Two-stage two-dimensional guillotine cutting problems with usable leftovers*

R. Andrade † E. G. Birgin† R. Morabito ‡

July 11, 2013

Abstract

In this study we are concerned with the non-exact two-stage two-dimensional guillotine cutting problem considering usable leftovers, in which stock plates remainders of the cutting patterns (non-used material or trim loss) can be used in the future, if they are large enough to fulfill future demands of items (ordered smaller plates). This cutting problem can be characterized as a residual bin-packing problem because of the possibility of putting back into stock residual pieces, since the trim loss of each cutting/packing pattern does not necessarily represent waste of material depending on its size. Two bilevel mathematical programming models to represent this non-exact two-stage two-dimensional residual bin-packing problem are presented. The models basically consist on cutting/packing the ordered items using a set of plates of minimum cost and, among all possible solutions of minimum cost, choosing one that maximizes the value of the generated usable leftovers. Because of special characteristics of these bilevel models, they can be reformulated as one-level mixed integer programming models. Results of some numerical experiments are presented to show that the models represent appropriately the problem and to illustrate their performances.

Key words: Two-stage two-dimensional guillotine cutting, residual bin-packing problem, residual cutting-stock problem, bilevel programming, MIP models, leftovers.

1 Introduction

Cutting problems are often found in different industrial processes where paper and aluminium rolls, glass and fibreglass plates, metal bars and sheets, hardboards, pieces of leather and cloth, etc., are cut in order to produce smaller pieces of ordered sizes and quantities. These problems are closely related to packing problems and they basically consist on determining the “best” way to cut large stock objects to produce small ordered items so that one or more objectives are optimized. Cutting and packing problems have been widely studied in the literature in the last decades. They appear in different classes and practical applications, as discussed in several surveys (Dyckhoff 1990, Dowsland and Dowsland 1992, Lodi et al. 2002, Wäscher et al. 2007), annotated bibliographies (Sweeney and Paternoster 1992, Dyckhoff et al. 1997), and special issues (Dyckhoff and Wäscher 1990, Martello

*This work was supported by PRONEX-CNPq/FAPERJ E-26/111.449/2010-APQ1, FAPESP (2010/10133-0, 2013/05475-7, and 2013/07375-0), and CNPq.

†Department of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, Rua do Matão, 1010, Cidade Universitária, 05508-090, São Paulo, SP, Brazil. e-mail: {randrade | egbirgin}@ime.usp.br

‡Department of Production Engineering, Federal University of São Carlos, Via Washington Luiz km. 235, 13565-905, São Carlos, SP - Brazil. e-mail: morabito@ufscar.br

1994a,b, Bischoff and Wäescher 1995, Arenales et al. 1999, Wang and Wäscher 2002, Hifi 2002, Oliveira and Wäescher 2007, Morabito et al. 2009, Kendall et al. 2010).

In a number of industrial cutting processes as, for example, in furniture and hardboard companies, the cutting equipment is able to produce only guillotine cuts on the plates (Gilmore and Gomory 1965, Farley 1983, Yanasse et al. 1991, Carnieri et al. 1994, Morabito and Arenales 2000, Morabito and Belluzzo 2005). A guillotine cut on a plate is a cut from one edge of the plate to the opposite edge, parallel to the remaining edge. In other words, the cut is of guillotine type if when applied to a rectangle it produces two new rectangles. Depending on the cutting equipment, the feasible two-dimensional cutting patterns for the plates can be produced by guillotine cuts in at most two stages. In the first stage, parallel longitudinal (horizontal) guillotine cuts are produced on a plate, without moving it, to produce a set of strips. In the second stage, these strips are pushed, one by one, and the remaining parallel transversal (vertical) guillotine cuts are made on each strip (see Figure 1). If there is no need for additional trimming (i.e. all items have the same height in each strip), the cutting pattern is called exact two-stage guillotine (Figure 1a); otherwise, it is called non-exact (Figure 1b). Other studies dealing with cutting problems in furniture and hardboard industries are found in, e.g. Foronda and França (1991), Yanasse et al. (1991), Morabito and Garcia (1998), Gramani and Franca (2006), Gramani et al. (2009), Macedo et al. (2010), Silva et al. (2010) and Alem and Morabito (2012).

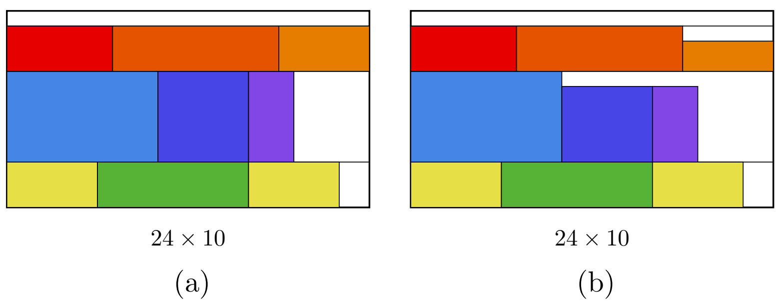

Figure 1: Two-stage cutting pattern: (a) exact case, (b) non-exact case.

