Noname manuscript No.
(will be inserted by the editor)

Three-dimensional guillotine cutting problems with constrained patterns: MILP formulations and a bottom-up algorithm

Mateus Martin · José Fernando Oliveira · Elsa Silva · Reinaldo Morabito* · Pedro Munari

Received: date / Accepted: date

Abstract In this paper, we address the Constrained Three-dimensional Guillotine Cutting Problem (C3GCP), which consists of cutting a larger cuboid block (object) to produce a limited number of smaller cuboid pieces (items) using orthogonal guillotine cuts only. This way, all cuts must be parallel to the object's walls and generate two cuboid sub-blocks, and there is a maximum number of copies that can be manufactured for each item type. The C3GCP arises in industrial manufacturing settings, such as the cutting of steel and foam for mattresses. To model this problem, we propose a new compact mixed-integer non-linear programming (MINLP) formulation by extending its two-dimensional version, and develop a mixed-integer linear programming (MILP) version. We also propose a new model for a particular case of the problem which considers 3-staged patterns. As a solution method, we extend the algorithm of Wang (1983) to the three-dimensional case. We emphasise that the C3GCP is different from 3D packing problems, namely from the Container Loading Problem, because of the guillotine cut constraints. All proposed approaches are evaluated through computational experiments using benchmark instances. The results show that the approaches are effective on different types of instances, mainly when the maximum number of copies per item type is small, a situation typically encountered in practical settings with low demand for each item type.

Keywords Cutting and packing · Constrained three-dimensional cutting · Non-staged and 3-staged patterns · Mixed-integer linear programming models · Bottom-up packing

Mateus Martin
Department of Production Engineering, Federal University of São Carlos, Brazil
INESC TEC, Portugal
E-mail: mateus.pmartin@gmail.com

José Fernando Oliveira
INESC TEC, Faculty of Engineering, University of Porto, Portugal
E-mail: jfo@fe.up.pt

Elsa Silva
INESC TEC, Portugal
E-mail: emsilva@inesctec.pt

*Corresponding author: Reinaldo Morabito
Department of Production Engineering, Federal University of São Carlos,
Via Washington Luiz km. 235, 13565-905, São Carlos, SP - Brazil. Phone/fax: 55-16-33519516/33518240
E-mail: morabito@ufscar.br

Pedro Munari
Department of Production Engineering, Federal University of São Carlos, Brazil
E-mail: munari@dep.ufscar.br

1 Introduction

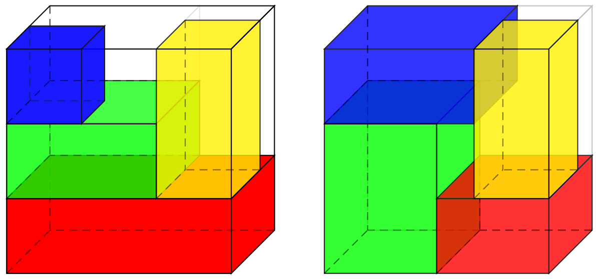
The Constrained Three-dimensional Guillotine Cutting Problem (C3GCP) consists of cutting a large cuboid block (object) to manufacture small cuboid pieces (items). The size of the object is $\bar{L} \times \bar{W} \times \bar{H}$, while each item type $i \in I = \{1, \ldots, m\}$ is characterised by its size $l_i \times w_i \times h_i$, value p_i (e.g. area or other) and the maximum number of u_i copies to be manufactured. The objective is to select and cut the most valuable sub-set of items from the object through orthogonal guillotine cuts while respecting a constrained pattern, as discussed next. Orthogonality means that each cut item has its sides parallel to the walls of the object. The guillotine restriction means that each cut item can only be obtained through a sequence of edge-to-edge cuts, that is, a guillotine cut on the object (or residual object) always generates two sub-objects. If this sequence of orthogonal guillotine cuts is restricted to a limited number of 90° rotations, the cutting pattern is called d-staged where $d \in \mathbb{Z}_+$, and non-staged otherwise. Constrained patterns limit the number of copies produced for each item type, as also happens in real-world problems when overproduction is not allowed. According to the typology of Wäscher et al (2007) for Cutting & Packing problems, the C3GCP is classified as a variant of the Three-dimensional Single Large Placement Problem (SPLOPP), where the guillotine and constrained restrictions are additional requirements. Any feasible solution of this problem should satisfy the following requirements:

(i) the cut items must be inside the object, must not intersect each other, and must have their edges parallel to the walls of the object (geometric constraint);
(ii) all cuts are of guillotine-type (technological constraint);
(iii) the cutting pattern is constrained (production constraint).

Three-dimensional guillotine cutting problems have been rarely addressed in the literature (Bortfeldt and Wäscher, 2013), despite their wide applicability in industrial applications, such as the cutting of steel, marble stones, foam for mattresses and others. Another application example, pointed out in the pioneering paper by Gilmore and Gomory (1965), involves the cutting of graphite blocks for anodes. Particularly, the guillotine cutting of steel is an essential industrial process in steel solution companies that cut stocked blocks of tens of tons to produce small pieces of up to a few tens of grams. This process takes into account the cost and quality of the stocked blocks, the demand for small pieces, and the times for production and delivery. The operation is typically very critical as cutting a stocked block of a few metres in each dimension can take hours to generate two smaller sub-blocks. These considerations indicate a trade-off between higher utilisation rates and shorter production times. In this paper, we propose approaches that take into account these two goals for non-staged and 3-staged cutting patterns.

The Container Loading Problem (CLP) is closely related to the C3GCP, and often considers additional requirements in the context of logistical environments, such as weight distribution, loading priorities, stacking, positioning and stability (Bortfeldt and Wäscher, 2013). The CLP is NP-Hard (Scheithauer, 1992) and mainly addressed by heuristics, which in container loading settings are extensively compared by Zhao et al (2016). Although guillotinable packing patterns are not desirable and should be avoided as they hinder cargo stability, there are solution approaches such as wall-building and layer-building (Toffolo et al, 2017) that generate guillotine cuts as a side-effect. Recently, Silva et al (2019) provided a comparative study of the most significant exact methods for the CLP, such as the mathematical formulations of relative positions (Chen et al, 1995) and discretised objects (Junqueira et al, 2012b,a). Fig. 1 depicts examples of three-dimensional cutting patterns, where the coloured boxes are copies of item types and the remaining volume is waste. It should be noted that while both patterns are feasible for the CLP, only the pattern in Fig. 1a is feasible for the C3GCP.

There are a few solution approaches to the C3GCP and related problems which are mainly based on Dynamic Programming (DP) techniques. Morabito and Arenales (1994) generalised the Gilmore and Gomory (1965)'s DP formulation for two-dimensional problems with 2-staged patterns to the three-dimensional case by considering layers and stacks. They also developed an AND/OR-graph strategy with improved bounds and heuristics to reduce the solution space. Hifi (2002) proposed an algorithm based on an improved version of Gilmore and Gomory (1965)'s DP formulation and several approximate algorithms. Later on, Hifi (2004) adapted Gilmore and Gomory (1966)'s DP formulation to the three-dimensional case and also developed a depth-first search exact algorithm. De Queiroz et al (2012)



(a) Guillotine. (b) Non-guillotine.

Fig. 1: Examples of three-dimensional cutting patterns.

addressed the C3GCP by extending Beasley (1985)'s DP formulations to the non-staged and staged cases. They also addressed the Cutting Stock and Strip Packing versions of the C3GCP by adapting the previous algorithms for heuristic solutions. It should be noted that DP-based algorithms generally consider unconstrained patterns (i.e. patterns that do not limit the maximum number of copies for each item type); these approaches may become intractable in the constrained case because of the huge number of states (i.e. the curse of dimensionality).

Martello et al (2007) addressed the problem of orthogonally packing a given set of boxes into the minimum number of containers. The authors developed an exact branch-and-bound method that considers an algorithm based on Constraint Programming (CP) for filling a single bin using types of non-guillotine patterns. Egeblad and Pisinger (2009) developed algorithms based on Simulated Annealing for the two- and three-dimensional versions of the C3GCP without the guillotine restriction. To represent the cutting patterns, these approaches use a sequence-pair representation and rectangle packing in the two-dimensional version, and a sequence-triple representation and box packing in the three-dimensional version. Amossen and Pisinger (2010) proposed an approach for the multi-dimensional Bin Packing Problem, with computational experiments for the two- and three-dimensional cases. They considered both non-guillotine and guillotine patterns. More recently, Liu et al (2014) addressed a version of the C3GCP that guarantees full support and orientation constraints by proposing a heuristic binary search tree algorithm. Their algorithm is divided into steps wherein the boxes are grouped into strips, and then the strips are further grouped into layers; several one-dimensional knapsack problems are solved in these steps.

The contributions of this paper are three-fold: (i) to completely model the problem, we extend to the three-dimensional case a compact Mixed-Integer Non-Linear Programming (MINLP) formulation, initially proposed in Martin et al (2019) for the Constrained Two-dimensional Guillotine Cutting Problem (C2GCP), and develop a linearization that leads to a Mixed-Integer Linear Programming (MILP) formulation of the C3GCP; (ii) as the adjustment of cutting machines tends to be costly and time-consuming, we propose an Integer Linear Programming (ILP) model for the C3GCP with 3-staged patterns, a case of particular interest in different practical settings; (iii) finally, as a solution method, we extend the bottom-up algorithm proposed by Wang (1983) to the three-dimensional case with non-staged as well as 3-staged cutting patterns.

The remainder of the paper is organised as follows. In Section 2, we propose the mathematical models for the C3GCP with non-staged and 3-staged cutting patterns. In Section 3, we show how to adapt the bottom-up algorithm of Wang (1983) to the three-dimensional case, and discuss some further enhancements. The results of computational experiments are presented in Section 4 regarding the solution quality and processing times of the proposed approaches through two sets of benchmark instances from the literature. Final remarks and future research are discussed in Section 5.

2 Mathematical models for the C3GCP

We are not aware of mathematical models for the C3GCP. The development of models contributes to the characterisation of the problems, and motivates the proposition of decomposition-based solution methods that can enable the solution of larger problem instances. To the best of our knowledge, there are four MILP formulations for the C2GCP (Ben Messaoud et al, 2008; Furini et al, 2016; Martin et al, 2020, 2019). These formulations are likely to be extendable to the three-dimensional case as well, although some would lead to models with too many variables and constraints, which tends to undermine the approach in the context of general-purpose optimisation solvers. In Section 2.1, we extend Martin et al (2019)'s compact formulation to the C3GCP, given that it is characterised for fewer variables and constraints with respect to the other formulations. In Section 2.2, we propose an ILP model for the version of C3GCP that strictly considers 3-staged patterns. It should be noted that d-staged cutting patterns seek shorter production times at the expense of lower material utilisation by rotating the cutting saw just up to d times.

2.1 MINLP model for non-staged patterns

The bottom-up packing approach of successive horizontal and vertical builds of small rectangles was proposed by Wang (1983) in the context of two search tree algorithms for the C2GCP. Recently, Martin et al (2019) addressed the same problem by proposing pseudo-polynomial and compact models which are also based on the bottom-up packing approach. In what follows, we extend their compact model to the C3GCP. We chose their compact model instead of their pseudo-polynomial model because the former outperformed the latter in their computational experiments with larger problem instances.

The C3GCP allows three types of builds in the bottom-up packing approach, as depicted in Fig. 2. Taking into consideration two copies of item types of sizes $l_1 \times w_1 \times h_1$ and $l_2 \times w_2 \times h_2$, the horizontal build of such items provides a block of size $(l_1 + l_2) \times \max\{w_1; w_2\} \times \max\{h_1; h_2\}$; the depth build provides a block of size $\max\{l_1; l_2\} \times (w_1 + w_2) \times \max\{h_1; h_2\}$; and the vertical build provides a block of size $\max\{l_1; l_2\} \times \max\{w_1; w_2\} \times (h_1 + h_2)$. A build in the bottom-up packing approach envelops two small blocks to generate a larger block. It generally does not differentiate the relative placement of such small blocks into the larger block, because the approach relies on the sizes of blocks. As guillotinable patterns are sought, the volume not occupied by copies of item types in a build is considered (inner) waste, i.e., it will not be occupied by any other copy of item types in future builds.

To define the model, we need to determine how many blocks (i.e. sub-patterns) could be built up based on the copies of item types. Let parameter $\hat{n}$ be the number of copies of the item types in a given optimal solution of an ordinary instance of the C3GCP. For such a solution, without loss of optimality, we could consider $\hat{n} - 1$ sub-patterns generated by successive horizontal, depth and/or vertical builds of the copies of the item types. Let $\bar{n}$ be an upper bound to parameter $\hat{n}$ (for instance, $\hat{n} = \sum_{i \in I} u_i$), given that parameter $\hat{n}$ is difficult to be known before solving the problem instance.

The model considers that each block, if any, is always a sub-pattern generated by combining copies of item types or smaller blocks. Let $J = \{1, \ldots, \overline{n} - 1\}$ be the set of possible sub-patterns (i.e. blocks). Let $O = \{h, d, v\}$ be the set of possible orientations for building a sub-pattern, where h, d and v are for horizontal, depth and vertical builds, respectively. Let $Q_i = \{1, \ldots, \min\{2, u_i\}\}$ be the set of possible copies of item type $i \in I$ to be contained in any sub-pattern $j \in J$. There are six sets of variables in this model, which are defined in Equations (1) to (6).

$$z_{jip} = \begin{cases} 1, & \text{if block } j \text{ contains the } p\text{-th copy of item type } i, \\ 0, & \text{otherwise,} \end{cases} \qquad j \in J, i \in I, p \in Q_i. \quad (1)$$

$$x_{jo} = \begin{cases} 1, & \text{if block } j \text{ is built up with orientation } o, \\ 0, & \text{otherwise,} \end{cases} \qquad j \in J, o \in O. \quad (2)$$

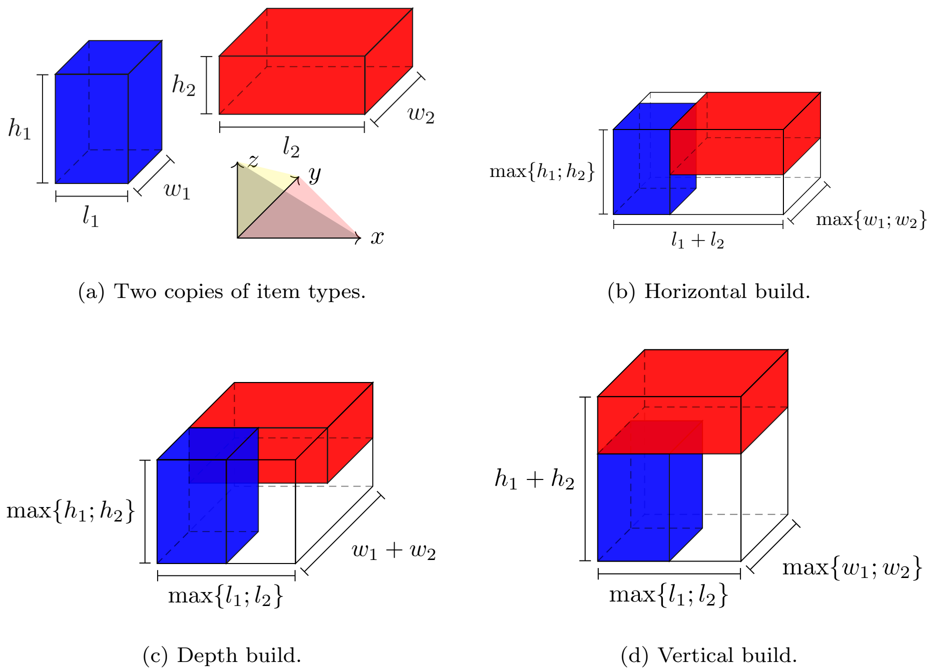


Fig. 2: Examples of builds for the bottom-up approach.

