Solving the three-dimensional open-dimension rectangular packing problem: a constraint programming model

Mateus Martin[a,*], Thiago Alves de Queiroz[b], Reinaldo Morabito[c]

[a] *Departamento de Engenharia de Produção, Universidade Federal Fluminense, Rua Domingos Silvério 135, 25650-050, Petrópolis-RJ, Brazil*

[b] *Instituto de Matemática e Tecnologia, Universidade Federal de Catalão, Av. Dr. Lamartine P. Avelar 1120, 75704-020, Catalão-GO, Brazil*

[c] *Departamento de Engenharia de Produção, Universidade Federal de São Carlos, Via Washington Luiz km 235, 13565-905, São Carlos-SP, Brazil*

Abstract

In this paper, we address the three-dimensional open-dimension rectangular packing problem (3D-ODRPP). This problem addresses a set of rectangular boxes of given dimensions and a rectangular container of open dimensions. The objective is to pack all boxes orthogonally into the container while minimizing the container volume. Real-world applications of the 3D-ODRPP arise in production systems with operations of shipping or moving. The literature has presented mainly mixed-integer programming (MIP) formulations and their linearization techniques for the problem allied with general-purpose optimization solvers. To model and solve the 3D-ODRPP, we propose a constraint programming model based on a position-free modeling approach with logic operators. We ran computational experiments to assess the performance of the proposed model compared to the benchmark MIP models from instances of the literature. The results show our approach is competitive in different sets of problem instances in terms of reaching optimality as well as providing satisfactory feasible solutions quickly.

Keywords: Cutting and Packing, Three-dimensional rectangular packing, Open-dimension problems, Constraint Programming

1. Introduction

The three-dimensional open-dimension rectangular packing problem (3D-ODRPP) addresses a set $I = \{1, \ldots, n\}$ of rectangular boxes of fixed dimensions and a single rectangular container of open dimensions. Each box $i \in I$ has a length l_i, width w_i, and height h_i, while the variables X, Y, and Z represent the container's dimensions along the Cartesian axes, i.e., the x-axis, y-axis, and z-axis, respectively. The objective is to pack all the boxes into the container while minimizing the volume XYZ of the container. The boxes may be freely rotated by 90 degrees if their walls are parallel to the container walls. Real-world applications of the problem are related to the design of shipping containers of minimal volume, which is common in the manufacturing, shipping, and warehouse industries (Junqueira & Morabito, 2017). For instance, Tsai et al. (2015) presented practical packing scenarios of electronic equipment for transportation to customers or vendors. According to the typology of Wäscher et al. (2007) for cutting and packing problems, the 3D-ODRPP can be categorized as an Open-Dimension Problem with the two geometric conditions, that is, no overlap between any pair of boxes (non-overlapping) and the boxes must be contained within the container (containment).

*Corresponding author.

Email addresses: `mpmartin@id.uff.br` (Mateus Martin), `taq@ufcat.edu.br` (Thiago Alves de Queiroz), `morabito@dep.ufscar.br` (Reinaldo Morabito)

Table 1 presents an illustrative problem instance of the 3D-ODRPP with $n = 4$ boxes. A solution for this problem instance is shown in Fig. 1. Notice that the 3D-ODRPP arises in a planning phase that usually neglects practical requirements for packing problems such as stability and cargo weight. However, if necessary, during the execution phase of packing the boxes into the container, one can use tools such as bubble wrap or polystyrene to guarantee cargo conservation.

Table 1: Illustrative example for the 3D-ODRPP with $n = 4$ boxes.

Data	Items			
	1	2	3	4
l_i	30	24	6	20
w_i	12	15	10	8
h_i	10	10	20	10

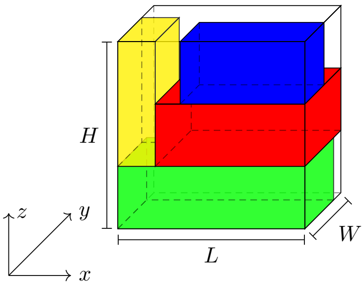


Figure 1: A solution for the example presented in Table 1 with $L = l_1 = 30$, $W = w_2 = 15$, and $H = h_1 + h_3 = 30$.

The literature of the 3D-ODRPP is relatively recent and small. The problem has been mainly modeled by mixed-integer programming (MIP) formulations and solved by their corresponding approximated or equivalent linear versions, all solved with general-purpose optimization solvers. There are two main types of MIP modeling approaches: the position-free and grid-position approaches. Tsai et al. (2015) seem to be the first to propose a position-free model for the 3D-ODRPP, considering the formulation proposed in Chen et al. (1995) for the container loading problem. They approximated the nonlinear objective function (i.e., XYZ) using logarithmic functions and piecewise linear techniques. From that, Lin et al. (2017) and Huang & Hwang (2018) derived improved position-free models by reducing the number of redundant binary variables and/or enhancing the linearization technique. For instance, Huang & Hwang (2018) proposed a linear equivalent set of expressions to the nonlinear objective function based on an improved special ordered set of type 1 (SOS1) formulation. As far as the grid-position approach is concerned, Junqueira & Morabito (2017) proposed a model for the 3D-ODRPP, considering the formulation proposed in Beasley (1985) for a two-dimensional cutting problem. They also extended this formulation to address stability requirements. This model considers the boxes to be aggregated into box types when they share identical sizes.

For recent approaches focusing on related problems and/or heuristic approaches for the 3D-ODRPP, the reader is referred to the works of Mundim et al. (2017), who tackled a two-dimensional problem, aiming to minimize the total area when packing irregularly shaped items; Truong et al. (2020a), who studied the three-dimensional problem, presenting a mixed integer programming model that adopts the full base support constraint to guarantee vertical stability; Truong et al. (2020b), who extended the model in Truong et al. (2020a) also to consider partial base support of items; Vieira et al. (2021), who address a shoe packing problem that involves packing shoes into boxes and loading these boxes into three-dimensional open-dimension containers, proposing integer programming models; and Truong et al. (2021), who talked the three-dimensional problem, developing a genetic algorithm that determines the packing sequence and the orientation of items, while using a greedy heuristic to determine the placement of items. In a recent paper, Iori et al. (2021) proposed an extensive review of cutting and packing problems, focusing mainly on orthogonal cutting and packing problems of items with two dimensions. They reviewed papers on strip packing, bin packing, knapsack, orthogonal packing, cutting stock problems, and their relevant variants, discussing the main heuristics and exact methods already proposed in the literature.

The main contributions presented in this paper are: (i) the proposition of a constraint programming (CP) model for the 3D-ODRPP based on a position-free modeling approach with logic operators; and, (ii) several experiments with two general-purpose CP solvers and one general-purpose mixed-integer linear programming (MILP) solver on four sets of benchmark instances to evaluate the computational performance of the proposed approach. We consider as a benchmark the linear version of the MIP models of Junqueira

& Morabito (2017) (a grid-position model) and Huang & Hwang (2018) (a position-free model), which are the current exact state-of-the-art model-based approaches for the 3D-ODRPP. The linearization technique proposed in Huang & Hwang (2018) is used to write the objective function of the grid-position approach of Junqueira & Morabito (2017) to have exact benchmark approaches only. Our approach is the first CP model for the 3D-ODRPP in the literature. Besides that, its results are very competitive in different sets of problem instances in terms of reaching optimality and providing improved feasible solutions quickly. In addition, we develop an exact algorithm for the 3D-ODRPP based on the proposed CP model as a feasibility checker program.

The remainder of the paper is structured as follows. For the sake of completeness, we briefly describe the benchmark MIP models of Junqueira & Morabito (2017) and Huang & Hwang (2018) in Section 2. In Section 3, we present the proposed CP model for the 3D-ODRPP. The computational experiments performed to evaluate the proposed approach are reported in Section 4, which considers four sets of benchmark instances from the literature. We present final remarks and possible paths for future research in Section 5.

2. Benchmark MIP models

We briefly describe the MIP models of Junqueira & Morabito (2017) and Huang & Hwang (2018) in Sections 2.1 and 2.2, respectively – the notation of each model is limited to its section. For a detailed description of these formulations, we refer to the original works, especially to Huang & Hwang (2018) concerning the linearization technique for the nonlinear objective function to minimize the container volume XYZ.

We next recall the notation of the 3D-ODRPP to be used when presenting the models with the addition of lower and upper bounds on the container dimensions:

n	number of boxes to be packed;
$I = \{1, \dots, n\}$	set of boxes to be packed;
l_i, w_i, h_i	length (x-axis), width (y-axis), and height (z-axis) of box $i \in I$, respectively;
$\underline{X}$, $\overline{X}$	lower and upper bounds on the container dimension in the x-axis, respectively;
$\underline{Y}$, $\overline{Y}$	lower and upper bounds on the container dimension in the y-axis, respectively;
$\underline{Z}$, $\overline{Z}$	lower and upper bounds on the container dimension in the z-axis, respectively;
X, Y, Z	variables representing the container dimensions in the x-axis, y-axis, and z-axis, respectively.

We notice that (i) one can estimate these lower and upper bounds on the container dimensions from techniques that exploit the input data (Lin et al., 2017) or even define them according to a managerial decision (Junqueira & Morabito, 2017); and (ii) variables X, Y, and Z are common to the MIP benchmark formulations and the proposed CP model. Without loss of optimality, we assume that the input data only have positive integers. Note that the accuracy of measuring equipment in practical environments is limited and that there is always the possibility of scaling decimal numbers to integers.