In this study we deal with the non-exact two-stage two-dimensional guillotine cutting problem of how to cut a set of rectangular objects with known sizes and quantities to make exactly a set of rectangular items with specified sizes and demands to be fulfilled. We assume that the assortment of ordered items can be strongly heterogeneous, i.e. the set of items can be characterized by the fact that only very few items are of identical size. We are particularly concerned with the special case of the problem in which the non-used material of the cutting patterns (object remainder or leftover) can be used in the future, if it is large enough to fulfill future items demands. In other words, we consider that the trim loss of a cutting pattern does not necessarily represent waste of material. If this trim loss is of a reasonable size, it can be stocked and used again as input (then called a residual piece or a retail or a leftover) in subsequent cutting processes. Otherwise, if the trim loss is considered too small to be used in the future, it represents material waste and is discarded as scrap. Therefore, the problem is seen as a non-exact two-stage two-dimensional guillotine cutting problem with usable leftovers. Note that the assortment of stock objects of this problem is considered heterogeneous as different leftovers of previous cutting processes are put back into stock.

According to the typology of Wäscher et al. (2007), this cutting problem can be characterized as a "residual bin-packing problem" because of the heterogeneity of the stock objects (based on the possibility of putting back into stock new residual pieces) and the assumption of strongly heterogeneous assortment of small items. Otherwise, if the assortment of items were weakly heterogeneous, the problem would be considered as a "residual cutting-stock problem". In fact, the models presented in this study apply to both problems, since identical items are treated differently than non-identical items in order to avoid symmetric solutions. We observe that the simple objective of minimizing the cost or trim loss of the objects cut may not be appropriate for this problem. The use of leftovers in cutting and packing problems was apparently first discussed in Brown (1971), but studies dealing with this subject began mainly after the work of Dyckhoff (1981). One-dimensional cutting/packing problems that allow the provision of residual pieces have been studied by different authors, such as in Roodman (1986), Scheithauer (1991), Gradisar et al. (1999), Sinuany-Stern and Weiner (1994), Gradisar and Trkman (2005), Trkman and Gradisar (2007), Cherri et al. (2009, 2012), Dimitriadis and Kehris (2009), Cui and Yang (2010), Gradisar et al. (2011), Bang-Jensen and Larsen (2012). Examples of applications of one-dimensional cutting problems with usable leftovers were reported in, e.g. the textile industry (Gradisar et al. 1997), the agricultural light aircraft manufacturing (Abuabara and Morabito 2009), and the wood-processing industry (Koch et al. 2009). To the best of our knowledge, all studies reported in the literature focused in one-dimensional residual bin-packing problems, in which only one dimension (e.g., the widths of the objects and items) is relevant. We are not aware of other studies dealing with residual bin-packing problems involving two or more dimensions.

The paper is organized as follows. In Section 2 we present two MIP models for the two-stage two-dimensional bin-packing problem without considering leftovers. These models are highly based on models introduced in Lodi and Monaci (2003) for the two-stage two-dimensional knapsack problem. Then, in Section 3, we present two bilevel models for the two-stage two-dimensional residual bin-packing problem and their one-level MIP reformulations. In Section 4 we report and analyse the numerical results obtained by solving the models using the branch-and-cut method of CPLEX. Finally, in Section 5 we present concluding remarks and discuss perspectives for future research.

2 Two-stage bin-packing models without leftovers

In this section we present two MIP models for the non-exact two-stage two-dimensional bin-packing problem without considering leftovers. These models are straightforward extensions of models M1 and M2 introduced in Lodi and Monaci (2003) for the two-stage two-dimensional knapsack problem. Other two-stage two-dimensional knapsack models could be considered as, for example, the ones discussed in Yanasse and Morabito (2012). Only a few studies are found in the literature dealing with two-stage two-dimensional bin-packing problems. Most of the studies are concerned with either the two-stage two-dimensional knapsack problem or the two-stage two-dimensional cutting stock problem. An example is the method developed by Gilmore and Gomory (1965), based on the simplex method with a column generation procedure to generate two-stage cutting patterns. This procedure involves two phases. In the first phase cutting patterns are determined for each longitudinal strip, while the second phase decides how many times each strip should be used. This method works well if the assortment of ordered items is weakly heterogeneous, i.e. the small items can be grouped into relatively few classes and the quantity of items in each class is "sufficiently" large, as it is the case of two-stage two-dimensional cutting stock problems (Wäscher et al. 2007).