$$y_{jk} = \begin{cases} 1, & \text{if block } j \text{ contains block } k, \\ 0, & \text{otherwise,} \end{cases} \qquad j \in J, k \in J, j < k. \quad (3)$$

$$L_j\text{: length of block } j, \qquad j \in J. \quad (4)$$

$$W_j\text{: width of block } j, \qquad j \in J. \quad (5)$$

$$H_j\text{: height of block } j, \qquad j \in J. \quad (6)$$

Having defined all variables and parameters, we state a compact MINLP formulation for the C3GCP, given by Model (7). Although these authors are unaware of applications, it is worth noting that this model is also extendable to the n-dimensional case ($n > 3$) by simply adding one new orientation in set O and one set of constraints related to the block dimension for each additional dimension. Indeed, Model (7) differs from the model of Martin et al (2019) because it has the third orientation for builds, as well an additional dimension for the sub-patterns.

$$\textbf{Max} \sum_{j \in J} \sum_{i \in I} \sum_{p \in Q_i} p_i z_{jip}, \tag{7a}$$

s.t.

$$\sum_{j \in J} \sum_{p \in Q_i} z_{jip} \leq u_i, \qquad i \in I, \tag{7b}$$

$$\sum_{o \in O} x_{jo} \leq 1, \qquad j \in J, \tag{7c}$$

$$\sum_{i \in I} \sum_{p \in Q_i} z_{1ip} + \sum_{k \in K, 1 < k} y_{1k} \leq 2 \sum_{o \in O} x_{1o}, \tag{7d}$$

$$\sum_{i\in I}\sum_{p\in Q_i} z_{jip} + \sum_{k\in K, j<k} y_{jk} = 2\sum_{o\in O} x_{jo}, \qquad j \in J\backslash\{1\}, \quad (7e)$$

$$\sum_{j\in J, j<k} y_{jk} = \sum_{o\in O} x_{ko}, \qquad k \in J\backslash\{1\}, \quad (7f)$$

$$L_j = (\sum_{i\in I}\sum_{p\in Q_i} l_i z_{jip} + \sum_{k\in J, j<k} L_k y_{jk}) * x_{jh} +$$

$$\max\{(i,p) \in (I \times Q_i) : l_i z_{jip}; k \in J, j < k : L_k y_{jk}\} * (x_{jd} + x_{jv}), \qquad j \in J, \quad (7g)$$

$$W_j = (\sum_{i\in I}\sum_{p\in Q_i} w_i z_{jip} + \sum_{k\in J, j<k} W_k y_{jk}) * x_{jd} +$$

$$\max\{(i,p) \in (I \times Q_i) : w_i z_{jip}; k \in J, j < k : W_k y_{jk}\} * (x_{jh} + x_{jv}), \qquad j \in J, \quad (7h)$$

$$H_j = (\sum_{i\in I}\sum_{p\in Q_i} h_i z_{jip} + \sum_{k\in J, j<k} H_k y_{jk}) * x_{jv} +$$

$$\max\{(i,p) \in (I \times Q_i) : h_i z_{jip}; k \in J, j < k : H_k y_{jk}\} * (x_{jh} + x_{jd}), \qquad j \in J, \quad (7i)$$

$$z_{jip} \in \{0,1\}, \qquad j \in J, i \in I, p \in Q_i, \quad (7j)$$

$$y_{jk} \in \{0,1\}, \qquad j \in J, k \in J, j < k, \quad (7k)$$

$$x_{jo} \in \{0,1\}, \qquad j \in J, o \in O, \quad (7l)$$

$$0 \le L_j \le \bar{L}, \qquad j \in J, \quad (7m)$$

$$0 \le W_j \le \bar{W}, \qquad j \in J, \quad (7n)$$

$$0 \le H_j \le \bar{H}, \qquad j \in J. \quad (7o)$$

The objective function (7a) consists of maximising the total sum of the value of the selected copies of the item types. Constraints (7b) ensure that each item type $i \in I$ is selected up to its maximum number of copies allowed to be manufactured. Constraints (7c) ensure that each block, if any, is a sub-pattern built up from a horizontal, depth or vertical orientation.

Constraints (7d) and (7e) guarantee that if sub-pattern $j \in J$ is built up (i.e. when $\sum_{o\in O} x_{jo} = 1$), then it should contain exactly either two copies of item types, or two sub-patterns, or a copy of one item type and one sub-pattern. Note that in such case the right-hand-side (RHS) of the corresponding constraint becomes 2. In contrast, if block $j \in J$ is not a sub-pattern (i.e. when $\sum_{o\in O} x_{jo} = 0$), then it cannot contain copies of item types nor sub-patterns. Particularly, constraints (7d), (7e) and (7f) also ensure that sub-pattern $j = 1$ is the last one built up, i.e. it is the final cutting pattern solution. Constraint (7d) is the special case of constraints (7e) for sub-pattern $j = 1$, as it must allow a solution formed by just one copy of any item type.

Constraints (7f) guarantee that block $k \in J\backslash\{1\}$ is a sub-pattern if and only if it is contained in a larger sub-pattern $j \in J$, $j < k$. These constraints also limit a block $k \in J\backslash\{1\}$ to be contained at most in one sub-pattern $j \in J$, $j < k$, according to constraints (7c).

Constraints (7g) ensure that variable L_j assumes the size of the block that envelopes the copies of item types and/or sub-patterns that built it up, according to orientation $o \in O$. For instance, if sub-pattern $j \in J$ is a horizontal build (i.e. $x_{jh} = 1$ and $x_{jd} + x_{jv} = 0$), then variable L_j assumes the sum of the lengths of the copies of item types and/or sub-patterns that build such a sub-pattern j. However, if block $j \in J$ is a depth or vertical build (i.e. $x_{jh} = 0$ and $x_{jd} + x_{jv} = 1$), then variable L_j assumes the largest length of the copies of item types and/or sub-patterns that build such a sub-pattern j by using the max$\{.\}$ function. If a block $j \in J$ contains another sub-pattern $k \in J$ (i.e. $y_{jk} = 1$), the length, width and height of such sub-pattern k can be defined as $L_k y_{jk}$, $W_k y_{jk}$ and $H_k y_{jk}$, respectively, which is useful to define the sizes of sub-pattern j. However, if block $j \in J$ does not contain sub-pattern $k \in J$ (i.e. $y_{jk} = 0$), the products of $L_k y_{jk}$, $W_k y_{jk}$ and $H_k y_{jk}$ are all equal to zero. Constraints (7h) and (7i) are similar to constraints (7g) but related to variables W_j and H_j, respectively.

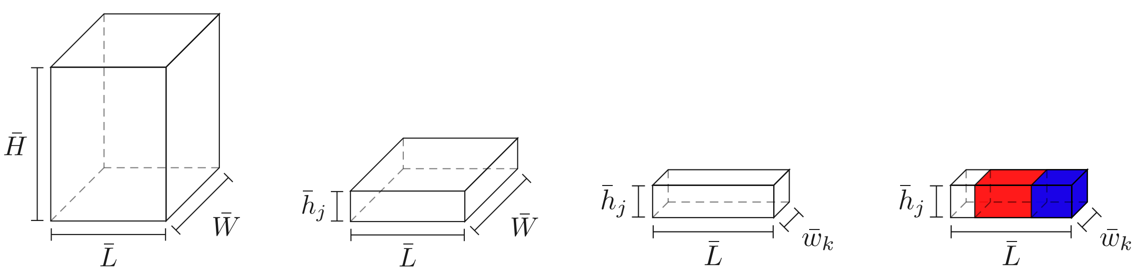
Constraints (7j) to (7o) define the domain of the variables. Similarly to Martin et al (2019), we assume that sub-pattern $j = 1$ is always built up and thus $\sum_{o\in O} x_{1o} = 1$ is also imposed in the model. Model (7) is non-linear because of the product of variables and the use of the max$\{.\}$ function in constraints (7g) to (7i). It does not require the input data (i.e. $\bar{L}$, $\bar{W}$, $\bar{H}$, l_i, w_i and h_i, for each $i \in I$) to be integers.

In Appendix A, we show how to obtain a MILP formulation from Model (7) through well-known linearisation strategies, which enables the use of general-purpose MILP solvers. In Appendix B, we present the pseudo-code of an algorithm for generating the actual item positions, given a solution of Model (7) or of its MILP version.

2.2 ILP model for 3-staged patterns

Some industrial environments limit their cutting operations up to $d \in \mathbb{Z}_+$ guillotine stages to seek shorter production times, to the detriment of lower material utilisation. As the guillotine cutting of blocks may be time-consuming, we propose in this section an ILP formulation that strictly considers 3-staged cutting patterns, mainly motivated by the models of Lodi and Monaci (2003), Yanasse and Morabito (2006), Yanasse and Morabito (2008), Silva et al (2010) and Macedo et al (2010) for types of for two-dimensional guillotine cutting problems.

Firstly, we propose a model that considers vertical cuts at the first stage (i.e. parallel to axis xy), depth cuts at the second stage (i.e. parallel to axis xz), and horizontal cuts at the third stage (i.e. parallel to axis yz). Next, we discuss how to adapt the model to other sequences of guillotine cuts. Fig. 3 depicts the cuts of the vertical-depth-horizontal sequence; the coloured blocks in Fig. 3d are copies of item types, while the white volume is waste. Hence, from object $\bar{L} \times \bar{W} \times \bar{H}$, the cuts of the first stage generate sub-objects of dimensions $\bar{L} \times \bar{W} \times \bar{h}_j$. The cuts of the second stage generate strips of dimensions $\bar{L} \times \bar{w}_k \times \bar{h}_j$. At the last stage, the copies of item types are cut from these strips. Let $J = \{\bar{h}_1, \bar{h}_2, \ldots, \bar{h}_{|J|}\}$ be the set of all different heights (and their copies) of item types $i \in I$, and let $K = \{\bar{w}_1, \bar{w}_2, \ldots, \bar{w}_{|K|}\}$ be the set of all different widths (and their copies) of item types $i \in I$. Let $I(j,k) = \{i \in I \mid h_i \leq \bar{h}_j,\ w_i \leq \bar{w}_k\}$ be the sub-set of the item types capable of fitting in strip $\bar{L} \times \bar{w}_k \times \bar{h}_j$. The number of times that different height $\bar{h}_j$ is duplicated in set J is given by $\min\{\sum_{i \in I \ | h_i = \bar{h}_j} u_i; \lfloor \bar{H}/\bar{h}_j \rfloor\}$, while the number of times that different width $\bar{w}_k$ is duplicated in set K is given by $\min\{\sum_{i \in I \ | w_i = \bar{w}_k} u_i; \lfloor \bar{W}/\bar{w}_k \rfloor\}$.



(a) Object. (b) 1st stg. (c) 2nd stg. (d) 3rd stg.

Fig. 3: Three-staged pattern for the C3GCP with vertical-depth-horizontal cuts.

The model has three types of decision variables, which are defined in Equations (8) to (10). Variables x_j are associated with the sub-objects cut at the first stage, and variables y_{jk} with the strips cut at the second stage. Variables z_{jki} point out the number of copies of item type $i \in I(j,k)$ cut from strip $\bar{L} \times \bar{w}_k \times \bar{h}_j$ at the third stage.

$$x_j = \begin{cases} 1, & \text{if sub-object } \bar{L} \times \bar{W} \times \bar{h}_j \text{ is cut,} \\ 0, & \text{otherwise,} \end{cases} \qquad j \in J. \quad (8)$$

$$y_{jk} = \begin{cases} 1, & \text{if strip } \bar{L} \times \bar{w}_k \times \bar{h}_j \text{ is cut,} \\ 0, & \text{otherwise,} \end{cases} \qquad j \in J, k \in K. \quad (9)$$

$$z_{jki} : \text{number of copies of item type } i \text{ cut from strip } \bar{L} \times \bar{w}_k \times \bar{h}_j, \qquad j \in J, k \in K, i \in I(j,k). \quad (10)$$

Having defined all needed variables and parameters, we state an ILP formulation that strictly considers 3-staged cutting patterns for the C3GCP, given by Model (11).

$$\textbf{Max} \sum_{j \in J} \sum_{k \in K} \sum_{i \in I(j,k)} p_i z_{jki}, \quad (11a)$$

s.t.

$$\sum_{j \in J} \bar{h}_j x_j \leq \bar{H}, \quad (11b)$$

$$\sum_{k \in K} \bar{w}_k y_{jk} \leq \bar{W} x_j, \qquad j \in J, \quad (11c)$$

$$\sum_{i \in I(j,k)} l_i z_{jki} \leq \bar{L} y_{jk}, \qquad j \in J, k \in K, \quad (11d)$$

$$x_j \in \{0,1\}, \qquad j \in J, \quad (11e)$$

$$y_{jk} \in \{0,1\}, \qquad j \in J, k \in K, \quad (11f)$$

$$z_{jki} \in \mathbb{Z}_+, \qquad j \in J, k \in K, i \in I(j,k). \quad (11g)$$

The objective function (11a) consists of maximising the total value of the cut items. Constraint (11b) ensures that the sum of heights $\bar{h}_j$ of the sub-objects does not exceed the height of the object. Constraints (11c) ensure that the sum of widths $\bar{w}_k$ of the strips does not exceed the width of sub-object $j \in J$. Constraints (11d) ensure that the sum of lengths l_i of the copies of item types $i \in I$ does not exceed the length of the strip. All these three blocks of constraints are knapsack constraints. Constraints (11e) to (11g) define the domain of the variables.

Notice that a 3-staged cutting pattern for the C3GCP may consider any of the three cut orientations at the first stage. The type of cut of the next stage (second stage) can be one of these three types of cut, but not the one held at the previous stage (first stage). Therefore, they can be: depth or vertical cuts when the first stage held horizontal cuts; horizontal or vertical cuts when the first stage held depth cuts; or, horizontal or depth cuts when the first stage held vertical cuts. Again, the type of cut of the next stage (third stage) can be one of the three types of cut, except the one held at the previous stage (second stage). In this way, there are three types of cuts for the first stage, two for the second stage, and two for the third stage, in a total of 12 (=3×2×2) permutations. Therefore, Model (11) allied to a general-purpose solver should be run 12 times with these different permutations to find the best 3-staged cutting pattern for the C3GCP. Fig. 4 shows how an object can be cut according to these 12 guillotine sequences.

To adapt Model (11) to any guillotine sequence, as depicted in Fig. 4, set J can be understood as the set of all different dimensions (and their copies) regarding the first stage, and set K as the set of all different dimensions (and their copies) regarding the second stage. For instance, sequence horizontal-vertical-depth considers set J as the set of all different lengths (and their copies) and set K as the set of all different heights (and their copies). In this way, the constants in the RHS of constraints (11b)

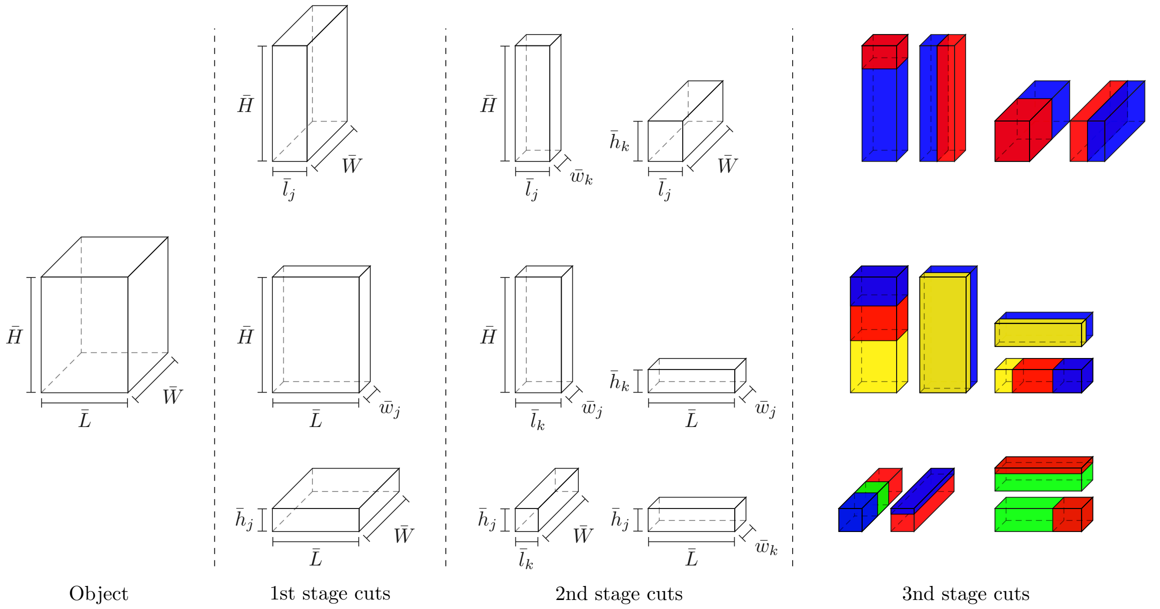


Fig. 4: Guillotine sequences of 3-staged patterns for the C3GCP.

to (11d) should be $\bar{L}$, $\bar{H}$ and $\bar{W}$, respectively, and w_i should replace l_i as the coefficient of variables z_{jki} in constraints (11d). Additionally, we note that this framework, based on 12 different executions of Model (11), cannot obtain all kinds of 3-staged cutting patterns for the C3GCP. Particularly, the six guillotine sequences with an identical type of cut in the first and third stages may require in set J segments that represent integer linear combinations of the corresponding dimension under analysis, i.e. the discretisation of normal sets (Herz, 1972; Christofides and Whitlock, 1977). To illustrate one of these cases, Fig. 5 depicts a cutting pattern with vertical-horizontal-vertical cuts, where the height $\bar{h}_j$ of the sub-object $\bar{L} \times \bar{W} \times \bar{h}_j$ is the sum of the height of two copies of small items, represented by the red and the blue boxes. We further discuss this topic in Section 4.