2.1. Formulation of Junqueira & Morabito (2017)

As a grid-position approach, this formulation aggregates the n boxes to be packed into $m \leq n$ box types when the boxes share identical sizes. Each box type $j \in \{1, \dots, m\}$ has a length l_j, width w_j, height h_j, and number b_j of copies to be packed, such that $n = \sum_{j=1}^{m} b_j$. The container is geometrically represented as a grid of points. At first, let us assume a unitary discretization of the container, that is, we define sets $X^c = \{0, 1, \dots, \overline{X}\}$, $Y^c = \{0, 1, \dots, \overline{Y}\}$, and $Z^c = \{0, 1, \dots, \overline{Z}\}$. In addition, let $J = \{1, \dots, 6m\}$ be a set representing the six different orientations of the box types – see Figure 2 for an illustration of the six orientations to pack a box orthogonally into the container. For instance, the orientations of the first box type are $j = 1, \dots, 6$, and the last box type are $j = 6m - 5, \dots, 6m$. By abuse of notation, let us assume l_i, w_i, and h_i be the length, width, and height of a box type in its orientation $i \in J$. Finally, we define the sets of allocation points for each box type $i \in J$ as $X_i^c = \{x \in X^c \mid x + l_i \leq \overline{X}\}$, $Y_i^c = \{y \in Y^c \mid y + w_i \leq \overline{Y}\}$, and $Z_i^c = \{z \in Z^c \mid z + h_i \leq \overline{Z}\}$. The main decision of the formulation concerns the allocation variables, which are defined in what follows:

λ_{ixyz} binary variable which equals (1) if box type $i \in J$ is packed at point (x, y, z), with $x \in X_i^c$, $y \in Y_i^c$, and $z \in Z_i^c$, and (0) otherwise.

A grid-position model based on Junqueira & Morabito (2017) is given by Model (1).

$$
\begin{aligned}
\textbf{Min} \quad & XYZ, && && \text{(1a)}\\
\textbf{s.t.} \quad & \sum_{i \in J} \sum_{\substack{\{x \in X_i^c,\\ x'-l_i+1 \le x \le x'\}}} \sum_{\substack{\{y \in Y_i^c,\\ y'-w_i+1 \le y \le y'\}}} \sum_{\substack{\{z \in Z_i^c,\\ z'-h_i+1 \le z \le z'\}}} \lambda_{ixyz} \le 1, && x' \in X^c, y' \in Y^c, z' \in Z^c, && \text{(1b)}\\
& \sum_{i=6j-5}^{6j} \sum_{x \in X_i^c} \sum_{y \in Y_i^c} \sum_{z \in Z_i^c} \lambda_{ixyz} = b_j, && j \in \{1, \ldots, m\}, && \text{(1c)}\\
& (x + l_i) \cdot \lambda_{ixyz} \le X, && i \in J, x \in X_i^c, y \in Y_i^c, z \in Z_i^c, && \text{(1d)}\\
& (y + w_i) \cdot \lambda_{ixyz} \le Y, && i \in J, x \in X_i^c, y \in Y_i^c, z \in Z_i^c, && \text{(1e)}\\
& (z + h_i) \cdot \lambda_{ixyz} \le Z, && i \in J, x \in X_i^c, y \in Y_i^c, z \in Z_i^c, && \text{(1f)}\\
& \lambda_{ixyz} \in \{0, 1\}, && i \in J, x \in X_i^c, y \in Y_i^c, z \in Z_i^c, && \text{(1g)}\\
& X, Y, Z \ge 0. && && \text{(1h)}
\end{aligned}
$$

The objective function (1a) minimizes the volume of the container. Constraints (1b) guarantee the non-overlap of the packed boxes by ensuring each point (x', y', z') of the geometric representation of the container is occupied by at most one box, already considering the six possible orientations of each box type. Constraints (1c) enforce all copies b_j of box type $j \in \{1, \ldots, m\}$ to be packed; note that the first summation in the left-hand side of the constraints considers the six possible orientations of the corresponding box type. Constraints (1d), (1e), and (1f) enforce the packed boxes to lie entirely within the container in the x-axis, y-axis, and z-axis, respectively. Constraints (1g) and (1h) define the domain of the variables.

We highlight that Junqueira & Morabito (2017) considered the sets of normal points (Herz, 1972; Christofides & Whitlock, 1977) instead of a complete unitary discretization for representing the container, seeking for a smaller number of variables and constraints, which tends to be useful in the context of general-purpose solvers. To do that, one should replace sets X^c, Y^c, and Z^c by the corresponding sets of normal points, defined in Section 3.1.

2.2. Formulation of Huang & Hwang (2018)

As a position-free approach, this formulation considers the boxes to be different from each other even when they share identical sizes. Beyond the container variables, the formulation relies on fifteen types of decision variables, which are defined in what follows:

(x_i, y_i, z_i) variables representing the left-front-bottom corner of item $i \in I$;

$\mathcal{L}_{xi}, \mathcal{L}_{xy}, \mathcal{L}_{xz}$ binary variables indicating whether the length of box $i \in I$ is parallel to the x-axis, y-axis or z-axis, respectively:

$\mathcal{W}_{xi}, \mathcal{W}_{xy}, \mathcal{W}_{xz}$ binary variables indicating whether the width of box $i \in I$ is parallel to the x-axis, y-axis or z-axis, respectively:

$\mathcal{H}_{xi}, \mathcal{H}_{xy}, \mathcal{H}_{xz}$ binary variables indicating whether the height of box $i \in I$ is parallel to the x-axis, y-axis or z-axis, respectively:

$\alpha_{ij}, \beta_{ij}, \delta_{ij}$ binary variables indicating the relative positions of two boxes $i, j \in I$, $i < j$, where:
$(\alpha_{ij}, \beta_{ij}, \delta_{ij}) = (0, 0, 1)$ if box i is to the left of box j,
$(\alpha_{ij}, \beta_{ij}, \delta_{ij}) = (0, 1, 0)$ if box i is to the right of box j,
$(\alpha_{ij}, \beta_{ij}, \delta_{ij}) = (1, 0, 0)$ if box i is behind box j,
$(\alpha_{ij}, \beta_{ij}, \delta_{ij}) = (0, 1, 1)$ if box i is in front of box j,
$(\alpha_{ij}, \beta_{ij}, \delta_{ij}) = (1, 0, 1)$ if box i is below box j,
$(\alpha_{ij}, \beta_{ij}, \delta_{ij}) = (1, 1, 0)$ if box i is above box j.

A position-free model based on Huang & Hwang (2018) is given by Model (2).

$$
\begin{aligned}
\textbf{Min}\quad & XYZ, && && \text{(2a)}\\
\textbf{s.t.}\quad & x_i + l_i\mathcal{L}_{xi} + w_i\mathcal{W}_{xi} + h_i\mathcal{H}_{xi} \le x_j + M(\alpha_{ij} + \beta_{ij} + 1 - \delta_{ij}), && i,j \in I, i<j, && \text{(2b)}\\
& x_j + l_j\mathcal{L}_{xj} + w_j\mathcal{W}_{xj} + h_j\mathcal{H}_{xj} \le x_i + M(\alpha_{ij} + 1 - \beta_{ij} + \delta_{ij}), && i,j \in I, i<j, && \text{(2c)}\\
& y_i + l_i\mathcal{L}_{yi} + w_i\mathcal{W}_{yi} + h_i\mathcal{H}_{yi} \le y_j + M(1 - \alpha_{ij} + \beta_{ij} + \delta_{ij}), && i,j \in I, i<j, && \text{(2d)}\\
& y_j + l_j\mathcal{L}_{yj} + w_j\mathcal{W}_{yj} + h_j\mathcal{H}_{yj} \le y_i + M(\alpha_{ij} + 2 - \beta_{ij} - \delta_{ij}), && i,j \in I, i<j, && \text{(2e)}\\
& z_i + l_i\mathcal{L}_{yi} + w_i\mathcal{W}_{zi} + h_i\mathcal{H}_{zi} \le z_j + M(2 - \alpha_{ij} + \beta_{ij} - \delta_{ij}), && i,j \in I, i<j, && \text{(2f)}\\
& z_j + l_j\mathcal{L}_{yj} + w_j\mathcal{W}_{zj} + h_j\mathcal{H}_{zj} \le z_i + M(2 - \alpha_{ij} - \beta_{ij} + \delta_{ij}), && i,j \in I, i<j, && \text{(2g)}\\
& 1 \le \alpha_{ij} + \beta_{ij} + \delta_{ij} \le 2, && i,j \in I, i<j, && \text{(2h)}\\
& x_i + l_i\mathcal{L}_{xi} + w_i\mathcal{W}_{xi} + h_i\mathcal{H}_{xi} \le X, && i \in I, && \text{(2i)}\\
& y_i + l_i\mathcal{L}_{yi} + w_i\mathcal{W}_{yi} + h_i\mathcal{H}_{yi} \le Y, && i \in I, && \text{(2j)}\\
& z_i + l_i\mathcal{L}_{yi} + w_i\mathcal{W}_{zi} + h_i\mathcal{H}_{zi} \le Z, && i \in I, && \text{(2k)}\\
& \mathcal{L}_{xi} + \mathcal{L}_{yi} + \mathcal{L}_{zi} = 1, && i \in I, && \text{(2l)}\\
& \mathcal{W}_{xi} + \mathcal{W}_{yi} + \mathcal{W}_{zi} = 1, && i \in I, && \text{(2m)}\\
& \mathcal{H}_{xi} + \mathcal{H}_{yi} + \mathcal{H}_{zi} = 1, && i \in I, && \text{(2n)}\\
& \mathcal{L}_{xi} + \mathcal{W}_{xi} + \mathcal{H}_{xi} = 1, && i \in I, && \text{(2o)}\\
& \mathcal{L}_{yi} + \mathcal{W}_{yi} + \mathcal{H}_{yi} = 1, && i \in I, && \text{(2p)}\\
& \mathcal{L}_{zi} + \mathcal{W}_{zi} + \mathcal{H}_{zi} = 1, && i \in I, && \text{(2q)}\\
& Z \le Y \le X, && && \text{(2r)}\\
& x_i, y_i, z_i \ge 0, && i \in I, && \text{(2s)}\\
& \mathcal{L}_{xi}, \mathcal{L}_{yi}, \mathcal{L}_{yi}, \mathcal{W}_{xi}, \mathcal{W}_{yi}, \mathcal{W}_{zi}, \mathcal{H}_{xi}, \mathcal{H}_{yi}, \mathcal{H}_{zi} \in \{0,1\}, && i \in I, && \text{(2t)}\\
& \alpha_{ij}, \beta_{ij}, \delta_{ij} \in \{0,1\}, && i,j \in I, i<j, && \text{(2u)}\\
& 0 \le X \le \bar{X}, 0 \le Y \le \bar{Y}, 0 \le Z \le \bar{Z}. && && \text{(2v)}
\end{aligned}
$$