Solution approaches for two-stage two-dimensional cutting stock problems based on two phases are common in the literature, as for example in Farley (1983), Riehme et al. (1996), Hifi (1997), Morabito and Garcia (1998), Yanasse and Katsurayama (2005). For the case in which the set of items has few items of identical size, authors have proposed alternatives for the rounding of relaxed solutions of the simplex method (Wäscher and Gau 1996, Poldi and Arenales 2006) or greedy heuristics combined with column generation procedures (Hinxman 1980, Poldi and Arenales 2009) that may not work well for two-stage two-dimensional bin-packing problems. Other studies dealing with stage guillotine cutting problems can be found in Beasley (1985), Hadjiconstantinou and Christofides (1995), Morabito and Arenales (1996), Riehme et al. (1996), Hifi (1997), Hifi and Roucairol (2001), Lodi and Monaci (2003), Cui et al. (2005), Belov and Scheithauer (2006), Cintra et al. (2008), Silva et al. (2010), Morabito and Pureza (2010).

2.1 Two-stage bin-packing model $\mathcal{M}_1$

Let us consider p large rectangular objects, each object ℓ with width W_ℓ, height H_ℓ and cost per unit of area c_ℓ ($\ell = 1, \dots, p$) and n small rectangular items, each item i with width w_i and height h_i ($i = 1, \dots, n$). The non-exact two-stage two-dimensional bin-packing problem can be defined as the problem of packing (cutting) all n items into (from) a chosen subset of objects, so that the obtained packing (cutting) pattern for each object is feasible (the packed items do not overlap and fit inside the object according to a non-exact two-stage guillotine pattern) and the cost of the used objects is minimized. We consider only the case in which the first stage cuts on the plate are horizontal, i.e., they are parallel to the object width. No item rotations are allowed and there are no other constraints related to the positioning of the items within the objects.

Without loss of generality, we assume that $h_1 \geq h_2 \geq \cdots \geq h_n$. We also assume that the cuts on the objects are infinitely thin; otherwise, we consider that the saw thickness was added to the dimensions of the objects and items, without loss of generality (Gilmore and Gomory 1965, Morabito and Arenales 2000). Moreover, we assume that all dimensions of the objects and items, as well as the objects unit costs, are integer numbers. This is not a very restrictive assumption to deal with problem instances in practice since, due to the finite precision of the cutting and measuring tools and due to the finite precision used in any currency considered to define the objects' costs, they can be easily satisfied by a change of scale. The model presented below can be seen as a simple extension of the two-stage knapsack model M1 introduced in Lodi and Monaci (2003, p.261) to deal with the two-stage bin-packing problem. The original objective function of the model is modified in order to appropriately consider more than one object. For this, we define the binary variables u_ℓ ($\ell = 1, \dots, p$), which indicate if object ℓ is used or not:

$$u_\ell = \begin{cases} 1, & \text{if object } \ell \text{ is used,} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The other binary variables $x_{ik\ell}$ ($k = 1, \dots, n,\ i = k, \dots, n,\ \ell = 1, \dots, p$) indicate the object and strip from which the item is cut:

$$x_{ik\ell} = \begin{cases} 1, & \text{if item } i \text{ is cut from strip } k \text{ of object } \ell, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Following the terminology used in Lodi and Monaci (2003), given an object ℓ, a shelf is defined as a strip of the object with width W_ℓ and height equal to the height of the heightest item packed (cut) in (from) it. It is assumed that all items in a shelf have their bottom (inferior side) on the shelf floor. The roof of the shelf, determined by the top (superior side) of the item of largest height, defines the floor of the next shelf. The shelf concept is illustrated in Figure 2. In this figure, there is one object and three shelves: 1, 4, and 6. Items 1, 2, 3 and 5 are in shelf 1, items 4 and 7 are in shelf 4, and items 6, 8 and 9 are in shelf 6. Note that the number of each shelf is defined as the number of the first item packed in it. In the model presented below, we consider that we can have up to n shelves, each one defined by an item. We say that a shelf k is *open* (or used) if item k is the smallest-index item assigned to (or packed in) the shelf. In this case, if item k is on shelf k

and shelf k is assigned to object ℓ, we have $x_{kk\ell} = 1$. Note that any optimal two-stage cutting pattern has an equivalent cutting pattern where the item of largest height in each shelf is the first item placed to the left of the shelf (as depicted in Figure 2). A feasible two-stage guillotine cutting pattern is composed of shelves and each item allocated to a shelf is cut in at most two stages (plus the *trimming*). A model named $\mathcal{M}_1$ for the non-exact two-stage bin-packing problem (without leftovers) is given by:

$$
\begin{array}{llll}
\underset{u,x}{\text{Min}} & \sum_{\ell=1}^{p} c_\ell W_\ell H_\ell u_\ell & & \quad (3)\\
\text{s.t.} & \sum_{k=1}^{n} h_k x_{kk\ell} \leq H_\ell u_\ell, & \ell = 1,\ldots,p, & \quad (4)\\
& \sum_{i=k+1}^{n} w_i x_{ik\ell} \leq (W_\ell - w_k) x_{kk\ell}, & k = 1,\ldots,n,\ \ell = 1,\ldots,p, & \quad (5)\\
& \sum_{\ell=1}^{p} \sum_{k=1}^{i} x_{ik\ell} = 1, & i = 1,\ldots,n, & \quad (6)\\
& \sum_{\ell=1}^{p} x_{k+1,k+1,\ell} \leq \sum_{\ell=1}^{p} x_{kk\ell}, & j = 1,\ldots,m,\ k \in [\alpha_{j-1}+1, \alpha_j - 1], & \quad (7)\\
& \sum_{\ell=1}^{p} \sum_{i=k+2}^{\alpha_j} x_{i,k+1,\ell} \leq \sum_{\ell=1}^{p} \sum_{i=k+1}^{\alpha_j} x_{ik\ell}, & j = 1,\ldots,m,\ k \in [\alpha_{j-1}+1, \alpha_j - 1], & \quad (8)\\
& u_\ell \in \{0,1\}, & \ell = 1,\ldots,p, & \quad (9)\\
& x_{ik\ell} \in \{0,1\}, & k = 1,\ldots,n,\ i = k,\ldots,n,\ \ell = 1,\ldots,p. & \quad (10)
\end{array}
$$

The objective function (3) minimizes the total cost of the objects used (note that, if $c_\ell = 1$ for all ℓ, (3) minimizes the total object area cut). Constraint (4) ensures that, for each used object, the sum of the heights of the open shelves is not greater than the object height, and that open shelves are attributed to used objects only. Constraint (5) ensures that, for each object, the sum of the widths of the items allocated to each shelf is not greater than the object width, and that an item can be allocated to a shelf only if the shelf is open. Constraint (6) ensures that the demand of each item is met. Constraints (7) and (8) are redundant symmetry-breaking constraints and refer to identical items. Without loss of generality, we assume that identical items (items with the same width and height) are numbered consecutively. We also assume that there are m different types of items and we define $\alpha_0 \equiv 0$ and α_j as last index of items of the j-th type. It means that the indices of items of type j range from $\alpha_{j-1}+1$ to α_j. Symmetry-breaking constraint (7) says that an item that is not the first of its type can open a shelf only if the previous item (of the same type) opens a shelf too. Symmetry-breaking constraint (8) says that if two consecutive items k and $k+1$ of the j-th type open a shelf, the number of items of type j on shelf $k+1$ must be less than or equal to the number of items of type j on shelf k. Constraints (9) and (10) define the domain of variables u_ℓ and $x_{ik\ell}$.

2.2 Two-stage bin-packing model $\mathcal{M}_2$

In Lodi and Monaci (2003, p.262) another model for the non-exact two-stage two-dimensional knapsack problem was presented, named M2, which considers identical items as items of the same

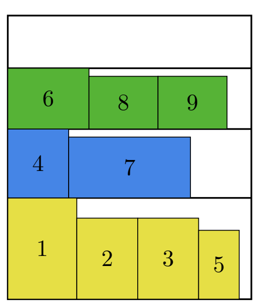

Figure 2: Illustration of the concept of shelves.

group or type. In the following we present a slightly modified version of this model to solve the two-stage two-dimensional bin-packing problem. The model assumes that there are n items of m different types. Items of type i have width $\bar{w}_i$ and height $\bar{h}_i$. We assume, without loss of generality, that $\bar{h}_1 \geq \bar{h}_2 \geq \cdots \geq \bar{h}_m$. The demanded quantity of items of type i is given by b_i. The additional model parameters (that can be easily computed from the other ones) are: α_i $(i = 0, \ldots, m)$ and β_k $(k = 1, \ldots, n)$, with $\alpha_0 \equiv 0$, $\alpha_i \equiv \sum_{s=1}^{i} b_s$ $(i = 1, \ldots, m)$ (that coincides with the definition introduced in the previous subsection), and $\beta_k \equiv \min\{i \mid \alpha_i \geq k\}$ $(k = 1, \ldots, n)$. Note that: (i) any item of type i can be packed in any shelf in the interval $[1, \alpha_i]$, i.e. α_i indicates the highest shelf index to allocate items of type i and (ii) indices in the interval $[\alpha_{i-1} + 1, \alpha_i]$ can be interpreted as the indices of the shelves characterized by items of type i. Moreover, each shelf k can pack items of types $[\beta_k, m]$, i.e. parameter β_k can be seen as the index of the item type of largest height (lowest index) that can be allocated in shelf k. In other words, β_k is the index of the item type that defines shelf k. Note that there is a shelf for each item.