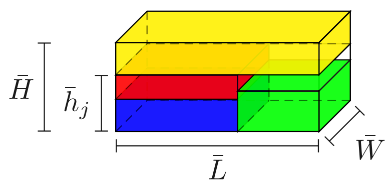


Fig. 5: Cutting pattern with vertical-horizontal-vertical cuts, where height $\bar{h}_j$ is the sum of the heights of the red and the blue small item types.

3 A bottom-up algorithm for the C3GCP

In this section, we propose a solution method by extending Wang (1983)'s bottom-up search tree algorithm to the C3GCP. This algorithm generates new partial solutions by successively merging copies of item types and/or partial solutions generated in previous iterations of the tree. To avoid the explosive growth of partial solutions, the algorithm rejects partial solutions with waste greater than or equal to a threshold, which is an aspiration level for accepting new partial solutions. The author proposed two algorithms that differ with respect to such aspiration levels. Wang's first algorithm rejects partial solutions with inner waste greater than a percentage of the area of the large object, while Wang's second algorithm rejects partial solutions with inner waste greater than a percentage of the area of the partial solution itself. The author also proved optimality conditions for both algorithms in order to declare certificates of optimality or estimate how close a solution is to the optimal solution. It should be noted that these algorithms are tailored for non-weighted instances, i.e. instances in which the value p_i of each item type $i \in I$ is its area $l_i w_i$, in the original paper, or its volume $l_i w_i h_i$, in the current paper.

Let the following definitions be introduced:

(i) Parameter β, $0 \leq \beta \leq 1$, is used for defining the maximum acceptable waste percentage in a partial solution;
(ii) Set $F^{(k)}$ contains all partial solutions generated in iteration k;
(iii) Set $L^{(k)}$ contains all partial solutions generated up to iteration k;
(iv) A partial solution T is characterised by its size $L_T \times W_T \times H_T$, waste T_t and the number of copies for each item type $i \in I$;
(v) The total waste of a partial solution T is $T_t = T_i + T_o$, where its inner waste is $T_i = L_T W_T H_T - V$ (V is the volume of all its copies of item types), and its outer waste is $T_o = \bar{L}\bar{W}\bar{H} - L_T W_T H_T$, i.e. when placing such partial solution into the object, as depicted by the blank volumes in Fig. 6.

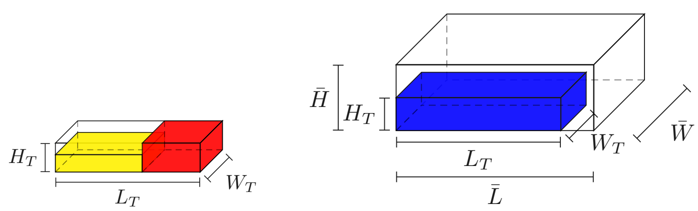
We state a bottom-up algorithm for the C3GCP, given by Algorithm 1. This algorithm is a straightforward adaptation of Wang's first algorithm to the three-dimensional case since the criterion for accepting partial solutions is based on a percentage of the volume of the large object.

The value of parameter β is critical for the trade-off between obtaining quality solutions and reasonable processing times. When its value increases, the solution space also increases, which contributes to obtaining solutions with higher utilisation rates, to the detriment of processing times. In contrast, when its value decreases, the utilisation rates and processing times also tend to decrease. Theorem 1 adapts the result originally proposed by Wang for the two-dimensional case to the C3GCP. It provides a sufficient condition for the optimality of the best solution T obtained by the algorithm, linking the value of parameter β used for generating such partial solution and its total waste T_t.

Theorem 1 *If waste T_t of the best solution T obtained at the end of the Algorithm 1, generated with fixed β, satisfies $T_t \leq \beta\bar{L}\bar{W}\bar{H}$, then T is an optimal solution.*

Proof Let $\mathcal{O}$ be the set of optimal solutions, i.e., solutions with minimum total waste T_t^* that satisfy the constraints of the C3GCP, i.e., the geometric, guillotine and production constraints. Let β^* be the smallest value of β that allows the creation of at least one solution of $\mathcal{O}$. Assume $T_t \leq \beta\bar{L}\bar{W}\bar{H}$. If $T \notin \mathcal{O}$, then $\beta < \beta^*$, and hence $T_t^* \geq \beta^*\bar{L}\bar{W}\bar{H} > \beta\bar{L}\bar{W}\bar{H} \geq T_t$, contradicting the optimality of T_t^*. Thus $T \in \mathcal{O}$.

The time limit was also considered as stop criterion for Algorithm 1 (loop of lines 3-5) in the computational experiments of Section 4. When such time limit is reached, the certificate of optimality



(a) Inner waste. (b) Outer waste.

Fig. 6: Placing a partial solution T into the large object.

Algorithm 1: Bottom-up algorithm for the C3GCP.

Input: Instance.

1 Choose a value for parameter β, $0 \leq \beta \leq 1$;
2 Define set $F^{(0)} \leftarrow \{1, 2, \ldots, m\}$, set $L^{(0)} \leftarrow F^{(0)}$, and $k \leftarrow 0$;
3 **while** $F^{(k)}$ *is not empty* **do**
4 $\quad k \leftarrow k + 1$;
5 $\quad$ Generate set $F^{(k)}$ of all partial solutions T by combining all elements of $F^{(k-1)}$ with all elements of $L^{(k-1)}$ satisfying:
$\quad\quad$ (i) T is formed by horizontal, depth or vertical builds of such elements;
$\quad\quad$ (ii) waste T_t does not exceed $\beta \bar{L}\bar{W}\bar{H}$;
$\quad\quad$ (iii) the number of copies of each item type $i \in I$ does not exceed u_i;
$\quad L^{(k)} \leftarrow L^{(k-1)} \cup F^{(k)}$;
6 $M \leftarrow k - 1$;
7 Choose the partial solution of set $L^{(M)}$ that has the smallest waste T_t;

Output: A solution for the C3GCP.

of Theorem 1 does not hold. In what follows, we present three features for Algorithm 1. Two of them are based on the approaches of Vasko (1989) and Oliveira and Ferreira (1990), which seek to accelerate the convergence of the algorithm. In Section 3.3, we show how to strictly consider 3-staged patterns in Algorithm 1.

3.1 Completeness of partial solutions

A partial solution that cannot be used in a horizontal build (step 5(i) of Algorithm 1) because using it with any other partial solution (for instance, the item type with the smallest length) would exceed the object's length, is said to be "horizontally complete". Similarly, a partial solution that cannot be used in a depth (resp., vertical) build because using it with any other partial solution would exceed the width (resp., height) of the object, is said to be "in depth complete" (resp., "vertically complete"). The enhancements of this feature are summarised below:

- A horizontally (resp., in depth, vertical) complete partial solution is not considered in future horizontal (resp., depth, vertical) builds in the building process;
- After a partial solution is generated, it is checked for horizontal, in depth and vertical completeness. When a partial solution T is horizontally complete, its length L_T is set to be the object's length $\bar{L}$, and its inner waste T_i is adjusted by adding $(\bar{L} - L_T)W_T H_T$. A similar behaviour holds in in depth and vertically complete partial solutions for the corresponding orientations;
- For a horizontally, in depth and vertically complete partial solution T, if $T_t < \beta\bar{L}\bar{W}\bar{H}$, then β is updated to $\beta = T_t/(\bar{L}\bar{W}\bar{H})$. All partial solutions in $F^{(k)}$ with waste greater than this new updated value of β multiplied by the volume of the object can be deleted, without loss of optimality;

Each item type $i \in I$ is checked for horizontal, in depth and vertical completeness when defining set $F^{(0)}$, with adjustments in its inner waste, if any. For a horizontally, in depth and vertically complete item type, if $T_t > \beta\bar{L}\bar{W}\bar{H}$, it is not included in $F^{(0)}$, otherwise, β is updated. It should be noted that if a partial solution T is said to be horizontally, in depth and vertically complete, then its outer waste T_o is zero.

3.2 Estimation of the outer waste T_o

The aspiration level proposed by Wang only considers the inner waste of partial solutions. Oliveira and Ferreira (1990) proposed an adaptation of the aspiration level of Wang's first algorithm by considering an estimation of the outer waste T_o of a partial solution T, which should be rejected if $T_t = T_i + T_o >$

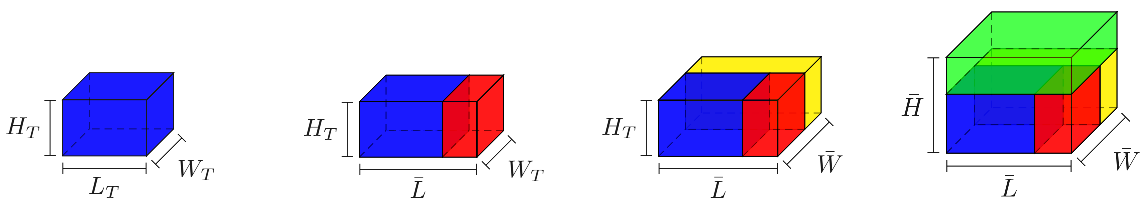
$\beta \bar{L}\bar{W}\bar{H}$. In other words, their aspiration level tries to more quickly identify partial solutions that will not satisfy the optimality condition. In this way, T_o corresponds to the best (minimum) outer waste of a larger partial solution that contains partial solution T in its bottom-left-front corner. The authors estimated T_o by making use of Gilmore and Gomory (1965)'s Dynamic Programming (DP) formulation of the unconstrained version of the C2GCP. In what follows, we heuristically estimate T_o by making use of De Queiroz et al (2012)'s DP formulation of the unconstrained version of the C3GCP.

Fig. 7a depicts an example of a partial solution T of size $L_T \times W_T \times H_T$. We consider it as a sub-part of a larger solution formed by three additional builds, i.e. one build for each dimension. Particularly, Fig. 7d shows a partial solution in which partial solution T is taken in a horizontal-depth-vertical (H-D-V) sequence. The other five possible sequences to fill the volume of the object are: horizontal-vertical-depth, depth-horizontal-vertical, depth-vertical-horizontal, vertical-horizontal-depth and vertical-depth-horizontal. The value of outer waste T_o can be estimated with respect to a H-D-V sequence as the sum of the volumes of the red, yellow and green partial solutions of Fig. 7d minus their corresponding optimal values, obtained from De Queiroz et al (2012)'s DP formulation of the unconstrained version of the C3GCP. We propose to consider the minimum value of the six sequences of permutations, depicted in Fig. 8, as the estimation for the outer waste T_o for a partial solution T. The red, yellow and green partial solutions always fill the first, second and third dimensions, respectively. It should be noted that this feature is with loss of optimality because partial solution T could be considered in cutting patterns with more than three additional builds, which could lead to smaller values of T_o. However, it can drastically reduce the solution space in some scenarios, which contributes to the convergence of the algorithm, as shown in the computational results of Section 4.

3.3 Bottom-up algorithm for 3-staged patterns

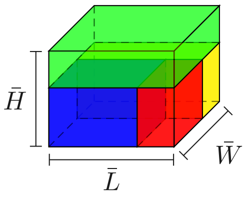
The literature on d-staged cutting patterns usually relies on the top-down approach, in which the large object is cut down to the small items after the generation of piles and/or strips. Fig. 4 highlights this approach in Model (11). In contrast, d-staged patterns are rarely addressed in bottom-up approaches, which grow solutions from the small items up to the size of the large object. To the best of our knowledge, Martin et al (2019) were the first to address d-staged patterns in the bottom-up approach by proposing an additional set of constraints in their models. Those constraints are based on counting the minimum number of guillotine stages for each sub-pattern. This idea does not seem to be straightforwardly adaptable to Algorithm 1 – both approaches are based on a tree structure –, because the C3GCP has three possible orientations (i.e. h, d and v) which generate several types of sub-patterns and complicate the counting.

Therefore, we adapt Algorithm 1 to strictly consider 3-staged patterns by pre-enumerating through inspection all possible types of d-patterns, with $d \leq 3$, generated from horizontal, in depth and vertical builds. There are: (i) one 0-staged pattern formed by just one copy of item type (i.e. O); (ii) three types of 1-staged patterns formed after a build from two small items (i.e. H, D and V); (iii) six types of 2-staged patterns (i.e. H-D, H-V, D-H, D-V, V-D and V-H); and (iv) twelve types of 3-staged patterns (i.e. H-D-H, H-D-V, H-V-D, H-V-H, D-H-D, D-H-V, D-V-D, D-V-H, V-D-H, V-D-V, V-H-D

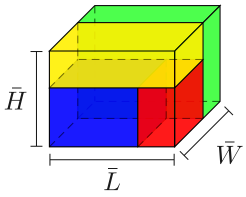


(a) Partial solution. (b) Horizontal build. (c) Depth build. (d) Vertical build.

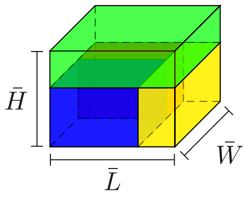
Fig. 7: Estimating outer waste T_o in a horizontal-depth-vertical sequence.



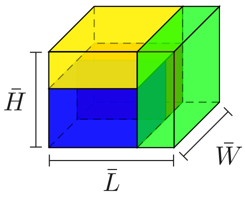
(a) H-D-V.



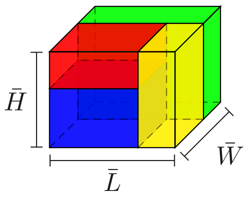
(b) H-V-D.



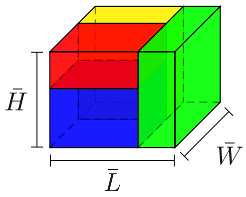
(c) D-H-V.



(d) D-V-H.



(e) V-H-D.



(f) V-D-H.

Fig. 8: Six permutations for estimating outer waste T_o.

and V-H-V). These 22 (=1+3+6+12) types of sub-pattern were analysed across each other for each possible build (i.e. h, d and v). For instance, the horizontal build of the H-D and H-D-H sub-pattern types generates the H-D-H sub-pattern type; however, the in depth and the vertical builds of these two sub-patterns are forbidden, because each of the resulting sub-patterns would require more than three guillotine stages. The inspection process generated a look-up table which is used to verify whether or not a build is allowed to occur as an additional criterion in line 5 of Algorithm 1, and which is the type of sub-pattern generated when it is possible. It should be noted that each partial solution is now additionally characterised by the information about the type of sub-pattern. These look-up tables are described in Appendix C.