The objective function (2a) minimizes the container volume. Constraints (2b) to (2h) guarantee the non-overlap of any pair of boxes $i, j \in I$, $i < j$. Constraints (2i), (2j), and (2k) enforce that each box $i \in I$ lies entirely within the container in the x-axis, y-axis, and z-axis, respectively. Constraints (2l) ensure that the length of box $i \in I$ is parallel to one of the x-axis, y-axis, and z-axis, respectively. Similarly, we have constraints (2m) and (2n) for the width and height of box $i \in I$, respectively. Constraints (2o) ensure that only one of the length, width, and height of box $i \in I$ is parallel to the x-axis, respectively. Similarly, we have constraints (2p) and (2q) models for the y-axis and z-axis, respectively. Valid inequalities (2r) impose that the dimension of the container in the x-axis (resp., z-axis) is the largest (resp., smallest) among all (Lin et al., 2017); these inequalities do not cause the loss of optimality, since the boxes are free to be rotated. Constraints (2s) to (2v) define the variables domain.

We highlight that Huang & Hwang (2018) improved Model (2) by exploring the dependencies among constraints (2l) to (2q). They replaced these six sets of constraints with five sets and reduced their corresponding nine sets of binary variables into four sets. In particular, variables $\mathcal{L}_{xi}$, $\mathcal{L}_{yi}$, $\mathcal{L}_{zi}$, $\mathcal{W}_{xi}$, and $\mathcal{H}_{xi}$ can be replaced in the formulation by expressions written in terms of variables $\mathcal{W}_{yi}$, $\mathcal{W}_{zi}$, $\mathcal{H}_{yi}$, and $\mathcal{H}_{zi}$, $i \in I$. Notice that Models (1) and (2) are nonlinear due to the product of variables in their objective functions. However, Huang & Hwang (2018) proposed a linear equivalent set of expressions to objective function (2a) based on an improved special ordered set of type 1 (SOS1) formulation. More details are given by the authors in the original paper.

3. A constraint programming model for the 3D-ODRPP

The proposed CP model for the 3D-ODRPP relies on the position-free modeling approach with logic operators. Nevertheless, the model can be seen as intermediate between the position-free and grid-position approaches since we can exploit the benefits of both approaches, as discussed in Section 3.1. There are four families of decision variables in the proposed model. They are defined in what follows:

X, Y, Z	variables representing the container's dimensions in the x-axis, y-axis, and z-axis, respectively;
x_i, y_i, z_i	variables representing the allocation point of box $i \in I$;
d_i^x, d_i^y, d_i^z	variables representing the dimensions of box $i \in I$ in the x-axis, y-axis, z-axis, respectively;
$r_i = \{1, \ldots, 6\}$	variable that assumes one of the six possible rotations of box $i \in I$.

A box is free to be rotated in a solution to the 3D-ODRPP if its walls are parallel to the container's walls. If a box $i \in I$ were not free to be rotated, then $d_i^x = l_i$, $d_i^y = w_i$, and $d_i^z = h_i$ (i.e., $r_i = 1$) would be the single scheme. However, each box $i \in I$ can be rotated in six ways to be packed into the container, as depicted in Fig. 2. For instance, one can see in the figure that $r_i = 5$ means that $d_i^x = h_i$, $d_i^y = l_i$, and $d_i^z = w_i$.

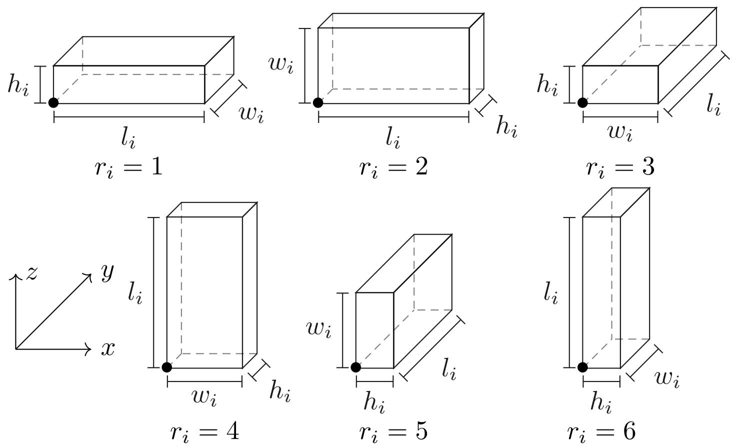


Figure 2: Relations of variable r_i and the dimensions of box $i \in I$ according to the input data.

The proposed CP model for the 3D-ODRPP is given by Model (3).

$$\textbf{Min} \quad XYZ, \tag{3a}$$

$$\textbf{s.t.} \quad x_i + d_i^x \leq X, \qquad i \in I, \tag{3b}$$

$$y_i + d_i^y \leq Y, \qquad i \in I, \tag{3c}$$

$$z_i + d_i^z \leq Z, \qquad i \in I, \tag{3d}$$

$$\begin{aligned} & x_i + d_i^x \leq x_j \vee x_j + d_j^x \leq x_i \vee y_i + d_i^y \leq y_j \\ & \vee\, y_j + d_j^y \leq y_i \vee z_i + d_i^z \leq z_j \vee z_j + d_j^z \leq z_i, \end{aligned} \qquad i, j \in I, i < j, \tag{3e}$$

$$r_i = 1 \implies d_i^x = l_i \wedge d_i^y = w_i \wedge d_i^z = h_i, \qquad i \in I, \tag{3f}$$

$$r_i = 2 \implies d_i^x = l_i \wedge d_i^y = h_i \wedge d_i^z = w_i, \qquad i \in I, \tag{3g}$$

$$r_i = 3 \implies d_i^x = w_i \wedge d_i^y = l_i \wedge d_i^z = h_i, \qquad i \in I, \tag{3h}$$

$$r_i = 4 \implies d_i^x = w_i \wedge d_i^y = h_i \wedge d_i^z = l_i, \qquad i \in I, \tag{3i}$$

$$r_i = 5 \implies d_i^x = h_i \wedge d_i^y = l_i \wedge d_i^z = w_i, \qquad i \in I, \tag{3j}$$

$$r_i = 6 \implies d_i^x = h_i \wedge d_i^y = w_i \wedge d_i^z = l_i, \qquad i \in I, \quad (3k)$$
$$X, Y, Z \in \mathbb{Z}_+, \qquad (3l)$$
$$x_i, y_i, z_i \in \mathbb{Z}_+, \qquad i \in I, \quad (3m)$$
$$d_i^x, d_i^y, d_i^z \in \mathbb{Z}_+, \qquad i \in I, \quad (3n)$$
$$r_i \in \{1, 2, 3, 4, 5, 6\}, \qquad i \in I. \quad (3o)$$

The objective function (3a) minimizes the volume of the container. Constraints (3b) ensure the containment condition in the x-axis, i.e., each box $i \in I$ must be entirely within the length of the container. Similarly, constraints (3c) and (3d) ensure the containment condition in the y-axis and z-axis, respectively. Constraints (3e) guarantee the non-overlap between any pair of boxes $i, j \in I$, $i < j$. They are "or constraints" that require the fulfillment of at least one out of six conditions: box i is to the left/right of (x-axis), in front of/behind (y-axis), or below/above (z-axis) box j. Using logic operators, constraints (3f) to (3k) are responsible for linking the value of variables d_i^x, d_i^y, and d_i^z according to the rotation r_i of box $i \in I$ and the input data. Constraints (3l), (3m), (3n), and (3o) define the domain of the variables. Notice that Model (3) is nonlinear due to the objective function (3a) and constraints (3e) to (3k).