The former variables $x_{ik\ell}$ $(k = 1, \ldots, n,\ i = k, \ldots, n,\ \ell = 1, \ldots, p)$ are also used in this model, but with a slightly different meaning. Now they are integer (instead of binary) and relate items and shelves in the following way:

$$x_{ik\ell} = \begin{cases} \text{quantity of items of type } i \text{ allocated to shelf } k \text{ of object } \ell, & \text{if } i \neq \beta_k, \\ \text{quantity of additional items of type } i \text{ (other than the one} \\ \text{that defines the shelf) allocated to shelf } k \text{ of object } \ell, & \text{if } i = \beta_k, \end{cases} \tag{11}$$

with $i = 1, \ldots, m,\ k \in [1, \alpha_i],\ \ell = 1, \ldots, p$. The model also considers the binary variables $q_{k\ell}$ $(k = 1, \ldots, n,\ \ell = 1, \ldots, p)$, which indicate if the shelf is open (used) or not:

$$q_{k\ell} = \begin{cases} 1, & \text{if shelf } k \text{ of object } \ell \text{ is open,} \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

The remaining model parameters and variables are the same as the model of the previous section. The second model for the two-stage bin-packing problem (without leftovers), named $\mathcal{M}_2$, is given by:

$$\begin{array}{llcll} \min\limits_{u,x,q} & \sum_{\ell=1}^{p} c_\ell W_\ell H_\ell u_\ell & & & (13) \\ \text{s.t.} & \sum_{k=1}^{n} \bar{h}_{\beta_k} q_{k\ell} & \leq & H_\ell u_\ell, \quad \ell = 1, \ldots, p, & (14) \\ & \sum_{i=\beta_k}^{m} \bar{w}_i x_{ik\ell} & \leq & (W_\ell - \bar{w}_{\beta_k}) q_{k\ell}, \quad k = 1, \ldots, n,\ \ell = 1, \ldots, p, & (15) \\ & \sum_{\ell=1}^{p} \left(\sum_{k=1}^{\alpha_i} x_{ik\ell} + \sum_{k=\alpha_{i-1}+1}^{\alpha_i} q_{k\ell} \right) & = & b_i, \quad i = 1, \ldots, m, & (16) \\ & \sum_{\ell=1}^{p} q_{k\ell} & \leq & 1, \quad k = 1, \ldots, n, & (17) \\ & \sum_{\ell=1}^{p} q_{k+1,\ell} & \leq & \sum_{\ell=1}^{p} q_{k\ell}, \quad i = 1, \ldots, m,\ k \in [\alpha_{i-1} + 1, \alpha_i - 1], & (18) \\ & \sum_{\ell=1}^{p} x_{i,k+1,\ell} & \leq & \sum_{\ell=1}^{p} x_{ik\ell}, \quad i = 1, \ldots, m,\ k \in [\alpha_{i-1} + 1, \alpha_i - 1], & (19) \\ & \sum_{\ell=1}^{p} \sum_{s=k}^{\alpha_i} x_{is\ell} & \leq & b_i - (k - \alpha_{i-1}), i = 1, \ldots, m,\ k \in [\alpha_{i-1} + 1, \alpha_i], & (20) \\ & x_{ik\ell} & \leq & b_i, \quad i = 1, \ldots, m,\ k \in [1, \alpha_i],\ \ell = 1, \ldots, p, & (21) \\ & u_\ell & \in & \{0, 1\}, \quad \ell = 1, \ldots, p, & (22) \\ & x_{ik\ell} & \in & \mathbb{N}_{\geq 0}, \quad i = 1, \ldots, m,\ k \in [1, \alpha_i],\ \ell = 1, \ldots, p, & (23) \\ & q_{k\ell} & \in & \{0, 1\}, \quad k = 1, \ldots, n,\ \ell = 1, \ldots, p. & (24) \end{array}$$

Constraint (14) ensures that, for each object, the sum of the heights of the open shelves is less than or equal to the object height and that shelves are opened in used objects only. Constraint (15) ensures that, for each object, the sum of the widths of the items allocated to each shelf is not greater than the object width. Constraint (16) ensures that the demand of each item is met. Constraint (17) ensures that each shelf can be opened only once (i.e. we cannot have a shelf k opened in two different objects). Constraints (18) and (19) are redundant symmetry-breaking constraints equivalent to (7) and (8) from model $\mathcal{M}_1$. Constraint (20) is a redundant constraint considered in Lodi and Monaci (2003) to improve the quality of the lower bounds of the LP relaxation of the model (removing the integrality constraints). Constraints (21–23) define the domain of the variables.