4 Computational experiments

Computational experiments were performed to assess the solution quality and processing times of the proposed models and solution method. The linear version of Model (7) of Section 2.1 for non-staged cutting patterns, reformulated as a linear program in Appendix A, with the corresponding valid inequalities, is hereafter called the `Non-staged Model`. Model (11) of Section 2.2 for 3-staged cutting patterns is hereafter called the `3-staged Model`, while the bottom-up algorithm with enhanced completeness of partial solutions of Section 3.1 is hereafter called the `Bottom-up Alg`. As discussed previously, the `3-staged Model` seeks to propose the best solution among twelve guillotine sequences. `Bottom-up Alg+TE` denotes when the trim estimation of the outer volume is additionally considered. When such algorithms strictly consider 3-staged patterns, as discussed in Section 3.3, they are denoted by `3-staged Bottom-up Alg` and `3-staged Bottom-up Alg+TE`, respectively.

This section is divided into two parts. Each of them considers a set of benchmark instances from the literature. Set A comprises the 12 instances of De Queiroz et al (2012), while set B comprises the 60 instances of Egeblad and Pisinger (2009). These instances are available at `http://www.loco.ic.unicamp.br/instances/3duk.html` and `http://hjemmesider.diku.dk/~pisinger/codes.html`. They are further detailed at the beginning of each section.

For benchmark purposes, we compare the proposed approaches with the DP formulations of De Queiroz et al (2012) for non-staged and 3-staged patterns in the unconstrained version of the C3GCP (i.e. when $u_i \geq \lfloor \bar{L}/l_i \rfloor \lfloor \bar{W}/w_i \rfloor \lfloor \bar{H}/h_i \rfloor$ for each $i \in I$) in Section 4.1. To the best of our knowledge, Morabito and Arenales (1994) have the only approach in the literature that deals with the C3GCP in the constrained guillotine case. The authors considered the guillotine requirement when proposing a solution method for the CLP. However, they reported computational results with just one instance in the constrained case. Therefore, we compare our approaches among themselves in the constrained case (i.e. when $u_i \leq \lfloor \bar{L}/l_i \rfloor \lfloor \bar{W}/w_i \rfloor \lfloor \bar{H}/h_i \rfloor$ for each $i \in I$, with the inequality holding for at least one item type) in Section 4.2. All approaches, namely the `Non-staged Model`, `3-staged Model`, `Bottom-up Alg`, `Bottom-up Alg+TE`, `3-staged Bottom-up Alg`, `3-staged Bottom-up Alg+TE` and the two DP formulations of De Queiroz et al (2012), were coded in C++, using the Concert Studio library on top of the IBM CPLEX Optimization Studio, v12.8, if necessary. The experiments were carried out on a PC with an Intel Xeon E5-2680v2 (2.8 GHz), limited to 20 threads, 32 GB of RAM, running the CentOS Linux 7.2.1511 Operating System. We limited each execution of the solver to 3,600 seconds for the `Non-staged Model` and `3-staged Model`. The time limit of the four versions of the bottom-up algorithm (i.e. the `Bottom-up Alg`, `Bottom-up Alg+TE`, `3-staged Bottom-up Alg` and `3-staged Bottom-up Alg+TE`) was 600 seconds by seeking to avoid memory breakdowns when optimality is not proven.

It is worth recalling that the `Non-staged Model` requires the definition of upper bounds $\bar{n}$, UB_1 and UB_2. Each of these parameters was obtained by solving one-dimensional knapsack problems. Similarly to Martin et al (2019), we solved Model (12) using the CPLEX to determine parameter $\bar{n} = \sum_{i \in I} q_i^*$, where q_i^* is the optimal solution obtained for this model.

$$\bar{n} = \mathbf{Max} \sum_{i \in I} q_i, \mathbf{s.t.} \sum_{i \in I} (l_i w_i h_i) q_i \leq \bar{L}\bar{W}\bar{H}, q_i \leq u_i, \qquad q_i \in \mathbb{Z}_+, i \in I. \tag{12}$$

For parameters UB_1 and UB_2, we solved similar models which consider the item types' volumes or values as coefficients in the objective function, respectively (i.e. $UB_1 = \sum_{i \in I} l_i w_i h_i q_i^*$ and $UB_2 = \sum_{i \in I} v_i q_i^*$). In fact, for each of these three parameters, we solved several one-dimensional knapsack problems in the context of a conservative scale function, as considered by Egeblad and Pisinger (2009), by seeking to obtain tight upper bounds. These one-dimensional knapsack problems differ with respect to the dimensions of the item types, which are redefined according to a predefined conservative scale function. The premise is that the minimum optimal value among these one-dimensional knapsack problems is a valid upper bound to the original instance (Fekete and Schepers, 2004).

The initial value of parameter β provided to the four versions of the bottom-up algorithm (line 1 of Algorithm 1) was generated by solving the `3-staged Model` with the CPLEX. We considered only the two permutations (out of twelve) with vertical first stage cuts, as depicted in Fig. 3b. For each run of the solver, the time limit was set to 60 seconds.

4.1 Results of instances of set A

De Queiroz et al (2012) proposed the *gcut_3d* instances for the unconstrained version of the C3GCP by adapting the 12 *gcut* instances of Beasley (1985), which were initially proposed for the two-dimensional case. Firstly, Beasley uniformly sampled the length and width of these instances in the intervals $[0.25\bar{L}, 0.75\bar{L}]$ and $[0.25\bar{W}, 0.75\bar{W}]$, respectively. Then, those authors added the third dimension for each item type by randomly choosing it from the dimensions already used for the item types. These instances are non-weighted, i.e. the value of each item type is its volume ($p_i = l_i w_i h_i$ for $i \in I$). For each of them, the dimensions of the object ($\bar{L}, \bar{W} and \bar{H}$), number of item types (m) and total number of items ($n = \sum_{i \in I} u_i$) are described in Table 1. We generated the *gcut_3d_cons* instances by adapting the *gcut_3d* instances to the most possible constrained case (i.e. when $u_i = 1$ for each $i \in I$); such instances are also considered in what follows. The percentage ratio of the sum of the volume of all copies of item types over the object's volume, i.e. $\sum_{i \in I}(l_i w_i h_i u_i)/(\bar{L} \times \bar{W} \times \bar{H}) * 100$, is reported in column V_{cons} for the corresponding constrained case. This ratio indicates that the selection decision associated with the C3GCP remains even under the most possible constrained case for these instances.

Table 2 shows the results of the approaches proposed in this study and the benchmark approach for non-staged patterns, in the unconstrained case. For each approach and instance, the table reports the material utilisation as a percentage (column sol[%]), and the processing time in seconds (column

Table 1: Characteristics of instances of set A.

Instances	$\bar{L}$	$\bar{W}$	$\bar{H}$	m	n	V_{cons}[%]
gcut1_3d	250	250	250	10	57	117.64
gcut2_3d	250	250	250	20	113	198.66
gcut3_3d	250	250	250	30	207	254.21
gcut4_3d	250	250	250	50	274	572.18
gcut5_3d	500	500	500	10	60	104.04
gcut6_3d	500	500	500	20	82	270.97
gcut7_3d	500	500	500	30	132	400.62
gcut8_3d	500	500	500	50	270	524.15
gcut9_3d	1000	1000	1000	10	82	87.75
gcut10_3d	1000	1000	1000	20	75	307.60
gcut11_3d	1000	1000	1000	30	133	316.77
gcut12_3d	1000	1000	1000	50	251	587.08

time[s]). For the `Non-staged Model`, the table also reports the number of variables (column var), number of constraints (column cons) and the gap as a percentage (column gap[%]). The gap is calculated as $(UB - sol)/(UB + 10^{-10}) * 100$, where UB is the best upper bound and sol is the value of the best integer solution, both found by the solver using the formulation. Upper bound $\bar{n}$ is also reported for the `Non-staged Model`. For the `Bottom-up Alg` and `Bottom-up Alg+TE`, the table also reports the number of partial solutions generated (column nGer), inserted in sets $L^{(k)}$ (column nIns) and in set $L^{(M)}$ (column nEnd). We use "tl" to denote when the time limit was reached. For each approach, the optimal solutions found are highlighted in bold, even when optimality is not proven by such approach.

Firstly, we analyse the results of the DP formulation of De Queiroz et al (2012) for non-staged unconstrained patterns. Optimality was proven by the algorithm for all instances in set A in a few tenths of a second. As proposed by the authors, we used the discretisation sets of reduced raster points, which, according to them, are without loss of generality for the unconstrained case. These instances are characterised by item types with large dimensions with respect to the sizes of the object, which lead to discretisation sets with fewer elements, and that in turns contribute to the fast convergence of the algorithm. These results point out the effectiveness of DP formulations for unconstrained guillotine cutting problems (Cintra et al, 2008; De Queiroz et al, 2012).

The results of the `Non-staged Model` in Table 2 show that the number of variables and constraints is relatively small. However, the time limit was reached for all these instances, with an average gap of 15.73%. Particularly, the optimal solution of instance gcut10_3d was found, but not proven within the time limit (gap of 14.84%). These gaps are mainly related to the weak LP-relaxation of the model, and the upper bound values are identical to the values provided by parameter UB_1. For instance, if we consider the values of the optimal solution obtained by the DP formulation instead of the LP-relaxation when calculating the gaps, the average (optimality) gap would be 5.11%.

The `Bottom-up Alg` was able to find solutions and prove optimality in all instances of set A with an average processing time of 86.05 seconds. The number of partial solutions generated (column nGer) seems to quickly increase when the number of item types also increase. However, the number of partial solutions included in sets $L^{(k)}$ (column nIns) is much smaller due to the aspiration level of the algorithm, which rejects most of them. Particularly, the number of partial solutions in set $L^{(M)}$ (column nEnd) is even smaller, because no duplicate partial solutions are allowed, and parameter β may be improved during the search by the completeness of partial solutions. The analysis of the results of the heuristic approach `Bottom-up Alg+TE` shows that this version of the algorithm was able to find solutions with an average optimality gap of 0.23% and with a time improvement of 67.20% over the `Bottom-up Alg`. It should be noted that optimal solutions were found in 8 out of 12 instances. Moreover, the number of partial solutions handled was drastically reduced with respect to the `Bottom-up Alg` by using the estimation of the outer volume of the partial solutions. For these unconstrained instances, the CPLEX with the `3-staged Model` provided the `Bottom-up Alg` and `Bottom-up Alg+TE` with an average initial value for parameter β of 0.1344 (i.e. with an average material utilisation rate of 86.56%).

Table 3 shows the results for non-staged constrained patterns with gcut_3d instances adapted to the most possible constrained case (i.e. when $u_i = 1$ for $i \in I$). For these experiments, we do not report results for the benchmark approach, given that it only addresses the unconstrained case, and its extension to the constrained case is not straightforward. The `Bottom-up Alg` was able to find and prove optimality in all adapted instances of set A with an average processing time of 69.07 seconds. The variations in the number of partial solutions in columns nGer, nIns and nEnd with respect to those of Table 2 are mainly related to: (i) the number of copies of item types available (i.e. $n = \sum_{i \in I} u_i$); and, (ii) the value of parameter β, which is larger in the constrained case. For these constrained instances, the CPLEX with the `3-staged Model` provided the `Bottom-up Alg` and `Bottom-up Alg+TE` with an average initial value for parameter β of 0.2656 (i.e. with an average material utilisation rate of 73.44%). The analysis of the results of the heuristic approach `Bottom-up Alg+TE` shows that this version of the algorithm was able to find the optimal solutions for all adapted instances, with a time improvement of 84.72% over the `Bottom-up Alg`. Regarding the model, the CPLEX with the `Non-staged Model` was able to prove optimality for instances cons_gcut1_3d and cons_gcut5_3d. For three additional instances (cons_gcut6_3d, cons_gcut9_3d and cons_gcut10_3d), optimal solutions were found but not proven within the time limit. The average optimality gap was 3.18%, and the average processing time was 3,085.49 seconds. As before, the gaps shown in Table 3 are mainly related to the weak LP-relaxation of the model. For instance, the value of the solutions of instances cons_gcut6_3d and cons_gcut10_3d are indeed optimal, but their estimated gaps are over 20%.

Tables 4 and 5 show the results of the approaches proposed in this study and the benchmark approach for 3-staged patterns in the unconstrained and constrained cases, respectively. Similarly to the results of Table 2, the DP formulation of De Queiroz et al (2012) for 3-staged patterns was able to prove optimality for all instances in set A in the unconstrained case in a few tenths of a second. For the `3-staged Model`, the values in columns var, cons, sol[%] and gap[%] are related to the best solution among the twelve guillotine sequences, whilst the corresponding total processing time is shown in column time[s]. The total time limit was 43,200 (=12*3,600) seconds for the `3-staged Model`. In Tables 4 and 5, the results of the `3-staged Model` show that the numbers of variables and constraints are also relatively small, but increase quickly when the number of item types also increases. Moreover, for each instance, the optimality was proven for all twelve guillotine sequences, with an average processing time of 476.84 seconds in Table 4 and 86.90 seconds in Table 5.

The `3-staged Bottom-up Alg` was able to find solutions and prove optimality in 11 out of 12 instances with an average processing time of 129.42 seconds and 169.64 seconds in Tables 4 and 5, respectively. The optimality of instances gcut03_3d and cons_gcut04_3d was not proven by this version of the bottom-up algorithm; however, when a time limit of 1,800 seconds (instead of 600 seconds) was considered, such solutions were proven to be optimal. As expected, the number of partial solutions generated for 3-staged patterns is smaller than what was obtained in corresponding cases for non-staged patterns. In contrast, the average processing times are slightly longer, mainly because the average material utilisation rate (i.e. sol[%]) is lower for 3-staged patterns than for non-staged patterns, which results in larger values of parameter β during the search and thus in larger feasible spaces. Regarding the `3-staged Bottom-up Alg+TE`, this version of the bottom-up algorithm was able to find optimal solutions in 11 out of 12 instances in the unconstrained case, and in all instances in the constrained case, with a time improvement over the `3-staged Bottom-up Alg` of 78.41% and 94.03%, respectively. It should be noted that the value of the optimal solution of 3-staged and non-staged patterns was equal in three instances in the unconstrained case (gcut01_3d, gcut10_3d and gcut11_3d), and in four instances in the constrained case (cons_gcut01_3d, cons_gcut05_3d, cons_gcut10_3d and cons_gcut12_3d).

In summary, the main results of this section are:

(i) the DP formulations of De Queiroz et al (2012), which were tailored to the unconstrained case, outperformed the proposed approaches in these unconstrained problems;
(ii) the `Bottom-up Alg` outperformed the `Non-staged Model` both in solution quality and processing time in the unconstrained and constrained cases. Particularly, the weak LP-relaxation of the `Non-staged Model` harms its search, but mainly the proof of optimality;

(iii) the `3-staged Model` performed better than the `3-staged Bottom-up Alg` with respect to processing times in the constrained case, but the opposite happened in the unconstrained case; and
(iv) the trim estimation in the `Bottom-up Alg+TE` and `3-staged Bottom-up Alg+TE` significantly accelerated the convergence of the algorithms.

We highlight the quality of the initial feasible solutions generated by the CPLEX with the `3-staged Model` for the four versions of the bottom-up algorithm. For non-staged patterns, the average optimality gap of these initial solutions was smaller than 3.50%, while the average optimality gap was smaller than 1.75% for 3-staged patterns. For both cases, the average processing time was lower than 25 seconds.

Table 2: Results of instances of set A for non-staged patterns in the unconstrained case.