3.1. Enhancing the constraint programming model

In what follows, we use valid inequalities and techniques to reduce, without loss of optimality, the feasible solution space seeking to improve the performance of the proposed model in the context of a general-purpose CP solver. Valid inequality (4a) imposes that the sum of the volume of the boxes is a lower bound for the volume of the container (Tsai et al., 2015). Valid inequalities (4b) impose that the dimension of the container in the x-axis (resp. z-axis) is the largest (resp. smallest) among all (Lin et al., 2017); these inequalities are valid since boxes are free to be rotated. Expressions (4c), (4d), and (4e) are "and constraints" that forbid some schemes in the rotation r_i of a box $i \in I$ when this box shares the same size in two dimensions. For instance, these expressions enforce that $r_i = 1$ if box $i \in I$ has $l_i = w_i = h_i$.

$$XYZ \geq \sum_{i \in I} l_i w_i h_i, \qquad (4a)$$
$$X \geq Y \geq Z, \qquad (4b)$$
$$r_i \neq 3 \wedge r_i \neq 4 \wedge r_i \neq 6, \qquad i \in I, l_i = w_i, \quad (4c)$$
$$r_i \neq 4 \wedge r_i \neq 5 \wedge r_i \neq 6, \qquad i \in I, l_i = h_i, \quad (4d)$$
$$r_i \neq 2 \wedge r_i \neq 5 \wedge r_i \neq 6, \qquad i \in I, w_i = h_i. \quad (4e)$$

We highlight that, although Model (3) relies on a position-free modeling approach, we can exploit techniques developed for the grid-position modeling approach, given that the CP paradigm explores the domain of the variables during the search. Similarly to Junqueira & Morabito (2017), we consider the allocation point of a box $i \in I$ to be left-front-bottom corner according to the sets of normal points. Since each box can be rotated, let d_i be a $3n$-array that receives the values of l_i, w_i, and h_i, $\forall i \in I$. Let X^N, Y^N, and Z^N be the set of normal patterns in the x-axis, y-axis, and z-axis, respectively, as defined in expressions (5a), (5b), and (5c) (Herz, 1972; Christofides & Whitlock, 1977).

$$X^N = \left\{ p_x \mid p_x = \sum_{i=1}^{3n} d_i \tau_i, \ 0 \leq p_x \leq \overline{X} - \min_{i=1,\dots,3n} \{d_i\}, \tau_i \in \{0,1\}, i = 1, \dots, 3n \right\}, \qquad (5a)$$
$$Y^N = \left\{ p_y \mid p_y = \sum_{i=1}^{3n} d_i \tau_i, \ 0 \leq p_y \leq \overline{Y} - \min_{i=1,\dots,3n} \{d_i\}, \tau_i \in \{0,1\}, i = 1, \dots, 3n \right\}, \qquad (5b)$$
$$Z^N = \left\{ p_z \mid p_z = \sum_{i=1}^{3n} d_i \tau_i, \ 0 \leq p_z \leq \overline{Z} - \min_{i=1,\dots,3n} \{d_i\}, \tau_i \in \{0,1\}, i = 1, \dots, 3n \right\}. \qquad (5c)$$

Note that the domain of an allocation point (x_i, y_i, z_i) of each box $i \in I$ can be reduced to those points in the set of the normal patterns. Thus, without loss of optimality, expression (3m) is replaced with expression (6a).

$$x_i \in X^N, y_i \in Y^N, z_i \in Z^N, \qquad i \in I. \quad (6a)$$

Similarly, we can use the set of normal points to reduce the domain of the variables X, Y, and Z representing the container's dimensions. However, since these dimensions are not allocation points but sizes, we need to adjust these sets, as defined in expressions (7a) to (7c). These adjusted sets could be seen as the set of useful numbers (Cunha et al., 2020). Thus, expression (3l) is replaced with expressions (7a) to (7c).

$$X \in \{p_x + d_i \mid p_x \in X^N, \underline{X} \leq p_x + d_i \leq \overline{X}, i = 1, \ldots, 3n\}, \quad (7a)$$

$$Y \in \{p_y + d_i \mid p_y \in Y^N, \underline{Y} \leq p_y + d_i \leq \overline{Y}, i = 1, \ldots, 3n\}, \quad (7b)$$

$$Z \in \{p_z + d_i \mid p_z \in Z^N, \underline{H} \leq p_z + d_i \leq \overline{Z}, i = 1, \ldots, 3n\}. \quad (7c)$$

More details on grids of points, including discussions on the optimality and a dynamic programming algorithm to calculate the normal patterns, can be found in Cunha et al. (2020). It is important to mention that a more refined grid could result in the loss of the optimal solution for 3D-ODRPP, e.g., by using the grid of reduced raster points as discussed by Junqueira & Queiroz (2022).

4. Computational experiments

We consider experiments to evaluate the computational performance of the proposed CP model. The proposed and benchmark approaches were coded in C++. We used IBM CPLEX Optimization Studio v.22.1 as a MILP and CP solver and LocalSolver as an alternative and heuristic solver. All experiments were conducted on a PC with Intel Xeon X5660 2.8 GHz, 96 GB of RAM, under Ubuntu 22.04 LTS as operating system. Each run of the solver was limited to 3,600 seconds unless stated otherwise. We use the letters "tl" in the next tables to indicate when this time limit was reached for a given instance or class of instances.

4.1. Solution approaches and problem instances

We refer to Model (3) with the enhancements discussed in Section 3.1 as Exact-CPM if solved by CPLEX and Heuri-CPM if solved by LocalSolver. We considered as a benchmark the linear version of the MIP models of Junqueira & Morabito (2017) and Huang & Hwang (2018), which are denoted by Grid-ILP and PosF-ILP, respectively. These benchmark models were solved with the aim of CPLEX. As mentioned before, we extended the linearization technique proposed in Huang & Hwang (2018) for the objective function to the grid-position approach of Junqueira & Morabito (2017).

We report information on the five sets of instances used in the experiments in Table 2, which makes 156 problem instances in total. We consider the same problem instances of Tsai et al. (2015) and Junqueira & Morabito (2017) in Section 4.2. In Sections 4.3 and 4.4, we consider the problem instances of Huang & Hwang (2018) and Egeblad & Pisinger (2009) (ep3 instances), respectively. The lower and upper bounds on the container dimensions were taken from the supplementary material of Huang & Hwang (2018) concerning instance sets #A and #C, and the definitions of Junqueira & Morabito (2017) concerning instance set #B. The ep3 instances were initially generated for the three-dimensional knapsack problem; thus, we neglected the information on the boxes' profit. For sets #D1 and #D2, we define the lower bounds on the container dimensions as $\min_{i \in I}\{l_i; w_i; h_i\}$ (i.e., the smallest dimension among all the boxes). These sets #D1 and #D2 differ concerning the upper bounds on the container dimensions. For set #D1, we use the original dimension of the knapsack on the instance multiplied by a factor of 4. For set #D2, we assume a container with identical upper bounds $\overline{X} = \overline{Y} = \overline{Z}$, where $\overline{X} = \lceil \sqrt[3]{1.35 \sum_{i \in I} l_i w_i h_i} \rceil$.

Table 2: Sets of benchmark instances.

Sets	Number of instances	Source	Number of boxes (n)	Number of box types (m)
#A	10 instances	Tsai et al. (2015)	4 to 9	4 to 6
#B	16 instances	Junqueira & Morabito (2017)	9 to 32	5 to 10
#C	10 instances	Huang & Hwang (2018)	6 to 16	6 to 16
#D1 and #D2	60+60 instances	Egeblad & Pisinger (2009)	20 to 60	5 to 60

It is important to mention that we have implemented and tested other versions of the CP model to 3D-ODRPP. None outperformed the model presented in Section 3. In the first version, we iterate over the container dimensions X, Y, and Z by performing a search over the sets of normal patterns. To each container (X, Y, Z), we invoke the CP model in Section 3, without its objective function, to check the packing feasibility (i.e., whether it is possible to pack all items inside the given container of dimensions (X, Y, Z)). The second version considers solving a relaxed two-dimensional problem before checking the packing feasibility (Nascimento et al., 2021). This relaxation consists of checking whether all items can be arranged inside the given container, satisfying the following constraints: (i) for each line perpendicular to the xy-plane and crossing the normal point (s, t), the sum of the heights of those items intersecting such a line must be less than or equal to the container height Z; (ii) similarly, we require, for each line that is perpendicular to the xz- and yz-planes and crossing the normal points (s, u) and (t, u), respectively, the sum of the widths and lengths of those items intersecting such lines must be less than or equal to the container width Y and length X. If there is no solution for this relaxation, there is no need to check the packing feasibility, and consequently, we avoid unnecessary calls to the latter. In the third version, we include this relaxation in the CP model of Section 3. All these versions required at least twice the computing time required to solve the CP model as proposed in Section 3.

