the number of items, the item size geometry and the item generation type.
- Column 5. The objective function of the upper bound $1D$.
- Columns 6-9. The objective function of the best solution found by H_{EP}, the Model 3BKP-M, our heuristic 3BKP-U with and without the balancing constraints activated.
- Columns 10-14. The percentage gap between the upper bound upper bound $1D$ and objective function of the best solution found by H_{EP}, the Model 3BKP-M, our heuristic 3BKP-U with and without the balancing constraints activated. In the case of 3BKP-U with balancing constraints, we consider the weights of Set1.

The computational times are not reported, being fixed for each method to 120 seconds for H_{EP}, 200 seconds for 3BKP-M and 5 seconds for 3BKP-U. From the results we can notice how the model is not competitive, with a gap almost doubled than H_{EP}. However, the model is much more flexible than the heuristics, making possible to easily introduce additional constraints like fixed positions for the items, forbidden rotations and precedence constraints in items loading. Moreover, giving to the model a time limit equal to 1000 seconds, the gap can be reduced, even if it is still about 10% more than H_{EP}. If we compare H_{EP} with 3BKP-U with the balancing constraints activated, we can notice how the results of 3BKP-U are about 3% worst than H_{EP}. However, this gap is given by the balancing constraints. In fact, if we remove the balancing constraints we obtain a mean gap of 16%, which is about 2% less than Egeblad and Pisinger results. These results are more impressive if we consider that 3BKP-U require a computational time which is about 2 order of magnitude less than H_{EP}. We also tried to increase the computational time of 3BKP-U in order to obtain better results, but the computational experience show that the increase of efficacy is negligible.

VI. Conclusions

In this paper, we introduced the Three-Dimensional Knapsack Problem with Balancing Constraints Problem, the extension of the standard Three-Dimensional Knapsack Problem Problem (3KP) where additional constraints related to the Center of Mass of the three-dimensional packing are given. A MIP formulation of