	Non-staged Model						Bottom-up Alg					Bottom-up Alg+TE					De Queiroz et al (2012)	
Instance	$\bar{n}$	var	cons	sol[%]	gap[%]	time[s]	sol[%]	time[s]	nGer	nIns	nEnd	sol[%]	time[s]	nGer	nIns	nEnd	sol[%]	time[s]
gcut01_3d.txt	18	952	2,664	77.57	22.43	tl	**80.57**	0.55	68,265	3,737	929	**80.57**	0.52	2,714	193	20	**80.57**	0.10
gcut02_3d.txt	27	2,444	7,042	80.93	19.07	tl	**84.88**	113.29	26,303,103	127,381	36,213	**84.88**	9.80	32,146	561	35	**84.88**	0.10
gcut03_3d.txt	27	3,016	8,768	87.49	12.51	tl	**92.48**	519.92	166,244,800	165,048	15,437	**92.48**	42.63	47,178	550	48	**92.48**	0.13
gcut04_3d.txt	33	5,184	15,220	89.00	11.00	tl	**95.43**	98.87	22,760,214	25,908	7,670	95.04	69.58	5,231	99	62	**95.43**	0.37
gcut05_3d.txt	21	1,280	3,612	81.77	18.23	tl	**84.34**	0.66	146,386	3,137	1,446	**84.34**	0.49	683	53	24	**84.34**	0.10
gcut06_3d.txt	22	1,743	4,999	78.96	21.04	tl	**84.84**	4.05	282,406	9,543	2,284	**84.84**	1.92	9,994	269	27	**84.84**	0.10
gcut07_3d.txt	22	2,100	6,080	79.98	20.02	tl	**88.11**	15.95	4,352,043	19,101	6,057	**88.11**	12.75	9,605	226	41	**88.11**	0.10
gcut08_3d.txt	22	3,003	8,809	87.90	12.10	tl	**93.20**	112.96	26,912,145	38,971	15,691	92.89	69.67	5,974	112	69	**93.20**	0.31
gcut09_3d.txt	18	969	2,715	82.45	17.55	tl	**93.16**	0.89	137,163	2,249	746	**93.16**	1.80	1,033	71	41	**93.16**	0.10
gcut10_3d.txt	12	715	2,035	**85.16**	14.84	tl	**85.16**	1.17	22,615	1,196	579	**85.16**	3.80	439	34	30	**85.16**	0.10
gcut11_3d.txt	18	1,598	4,622	90.78	9.22	tl	**91.44**	6.16	1,375,346	12,269	5,380	90.78	5.71	1,453	51	48	**91.44**	0.10
gcut12_3d.txt	27	3,952	11,596	89.24	10.76	tl	**92.65**	158.08	24,553,935	33,038	12,017	91.50	120.01	4,007	99	90	**92.65**	0.28
Averages	22.25	2,246.33	6,513.50	84.27	15.73	tl	88.86	86.05	22,763,201.75	36,798.17	8,704.08	88.65	28.22	10,038.08	193.17	44.58	88.86	0.16

Table 3: Results of instances of set A for non-staged patterns in the constrained case.

	Non-staged Model						Bottom-up Alg					Bottom-up Alg+TE				
Instance	$\bar{n}$	var	cons	sol[%]	gap[%]	time[s]	sol[%]	time[s]	nGer	nIns	nEnd	sol[%]	time[s]	nGer	nIns	nEnd
cons_gcut01_3d.txt	8	196	516	**62.73**	0.00	545.57	**62.73**	0.33	2,006	625	328	**62.73**	0.61	1,001	205	139
cons_gcut02_3d.txt	13	576	1,606	72.90	27.10	tl	**73.83**	12.74	523,935	36,729	16,197	**73.83**	17.24	117,126	3,999	2,226
cons_gcut03_3d.txt	17	1,056	3,008	77.32	22.68	tl	**87.09**	212.29	26,488,139	117,588	24,315	**87.09**	21.64	572,098	3,168	318
cons_gcut04_3d.txt	18	1,496	4,336	84.00	16.00	tl	**90.86**	263.36	65,271,546	90,607	18,641	**90.86**	13.44	1,113,256	3,523	293
cons_gcut05_3d.txt	8	196	516	**56.86**	0.00	480.26	**56.86**	1.34	3,337	1,609	711	**56.86**	0.47	2,974	818	483
cons_gcut06_3d.txt	10	378	1,048	**76.28**	23.72	tl	**76.28**	13.87	13,778	2,764	1,045	**76.28**	9.22	8,158	655	78
cons_gcut07_3d.txt	13	696	1,976	80.59	19.41	tl	**80.88**	19.28	612,818	11,305	7,527	**80.88**	10.90	23,364	485	394
cons_gcut08_3d.txt	17	1,376	3,988	80.28	19.72	tl	**89.01**	169.89	40,316,874	92,327	21,816	**89.01**	23.24	1,665,605	5,458	391
cons_gcut09_3d.txt	9	240	636	**64.14**	8.72	tl	**64.14**	0.57	20,993	6,321	1,606	**64.14**	0.47	16,943	4,044	1,087
cons_gcut10_3d.txt	9	320	886	**71.63**	28.37	tl	**71.63**	1.24	4,902	1,521	672	**71.63**	1.11	2,911	347	100
cons_gcut11_3d.txt	14	780	2,216	74.90	25.10	tl	**78.21**	69.71	2,534,631	128,253	38,890	**78.21**	8.42	495,506	11,535	2,589
cons_gcut12_3d.txt	16	1,260	3,652	84.78	15.22	tl	**87.68**	64.15	9,452,812	31,233	23,326	**87.68**	19.90	21,198	250	243
Averages	12.67	714.17	2,032.00	73.87	17.17	3,085.49	76.60	69.07	12,103,814.25	43,406.83	12,922.83	76.60	10.55	336,678.33	2,873.92	695.08

Table 4: Results of instances of set A for 3-staged patterns in the unconstrained case.

	3-staged Model					3-staged Bottom-up Alg					3-staged Bottom-up Alg+TE					De Queiroz et al (2012)	
Instance	var	cons	sol[%]	gap[%]	time[s]	sol[%]	time[s]	nGer	nIns	nEnd	sol[%]	time[s]	nGer	nIns	nEnd	sol[%]	time[s]
gcut01_3d.txt	436	145	**80.57**	0.00	129.89	**80.57**	2.04	66,751	4,755	1,611	**80.57**	0.47	3,374	223	23	**80.57**	0.10
gcut02_3d.txt	3,661	902	**84.35**	0.00	330.88	**84.35**	270.37	11,645,372	123,257	53,369	**84.35**	10.96	34,078	727	160	**84.35**	0.11
gcut03_3d.txt	12,102	2,103	**89.11**	0.00	613.41	**89.11**	tl	66,855,129	166,898	117,600	**89.11**	43.19	32,036	474	399	**89.11**	0.99
gcut04_3d.txt	35,310	3,845	**94.31**	0.00	2,089.49	**94.31**	162.40	14,581,874	32,498	25,699	**94.31**	69.89	5,412	101	98	**94.31**	3.31
gcut05_3d.txt	799	279	**83.83**	0.00	129.34	**83.83**	1.22	117,544	3,626	2,739	**83.83**	0.44	827	62	59	**83.83**	0.10
gcut06_3d.txt	2,834	642	**81.84**	0.00	221.94	**81.84**	5.56	236,702	11,130	5,374	**81.84**	1.81	11,847	290	55	**81.84**	0.10
gcut07_3d.txt	13,460	1,971	**87.24**	0.00	317.35	**87.24**	26.05	2,814,318	20,613	10,352	**87.24**	8.14	11,081	249	75	**87.24**	0.14
gcut08_3d.txt	59,042	6,530	**92.56**	0.00	1,058.07	**92.56**	182.72	14,805,081	46,189	30,008	92.33	69.35	6,405	116	111	**92.56**	2.60
gcut09_3d.txt	692	233	**93.16**	0.00	133.86	**93.16**	6.10	197,381	3,251	1,922	**93.16**	0.67	1,443	89	77	**93.16**	0.10
gcut10_3d.txt	2,366	616	**85.16**	0.00	237.57	**85.16**	6.31	29,697	1,490	1,022	**85.16**	4.20	498	36	36	**85.16**	0.10
gcut11_3d.txt	11,629	1,871	**90.78**	0.00	27.02	**90.78**	10.61	1,135,592	14,832	10,550	**90.78**	6.19	1,458	50	48	**90.78**	0.28
gcut12_3d.txt	40,676	4,256	**91.23**	0.00	433.31	**91.23**	217.42	12,837,596	38,524	31,925	**91.23**	120.01	3,835	96	95	**91.23**	2.12
Averages	15,250.58	1,949.42	87.85	0.00	476.84	87.85	129.42	10,443,586.42	38,921.92	24,347.58	87.83	27.94	9,357.83	209.42	103.00	87.85	0.84

Table 5: Results of instances of set A for 3-staged patterns in the constrained case.

	3-staged Model					3-staged Bottom-up Alg					3-staged Bottom-up Alg+TE				
Instance	var	cons	sol[%]	gap[%]	time[s]	sol[%]	time[s]	nGer	nIns	nEnd	sol[%]	time[s]	nGer	nIns	nEnd
cons_gcut01_3d.txt	292	94	**62.73**	0.00	37.68	**62.73**	0.40	2,303	694	536	**62.73**	0.30	1,137	209	174
cons_gcut02_3d.txt	1,899	365	**73.19**	0.00	45.56	**73.19**	28.86	495,973	33,255	24,350	**73.19**	10.45	109,784	3,679	3,120
cons_gcut03_3d.txt	5,619	765	**83.58**	0.00	76.09	**83.58**	589.61	17,063,870	121,783	95,390	**83.58**	15.59	492,418	3,313	2,742
cons_gcut04_3d.txt	23,431	1,887	**87.30**	0.00	106.07	**87.30**	tl	41,231,549	125,989	98,647	**87.30**	15.54	1,043,946	4,568	3,448
cons_gcut05_3d.txt	333	100	**56.86**	0.00	35.83	**56.86**	46.81	4,250	1,870	1,384	**56.86**	0.50	3,781	903	752
cons_gcut06_3d.txt	2,472	442	**71.25**	0.00	82.38	**71.25**	11.32	17,579	3,435	2,390	**71.25**	1.09	10,212	721	325
cons_gcut07_3d.txt	5,163	702	**80.59**	0.00	38.87	**80.59**	18.71	631,441	11,887	10,266	**80.59**	6.33	25,803	486	464
cons_gcut08_3d.txt	32,447	2,455	**86.78**	0.00	164.88	**86.78**	387.90	32,472,683	95,619	57,999	**86.78**	26.38	1,689,756	5,556	1,795
cons_gcut09_3d.txt	423	113	**61.45**	0.00	43.21	**61.45**	5.01	24,816	7,511	4,007	**61.45**	1.46	19,527	4,543	2,706
cons_gcut10_3d.txt	1,770	361	**71.63**	0.00	32.33	**71.63**	5.69	5,450	1,623	877	**71.63**	8.43	3,117	348	100
cons_gcut11_3d.txt	7,979	962	**74.92**	0.00	231.51	**74.92**	210.78	2,126,313	136,772	85,807	**74.92**	15.79	449,941	12,308	9,002
cons_gcut12_3d.txt	27,089	2,079	**87.68**	0.00	148.34	**87.68**	88.44	6,590,510	31,101	27,025	**87.68**	18.79	22,116	250	246
Averages	9,076.42	860.42	74.83	0.00	86.90	74.83	169.64	8,388,894.75	47,628.25	34,056.50	74.83	10.05	322,628.17	3,073.67	2,072.83

4.2 Results of instances of set B

Egeblad and Pisinger (2009) randomly generated the 60 *ep3* instances for the three-dimensional knapsack problem. These instances contain 20, 40 or 60 items, which are divided into 5 different classes. The dimensions of the item types were uniformly sampled in the intervals described in Table 6. Regarding the maximum number of copies to be cut (i.e. u_i for $i \in I$), they are clustered in five item types (i.e. the number of items divided by 5) or random (i.e. one copy per item type). The volume of the large object is equal to 50% or 90% of the total volume of copies of the item types. The height $\bar{H}$ of the object is twice its length $\bar{L}$, which is equal to its width $\bar{W}$. The authors proposed the naming convention *ep3-n-c-t-p*, where $n \in \{20, 40, 60\}$ is the number of items, $c \in \{F, L, C, U, D\}$ describes the class, $t \in \{C, R\}$ describes if it is clustered or random, and $p \in \{50, 90\}$ describes the size of the object in percentage of the total sum of the item's volume. These instances are weighted, given that the value p_i of each item type $i \in I$ is its volume times a random number from $\{1, 2, 3\}$.

Table 6: Characteristics of instances of set B.

Class	Description	Length	Width	Height
F	items are flat	[50;100]	[25;60]	[50;100]
L	items are long	$[1; \frac{2}{3}.100]$	[50;100]	$[1; \frac{2}{3}.100]$
C	items are cubes	[1;100]	Equal to length	Equal to length
U	Largest dimension is no more than 200% of the smallest	[50;100]	[50;100]	[50;100]
D	Largest dimension can be up to 50 times the smallest	[1;50]	[1;50]	[1;50]

Table 7 shows the results of the `Non-staged Model` and `3-staged Model` for the weighted instances of set B. It should be recalled that the bottom-up algorithm can only deal with non-weighted instances. In this table, we report the values of the objective function in columns sol, instead of the material utilisation. Optimality was proven by the solver with the `Non-staged Model` only in 5 out of 60 instances, whilst optimality was proven with the `3-staged Model` in 51 out of 60 instances. Particularly, the instances with optimality proven with the `Non-staged Model` are characterised by having the smallest amount of items ($n = 20$) and the smallest type of large object ($p = 50$). For both models, the number of variables and constraints increases when the number of items n also increases. It should be noted that the average processing times of the `3-staged Model` quickly increase when the number of items also increases. Although the `Non-staged Model` deals with more general cutting patterns than the `3-staged Model`, the latter had a better average performance than the former regarding the values of the solutions. The average processing time of the `3-staged Model` for the clustered instances was 62.48 seconds, and 5,728.95 seconds for the random instances. We highlight that the `3-staged Model` comprises twelve runs of the solver, but high-quality feasible solutions are found in the first 30 seconds of the search, as discussed at the end of Section 4.1. Moreover, the average deviation of the solutions of the `3-staged Model` with respect to the best solutions obtained by Egeblad and Pisinger (2009) was 3.82%. This suggests that the `3-staged Model` may be considered to obtain good feasible solutions for the CLP.

We generate the *nw-ep3* instances by adapting the *ep3* instances of set B for the non-weighted case, i.e. by considering that the value of each item type is equal to its volume. Table 8 shows the results of the proposed approaches for such adapted instances. The optimal solution of most of these instances is close to 100.0% of material utilisation when presupposing an unlimited number of copies per item type (i.e. in the unconstrained case), so we chose to not report results for the `Bottom-up Alg+TE` and `3-staged Bottom-up Alg+TE`, because the impact of trim estimation of the outer volume would be marginal for these instances.

The analysis of the results of Table 8 shows that the performance of the `Non-staged Model` and `3-staged Model` is similar to the performance observed with the weighted instances in Table 7. The `Bottom-up Alg` had memory breakdown issues in 23 out of 60 instances and was able to find an optimal solution and prove its optimality in 18 out of 60 instances. The `3-staged Bottom-up Alg`

Table 7: Results of weighted instances of set B for non-staged and 3-staged patterns.