4.2. Results of the instance sets #A and #B

We report the results considering the instance set #A in Table 3. We report the name of the instance, the number of box types (m), and the number of boxes (n). In this set, boxes can be aggregated into box types when they share the same dimensions. Then, we provide input data aggregated into box types only to Grid-ILP as it is the only approach that uses this strategy. For each of the four approaches, we report the number of variables (vars), number of constraints (cons), value of the container dimensions (X, Y, Z), value of the container volume (XYZ), value of the lower bound (lb), optimality gap in percentage (gap[%]), and computing time in seconds (time[s]). For each instance, we set the gap as $100 \cdot (XYZ - lb)/XYZ$, where lb is the best lower bound found among the four approaches; thus, if the gap is equal to zero and the time limit is reached, then an optimal solution has been obtained, but its optimality has not been proven within the time limit. We highlight that the linear objective function considered in Grid-ILP and PosF-ILP uses logarithmic functions that, even not losing optimality, affect the values of the primal and dual solutions, resulting in artificially smaller optimality gaps. For instance, an optimality gap of 2.50% obtained by the solver with Grid-ILP or PosF-ILP at the end of the search means $e^{2.50} = 12.18\%$, while a gap of 1.50% means indeed $e^{1.50} = 4.48\%$. To guarantee a fair comparison with Exact-CPM and Heuri-CPM, we adjust the value of the gaps of Grid-ILP and PosF-ILP with the exponential function. In addition, the entry "*" in the gap columns means the approach requires excessive memory (an out-of-memory error). Moreover, if it is equal to 100.00, then the solver with the approach failed to obtain a feasible solution within the time limit. For each instance, we bold the approach(es) with the best optimality gap and computing time.

Concerning the results in Table 3, for the instance set #A, the solver with Grid-ILP obtains optimal solutions for 2 out of 10 instances, while with PosF-ILP this happens for 10 instances, with Exact-CPM it is for 8 instances, and with Heuri-CPM it is for none of the instances. The time limit is reached by the solver with Grid-ILP in 4 instances, with PosF-ILP in none of the instances, with Exact-CPM in 2 instances, and with Heuri-CPM in all instances. We notice that the solver with Grid-ILP runs out of memory in 4 instances and fails to obtain a feasible solution within the time limit in 4 instances. The best computing time among the four approaches is reached by the solver with PosF-ILP in 3 instances and with Exact-CPM

Table 3: Results for the instance set #A from Tsai et al. (2015).

Instance	m	n	Approach	vars	cons	(X, Y, Z)	XYZ	lb	gap[%]	time[s]
T01	4	4	Grid-ILP	292,386	681,763	(*,*,*)	*	*	100.00	tl
			PosF-ILP	144	104	(28,26,6)	4,368	4368	**0.00**	1.32
			Exact-CPM	31	45	(28,26,6)	4,368	4368	**0.00**	**0.17**
			Heuri-CPM	31	45	(40,19,6)	4,560	3616	4.21	tl
T02	5	5	Grid-ILP	1,014,077	2,542,500	(*,*,*)	*	*	100.00	tl
			PosF-ILP	188	145	(37,16,8)	4,736	4736	**0.00**	2.53
			Exact-CPM	38	58	(37,16,8)	4,736	4736	**0.00**	**0.97**
			Heuri-CPM	38	58	(37,16,8)	4,736	3829	**0.00**	tl
T03	6	6	Grid-ILP	4,290,237	10,617,781	(*,*,*)	*	*	100.00	tl
			PosF-ILP	264	195	(42,13,10)	5,460	5460	**0.00**	7.43
			Exact-CPM	45	72	(42,13,10)	5,460	5460	**0.00**	**7.16**
			Heuri-CPM	45	72	(42,13,10)	5,460	4629	**0.00**	tl
T04	6	7	Grid-ILP	1,265,857	3,904,579	(*,*,*)	*	*	100.00	tl
			PosF-ILP	287	249	(31,18,12)	6,696	6696	**0.00**	**80.06**
			Exact-CPM	52	87	(31,18,12)	6,696	6696	**0.00**	370.39
			Heuri-CPM	52	87	(31,18,12)	6,696	5829	**0.00**	tl
T05	4	8	Grid-ILP	15,374	34,699	(9,8,5)	360	360	**0.00**	25.15
			PosF-ILP	178	306	(9,8,5)	360	360	**0.00**	15.50
			Exact-CPM	59	127	(9,8,5)	360	360	**0.00**	**0.02**
			Heuri-CPM	59	127	(9,8,5)	360	294	**0.00**	tl
T06	4	9	Grid-ILP	20,985	47,283	(10,8,6)	480	480	**0.00**	100.40
			PosF-ILP	216	379	(12,8,5)	480	480	**0.00**	165.73
			Exact-CPM	66	147	(12,8,5)	480	480	**0.00**	**0.70**
			Heuri-CPM	66	147	(10,8,6)	480	358	**0.00**	tl
T07	4	4	Grid-ILP	*	*	(*,*,*)	*	*	*	*
			PosF-ILP	269	108	(127,57,30)	217,170	217170	**0.00**	0.38
			Exact-CPM	31	45	(127,57,30)	217,170	217170	**0.00**	**0.16**
			Heuri-CPM	31	45	(72,57,55)	225,720	186131	3.79	tl
T08	5	5	Grid-ILP	*	*	(*,*,*)	*	*	*	*
			PosF-ILP	421	150	(102,95,30)	290,700	290700	**0.00**	11.53
			Exact-CPM	38	58	(102,95,30)	290,700	290700	**0.00**	**7.15**
			Heuri-CPM	38	58	(102,95,30)	290,700	254756	**0.00**	tl
T09	5	6	Grid-ILP	*	*	(*,*,*)	*	*	*	*
			PosF-ILP	494	198	(92,81,50)	372,600	372600	**0.00**	**318.50**
			Exact-CPM	45	72	(92,81,50)	372,600	335958	**0.00**	tl
			Heuri-CPM	45	72	(90,85,50)	382,500	335956	2.59	tl
T10	6	7	Grid-ILP	*	*	(*,*,*)	*	*	*	*
			PosF-ILP	588	255	(101,89,50)	449,450	449450	**0.00**	**1,218.54**
			Exact-CPM	52	87	(101,89,50)	449,450	422789	**0.00**	tl
			Heuri-CPM	52	87	(169,56,50)	473,200	422788	5.02	tl

Note: (i) gap[%] equals "*" means the solver with the approach runs out of memory, (ii) gap[%] equals 100.00 means the solver with the approach fails to obtain a feasible solution within the time limit; and, (iii) gap[%] equals zero and the time limit reached means an optimal solution is found but its optimality is not proven within the time limit.

in 7 instances. Finally, the number of variables and constraints of Exact-CPM and Heuri-CPM are the smallest ones, followed by PosF-ILP, and lastly by Grid-ILP.

We report the results considering the instance set #B in Table 4. In these results, we observe that the number of proven optimal solutions of the solver with Grid-ILP is for 6 out of 16 instances, with PosF-ILP it is for only 1 instance, with Exact-CPM it is for 4 instances, and with Heuri-CPM it happens for none of the instances. The time limit is reached by the solver with Grid-ILP in 10 instances, with PosF-ILP in 15 instances, with Exact-CPM in 12 instances, and with Heuri-CPM in all 16 instances. The solver with Grid-ILP fails to obtain a feasible solution within the time limit in 3 instances and with PosF-ILP in 6 instances. In general, the solver with Grid-ILP has the best computing times, followed by Exact-CPM and lastly by PosF-ILP and Heuri-CPM. Similar to the previous section, the number of variables and constraints of Exact-CPM and Heuri-CPM are the smallest ones, followed by PosF-ILP, and lastly by Grid-ILP.

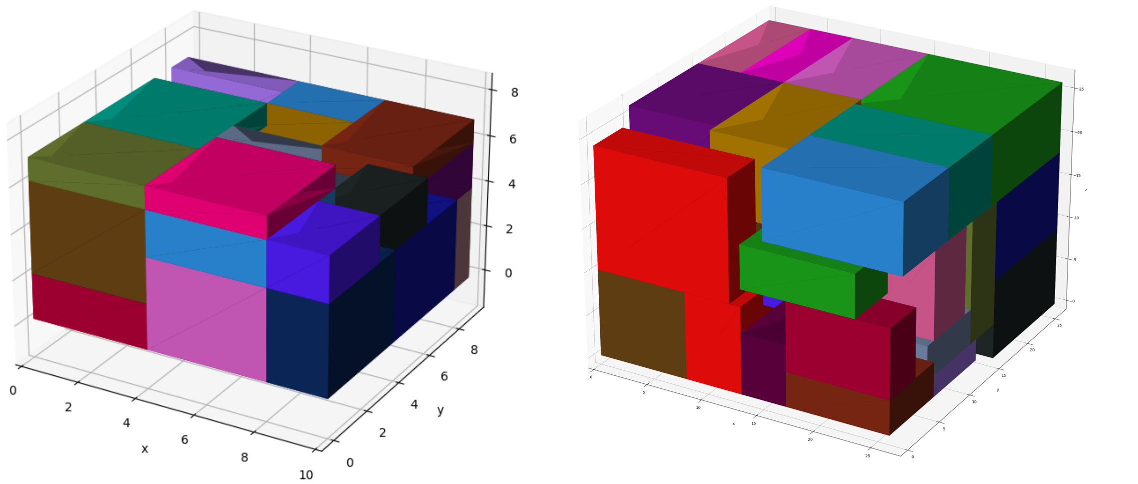
In summary, Exact-CPM is very competitive with Grid-ILP and PosF-ILP in the instance sets #A and #B in terms of optimal solutions and computing time. As mentioned, Exact-CPM can be seen as an intermediary model because it exploits the advantages of the position-free and grid-position approaches. For instance, its number of variables and constraints is small as a function of the number of boxes (n), like in a

Table 4: Results for the instance set #B from Junqueira & Morabito (2017).