Instance	n	IT Geom	IT Gen	1D	H_{EP}	Model	3BKP-U	3BKP-U UNB	H_{EP}	Model	3BKP-U UNB	3BKP-U
ep3d-20-C-C-50	20	C	C	1026348	633672	633672	633672	633672	7.0	7.0	7.0	7.0
ep3d-20-C-C-90	20	C	C	1834340	916241	916241	916241	916241	0.0	0.0	0.0	0.0
ep3d-20-C-R-50	20	C	R	2188245	1492413	1492413	1492413	1492413	0.0	0.0	0.0	0.0
ep3d-20-C-R-90	20	C	R	3925057	2497691	2497691	2449089	2497691	0.0	0.0	1.9	0.0
ep3d-20-D-C-50	20	D	C	395916	239532	316492	315964	315964	24.3	0.0	0.2	0.2
ep3d-20-D-C-90	20	D	C	718692	468112	559756	526480	526480	34.9	22.1	26.7	26.7
ep3d-20-D-R-50	20	D	R	240621	195937	206026	214227	214227	18.6	14.4	11.0	11.0
ep3d-20-D-R-90	20	D	R	414188	318848	368476	335123	335123	23.0	11.0	19.1	19.1
ep3d-20-F-C-50	20	F	C	2395087	1900250	1900250	1900250	1900250	20.7	20.7	20.7	20.7
ep3d-20-F-C-90	20	F	C	4304020	2989393	2510887	3118132	3118132	30.5	41.7	27.6	27.6
ep3d-20-F-R-50	20	F	R	2252037	1563997	1698795	1800399	1800399	30.5	24.6	20.1	20.1
ep3d-20-F-R-90	20	F	R	4099982	2918002	2353072	3232580	3232580	28.8	42.6	21.2	21.2
ep3d-20-L-C-50	20	L	C	1064487	834335	941069	891283	891283	21.6	11.6	16.3	16.3
ep3d-20-L-C-90	20	L	C	1894489	1589303	1546821	1619190	1619190	16.1	18.4	14.5	14.5
ep3d-20-L-R-50	20	L	R	718561	569900	633463	626381	626381	20.7	11.8	12.8	12.8
ep3d-20-L-R-90	20	L	R	1282710	1051084	1114602	1056699	1056699	17.9	12.9	17.5	17.5
ep3d-20-U-C-50	20	U	C	4495440	3088676	2796072	3127252	3127252	31.3	37.8	30.4	30.4
ep3d-20-U-C-90	20	U	C	8067424	5360280	4289980	6074000	6074000	33.5	46.7	24.6	24.6
ep3d-20-U-R-50	20	U	R	4413077	3509748	2677316	3087569	3509748	20.5	39.3	30.0	20.5
ep3d-20-U-R-90	20	U	R	8041072	6921250	4156121	6638762	6921250	13.9	48.3	17.4	13.9
ep3d-40-C-C-50	40	C	C	2065540	1265664	1265664	1265664	1265664	38.7	38.7	38.7	38.7
ep3d-40-C-C-90	40	C	C	3652448	2828160	2335561	2717385	2828160	21.8	35.4	24.8	21.8
ep3d-40-C-R-50	40	C	R	4102972	3002269	2503936	3008658	3008658	26.8	39.0	26.7	26.7
ep3d-40-C-R-90	40	C	R	7335602	5972946	3498247	4900577	5972946	4.1	43.8	21.3	4.1
ep3d-40-D-C-50	40	D	C	788124	539040	580512	630996	630996	31.6	26.3	19.9	19.9
ep3d-40-D-C-90	40	D	C	1423896	1126300	1032016	1124788	1126300	20.9	27.5	21.0	20.9
ep3d-40-D-R-50	40	D	R	399894	349470	248518	349152	349470	12.6	37.9	12.7	12.6
ep3d-40-D-R-90	40	D	R	728248	639819	510962	612487	639819	12.1	29.8	15.9	12.1
ep3d-40-F-C-50	40	F	C	4816926	3590244	2596874	3274502	3590244	25.5	46.1	32.0	25.5
ep3d-40-F-C-90	40	F	C	8664122	6435962	4039655	5760960	6435962	25.7	53.4	33.5	25.7
ep3d-40-F-R-50	40	F	R	4518343	3477469	2623783	3644680	3644680	23.0	41.9	19.3	19.3
ep3d-40-F-R-90	40	F	R	8199224	7336067	3560051	6386094	7336067	10.5	56.6	22.1	10.5
ep3d-40-L-C-50	40	L	C	2127316	1675122	1410197	1760700	1760700	21.3	33.7	17.2	17.2
ep3d-40-L-C-90	40	L	C	3819412	2943657	2054950	3032364	3032364	22.9	46.2	20.6	20.6
ep3d-40-L-R-50	40	L	R	1784686	1609648	1067546	1567893	1609648	9.8	40.2	12.1	9.8
ep3d-40-L-R-90	40	L	R	3224295	2699629	1722617	2689260	2699629	16.3	46.6	16.6	16.3
ep3d-40-U-C-50	40	U	C	8988536	7008136	4317064	7355808	7355808	22.0	52.0	18.2	18.2
ep3d-40-U-C-90	40	U	C	16241380	14065676	5580692	10819676	14065676	13.4	65.6	33.4	13.4
ep3d-40-U-R-50	40	U	R	8666294	7766238	4418573	7538465	7766238	10.4	49.0	13.0	10.4
ep3d-40-U-R-90	40	U	R	15531980	13077284	6217878	11120608	13077284	15.8	60.0	28.4	15.8
ep3d-60-C-C-50	60	C	C	3063219	1504980	1370916	1504980	1504980	50.9	55.2	50.9	50.9
ep3d-60-C-C-90	60	C	C	5517671	4475024	2590702	3892171	4475024	18.9	53.0	29.5	18.9
ep3d-60-C-R-50	60	C	R	6493464	5695120	2916398	4435949	5695120	12.3	55.1	31.7	12.3
ep3d-60-C-R-90	60	C	R	11675188	10209801	4213641	8729652	10209801	12.5	63.9	25.2	12.5
ep3d-60-D-C-50	60	D	C	1200408	1057032	801200	954856	1057032	11.9	33.3	20.5	11.9
ep3d-60-D-C-90	60	D	C	2143544	1843584	1440492	1488020	1843584	14.0	32.8	30.6	14.0
ep3d-60-D-R-50	60	D	R	538113	484363	323947	502275	502275	10.0	39.8	6.7	6.7
ep3d-60-D-R-90	60	D	R	966582	861655	433736	848299	861655	10.9	55.1	12.2	10.9
ep3d-60-F-C-50	60	F	C	7193700	6257697	3700025	5565875	6257697	13.0	48.6	22.6	13.0
ep3d-60-F-C-90	60	F	C	12913715	10412682	3714761	8298024	10412682	19.4	71.2	35.7	19.4
ep3d-60-F-R-50	60	F	R	6780100	6146420	3193484	5484876	6146420	9.3	52.9	19.1	9.3
ep3d-60-F-R-90	60	F	R	12301636	10866326	4154808	8884312	10866326	11.7	66.2	27.8	11.7
ep3d-60-L-C-50	60	L	C	3211612	2327139	1708786	2656622	2656622	27.5	46.8	17.3	17.3
ep3d-60-L-C-90	60	L	C	5736894	4832080	2773026	4422760	4832080	15.8	51.7	22.9	15.8
ep3d-60-L-R-50	60	L	R	2391507	2042317	1157278	2158105	2158105	14.6	51.6	9.8	9.8
ep3d-60-L-R-90	60	L	R	4304649	3872594	1748761	3568203	3872594	10.0	59.4	17.1	10.0
ep3d-60-U-C-50	60	U	C	13508800	12033592	4939888	10782744	12033592	10.9	63.4	20.2	10.9
ep3d-60-U-C-90	60	U	C	24342664	19787768	3868824	14683564	19787768	18.7	84.1	39.7	18.7
ep3d-60-U-R-50	60	U	R	12097660	10857656	5207450	9952696	10857656	10.2	57.0	17.7	10.2
ep3d-60-U-R-90	60	U	R	21893096	19304585	4374061	14216597	19304585	11.8	80.0	35.1	11.8
Mean									**18.2**	**40.0**	**21.0**	**16.0**