						Non-staged Model						3-staged Model				
Instance	$\bar{L}$	$\bar{W}$	$\bar{H}$	m	n	$\bar{n}$	var	cons	sol	gap[%]	time[s]	var	cons	sol	gap[%]	time[s]
ep3-20-C-C-50	120	241	120	5	20	11	340	908	**65,308**	0.00	2.99	449	188	**65,308**	0.00	7.64
ep3-20-C-C-90	146	293	146	5	20	15	588	1,604	80,124	17.76	tl	617	253	**80,124**	0.00	4.80
ep3-20-C-R-50	116	232	116	20	20	12	506	1,409	**62,364**	0.00	5.87	2,856	399	**62,364**	0.00	20.54
ep3-20-C-R-90	141	283	141	20	20	15	728	2,039	66,844	17.43	tl	2,858	403	**66,844**	0.00	19.16
ep3-20-D-C-50	59	119	59	5	20	11	340	908	**13,192**	0.00	16.58	381	177	**13,192**	0.00	6.12
ep3-20-D-C-90	72	144	72	5	20	15	588	1,604	27,848	9.37	tl	485	225	**27,848**	0.00	15.98
ep3-20-D-R-50	52	104	52	20	20	15	728	2,039	14,619	18.08	tl	2,080	384	**13,754**	0.00	7.53
ep3-20-D-R-90	63	126	63	20	20	18	986	2,777	19,609	18.93	tl	2,868	443	**19,265**	0.00	15.69
ep3-20-F-C-50	106	213	106	5	20	10	288	764	**71,816**	0.00	1,749.21	309	153	**71,816**	0.00	45.42
ep3-20-F-C-90	129	259	129	5	20	16	660	1,808	161,011	10.57	tl	376	181	**156,237**	0.00	93.07
ep3-20-F-R-50	104	208	104	20	20	9	320	887	**80,962**	0.00	305.96	2,483	385	**80,962**	0.00	2.96
ep3-20-F-R-90	127	254	127	20	20	16	810	2,273	151,147	10.45	tl	2,239	405	**148,204**	0.00	34.88
ep3-20-L-C-50	80	160	80	5	20	12	396	1,064	20,148	26.84	tl	148	91	**20,148**	0.00	44.77
ep3-20-L-C-90	97	194	97	5	20	16	660	1,808	44,820	13.79	tl	265	155	**44,426**	0.00	45.43
ep3-20-L-R-50	77	154	77	20	20	15	728	2,039	24,729	15.92	tl	2,154	379	**24,729**	0.00	6.31
ep3-20-L-R-90	94	188	94	20	20	19	1,080	3,047	30,582	18.02	tl	2,306	401	**30,096**	0.00	25.44
ep3-20-U-C-50	125	250	125	5	20	10	288	764	91,036	36.47	tl	77	47	**91,036**	0.00	13.39
ep3-20-U-C-90	152	305	152	5	20	16	660	1,808	118,252	22.63	tl	440	177	**132,291**	0.00	2.62
ep3-20-U-R-50	129	258	129	20	20	11	440	1,223	97,358	21.28	tl	1,992	366	**97,358**	0.00	6.93
ep3-20-U-R-90	157	314	157	20	20	18	986	2,777	171,709	16.62	tl	2,220	385	**171,709**	0.00	18.41
ep3-40-C-C-50	151	303	151	5	40	27	1,716	4,844	141,418	16.97	tl	1,075	469	**141,418**	0.00	3.29
ep3-40-C-C-90	184	369	184	5	40	34	2,640	7,532	234,469	7.11	tl	897	400	**243,447**	0.00	149.61
ep3-40-C-R-50	141	282	141	40	40	28	2,646	7,657	126,069	13.10	tl	19,944	1,490	**125,544**	0.00	574.35
ep3-40-C-R-90	172	344	172	40	40	36	3,990	11,593	209,076	7.85	tl	21,565	1,570	**213,452**	0.00	3,584.24
ep3-40-D-C-50	75	150	75	5	40	27	1,716	4,844	20,464	35.86	tl	359	172	**20,464**	0.00	4.85
ep3-40-D-C-90	91	182	91	5	40	34	2,640	7,532	54,480	14.34	tl	1,174	529	**55,092**	0.00	3.65
ep3-40-D-R-50	60	121	60	40	40	31	3,120	9,043	23,860	31.83	tl	14,050	1,342	**25,543**	0.00	504.53
ep3-40-D-R-90	74	148	74	40	40	38	4,366	12,697	45,457	13.86	tl	17,887	1,532	**46,189**	0.00	1,129.75
ep3-40-F-C-50	134	268	134	5	40	24	1,380	3,872	151,848	12.45	tl	81	50	**156,066**	0.00	1.72
ep3-40-F-C-90	163	326	163	5	40	34	2,640	7,532	397,479	14.05	tl	531	265	**405,370**	0.00	12.09
ep3-40-F-R-50	131	263	131	40	40	25	2,208	6,379	215,992	20.40	tl	13,846	1,377	**220,150**	0.00	281.36
ep3-40-F-R-90	160	320	160	40	40	37	4,176	12,139	316,456	19.37	tl	16,709	1,417	**347,868**	0.00	3,956.37
ep3-40-L-C-50	100	201	100	5	40	24	1,380	3,872	54,054	11.71	tl	300	183	**55,846**	0.00	15.63
ep3-40-L-C-90	122	245	122	5	40	34	2,640	7,532	93,240	18.04	tl	445	223	**94,884**	0.00	2.60
ep3-40-L-R-50	94	188	94	40	40	29	2,800	8,107	44,004	28.19	tl	14,555	1,459	**43,820**	0.00	479.65
ep3-40-L-R-90	114	229	114	40	40	38	4,366	12,697	70,995	16.98	tl	19,535	1,654	**70,560**	0.00	808.02
ep3-40-U-C-50	158	316	158	5	40	19	900	2,492	150,434	16.74	tl	240	142	**158,616**	0.00	5.33
ep3-40-U-C-90	192	384	192	5	40	34	2,640	7,532	365,880	17.24	tl	219	119	**388,456**	0.00	27.85
ep3-40-U-R-50	164	329	164	40	40	24	2,070	5,977	216,407	19.33	tl	15,298	1,411	**228,803**	0.00	530.57
ep3-40-U-R-90	200	400	200	40	40	37	4,176	12,139	321,741	19.86	tl	17,424	1,531	347,272	1.10	15,711.01
ep3-60-C-C-50	173	347	173	5	60	42	3,936	11,324	219,871	38.90	tl	876	400	**246,129**	0.00	35.40
ep3-60-C-C-90	211	422	211	5	60	52	5,916	17,144	455,340	14.76	tl	2,440	1,027	**499,572**	0.00	99.50
ep3-60-C-R-50	161	322	161	60	60	46	6,930	20,313	197,795	13.84	tl	63,840	3,257	216,117	2.09	12,414.96
ep3-60-C-R-90	196	392	196	60	60	55	9,288	27,279	354,804	12.94	tl	73,137	3,551	**362,563**	0.00	16,030.72
ep3-60-D-C-50	85	171	85	5	60	40	3,588	10,304	65,592	11.78	tl	1,360	625	**65,592**	0.00	3.37
ep3-60-D-C-90	104	209	104	5	60	46	4,680	13,508	104,128	7.45	tl	1,765	801	**105,600**	0.00	142.75
ep3-60-D-R-50	67	135	67	60	60	50	7,938	23,289	38,738	31.10	tl	42,655	2,881	**44,843**	0.00	5,060.08
ep3-60-D-R-90	82	164	82	60	60	58	10,146	29,817	58,778	19.60	tl	52,116	3,273	64,682	1.01	10,632.90
ep3-60-F-C-50	153	307	153	5	60	39	3,420	9,812	443,790	12.71	tl	137	81	**467,730**	0.00	165.95
ep3-60-F-C-90	186	373	186	5	60	53	6,136	17,792	451,524	23.20	tl	839	362	**506,304**	0.00	227.00
ep3-60-F-R-50	150	301	150	60	60	34	4,290	12,537	313,020	27.80	tl	48,223	2,972	360,432	0.00	10,963.49
ep3-60-F-R-90	183	367	183	60	60	56	9,570	28,113	552,028	17.43	tl	44,723	3,064	609,403	0.50	23,926.52
ep3-60-L-C-50	115	230	115	5	60	36	2,940	8,408	110,502	12.36	tl	413	211	**112,272**	0.00	155.10
ep3-60-L-C-90	140	280	140	5	60	55	6,588	19,124	97,347	21.90	tl	633	362	**110,796**	0.00	165.43
ep3-60-L-R-50	104	209	104	60	60	45	6,688	19,599	52,586	29.10	tl	51,189	3,153	54,347	0.00	8,519.43
ep3-60-L-R-90	127	255	127	60	60	58	10,146	29,817	97,577	16.86	tl	57,316	3,442	102,089	0.98	20,539.40
ep3-60-U-C-50	180	361	180	5	60	33	2,496	7,112	330,610	22.77	tl	219	119	**357,004**	0.00	187.34
ep3-60-U-C-90	220	440	220	5	60	55	6,588	19,124	245,476	24.42	tl	588	310	**272,664**	0.00	186.81
ep3-60-U-R-50	184	369	184	60	60	37	4,896	14,319	368,386	18.61	tl	49,683	3,204	373,634	2.64	10,351.85
ep3-60-U-R-90	224	449	224	60	60	56	9,570	28,113	498,608	17.57	tl	57,202	3,504	526,282	6.36	25,711.46
Averages						30.67	3,201.33	9,273.50	162,063.35	17.08	3,334.68	12,551.52	992.18	171,077.10	0.24	2,895.72

Table 8: Results of adapted non-weighted instances of set B for non-staged and 3-staged patterns.

						Non-staged patterns					3-staged patterns				
						`Non-staged Model`			`Bottom-up Alg`		`3-staged Model`			`3-staged Bottom-up Alg`	
Instance	$\bar{L}$	$\bar{W}$	$\bar{H}$	m	n	sol[%]	gap[%]	time[s]	sol[%]	time[s]	sol[%]	gap[%]	time[s]	sol[%]	time[s]
nw-ep3-20-C-C-50	120	241	120	5	20	**49.96**	0.00	2.01	**49.96**	15.82	**49.96**	0.00	95.97	49.96	tl
nw-ep3-20-C-C-90	146	293	146	5	20	**47.02**	21.50	tl	**47.02**	221.84	**47.02**	0.00	76.26	47.02	tl
nw-ep3-20-C-R-50	116	232	116	20	20	54.38	5.20	tl	54.38	tl	**54.38**	0.00	220.36	54.38	tl
nw-ep3-20-C-R-90	141	283	141	20	20	44.93	23.47	tl	*	*	**44.93**	0.00	8.61	44.82	tl
nw-ep3-20-D-C-50	59	119	59	5	20	**70.50**	0.00	21.91	**70.50**	1.38	**70.50**	0.00	1.91	**70.50**	22.61
nw-ep3-20-D-C-90	72	144	72	5	20	**56.78**	12.23	tl	**56.78**	26.40	**56.78**	0.00	97.04	56.78	tl
nw-ep3-20-D-R-50	52	104	52	20	20	77.49	22.13	tl	79.94	tl	**73.65**	0.00	8.55	73.65	tl
nw-ep3-20-D-R-90	63	126	63	20	20	71.67	26.13	tl	67.62	tl	**70.41**	0.00	237.18	66.31	tl
nw-ep3-20-F-C-50	106	213	106	5	20	**79.35**	0.20	tl	**79.35**	5.17	**79.35**	0.00	33.44	**79.35**	1.34
nw-ep3-20-F-C-90	129	259	129	5	20	67.20	9.51	tl	**69.29**	1.31	**65.76**	0.00	43.13	**65.76**	6.59
nw-ep3-20-F-R-50	104	208	104	20	20	**69.47**	0.00	483.37	**69.47**	8.23	67.96	0.00	46.17	**69.47**	17.25
nw-ep3-20-F-R-90	127	254	127	20	20	68.88	17.94	tl	69.35	tl	**68.17**	0.00	6.28	68.17	tl
nw-ep3-20-L-C-50	80	160	80	5	20	**83.01**	16.99	tl	**83.01**	2.80	**79.62**	0.00	6.11	**79.62**	0.78
nw-ep3-20-L-C-90	97	194	97	5	20	66.53	20.49	tl	**67.46**	4.56	**62.60**	0.00	2.30	**62.60**	12.47
nw-ep3-20-L-R-50	77	154	77	20	20	80.77	19.23	tl	**82.12**	14.85	**80.77**	0.00	5.91	**80.77**	35.32
nw-ep3-20-L-R-90	94	188	94	20	20	77.59	22.00	tl	77.59	tl	**75.22**	0.00	10.63	68.75	tl
nw-ep3-20-U-C-50	125	250	125	5	20	**61.99**	36.79	tl	**61.99**	1.77	**61.99**	0.00	117.41	**61.99**	7.38
nw-ep3-20-U-C-90	152	305	152	5	20	73.88	17.70	tl	**73.96**	12.92	**73.88**	0.00	47.49	**73.88**	8.87
nw-ep3-20-U-R-50	129	258	129	20	20	**71.78**	26.87	tl	**71.78**	13.88	**69.76**	0.00	50.28	**69.76**	1.77
nw-ep3-20-U-R-90	157	314	157	20	20	71.50	25.62	tl	75.68	tl	**75.56**	0.00	14.55	70.78	tl
nw-ep3-40-C-C-50	151	303	151	5	40	50.09	31.74	tl	*	*	**50.09**	0.00	6.87	48.94	tl
nw-ep3-40-C-C-90	184	369	184	5	40	74.87	6.30	tl	*	*	**75.09**	0.00	4.48	75.09	tl
nw-ep3-40-C-R-50	141	282	141	40	40	53.28	29.54	tl	58.24	tl	**61.61**	0.00	610.24	58.24	tl
nw-ep3-40-C-R-90	172	344	172	40	40	77.46	8.08	tl	*	*	**80.53**	0.00	2,331.06	*	*
nw-ep3-40-D-C-50	75	150	75	5	40	72.32	24.91	tl	76.18	tl	**76.04**	0.00	173.61	75.97	tl
nw-ep3-40-D-C-90	91	182	91	5	40	72.01	14.47	tl	75.98	tl	**74.96**	0.00	14.65	73.84	tl
nw-ep3-40-D-R-50	60	121	60	40	40	81.33	18.67	tl	80.84	tl	**83.95**	0.00	228.90	80.84	tl
nw-ep3-40-D-R-90	74	148	74	40	40	71.55	28.45	tl	*	*	**76.94**	0.00	634.22	76.83	tl
nw-ep3-40-F-C-50	134	268	134	5	40	64.39	30.85	tl	**73.49**	2.07	**73.49**	0.00	48.71	**73.49**	9.02
nw-ep3-40-F-C-90	163	326	163	5	40	66.54	20.04	tl	74.23	tl	**72.48**	0.00	49.42	69.46	tl
nw-ep3-40-F-R-50	131	263	131	40	40	70.57	29.43	tl	*	*	**74.95**	0.00	72.73	*	*
nw-ep3-40-F-R-90	160	320	160	40	40	82.66	17.33	tl	*	*	**83.61**	0.00	3,592.23	82.55	tl
nw-ep3-40-L-C-50	100	201	100	5	40	83.29	16.70	tl	**83.84**	4.62	**73.45**	0.00	81.23	**73.45**	69.57
nw-ep3-40-L-C-90	122	245	122	5	40	74.03	18.63	tl	82.94	tl	**74.83**	0.00	82.13	72.27	tl
nw-ep3-40-L-R-50	94	188	94	40	40	78.58	21.42	tl	73.81	tl	**79.84**	0.00	346.64	73.81	tl
nw-ep3-40-L-R-90	114	229	114	40	40	73.79	26.21	tl	*	*	**78.95**	0.00	883.09	77.10	tl
nw-ep3-40-U-C-50	158	316	158	5	40	70.32	29.68	tl	**73.44**	6.75	**72.38**	0.00	76.49	**72.38**	6.10
nw-ep3-40-U-C-90	192	384	192	5	40	74.37	19.99	tl	*	*	**75.33**	0.00	58.73	75.33	tl
nw-ep3-40-U-R-50	164	329	164	40	40	83.75	16.25	tl	87.11	tl	**88.44**	0.00	876.39	87.11	tl
nw-ep3-40-U-R-90	200	400	200	40	40	72.06	27.94	tl	*	*	80.64	0.00	12,373.89	77.08	tl
nw-ep3-60-C-C-50	173	347	173	5	60	84.72	15.28	tl	*	*	**86.23**	0.00	76.86	86.23	tl
nw-ep3-60-C-C-90	211	422	211	5	60	66.45	22.43	tl	*	*	**72.75**	0.00	107.43	71.02	tl
nw-ep3-60-C-R-50	161	322	161	60	60	67.34	24.70	tl	70.23	tl	78.22	0.00	6,459.50	70.23	tl
nw-ep3-60-C-R-90	196	392	196	60	60	78.82	16.47	tl	*	*	80.16	0.00	10,401.62	*	*
nw-ep3-60-D-C-50	85	171	85	5	60	75.09	22.97	tl	80.48	tl	**77.95**	0.00	3.04	77.95	tl
nw-ep3-60-D-C-90	104	209	104	5	60	70.20	7.14	tl	*	*	**70.25**	0.00	6.14	70.25	tl
nw-ep3-60-D-R-50	67	135	67	60	60	83.50	16.50	tl	79.67	tl	**83.71**	0.00	578.19	79.67	tl
nw-ep3-60-D-R-90	82	164	82	60	60	75.13	24.87	tl	*	*	82.48	0.00	4,752.76	76.63	tl
nw-ep3-60-F-C-50	153	307	153	5	60	66.91	33.09	tl	89.17	194.32	**88.47**	0.00	4.30	86.15	tl
nw-ep3-60-F-C-90	186	373	186	5	60	68.92	23.51	tl	*	*	**77.90**	0.00	7.43	73.97	tl
nw-ep3-60-F-R-50	150	301	150	60	60	82.17	17.83	tl	*	*	87.57	0.90	8,416.04	*	*
nw-ep3-60-F-R-90	183	367	183	60	60	80.49	19.51	tl	*	*	84.99	1.20	14,503.98	*	*
nw-ep3-60-L-C-50	115	230	115	5	60	73.84	26.16	tl	**90.82**	203.61	**85.27**	0.00	134.55	85.27	tl
nw-ep3-60-L-C-90	140	280	140	5	60	80.07	19.93	tl	*	*	**85.93**	0.00	81.07	78.23	tl
nw-ep3-60-L-R-50	104	209	104	60	60	76.26	23.74	tl	81.30	tl	84.67	0.00	6,497.80	81.30	tl
nw-ep3-60-L-R-90	127	255	127	60	60	75.78	24.22	tl	*	*	82.92	0.00	18,256.03	79.28	tl
nw-ep3-60-U-C-50	180	361	180	5	60	75.46	24.54	tl	**86.24**	108.46	**85.56**	0.00	118.68	85.56	tl
nw-ep3-60-U-C-90	220	440	220	5	60	68.72	31.28	tl	*	*	**83.02**	0.00	94.20	83.02	tl
nw-ep3-60-U-R-50	184	369	184	60	60	85.90	14.10	tl	*	*	87.24	0.00	8,921.68	84.89	tl
nw-ep3-60-U-R-90	224	449	224	60	60	74.97	25.03	tl	*	*	85.01	3.85	23,110.23	72.49	tl
Averages						71.71	19.90	3,428.45	73.66**	330.42**	74.77	0.10	2,103.45	72.37**	487.77**