Instance	m	n	Approach	vars	cons	(X, Y, Z)	XYZ	lb	gap[%]	time[s]
J01	5	9	Grid-ILP	3,054	7,875	(10,10,7)	700	700	**0.00**	**19.56**
			PosF-ILP	207	378	(10,10,7)	700	630	**0.00**	tl
			Exact-CPM	66	131	(10,10,7)	700	700	**0.00**	118.04
			Heuri-CPM	66	131	(10,10,7)	700	625	**0.00**	tl
J02	10	10	Grid-ILP	7,410	18,754	(10,10,7)	700	700	**0.00**	**48.43**
			PosF-ILP	241	458	(10,10,7)	700	630	**0.00**	tl
			Exact-CPM	73	150	(10,10,7)	700	630	**0.00**	tl
			Heuri-CPM	73	150	(10,10,7)	700	625	**0.00**	tl
J03	5	9	Grid-ILP	14,973	46,953	(20,20,18)	7,200	4994	30.15	tl
			PosF-ILP	232	381	(19,18,16)	5,472	5029	**8.09**	tl
			Exact-CPM	66	120	(19,18,16)	5,472	4998	**8.09**	tl
			Heuri-CPM	66	120	(19,19,16)	5,776	4994	12.93	tl
J04	10	12	Grid-ILP	43,335	126,248	(20,20,20)	8,000	4990	36.41	tl
			PosF-ILP	345	645	(20,20,15)	6,000	4990	15.22	tl
			Exact-CPM	87	180	(20,20,13)	5,200	5087	**2.17**	tl
			Heuri-CPM	87	180	(20,18,15)	5,400	4990	5.80	tl
J05	5	7	Grid-ILP	4,071	13,096	(30,25,25)	18,750	18750	**0.00**	**910.23**
			PosF-ILP	199	248	(29,27,24)	18,792	16128	0.22	tl
			Exact-CPM	52	88	(30,25,25)	18,750	18750	**0.00**	1,374.27
			Heuri-CPM	52	88	(30,27,24)	19,440	16027	3.55	tl
J06	10	11	Grid-ILP	14,982	37,488	(30,27,24)	19,440	17052	**12.28**	tl
			PosF-ILP	326	552	(30,28,24)	20,160	16767	15.42	tl
			Exact-CPM	80	169	(30,27,24)	19,440	16759	**12.28**	tl
			Heuri-CPM	80	169	(30,27,24)	19,440	16751	**12.28**	tl
J07	5	9	Grid-ILP	6,993	21,606	(50,47,34)	79,900	74060	7.31	tl
			PosF-ILP	302	387	(46,43,43)	85,054	72192	12.93	tl
			Exact-CPM	66	120	(47,43,38)	76,798	72183	**3.57**	tl
			Heuri-CPM	66	120	(46,43,43)	85,054	72171	12.93	tl
J08	10	12	Grid-ILP	7,356	18,854	(50,45,37)	83,250	77658	**6.72**	tl
			PosF-ILP	414	651	(*,*,*)	*	*	100.00	tl
			Exact-CPM	87	190	(50,45,37)	83,250	77458	**6.72**	tl
			Heuri-CPM	87	190	(50,45,37)	83,250	76694	**6.72**	tl
J09	5	7	Grid-ILP	3,504	9,108	(96,90,84)	725,760	725760	**0.00**	1,292.42
			PosF-ILP	345	253	(96,90,84)	725,760	725760	**0.00**	352.39
			Exact-CPM	52	96	(96,90,84)	725,760	725760	**0.00**	**0.66**
			Heuri-CPM	52	96	(96,96,84)	774,144	589599	6.25	tl
J10	10	10	Grid-ILP	13,176	35,368	(96,94,76)	685,824	611464	10.84	tl
			PosF-ILP	439	470	(99,94,73)	679,338	611464	9.98	tl
			Exact-CPM	73	143	(96,94,73)	658,752	611510	**7.17**	tl
			Heuri-CPM	73	143	(96,94,73)	658,752	611464	**7.17**	tl
J11	5	25	Grid-ILP	14,937	39,581	(10,9,7)	630	630	**0.00**	**201.51**
			PosF-ILP	1,117	2,620	(*,*,*)	*	*	100.00	tl
			Exact-CPM	178	555	(10,9,7)	630	630	**0.00**	3,196.08
			Heuri-CPM	178	555	(10,9,8)	720	608	12.50	tl
J12	10	28	Grid-ILP	31,845	81,572	(10,9,7)	630	630	**0.00**	**941.44**
			PosF-ILP	1,374	3,269	(*,*,*)	*	*	100.00	tl
			Exact-CPM	199	667	(10,10,7)	700	630	**10.00**	tl
			Heuri-CPM	199	667	(10,10,8)	800	615	21.25	tl
J13	5	30	Grid-ILP	86,919	258,600	(20,20,18)	7,200	4854	32.50	tl
			PosF-ILP	1,584	3,743	(*,*,*)	*	*	100.00	tl
			Exact-CPM	213	714	(20,19,16)	6,080	4860	20.07	tl
			Heuri-CPM	213	714	(19,18,17)	5,814	4854	**16.41**	tl
J14	10	32	Grid-ILP	165,045	470,198	(*,*,*)	*	*	100.00	tl
			PosF-ILP	1,786	4,248	(*,*,*)	*	*	100.00	tl
			Exact-CPM	227	799	(20,17,17)	5,780	4860	**15.92**	tl
			Heuri-CPM	227	799	(19,18,17)	5,814	4855	16.41	tl
J15	5	31	Grid-ILP	166,218	514,002	(*,*,*)	*	*	100.00	tl
			PosF-ILP	1,707	3,992	(*,*,*)	*	*	100.00	tl
			Exact-CPM	220	747	(30,29,24)	20,880	16461	21.16	tl
			Heuri-CPM	220	747	(29,26,26)	19,604	16457	**16.03**	tl
J16	10	28	Grid-ILP	349,209	1,009,855	(*,*,*)	*	*	100.00	tl
			PosF-ILP	1,426	3,272	(*,*,*)	*	*	100.00	tl
			Exact-CPM	199	642	(29,28,24)	19,488	16020	17.80	tl
			Heuri-CPM	199	642	(27,26,26)	18,252	16018	**12.23**	tl

Note: (i) gap[%] equals "*" means the solver with the approach runs out of memory, (ii) gap[%] equals 100.00 means the solver with the approach fails to obtain a feasible solution within the time limit; and, (iii) gap[%] equals zero and the time limit reached means an optimal solution is found but its optimality is not proven within the time limit.

position-free approach. Still, it also uses a grid to define (the domain of) its variables, like in a grid-position fashion. In addition, we note that a major difference concerning instance sets #A and #B is how tight the lower and upper bounds are on the container dimensions. In this sense, at first, we could say the instance set #A is easier to solve than the set #B because of the value of parameters m and n. However, those lower and upper bounds are tighter in the instance set #B than in the set #A concerning the container dimensions (X, Y, Z) in an optimal solution. Therefore, Grid-ILP seems to be very dependent on these bounds given its weak and good performance in instance sets #A and #B, respectively, while PosF-ILP is more dependent on the number of boxes (n). As far as Heuri-CPM is concerned, optimal and near-optimal solutions are found by LocalSolver; however, it failed to provide optimality within the time limit. In Fig. 3, we present two solutions for problem instances of set #B.



(a) a solution of $(X, Y, Z) = (10, 9, 7)$ to instance J11. (b) a solution of $(X, Y, Z) = (27, 26, 26)$ to instance J16.

Figure 3: Solutions for instances J11 and J16: (a) optimal solution provided for the first time to J11, obtained with Exact-CPM; (b) new best know solution provided for J16, obtained with Heuri-CPM.

4.3. Results of the instance set #C

In this section, we evaluate the performance of the approaches, with different values for the time limit, in reaching satisfactory feasible solutions quickly and/or proven optimal solutions. The solver with Grid-ILP runs out of memory in all these experiments; thus, we report results only to PosF-ILP, Exact-CPM, and Heuri-CPM. Again, these results show the dependence of Grid-ILP concerning the bounds on the container dimensions.

All results considering the instance set #C are given in Table 5. The instance set #C consists of 10 problem instances proposed in Huang & Hwang (2018): the first five instances are from an online desktop computer retailer in Taiwan, and the last five have some dimensions originally proposed with rational numbers (we scaled these instances up to get integers numbers). We report results considering the time limits $tl = 60, 600, 3600$ seconds. The solver with PosF-ILP and Exact-CPM obtains optimal solutions for instances H01, H02, and H03 in up to 60 seconds, while for instances H04 and H06, this happens in up to 600 seconds. Concerning instances H05, H07, H08, H09, and H10, the solver performance with Exact-CPM is superior to PosF-ILP regarding the quality of the feasible solutions and/or the computing time. For instance, the solution optimality to instance H08 is proven by the solver with Exact-CPM in up to 60 seconds, while it takes more than 600 seconds with PosF-ILP. We note the gap of the solver with Exact-CPM with tl= 60 and tl= 3600 seconds is smaller than that with PosF-ILP. This indicates the solver with Exact-CPM can find satisfactory feasible solutions quickly. Again, the solver with Heuri-CPM has not proven the optimality in all instances.