Model $\mathcal{M}_1$ has $2p$ continuous variables, $2p+np/2+n^2p/2$ binary variables, and $p+np+3n-2m$ constraints. Model $\mathcal{M}_2$ has $2p$ continuous variables, $2p+np$ binary variables, $p\sum_{k=0}^{m}\alpha_k$ integer variables, and $5p+np+4n-m+p\sum_{k=0}^{m}\alpha_k$ constraints. Note that both models have the same quantity of continuous variables, but differ in terms of the number of binary and integer variables. Model $\mathcal{M}_1$ has $O(n^2p)$ binary variables, while model $\mathcal{M}_2$ has only $O(np)$. Model $\mathcal{M}_1$ does not have integer variables, whereas model $\mathcal{M}_2$ has $p\sum_{k=0}^{m}\alpha_k$. Analyzing $\sum_{k=0}^{m}\alpha_k$, we have the following extreme cases: (i) if $m=1$, then $\sum_{k=0}^{m}\alpha_k=n$, (ii) if $m=n$ and $\bar{h}_1>\bar{h}_2>\cdots>\bar{h}_m$, then $\sum_{k=0}^{m}\alpha_k=n(n+1)/2$, (iii) if $m=n$ and $\bar{h}_1=\bar{h}_2=\cdots=\bar{h}_m$, then $\sum_{k=0}^{m}\alpha_k=mn=n^2$. In this way, in the worst case, model $\mathcal{M}_2$ has $O(n^2p)$ integer variables. Considering the constraints, we observe that model $\mathcal{M}_1$ has $O(np)$ constraints. Model $\mathcal{M}_2$ has $O(n^2p)$ constraints in the worst case. Note that in cases where m is a relatively small number compared to n, model $\mathcal{M}_2$ should have an amount of integer variables and constraints proportional to np. Therefore, as the ratio m/n decreases, we expect that model $\mathcal{M}_2$ becomes easier to solve than model $\mathcal{M}_1$.

3 Two-stage residual bin-packing models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$

In this section we present models for the two-stage bin-packing problem considering leftovers. We consider leftover as any trim loss of height not smaller than $d_{\min}$, obtained after producing the first-stage horizontal cuts in the object. We look for a solution that minimizes the costs of the used objects and, among all minimum cost solutions, we look for one that maximizes the sum of the values of the leftovers. As illustrative example, consider an instance with $p=2$ identical objects with $W_1=W_2=H_1=H_2=9$ and $c_1=c_2=1$, and $n=8$ items with $w_1=\cdots=w_4=4$, $w_5=\cdots=w_8=3$, $h_1=\cdots=h_6=4$, and $h_7=h_8=2$. Figures 3(a–b) represent two different feasible solutions (to models $\mathcal{M}_1$ and $\mathcal{M}_2$ that do not consider leftovers) with cost 162. Since the sum of the areas of the eight demanded items is larger than the area of a single object, at least two objects are needed to cut the items and, hence, both depicted feasible solutions are optimal. However, there is a feature that differentiate these two optimal solutions and that is not being captured by models $\mathcal{M}_1$ and $\mathcal{M}_2$. If we consider $d_{\min}$ equal to the smallest height of a demanded item, i.e. $d_{\min}=2$, we have that the solution depicted on Figure 3(b) has a leftover in one of its objects, while the solution depicted on Figure 3(a) has no (usable) leftovers.

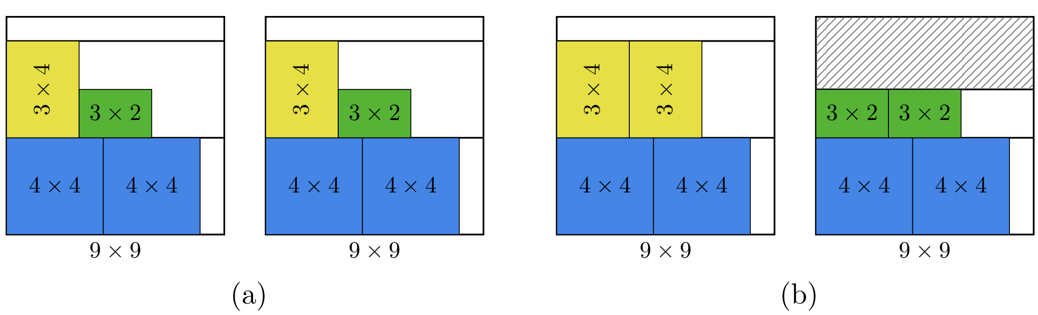

Figure 3: Illustration of the concept of leftovers in the two-stage bin-packing problem.

A natural modeling approach for considering leftovers would be to consider a bilevel mathematical programming problem. A bilevel mathematical programming problem (see, for example, (Dempe 2002)) is an optimization problem that maximizes or minimizes an objective function with some of the problem variables restricted to be a solution to another optimization problem. The two models presented below for the two-stage residual bin-packing problem are bilevel models with integer and continuous variables and linear objective functions and constraints. Before presenting the models, we explain how to model the leftovers. Let us consider s_ℓ as the height of the trim loss of object $\ell = 1, \dots, p$. The trim loss of object ℓ is considered as a leftover (i.e. a residual piece) if its height s_ℓ is such that $s_\ell \geq d_{\min}$, where $d_{\min} \geq 0$ is a given parameter. If the trim loss is not a leftover, it is considered as a waste (i.e. a scrap) and its value is null. Therefore, to model the leftovers area we define function $\bar{s}(s_\ell)$ as:

$$\bar{s}(s_\ell) = \begin{cases} W_\ell s_\ell, & \text{if } s_\ell \geq d_{\min}, \\ 0, & \text{otherwise.} \end{cases} \tag{25}$$

The value of the leftover is defined as its area times its unit area value, given by $\bar{c}_\ell > 0$. It would be reasonable to consider $\bar{c}_\ell \equiv c_\ell$ for $\ell = 1, \dots, p$, i.e. to set the unit area value of a leftover as the unit area cost of the corresponding object. Note that, in practice, if the unit area value of the leftovers is independent of the objects from which they were cut, one may simply consider $\bar{c}_\ell = 1$ for all ℓ.