**: This average was calculated with instances without memory overflow, and so it cannot be directly compared.

had memory breakdown issues in 5 out of 60 instances and was able to find an optimal solution and prove its optimality in 13 out of 60 instances. It should be noted that ep3 instances are characterised by moderate levels of material utilisation in the optimal solution. Thus, parameter β is significantly large, meaning that the aspiration criterion eliminates few partial solutions, and the memory required for the sets of partial solutions grows too much. We highlight that optimality was proven for instance nw-ep3-20-F-R-50 by the `3-staged Bottom-up Alg`. For such instance, the optimality was also proven by solver with the `3-staged Model`, however their optimal value differs, as `3-staged Model` does not address all kinds of 3-staged patterns. This instance was the only scenario in our computational experiments where considering the dimensions of the item types as first stage cut positions (i.e., restricted stage cuts) led to a non-optimal solution with respect to more general 3-staged patterns, as the one depicted in Fig. 5. For this reason, we chose not to highlight in bold the optimality of the solutions of the `3-staged Bottom-up Alg` according to proven optimal solutions of the CPLEX with the `3-staged Model`.

In summary, the main results of this section are: (i) the `3-staged Model` seems to outperform the `Non-staged Model` both in quality solution and processing time for most of the instances, although it addresses a more limited type of cutting pattern; (ii) the value of parameter β strongly affects the performance of the `Bottom-up Alg`, which was able to prove optimality for some instances even with low material utilisation rates.

5 Conclusions

In this paper, we addressed the Constrained Three-dimensional Guillotine Placement Problem (C3GCP). The C3GCP arises in different manufacturing settings, such as the cutting of steel, foam for mattresses, and marble stones. We proposed a new compact mixed-integer non-linear programming (MINLP) formulation and its mixed-integer linear programming (MILP) version to model the problem by extending a model previously proposed for the two-dimensional case. As some industrial environments limit their cutting operations to a few guillotine stages in order to simplify the cutting patterns and reduce the processing times to cut these patterns, we also proposed a new model that considers only 3-staged patterns. As a solution method for the C3GCP, we extended to the three-dimensional case the algorithm of Wang (1983) that successively combines small items until the size of the larger object is reached.

Computational experiments using benchmark instances from the literature indicated that the proposed approaches are appropriate for scenarios with a moderate number of copies of item types. Notably, the bottom-up algorithm was able to prove optimality in a few seconds in several instances, especially when the sizes of the item types are large with respect to the sizes of the object. The model proposed for 3-staged patterns showed effective performance in non-weighted and weighted instances, even considering more limited patterns than the non-staged model.

For future research, the models could be enhanced with the development of additional valid inequalities to seek better LP-bounds in the context of general-purpose optimisation solvers. The development of solution methods based on decomposition (e.g. Dantzig-Wolfe decomposition, Benders decomposition and Lagrangean relaxation) could enable the solution of larger problem instances. Regarding the bottom-up algorithm, analysing different types of guillotine cuts and improving the aspiration criterion could prove useful. Moreover, new heuristic strategies for rejecting non-promising feasible solutions could contribute to overcoming memory breakdown issues. A general framework for the problem considering usable leftovers during the cutting analysis would have practical benefits in a few industrial environments. It is worth recalling that the extents of the proposed compact formulation and of the proposed bottom-up algorithm, both for non-staged patterns, seem to be relatively straightforward for the n-dimensional case (i.e. when $n > 3$). Another possibility for all approaches is to incorporate uncertainty in the input data.

Acknowledgements

The authors would like to thank the National Council for Scientific and Technological Development (CNPq-Brazil) [grant number 200745/2018-2] and the São Paulo Research Foundation (FAPESP-Brazil) [grant numbers 16/08039-1, 16/01860-1, 13/07375-0] for the financial support. The research was also financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT - Fundao para a Cincia e a Tecnologia within project POCI-01-0145-FEDER-029609. Research developed using the computational resources of the Center of Mathematical Sciences Applied to Industry (CeMEAI) was financed by FAPESP. The authors are also grateful to engineer Hugo Ferreira for discussing aspects of the guillotine cutting in the steel industry.

A MILP model for non-staged patterns

We make use of additional variables and disjunctive constraints to reformulate Model (7) in order to obtain a MILP model. Equations (13), (14) and (15) define these additional variables.

y^L_{jk}: length of block k contained by block j, $\quad j \in J, k \in J, j < k.$ (13)

y^W_{jk}: width of block k contained by block j, $\quad j \in J, k \in J, j < k.$ (14)

y^H_{jk}: height of block k contained by block j, $\quad j \in J, k \in J, j < k.$ (15)

Model (16) is a compact MILP formulation for the C3GCP.

Max (7a),

s.t.

$$(7b) - (7f), (7j) - (7n),$$

$$0 \le y^L_{jk} \le \bar{L} y_{jk}, \quad j \in J, k \in J, j < k, \quad (16a)$$

$$L_k - \bar{L}(1 - y_{jk}) \le y^L_{jk} \le L_k, \quad j \in J, k \in J, j < k, \quad (16b)$$

$$0 \le y^W_{jk} \le \bar{W} y_{jk}, \quad j \in J, k \in J, j < k, \quad (16c)$$

$$W_k - \bar{W}(1 - y_{jk}) \le y^W_{jk} \le W_k, \quad j \in J, k \in J, j < k, \quad (16d)$$

$$0 \le y^H_{jk} \le \bar{H} y_{jk}, \quad j \in J, k \in J, j < k, \quad (16e)$$

$$H_k - \bar{H}(1 - y_{jk}) \le y^H_{jk} \le H_k, \quad j \in J, k \in J, j < k, \quad (16f)$$

$$L_j \ge \sum_{i \in I} \sum_{p \in Q_i} l_i z_{jip} + \sum_{k \in K, j<k} y^L_{jk} - 2\bar{L}(1 - x_{jh}), \quad j \in J, \quad (16g)$$

$$L_j \ge l_i z_{jip} - \bar{L}(1 - x_{jv} - x_{jp}), \quad j \in J, i \in I, p \in Q_i, \quad (16h)$$

$$L_j \ge y^L_{jk} - \bar{L}(1 - x_{jv} - x_{jp}), \quad j \in J, k \in K, j < k, \quad (16i)$$

$$W_j \ge w_i z_{jip} - \bar{W}(1 - x_{jh} - x_{jv}), \quad j \in J, i \in I, p \in Q_i, \quad (16j)$$

$$W_j \ge y^W_{jk} - \bar{W}(1 - x_{jh} - x_{jv}), \quad j \in J, k \in K, j < k, \quad (16k)$$

$$W_j \ge \sum_{i \in I} \sum_{p \in Q_i} w_i z_{jip} + \sum_{k \in K, j<k} y^W_{jk} - 2\bar{W}(1 - x_{jd}), \quad j \in J, \quad (16l)$$

$$H_j \ge h_i z_{jip} - \bar{H}(1 - x_{jh} - x_{jd}), \quad j \in J, i \in I, p \in Q_i, \quad (16m)$$

$$H_j \ge y^H_{jk} - \bar{H}(1 - x_{jh} - x_{jd}), \quad j \in J, k \in K, j < k, \quad (16n)$$

$$H_j \ge \sum_{i \in I} \sum_{p \in Q_i} h_i z_{jip} + \sum_{k \in K, j<k} y^H_{jk} - 2\bar{H}(1 - x_{jv}), \quad j \in J. \quad (16o)$$

Constraints (16a) to (16b) define that variable y^L_{jk} is L_k when $y_{jk} = 1$, and zero otherwise (i.e. $y_{jk} = 0$). Similarly, constraints (16c)–(16d) and (16e)–(16f) define this behaviour for variables y^W_{jk} and y^H_{jk}. Each non-linear constraint (7h) of Model (7) can be replaced, without loss of optimality, by the three corresponding linear constraints (16g) to (16i). It should be noted that when sub-pattern $j \in J$ is built up with orientation h (i.e. when $x_{jh} = 1$), only constraint (16g) is enabled, but if it is built up with orientations d or v (i.e. when $x_{jv} + x_{jp} = 1$), only constraints (16h) and (16i) are enabled. Constraints (16j)–(16l) and (16m)–(16o) are similar to (16g)–(16i), but impose this behaviour on variables W_j and H_j, respectively.

We also extend the valid inequalities proposed by Martin et al (2019) to the three-dimensional case in Constraints (17). Constraint (17a) limits the total volume of the cut items to upper bound parameter UB_1. Constraint (17b) limits the total value of the cut items (i.e. the objective function) to upper bound parameter UB_2. Both parameters UB_1 and UB_2 can be calculated as any relaxation of the C3GCP. Constraint (17c) limits the number of cut items to the upper bound $\bar{n}$. Constraints (17d), (17e) and (17f) ensure that variables L_j, W_j and H_j assume positive values only when the corresponding sub-pattern j exists (i.e. when $\sum_{o \in O} x_{jo} = 1$).

$$\sum_{j \in J} \sum_{i \in I} \sum_{p \in Q_i} (l_i w_i h_i) z_{jip} \leq UB_1, \tag{17a}$$

$$\sum_{j \in J} \sum_{i \in I} \sum_{p \in Q_i} p_i z_{jip} \leq UB_2, \tag{17b}$$

$$\sum_{j \in J} \sum_{i \in I} \sum_{p \in Q_i} z_{jip} \leq \bar{n}, \tag{17c}$$

$$L_j \leq \bar{L} \sum_{o \in O} x_{jo}, \qquad j \in J, \tag{17d}$$

$$W_j \leq \bar{W} \sum_{o \in O} x_{jo}, \qquad j \in J, \tag{17e}$$

$$H_j \leq \bar{H} \sum_{o \in O} x_{jo}. \qquad j \in J. \tag{17f}$$

B Placement of the model for non-staged patterns

Algorithm 2 is a pseudo-code for generating a cutting pattern from a solution of Model (7) or Model (16). Index j varies from 1 to $\bar{n} - 1$ in line 2, as the final cutting pattern (i.e. sub-pattern $j = 1$) is placed at position $(0, 0, 0)$. This algorithm is a straightforward adaptation of the pseudo-code presented by Martin et al (2019) for the C2GCP.

C Enumeration of guillotine stages to the bottom-up algorithm for 3-staged patterns

In Section 3.3, we adapted the bottom-up algorithm to strictly consider up to 3-staged patterns by enumerating all possible types of d-patterns, with $d \leq 3$, generated from horizontal, in depth and vertical builds. Tables 9, 10 and 11 are the look-up tables considered in line 5 of Algorithm 1 when merging horizontally, in depth and vertically, respectively. For instance, Table 9 shows that when horizontally merging a partial solution of type *xy* with a partial solution of type *yx*, the result is a new partial solution of type *xyx*. The blank entries in these tables point out that such build would generate a partial solution with more than 3 stages, which is forbidden. It should be noted that there are few cases in which merging two partial solutions results in two possible types of d-staged patterns. Moreover, when horizontally merging a partial solution of type y with a partial solution of type z, the result is a new partial solution of type *yzx* or a partial solution of type *zyx*. To keep optimality in those special cases, we generate two new partial solutions, i.e. one solution for each possible type of d-staged pattern, and insert both of them in list $F^{(k)}$.

References

Amossen RR, Pisinger D (2010) Multi-dimensional bin packing problems with guillotine constraints. Computers & Operations Research 37(11):1999–2006, DOI 10.1016/j.cor.2010.01.017

Algorithm 2: Placement from a solution of the model for non-staged patterns.

Input: Instance, values of variables z_{jip}, x_{jo}, y_{jk}, L_j, W_j, H_j.