Table 5: Results for the instance set #C from Huang & Hwang (2018).

Instance	m	n	Approach	$tl = 60$ s				$tl = 600$ s				$tl = 3600$ s			
				(X, Y, Z)	XYZ	lb	gap[%]	(X, Y, Z)	XYZ	lb	gap[%]	(X, Y, Z)	XYZ	lb	gap[%]
H01	8	9	PosF-ILP	(92,53,36)	175,536	175,536	**0.00**	-	-	-	-	-	-	-	-
			Exact-CPM	(92,53,36)	175,536	175,536	**0.00**	-	-	-	-	-	-	-	-
			Heuri-CPM	(65,62,54)	217,620	149,084	19.34	(92,57,36)	188,784	149,084	**7.02**	(96,53,36)	183,168	149,084	**4.17**
H02	6	10	PosF-ILP	(87,53,48)	221,328	221,328	**0.00**	-	-	-	-	-	-	-	-
			Exact-CPM	(87,53,48)	221,328	221,328	**0.00**	-	-	-	-	-	-	-	-
			Heuri-CPM	(81,57,54)	249,318	197,248	11.23	(87,53,52)	239,772	197,248	**7.69**	(87,53,52)	239,772	197,248	**7.69**
H03	7	11	PosF-ILP	(135,52,34)	238,680	238,680	**0.00**	-	-	-	-	-	-	-	-
			Exact-CPM	(135,52,34)	238,680	238,680	**0.00**	-	-	-	-	-	-	-	-
			Heuri-CPM	(86,86,51)	377,196	232,172	36.72	(86,86,51)	377,196	232,172	**36.72**	(88,81,34)	242,352	232,172	**1.52**
H04	11	12	PosF-ILP	(98,58,57)	323,988	270,724	16.44	(87,64,53)	295,104	295,104	**0.00**	-	-	-	-
			Exact-CPM	(87,64,53)	295,104	258,259	**8.26**	(87,64,53)	295,104	295,104	**0.00**	-	-	-	-
			Heuri-CPM	(86,86,53)	391,988	258,258	30.94	(92,91,41)	343,252	258,258	14.03	(107,53,53)	300,563	258,258	**1.82**
H05	16	16	PosF-ILP	(139,64,64)	569,344	354,058	37.81	(116,78,57)	515,736	354,058	31.35	(87,84,58)	423,864	372,765	12.06
			Exact-CPM	(122,60,57)	417,240	354,060	**15.14**	(87,69,67)	402,201	354,060	**11.97**	(87,69,67)	402,201	354,060	**7.32**
			Heuri-CPM	(131,66,57)	492,822	354,058	28.16	(185,59,41)	447,515	354,058	20.88	(126,65,54)	442,260	354,058	15.71
H06	6	6	PosF-ILP	(180,94,94)	1,590,480	1,509,605	**5.08**	(180,94,94)	1,590,480	1,590,480	**0.00**	-	-	-	-
			Exact-CPM	(180,94,94)	1,590,480	1,312,501	**5.08**	(180,94,94)	1,590,480	1,590,480	**0.00**	-	-	-	-
			Heuri-CPM	(180,180,65)	2,106,000	1,312,500	28.32	(205,90,90)	1,660,500	1,312,500	4.22	(180,94,94)	1,590,480	1,312,500	**0.00**
H07	7	7	PosF-ILP	(480,135,105)	6,804,000	5,407,700	20.52	(480,135,105)	6,804,000	5,407,700	14.19	(331,190,100)	6,289,000	5,409,185	13.99
			Exact-CPM	(212,170,162)	5,838,480	5,407,701	**7.38**	(212,170,162)	5,838,480	5,838,480	**0.00**	-	-	-	-
			Heuri-CPM	(350,165,120)	6,930,000	5,407,700	21.97	(280,180,120)	6,048,000	5,407,700	3.46	(280,180,120)	6,048,000	5,407,700	**3.46**
H08	8	8	PosF-ILP	(245,245,197)	11,824,925	9,160,923	13.20	(315,245,135)	10,418,625	9,855,179	**1.48**	(315,245,133)	10,264,275	10,264,275	**0.00**
			Exact-CPM	(315,245,133)	10,264,275	10,264,275	**0.00**	-	-	-	-	-	-	-	-
			Heuri-CPM	(565,245,91)	12,596,675	8,985,375	18.52	(350,245,133)	11,404,750	8,985,375	10.00	(320,245,133)	10,427,200	8,985,375	1.56
H09	9	9	PosF-ILP	(305,260,215)	17,049,500	13,972,200	18.05	(310,240,216)	16,070,400	13,972,200	13.06	(435,201,180)	15,738,300	14,204,071	9.75
			Exact-CPM	(375,215,190)	15,318,750	13,972,200	**8.79**	(375,215,188)	15,157,500	13,972,200	**7.82**	(375,215,188)	15,157,500	13,972,200	**6.29**
			Heuri-CPM	(325,290,180)	16,965,000	13,972,200	17.64	(305,285,190)	16,515,750	13,972,200	15.40	(405,260,150)	15,795,000	13,972,200	10.07
H10	10	10	PosF-ILP	(1297,694,643)	578,775,874	404,642,700	30.09	(2100,630,387)	512,001,000	404,642,700	20.97	(945,770,665)	483,887,250	404,642,700	16.38
			Exact-CPM	(1470,735,420)	453,789,000	404,642,701	**10.83**	(1218,855,420)	437,383,800	404,642,701	**7.49**	(1218,855,420)	437,383,800	404,642,701	**7.49**
			Heuri-CPM	(1295,945,420)	513,985,500	404,642,700	21.27	(1575,743,420)	491,494,500	404,642,700	17.67	(1890,638,385)	464,240,700	404,642,700	12.84

Note: symbol "-" means the experiment with the current time limit was not performed since the optimality was proven by the solver with the approach in a previous experiment with a smaller time limit.

4.4. Results of the instance sets #D1 and #D2

In this section, we analyze the performance of the approaches considering problem instances with larger numbers of boxes (n), for the case they can be aggregated or not into a small number of box types (m). In addition, we consider instances with containers of identical upper bounds seeking solutions of cubic-alike containers, which tend to be helpful for future packing operations. For the instance sets #D1 and #D2, we use the 60 instances ep3 proposed in Egeblad & Pisinger (2009), disregarding the box profit as this is irrelevant for the 3D-ODRPP. The results of these instances are reported in Tables 6 and 7, respectively. Each table entry has an average value for 10 instances, except columns "w/o solution" with the number of instances without a feasible solution. Again, the solver with Grid-ILP runs out of memory in all these experiments and fails to obtain a feasible solution, and with PosF-ILP in 82 out of 120 instances. Thus, we report results only to Exact-CPM and Heuri-CPM. The solvers reach the limit time of 3600 seconds with Exact-CPM and Heuri-CPM in all instances.

The analysis of the results in Tables 6 and 7 shows that the performance of Heuri-CPM is superior to Exact-CPM on average. In particular, the solver with Exact-CPM fails to obtain a feasible solution within the time limit in 36 out of 60 instances of the set #D2. We highlight that the gap of the solver with Exact-CPM grows faster than with Heuri-CPM as the number of boxes n also grows. Finally, considering only the 38 problem instances that the solvers found feasible solutions for PosF-ILP, Exact-CPM and Heuri-CPM in Table 6, the ratio of the average container volume with Exact-CPM or Heuri-CPM in comparison to PosF-ILP is less than 36.00%, that is, the container volume of the solutions of the solvers with Exact-CPM and Heuri-CPM is less than the half with PosF-ILP on average.

Table 6: Results for the instance set #D from Egeblad & Pisinger (2009).

		Exact-CPM								Heuri-CPM							
n	m	X	Y	Z	XYZ	gap[%]	lb	time[s]	w/o solution	X	Y	Z	XYZ	gap[%]	lb	time[s]	w/o solution
20	5	218.50	185.10	117.10	5,323,116.50	14.10	4,525,939.90	tl	0	211.00	158.80	134.10	5,280,795.10	14.29	4,525,938.40	tl	0
	20	243.60	185.30	98.30	5,142,433.40	15.25	4,383,201.40	tl	0	201.90	158.40	133.90	5,264,297.80	16.74	4,383,198.60	tl	0
40	5	304.80	266.90	135.20	11,422,796.00	20.82	9,051,878.40	tl	0	250.60	211.30	176.60	10,908,652.40	17.02	9,051,876.80	tl	0
	40	306.80	248.40	120.60	10,734,512.00	21.30	8,517,201.40	tl	0	245.10	204.10	165.20	10,222,773.40	16.68	8,517,200.60	tl	0
60	5	317.50	275.30	188.90	18,294,041.90	25.65	13,577,816.90	tl	0	279.90	239.80	205.00	16,204,252.20	16.21	13,577,815.20	tl	0
	60	342.20	287.70	145.90	16,511,589.70	25.57	12,331,860.00	tl	0	259.70	241.60	191.60	15,078,034.30	18.21	12,331,856.40	tl	0

Table 7: Results for the instance set #E by adapting from Egeblad & Pisinger (2009).