We can use (25) as constraints of a MIP by applying the big-M technique. To simplify the explanation, we first present a bilevel model based on model $\mathcal{M}_1$ to represent the two-stage residual bin-packing problem. Then we discuss how to use (25) as MIP constraints, as well as the remaining details of the model:

$$
\begin{array}{llllr}
\underset{z,T}{\text{Max}} & \sum_{\ell=1}^{p} \bar{c}_\ell T_\ell & & & (26) \\
\text{s.t.} & d_{\min} & \leq & s_\ell + M_\ell z_\ell, \quad \ell = 1, \dots, p, & (27) \\
& s_\ell & \leq & d_{\min} + M_\ell (1 - z_\ell), \quad \ell = 1, \dots, p, & (28) \\
& T_\ell & \leq & s_\ell W_\ell + \bar{M}_\ell z_\ell, \quad \ell = 1, \dots, p, & (29) \\
& T_\ell & \leq & \bar{M}_\ell (1 - z_\ell), \quad \ell = 1, \dots, p, & (30) \\
& T_\ell & \geq & 0, \quad \ell = 1, \dots, p, & (31) \\
& z_\ell & \in & \{0, 1\}, \quad \ell = 1, \dots, p, & (32) \\
& (u, x, s) & \in & \underset{u', x', s'}{\operatorname{argmin}} \sum_{\ell=1}^{p} c_\ell W_\ell H_\ell u'_\ell & (33)
\end{array}
$$

$$
\begin{array}{llllr}
\text{s.t.} & \sum_{k=1}^{n} h_k x'_{kk\ell} + s'_\ell & = & H_\ell u'_\ell, \quad \ell = 1, \dots, p, & (34) \\
& \sum_{i=k+1}^{n} w_i x'_{ik\ell} & \leq & (W_\ell - w_k) x'_{kk\ell}, \quad k = 1, \dots, n, \ \ell = 1, \dots, p, & (35) \\
& \sum_{\ell=1}^{p} \sum_{k=1}^{i} x'_{ik\ell} & = & 1, \quad i = 1, \dots, n, & (36) \\
& \sum_{\ell=1}^{p} x_{k+1,k+1,\ell} & \leq & \sum_{\ell=1}^{p} x_{kk\ell}, \quad j = 1, \dots, m, \ k \in [\alpha_{j-1} + 1, \alpha_j - 1], & (37) \\
& \sum_{\ell=1}^{p} \sum_{i=k+2}^{\alpha_j} x_{i,k+1,\ell} & \leq & \sum_{\ell=1}^{p} \sum_{i=k+1}^{\alpha_j} x_{ik\ell}, \ j = 1, \dots, m, \ k \in [\alpha_{j-1} + 1, \alpha_j - 1], & (38) \\
& u'_\ell & \in & \{0, 1\}, \quad \ell = 1, \dots, p, & (39) \\
& x'_{ik\ell} & \in & \{0, 1\}, \quad k = 1, \dots, n, \ i = k, \dots, n, \ \ell = 1, \dots, p, & (40) \\
& s'_\ell & \geq & 0, \quad \ell = 1, \dots, p. & (41)
\end{array}
$$

The inferior level problem (33–41) ensures that the total area cost of the objects used is minimized. The superior level problem (26–32) ensures that the sum of the values of the leftover areas generated from the objects cut is maximized, considering as feasible points the solutions to the inferior level problem that satisfy constraints (27–32). Variables s_ℓ ($\ell = 1, \dots, p$) indicate the height of the leftover of each object. Constraints (27–32), together with the objective function (26), formulate $\bar{s}(s_\ell)$ defined in (25) by means of the big-M technique. For this, we need to define parameters M_ℓ and $\bar{M}_\ell$ such that $M_\ell \geq H_\ell$ and $\bar{M}_\ell \geq H_\ell W_\ell$ ($\ell = 1, \dots, p$), and we use a binary variable z_ℓ for each object ℓ. If the height s_ℓ of the leftover of object ℓ is smaller than $d_{\min}$, the corresponding z_ℓ is forced to assume value one by (27). In this case, by (30–31), we have $T_\ell = 0$. If s_ℓ is greater than $d_{\min}$, the corresponding z_ℓ is forced to be zero by (28) and, by (29–31) and the objective function (26), T_ℓ assumes the value of the leftover area. If $s_\ell = d_{\min}$, z_ℓ may assume value zero or one and, by the objective function (26), T_ℓ assumes the value of the leftover area. Constraint (33) ensures that (u, x, s) is a solution to the inferior level problem, in which the cost of the objects used to satisfy the demand is minimized. The inferior level problem composed by the objective function (33) and constraints (34–41) is essentially model $\mathcal{M}_1$ defined in (3–10), except for constraint (34), that differs of (4) by the addition of the term s'_ℓ that incorporates the leftover height of each object, and constraint (41) that says that the leftovers heights are non-negative quantities. Constraint (34) ensures that the sum of the heights of the open shelves of an object plus the height of its leftover must be equal to the height of the object.