TABLE I

3KP: RESULTS OF 3BKP-M AND 3BKP-U WITHOUT BALANCING CONSTRAINTS COMPARED TO H_{EP}.

the problem as well as an efficient and accurate heuristic method are presented. Extensive computational results showed how the MIP model is able to find better bounds than other relaxations and the heuristic method is able to efficiently solve both instances explicitly designed for 3BKP, as well as to be competitive with methods explicitly designed for solving 3KP. Presently, we are extending the test instances in order to give a better insight of the relationship between solution quality and balancing constraints tightness.

ACKNOWLEDGMENTS

We express our gratitude to Prof. David Pisinger, who graciously provided us with sets of test instances, as well as their detailed results. Moreover, the authors are grateful to Roberto Marcellino for his contribution to a previous version of the paper. This project has been partially supported by the Ministero dell'Istruzione, Università e Ricerca (MIUR) (Italian Ministry of University and Research), under the Progetto di Ricerca di Interesse Nazionale (PRIN) 2007 "Optimization of Distribution Logistics", and the Natural Sciences and Engineering Council of Canada (NSERC), through its Industrial Research Chair and Discovery Grants programs, and by the partners of the Chair, CN, Rona, Alimentation Couche-Tard and the Ministry of Transportation of Québec.

REFERENCES

[1] R. Baldacci and M. A. Boschetti, "A cutting plane approach for the two-dimensional orthogonal non-guillotine cutting problem," *European Journal of Operational Research*, vol. doi: 10.1016/j.ejor.2005.11.060, 2007.

[2] J. E. Beasley, "Algorithms for two-dimensional unconstrained guillotine cutting," *Journal of the Operational Research Society*, vol. 36, p. 297306, 1985.

[3] ——, "An exact two-dimensional non-guillotine cutting stock tree search procedure," *Operations Research*, vol. 33, pp. 49–64, 1985.

[4] ——, "Or-library: distributing test problems by electronic mail," *Journal of the Operational Research Society*, vol. 41, pp. 1069–1072, 1990. [Online]. Available: http://people.brunel.ac.uk/ mastjjb/jeb/info.html

[5] J. O. Berkey and P. Y. Wang, "Two dimensional finite bin packing algorithms," *Journal of the Operational Research Society*, vol. 38, pp. 423–429, 1987.

[6] E. E. Bischoff and M. S. W. Ratcliff, "Issues in the development of approaches to container loading," *Omega*, vol. 23, pp. 377–390, 1995.

[7] M. A. Boschetti, E. Hadjiconstantinou, and A. Mingozzi, "New upper bounds for the two-dimensional orthogonal cutting stock problem," *IMA Journal of Management Mathematics*, vol. 13, p. 95119, 2002.