		Exact-CPM								Heuri-CPM							
n	m	X	Y	Z	XYZ	gap[%]	lb	time[s]	w/o solution	X	Y	Z	XYZ	gap[%]	lb	time[s]	w/o solution
20	5	180.33	178.33	177.00	6,228,329.67	52.60	4,927,192.33	tl	4	170.80	168.40	164.60	5,563,894.60	11.44	4,525,938.40	tl	0
	20	157.63	156.00	153.75	4,996,721.75	35.97	3,900,232.00	tl	2	165.20	163.40	162.40	5,512,871.20	20.49	4,383,198.60	tl	0
40	5	132.00	132.00	131.00	2,282,544.00	85.17	1,692,331.00	tl	8	213.00	211.20	209.00	11,261,853.60	19.62	9,051,876.80	tl	0
	40	136.50	136.00	135.00	2,876,932.00	69.99	2,126,812.00	tl	6	205.40	205.00	203.80	10,765,141.60	20.88	8,517,200.60	tl	0
60	5	*	*	*	*	100.00	*	tl	10	244.80	244.20	241.00	17,160,484.80	20.88	13,577,815.20	tl	0
	60	201.50	201.50	191.50	11,522,419.00	69.45	9,024,153.00	tl	6	231.80	231.60	229.80	15,537,273.60	20.63	12,331,856.40	tl	0

5. Concluding remarks

We address the three-dimensional open-dimension rectangular packing problem (3D-ODRPP). The problem gains relevance in the design of shipping containers of minimal volume, which is common in the manufacturing, shipping, and warehouse industries (Junqueira & Morabito, 2017). Besides that, this activity is going to play an important role in Industry 4.0 given the general concern in optimizing material usage and reducing waste.

The literature has presented mainly mixed-integer programming formulations and their linearization techniques for the 3D-ODRPP. We instead propose a simple but effective constraint programming (CP) model based on a position-free modeling approach with logic operators. Since the CP paradigm exploits the domain of the variables during the search, we also use techniques of the grid-position approach to reduce the domain of the variables. Using two general-purpose optimization solvers, the results show our approach is able to find optimal or near-optimal solutions quickly for the benchmark instances of the literature, outperforming benchmark approaches in multiple instances. Considering all the 36 instances used in previous related works, our approach Exact-CPM and Heuri-CPM obtains an optimal solution for 22 and 9 instances, with an overall average gap of 4.06% and 6.58%, respectively, both presenting a feasible solution for all instances. We observe that Grid-ILP and PosF-ILP are not able to solve 21 and 7 instances within the given time limit, while their average gaps (considering only the instances they obtain a feasible solution) are 9.08% and 3.93%, respectively. Concerning the 120 larger instances based on the ep3 instances, our approach Exact-CPM and Heuri-CPM obtains a feasible solution for 84 and 120 instances, respectively, while Grid-ILP and PosF-ILP obtain a feasible solution for 0 and 38 instances. Thus, our approach outperforms the literature models.

A path for future research is to extend the enhancements of the proposed CP model to consider more recent developments about the grid of points such as extending the sets of reduced raster points or meet-in-the-middle points and still obtaining an optimal solution. The challenge is to adapt these techniques without losing the optimality; notice that these techniques were initially proposed for problems with containers of fixed dimensions (Queiroz et al., 2015). Another path is the development of heuristic and/or exact algorithms for the problem considering the presence of packing-related constraints, like vertical stability and load balancing (Nascimento et al., 2021). Finally, one could investigate the relationship between the 3D-ODRPP and the variable-sized bin packing problem in the context of distribution center operations that seek to select pre-existing boxes of minimal volume capable of enfolding the items of a customer order placed via e-commerce.

Disclosure statement

No potential conflict of interest was reported by the authors.

Acknowledgment

The authors thank the financial support of the São Paulo Research Foundation (FAPESP-Brazil) [grant numbers 2016/01860-1 and 2022/05803-3]; the National Council for Scientific and Technological Development (CNPq-Brazil) [grants numbers 405702/2021-3, 405369/2021-2, and 311185/2020-7]; and the State of Goiás Research Foundation (FAPEG-Brazil). The research was carried out using the computational resources of the Center for Mathematical Sciences Applied to Industry (CeMEAI), funded by FAPESP-Brazil [grant number 2013/07375-0].

References

Beasley, J. E. (1985). An exact two-dimensional non-guillotine cutting tree search procedure. *Operations Research*, *33*, 49–64. doi:10.1287/opre.33.1.49.

Chen, C., Lee, S., & Shen, Q. (1995). An analytical model for the container loading problem. *European Journal of Operational Research*, *80*, 68–76. doi:10.1016/0377-2217(94)00002-T.

Christofides, N., & Whitlock, C. (1977). An Algorithm for Two-Dimensional Cutting Problems. *Operations Research*, *25*, 30–44. doi:10.1287/opre.25.1.30.

Cunha, J. G. A., De Lima, V. L., & De Queiroz, T. A. (2020). Grids for cutting and packing problems: a study in the 2d knapsack problem. *4OR - A Quarterly Journal of Operations Research*, *18*, 293–339. doi:10.1007/s10288-019-00419-9.

Egeblad, J., & Pisinger, D. (2009). Heuristic approaches for the two- and three-dimensional knapsack packing problem. *Computers & Operations Research*, *36*, 1026–1049. doi:10.1016/j.cor.2007.12.004.

Herz, J. C. (1972). Recursive Computational Procedure for Two-dimensional Stock Cutting. *IBM Journal of Research and Development*, *16*, 462–469. doi:10.1147/rd.165.0462.

Huang, Y.-H., & Hwang, F. J. (2018). Global optimization for the three-dimensional open-dimension rectangular packing problem. *Engineering Optimization*, *50*, 1789–1809. doi:10.1080/0305215X.2017.1411484.

Iori, M., de Lima, V. L., Martello, S., Miyazawa, F. K., & Monaci, M. (2021). Exact solution techniques for two-dimensional cutting and packing. *European Journal of Operational Research*, *289*, 399–415. doi:https://doi.org/10.1016/j.ejor.2020.06.050.

Junqueira, L., & Morabito, R. (2017). On solving three-dimensional open-dimension rectangular packing problems. *Engineering Optimization*, *49*, 733–745. doi:10.1080/0305215X.2016.1208010.

Junqueira, L., & Queiroz, T. A. (2022). The static stability of support factor-based rectangular packings: an assessment by regression analysis. *International Transactions in Operational Research*, *29*, 574–599. doi:https://doi.org/10.1111/itor.12750.

Lin, M.-H., Tsai, J.-F., & Chang, S.-C. (2017). A superior linearization method for signomial discrete functions in solving three-dimensional open-dimension rectangular packing problems. *Engineering Optimization*, *49*, 746–761. doi:10.1080/0305215X.2016.1211403.

Mundim, L. R., Andretta, M., & de Queiroz, T. A. (2017). A biased random key genetic algorithm for open dimension nesting problems using no-fit raster. *Expert Systems with Applications*, *81*, 358–371. doi:https://doi.org/10.1016/j.eswa.2017.03.059.

Nascimento, O. X., Queiroz, T. A., & Junqueira, L. (2021). Practical constraints in the container loading problem: Comprehensive formulations and exact algorithm. *Computers & Operations Research*, *128*, 105186. doi:https://doi.org/10.1016/j.cor.2020.105186.

Queiroz, T. A., Miyazawa, F. K., & Wakabayashi, Y. (2015). On the l-approach for generating unconstrained two-dimensional non-guillotine cutting patterns. *4OR - A Quarterly Journal of Operations Research*, *13*, 199–219. doi:https://doi.org/10.1007/s10288-014-0274-3.

Truong, C., Amodeo, L., Yalaoui, F., Hautefaye, J., & Birebent, S. (2020a). A product arrangement optimization method to reduce packaging environmental impacts. *IOP Conference Series: Earth and Environmental Science*, *463*, 012164. doi:10.1088/1755-1315/463/1/012164.

Truong, C.-T.-T., Amodeo, L., & Yalaoui, F. (2020b). A mathematical model for three-dimensional open dimension packing problem with product stability constraints. In B. Dorronsoro, P. Ruiz, J. C. de la Torre, D. Urda, & E.-G. Talbi (Eds.), *Optimization and Learning* (pp. 241–254). Cham: Springer International Publishing.

Truong, C. T. T., Amodeo, L., & Yalaoui, F. (2021). A genetic algorithm for the three-dimensional open dimension packing problem. In B. Dorronsoro, L. Amodeo, M. Pavone, & P. Ruiz (Eds.), *Optimization and Learning* (pp. 203–215). Cham: Springer International Publishing.

Tsai, J.-F., Wang, P.-C., & Lin, M.-H. (2015). A global optimization approach for solving three-dimensional open dimension rectangular packing problems. *Optimization*, *64*, 2601–2618. doi:10.1080/02331934.2013.877906.

Vieira, M. V. C., Ferreira, F., Duque, J. C. M., & Almeida, R. M. P. (2021). On the packing process in a shoe manufacturer. *Journal of the Operational Research Society*, *72*, 853–864. doi:10.1080/01605682.2019.1700765.

Wäscher, G., Haußner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, *183*, 1109–1130. doi:10.1016/j.ejor.2005.12.047.