1 $\alpha_1 \leftarrow 0, \beta_1 \leftarrow 0, \gamma_1 \leftarrow 0$;
2 **for** $j \in J, o \in O \mid x_{jo} = 1$ **do**
3 $\quad \mathtt{k}_1 \leftarrow -1, \mathtt{i}_1 \leftarrow -1, \mathtt{p}_1 \leftarrow -1, \mathtt{l}_1 \leftarrow -1, \mathtt{w}_1 \leftarrow -1, \mathtt{h}_1 \leftarrow -1$;
4 $\quad \mathtt{k}_2 \leftarrow -1, \mathtt{i}_2 \leftarrow -1, \mathtt{p}_2 \leftarrow -1$;
$\quad$ `// Lines 5 to 14 identify the components (smaller sub-patterns or copies of item types) that build sub-pattern` j
5 $\quad$ **for** $k \in J, j < k \mid y_{jk} = 1$ **do**
6 $\quad\quad$ **if** $\mathtt{l}_1 = -1$ **then**
7 $\quad\quad\quad \mathtt{k}_1 \leftarrow k, \mathtt{l}_1 \leftarrow L_k, \mathtt{w}_1 \leftarrow W_k, \mathtt{h}_1 \leftarrow H_k$;
8 $\quad\quad$ **else**
9 $\quad\quad\quad \mathtt{k}_2 \leftarrow k$;
10 $\quad$ **for** $i \in I, p \in Q_i \mid z_{jip} = 1$ **do**
11 $\quad\quad$ **if** $\mathtt{l}_1 = -1$ **then**
12 $\quad\quad\quad \mathtt{i}_1 \leftarrow i, \mathtt{p}_1 \leftarrow p, \mathtt{l}_1 \leftarrow l_i, \mathtt{w}_1 \leftarrow w_i, \mathtt{h}_1 \leftarrow h_i$;
13 $\quad\quad$ **else**
14 $\quad\quad\quad \mathtt{i}_2 \leftarrow i, \mathtt{p}_2 \leftarrow p$;
$\quad$ `// Lines 15 to 20 define the positions of sub-patterns` k_1 `and` k_2`, if any`
15 $\quad$ **if** $\mathtt{k}_1 > -1$ **then**
16 $\quad\quad \alpha_{\mathtt{k}_1} \leftarrow \alpha_j, \beta_{\mathtt{k}_1} \leftarrow \beta_j, \gamma_{\mathtt{k}_1} \leftarrow \gamma_j$;
17 $\quad$ **if** $\mathtt{k}_2 > -1$ **then**
18 $\quad\quad$ **if** $o = h$ **then** $\alpha_{\mathtt{k}_2} \leftarrow \alpha_j + \mathtt{l}_1, \beta_{\mathtt{k}_2} \leftarrow \beta_j, \gamma_{\mathtt{k}_2} \leftarrow \gamma_j$;
19 $\quad\quad$ **if** $o = d$ **then** $\alpha_{\mathtt{k}_2} \leftarrow \alpha_j, \beta_{\mathtt{k}_2} \leftarrow \beta_j + \mathtt{w}_1, \gamma_{\mathtt{k}_2} \leftarrow \gamma_j$;
20 $\quad\quad$ **if** $o = v$ **then** $\alpha_{\mathtt{k}_2} \leftarrow \alpha_j, \beta_{\mathtt{k}_2} \leftarrow \beta_j, \gamma_{\mathtt{k}_2} \leftarrow \gamma_j + \mathtt{h}_1$;
$\quad$ `// Lines 21 to 26 define the positions of copies of item types` i_1 `and` i_2`, if any`
21 $\quad$ **if** $\mathtt{i}_1 > -1$ **then**
22 $\quad\quad \mathtt{x}_{j\mathtt{i}_1\mathtt{p}_1} \leftarrow \alpha_j, \mathtt{y}_{j\mathtt{i}_1\mathtt{p}_1} \leftarrow \beta_j, \mathtt{z}_{j\mathtt{i}_1\mathtt{p}_1} \leftarrow \gamma_j$;
23 $\quad$ **if** $\mathtt{i}_2 > -1$ **then**
24 $\quad\quad$ **if** $o = h$ **then** $\mathtt{x}_{j\mathtt{i}_2\mathtt{p}_2} \leftarrow \alpha_j + \mathtt{l}_1, \mathtt{y}_{j\mathtt{i}_2\mathtt{p}_2} \leftarrow \beta_j, \mathtt{z}_{j\mathtt{i}_2\mathtt{p}_2} \leftarrow \gamma_j$;
25 $\quad\quad$ **if** $o = d$ **then** $\mathtt{x}_{j\mathtt{i}_2\mathtt{p}_2} \leftarrow \alpha_j, \mathtt{y}_{j\mathtt{i}_2\mathtt{p}_2} \leftarrow \beta_j + \mathtt{w}_1, \mathtt{z}_{j\mathtt{i}_2\mathtt{p}_2} \leftarrow \gamma_j$;
26 $\quad\quad$ **if** $o = v$ **then** $\mathtt{x}_{j\mathtt{i}_2\mathtt{p}_2} \leftarrow \alpha_j, \mathtt{y}_{j\mathtt{i}_2\mathtt{p}_2} \leftarrow \beta_j, \mathtt{z}_{j\mathtt{i}_2\mathtt{p}_2} \leftarrow \gamma_j + \mathtt{h}_1$;
27 **for** $j \in J, i \in I, p \in Q_i \mid z_{jip} = 1$ **do**
28 $\quad$ Block j contains the p-th copy of item type i placed at $(\mathtt{x}_{jip}, \mathtt{y}_{jip}, \mathtt{z}_{jip})$;

Output: Positions $(\mathtt{x}_{jip}, \mathtt{y}_{jip}, \mathtt{z}_{jip})$ of the cut items.

Beasley JE (1985) Algorithms for unconstrained two-dimensional guillotine cutting. Journal of the Operational Research Society 36(4):297–306, DOI 10.1057/jors.1985.51

Ben Messaoud S, Chu C, Espinouse ML (2008) Characterization and modelling of guillotine constraints. European Journal of Operational Research 191(1):110–124, DOI 10.1016/j.ejor.2007.08.029

Bortfeldt A, Wäscher G (2013) Constraints in container loading-A state-of-the-art review. European Journal of Operational Research 229(1):1–20, DOI 10.1016/j.ejor.2012.12.006

Chen C, Lee S, Shen Q (1995) An analytical model for the container loading problem. European Journal of Operational Research 80(1):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(1):30–44, DOI 10.1287/opre.25.1.30

Cintra GF, Miyazawa FK, Wakabayashi Y, Xavier EC (2008) Algorithms for two dimensional cutting stock and strip packing problems using dynamic programming and column generation. European Journal of Operational Research 191(1):59–83, DOI 10.1016/j.ejor.2007.08.007

Table 9: Type of the new partial solution generated by horizontally merging two partial solutions.

		0-stg	1-stg			2-stg						3-stg											
		o	x	y	z	xy	xz	yx	yz	zx	zy	xyx	xyz	xzx	xzy	yxy	yxz	yzx	yzy	zxy	zxz	zyx	zyz
0-stg	o	x	x	yx	zx	xyx	xzx	yx	yzx	zx	zyx	xyx		xzx				yzx				zyx	
1-stg	x	x	x	yx	zx	xyx	xzx	yx	yzx	zx	zyx	xyx		xzx				yzx				zyx	
	y	yx	yx	yx	yzx,zyx	xyx		yx	yzx	yzx,zyx	zyx	xyx						yzx				zyx	
	z	zx	zx	yzx,zyx	zx		xzx	yzx,zyx	yzx	zx	zyx			xzx				yzx				zyx	
2-stg	xy	xyx	xyx	xyx		xyx		xyx				xyx											
	xz	xzx	xzx		xzx		xzx			xzx				xzx									
	yx	yx	yx	yx	yzx,zyx	xyx		yx	yzx	yzx,zyx	zyx	xyx						yzx				zyx	
	yz	yzx	yzx	yzx	yzx			yzx	yzx	yzx								yzx					
	zx	zx	zx	yzx,zyx	zx		xzx	yzx,zyx	yzx	zx	zyx			xzx				yzx				zyx	
	zy	zyx	zyx	zyx	zyx			zyx		zyx	zyx											zyx	
3-stg	xyx	xyx	xyx	xyx		xyx		xyx				xyx											
	xyz																						
	xzx	xzx	xzx		xzx		xzx			xzx				xzx									
	xzy																						
	yxy																						
	yxz																						
	yzx	yzx	yzx	yzx	yzx			yzx	yzx	yzx								yzx					
	yzy																						
	zxy																						
	zxz																						
	zyx	zyx	zyx	zyx	zyx			zyx		zyx	zyx											zyx	
	zyz																						

Table 10: Type of the new partial solution generated by merging in depth two partial solutions.

		0-stg	1-stg			2-stg						3-stg											
		o	x	y	z	xy	xz	yx	yz	zx	zy	xyx	xyz	xzx	xzy	yxy	yxz	yzx	yzy	zxy	zxz	zyx	zyz
0-stg	o	y	xy	y	zy	xy	xzy	yxy	yzy	zxy	zy				xzy	yxy			yzy	zxy			
1-stg	x	xy	xy	xy	xzy,zxy	xy	xzy	yxy		zxy	xzy,zxy				xzy	yxy				zxy			
	y	y	xy	y	zy	xy	xzy	yxy	yzy	zxy	zy				xzy	yxy			yzy	zxy			
	z	zy	xzy,zxy	zy	zy	xzy,zxy	xzy		yzy	zxy	zy				xzy				yzy	zxy			
2-stg	xy	xy	xy	xy	xzy,zxy	xy	xzy	yxy		zxy	xzy,zxy				xzy	yxy				zxy			
	xz	xzy	xzy	xzy	xzy	xzy	xzy				xzy				xzy								
	yx	yxy	yxy	yxy		yxy		yxy								yxy							
	yz	yzy		yzy	yzy				yzy		yzy								yzy				
	zx	zxy	zxy	zxy	zxy	zxy				zxy	zxy									zxy			
	zy	zy	xzy,zxy	zy	zy	xzy,zxy	xzy		yzy	zxy	zy				xzy				yzy	zxy			
3-stg	xyx																						
	xyz																						
	xzx																						
	xzy	xzy	xzy	xzy	xzy	xzy	xzy				xzy				xzy								
	yxy	yxy	yxy	yxy		yxy		yxy								yxy							
	yxz																						
	yzx																						
	yzy	yzy		yzy	yzy				yzy		yzy								yzy				
	zxy	zxy	zxy	zxy	zxy	zxy				zxy	zxy									zxy			
	zxz																						
	zyx																						
	zyz																						

De Queiroz TA, Miyazawa FK, Wakabayashi Y, Xavier EC (2012) Algorithms for 3D guillotine cutting problems: Unbounded knapsack, cutting stock and strip packing. Computers & Operations Research 39(2):200–212, DOI 10.1016/j.cor.2011.03.011

Egeblad J, Pisinger D (2009) Heuristic approaches for the two- and three-dimensional knapsack packing problem. Computers & Operations Research 36(4):1026–1049, DOI 10.1016/j.cor.2007.12.004

Fekete SP, Schepers J (2004) A general framework for bounds for higher-dimensional orthogonal packing problems. Mathematical Methods of Operations Research 60(2):311–329, DOI 10.1007/s001860400376

Furini F, Malaguti E, Thomopulos D (2016) Modeling Two-Dimensional Guillotine Cutting Problems via Integer Programming. INFORMS Journal on Computing 28(4):736–751, DOI 10.1287/ijoc.2016.0710

Gilmore PC, Gomory RE (1965) Multistage Cutting Stock Problems of Two and More Dimensions. Operations Research 13(1):94–120, DOI 10.1287/opre.13.1.94

Gilmore PC, Gomory RE (1966) The Theory and Computation of Knapsack Functions. Operations Research 14(6):1045–1074, DOI 10.1287/opre.14.6.1045

Herz JC (1972) Recursive Computational Procedure for Two-dimensional Stock Cutting. IBM Journal of Research and Development 16(5):462–469, DOI 10.1147/rd.165.0462

Table 11: Type of the new partial solution generated by vertically merging two partial solutions.

		0-stg	1-stg			2-stg						3-stg											
		o	x	y	z	xy	xz	yx	yz	zx	zy	xyx	xyz	xzx	xzy	yxy	yxz	yzx	yzy	zxy	zxz	zyx	zyz
0-stg	o	z	xz	yz	z	xyz	xz	yxz	yz	zxz	zyz		xyz				yxz				zxz		zyz
1-stg	x	xz	xz	xyz,yxz	xz	xyz	xz	yxz	xyz,yxz	zxz			xyz				yxz				zxz		
	y	yz	xyz,yxz	yz	yz	xyz	xyz,yxz	yxz	yz		zyz		xyz				yxz						zyz
	z	z	xz	yz	z	xyz	xz	yxz	yz	zxz	zyz		xyz				yxz				zxz		zyz
2-stg	xy	xyz	xyz	xyz	xyz	xyz	xyz		xyz				xyz										
	xz	xz	xz	xyz,yxz	xz	xyz	xz	yxz	xyz,yxz	zxz			xyz				yxz				zxz		
	yx	yxz	yxz	yxz	yxz		yxz	yxz	yxz								yxz						
	yz	yz	xyz,yxz	yz	yz	xyz	xyz,yxz	yxz	yz		zyz		xyz				yxz						zyz
	zx	zxz	zxz		zxz		zxz			zx											zxz		
	zy	zyz		zyz	zyz				zyz		zyz												zyz
3-stg	xyx																						
	xyz	xyz	xyz	xyz	xyz	xyz	xyz		xyz				xyz										
	xzx																						
	xzy																						
	yxy																						
	yxz	yxz	yxz	yxz	yxz		yxz	yxz	yxz								yxz						
	yzx																						
	yzy																						
	zxy																						
	zxz	zxz	zxz		zxz		zxz			zxz											zxz		
	zyx																						
	zyz	zyz		zyz	zyz				zyz		zyz												zyz

Hifi M (2002) Approximate algorithms for the container loading problem. International Transactions in Operational Research 9(6):747–774, DOI 10.1111/1475-3995.00386

Hifi M (2004) Exact algorithms for unconstrained three-dimensional cutting problems: A comparative study. Computers & Operations Research 31(5):657–674, DOI 10.1016/S0305-0548(03)00019-4

Junqueira L, Morabito R, Sato Yamashita D (2012a) MIP-based approaches for the container loading problem with multi-drop constraints. Annals of Operations Research 199(1):51–75, DOI 10.1007/s10479-011-0942-z

Junqueira L, Morabito R, Sato Yamashita D (2012b) Three-dimensional container loading models with cargo stability and load bearing constraints. Computers & Operations Research 39(1):74–85, DOI 10.1016/j.cor.2010.07.017

Liu S, Tan W, Xu Z, Liu X (2014) A tree search algorithm for the container loading problem. Computers and Industrial Engineering 75(1):20–30, DOI 10.1016/j.cie.2014.05.024

Lodi A, Monaci M (2003) Integer linear programming models for 2-staged two-dimensional Knapsack problems. Mathematical Programming 94(2-3):257–278, DOI 10.1007/s10107-002-0319-9

Macedo R, Alves C, Valério de Carvalho J (2010) Arc-flow model for the two-dimensional guillotine cutting stock problem. Computers & Operations Research 37(6):991 – 1001, DOI 10.1016/j.cor.2009.08.005

Martello S, Pisinger D, Vigo D, Boef ED, Korst J (2007) Algorithm 864. ACM Transactions on Mathematical Software 33(1):7–es, DOI 10.1145/1206040.1206047

Martin M, Morabito R, Munari P (2019) A bottom-up packing approach for modeling the constrained two-dimensional guillotine placement problem. Computers & Operations Research DOI 10.1016/j.cor.2019.104851

Martin M, Birgin EG, Lobato RD, Morabito R, Munari P (2020) Models for the two-dimensional rectangular single large placement problem with guillotine cuts and constrained pattern. International Transactions in Operational Research 27(2):767–793, DOI 10.1111/itor.12703

Morabito R, Arenales M (1994) An and/or-graph approach to the container loading problem. International Transactions in Operational Research 1(1):59–73, DOI 10.1111/1475-3995.d01-8

Oliveira JF, Ferreira JS (1990) An improved version of Wang's algorithm for two-dimensional cutting problems. European Journal of Operational Research 44(2):256–266, DOI 10.1016/0377-2217(90)90361-E

Scheithauer G (1992) Algorithms for the container loading problem. In: Gaul W, Bachem A, Habenicht W, Runge W, Stahl WW (eds) Operations Research Proceedings 1991, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 445–452

Silva E, Alvelos F, Valério de Carvalho JM (2010) An integer programming model for two- and three-stage two-dimensional cutting stock problems. European Journal of Operational Research 205(3):699–708, DOI 10.1016/j.ejor.2010.01.039

Silva EF, Toffolo TAM, Wauters T (2019) Exact methods for three-dimensional cutting and packing: A comparative study concerning single container problems. Computers & Operations Research 109:12–27, DOI 10.1016/j.cor.2019.04.020

Toffolo TA, Esprit E, Wauters T, Vanden Berghe G (2017) A two-dimensional heuristic decomposition approach to a three-dimensional multiple container loading problem. European Journal of Operational Research 257(2):526–538, DOI 10.1016/j.ejor.2016.07.033

Vasko FJ (1989) A computational improvement to wang's two-dimensional cutting stock algorithm. Computers & Industrial Engineering 16(1):109 – 115, DOI 10.1016/0360-8352(89)90013-2

Wang PY (1983) Two Algorithms for Constrained Two-Dimensional Cutting Stock Problems. Operations Research 31(3):573–586, DOI 10.1287/opre.31.3.573

Wäscher G, Haußner H, Schumann H (2007) An improved typology of cutting and packing problems. European Journal of Operational Research 183(3):1109–1130, DOI 10.1016/j.ejor.2005.12.047

Yanasse HH, Morabito R (2006) Linear models for 1-group two-dimensional guillotine cutting problems. International Journal of Production Research 44(17):3471–3491, DOI 10.1080/00207540500478603

Yanasse HH, Morabito R (2008) A note on linear models for two-group and three-group two-dimensional guillotine cutting problems. International Journal of Production Research 46(21):6189–6206, DOI 10.1080/00207540601011543

Zhao X, Bennell JA, Bektaş T, Dowsland K (2016) A comparative review of 3D container loading algorithms. International Transactions in Operational Research 23(1-2):287–320, DOI 10.1111/itor.12094