[8] F. K. R. Chung, M. R. Garey, and D. S. Johnson, "On packing two-dimensional bins," *SIAM - Journal of Algebraic and Discrete Methods*, vol. 3, pp. 66–76, 1982.

[9] L. Colaneri, F. D. Croce, G. Perboli, and R. Tadei, *A Heuristic Procedure for the Automated Cargo Vehicle Loading Problem: A Case Study*. Kluwer, 2003, pp. 27–42.

[10] T. G. Crainic, G. Perboli, and R. Tadei, "Extreme point-based heuristics for three-dimensional bin packing," *INFORMS Journal on Computing*, vol. 20, pp. 368–384, 2008.

[11] E. den Boef, J. Korst, S. Martello, D. Pisinger, and D. Vigo, "Erratum to "the three-dimensional bin packing problem": Robot-packable and orthogonal variants of packing problems," *Operations Research*, vol. 53, no. 4, pp. 735–736, 2005.

[12] H. Dyckhoff, "A typology of cutting and packing problems," *European Journal of Operational Research*, vol. 44, pp. 145–159, 1990.

[13] J. Egeblad and D. Pisinger, "Heuristic approaches for the two- and three-dimensional knapsack packing problem," *Computers & Operations Research*, vol. 36, pp. 1026–1049, 2009.

[14] G. Fasano, "A mip approach for some practical packing problems: Balancing constraints and tetris-like items," *4OR: A Quarterly Journal of Operations Research*, vol. 2, no. 2, pp. 161–174, 2004.

[15] S. P. Fekete and J. Schepers, "A new exact algorithm for general orthogonal d-dimensional knapsack problems," *ESA '97, Springer Lecture Notes in Computer Science*, vol. 1284, pp. 144–156, 1997.

[16] ——, "On more-dimensional packing iii: exact algorithms," *Discrete Applied Mathematics*, 1997, submitted for publication.

[17] ——, "New classes of lower bounds for bin packing problems," *Math. Programming*, vol. 91, no. 1, pp. 11–31, 2001.

[18] J. A. George and D. F. Robinson, "A heuristic for packing boxes into a container," *Computers & Operations Research*, vol. 7, pp. 147–156, 1980.

[19] P. C. Gilmore and R. E. Gomory, "European journal of operational research," *Operations Research*, vol. 13, pp. 94–119, 1965.

[20] Gurobi Optimization Inc., *Gurobi Optimizer Reference Manual Version 4.0*. Houston: Gurobi Optimization, 2011.

[21] E. Hadjiconstantinou and N. Christofides, "An exact algorithm for general and orthogonal and two-dimensional knapsack problems," *European Journal of Operational Research*, vol. 83, pp. 39–56, 1995.

[22] L. Junqueira, R. Morabito, and D. S. Yamashita, "Three-dimensional container loading models with cargo stability and load bearing constraints," *Computers & Operations Research*, 2010.

[23] B. L. Kaluzny and R. H. A. D. Shaw, "Optimal aircraft load balancing," *International Transactions in Operational Research*, vol. 16, pp. 767–787, 2009.

[24] S. Martello, D. Pisinger, and D. Vigo, "The three-dimensional bin packing problem," *Operations Research*, vol. 48, no. 2, pp. 256–267, 2000.

[25] S. Martello, D. Pisinger, D. Vigo, E. den Boef, and J. Korst, "Algorithms for general and robot-packable variants of the three-dimensional bin packing problem," *ACM Transactions on Mathematical Software*, vol. 33, pp. 7–19, 2007.

[26] M. Mongeau and C. Bés, "Optimization of aircraft container loading," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, pp. 140–150, 2003.

[27] G. Perboli, "Bounds and heuristics for the packing problems," Ph.D. dissertation, Politecnico di Torino, Torino, Italy, available at http://www.orgroup.polito.it/People/perboli/phd-thesys.pdf, 2002.

[28] R. T. T. G. Crainic, G. Perboli, "Unipack: a new heuristic framework for multi-dimensional packing problems," in *Proceedings of MIC 2007 The Seventh Metaheuristics International Conference, MIC 2007*, 2007.

[29] G. Wäscher, H. Hauβner, and H. Schumann, "An improved typology of cutting and packing problems," *European Journal of Operational Research*, vol. 183, pp. 1109–1130, 2007.

[30] H. P. Williams, *Model Building in Mathematical Programming*. John Wiley & Sons, London, 1993.