It should be noted that the use of an object could be prioritized in the model by simply nulling its unit area cost c_ℓ. This could be interesting to prioritize the use of leftovers (generated in previous periods) in the cutting pattern generation in order to avoid buying new objects while leftovers accumulate in stock. This situation could be accomplished by setting $c_\ell = 0$, to prioritize the use of object ℓ, while keeping in $\bar{c}_\ell$ the original unit area cost of object ℓ, to correctly represent the value of its leftover.

In the following we shortly present the second bilevel model for the two-stage residual bin-packing problem based on model $\mathcal{M}_2$. The explanation of this model follows the same steps of the presentation of the previous bilevel model based on model $\mathcal{M}_1$. The model is given by:

$$
\begin{array}{lllll}
\displaystyle\operatorname*{Max}_{z,T} & \sum_{\ell=1}^{p} \bar{c}_\ell T_\ell & & & (42)\\
\text{s.t.} & d_{\min} & \le & s_\ell + M_\ell z_\ell, \qquad \ell = 1,\dots,p, & (43)\\
& s_\ell & \le & d_{\min} + M_\ell(1 - z_\ell), \qquad \ell = 1,\dots,p, & (44)\\
& T_\ell & \le & s_\ell W_\ell + \bar{M}_\ell z_\ell, \qquad \ell = 1,\dots,p, & (45)\\
& T_\ell & \le & \bar{M}_\ell(1 - z_\ell), \qquad \ell = 1,\dots,p, & (46)\\
& T_\ell & \ge & 0, \qquad \ell = 1,\dots,p, & (47)\\
& z_\ell & \in & \{0,1\}, \qquad \ell = 1,\dots,p, & (48)\\
& (u,x,q,s) & \in & \displaystyle\operatorname*{argmin}_{u',x',q',s'} \sum_{\ell=1}^{p} c_\ell W_\ell H_\ell u'_\ell & (49)
\end{array}
$$

$$
\begin{array}{lllll}
\text{s.t.} & \sum_{k=1}^{n} \bar{h}_{\beta_k} q'_{k\ell} + s'_\ell & = & H_\ell u'_\ell, \qquad \ell = 1,\dots,p, & (50)\\
& \sum_{i=\beta_k}^{m} \bar{w}_i x'_{ik\ell} & \le & (W_\ell - \bar{w}_{\beta_k}) q'_{k\ell}, \qquad k = 1,\dots,n,\ \ell = 1,\dots,p, & (51)\\
& \sum_{\ell=1}^{p} (\sum_{k=1}^{\alpha_i} x'_{ik\ell} + \sum_{k=\alpha_{i-1}+1}^{\alpha_i} q'_{k\ell}) & = & b_i, \qquad i = 1,\dots,m, & (52)\\
& \sum_{\ell=1}^{p} q'_{k\ell} & \le & 1, \qquad k = 1,\dots,n, & (53)\\
& \sum_{\ell=1}^{p} q'_{k+1,\ell} & \le & \sum_{l=1}^{p} q'_{k\ell}, \quad i = 1,\dots,m,\ k \in [\alpha_{i-1}+1, \alpha_i - 1], & (54)\\
& \sum_{\ell=1}^{p} x'_{i,k+1,\ell} & \le & \sum_{\ell=1}^{p} x'_{ik\ell}, \quad i = 1,\dots,m,\ k \in [\alpha_{i-1}+1, \alpha_i - 1], & (55)\\
& \sum_{\ell=1}^{p} \sum_{s=k}^{\alpha_i} x'_{is\ell} & \le & b_i - (k - \alpha_{i-1}), i = 1,\dots,m,\ k \in [\alpha_{i-1}+1, \alpha_i], & (56)\\
& x'_{ik\ell} & \le & b_i, \qquad i = 1,\dots,m,\ k \in [1, \alpha_i],\ \ell = 1,\dots,p, & (57)\\
& u'_\ell & \in & \{0,1\}, \qquad \ell = 1,\dots,p, & (58)\\
& x'_{ik\ell} & \in & \mathbb{N}_{\ge 0}, \qquad i = 1,\dots,m,\ k \in [1, \alpha_i],\ \ell = 1,\dots,p, & (59)\\
& q'_{k\ell} & \in & \{0,1\}, \qquad k = 1,\dots,n,\ \ell = 1,\dots,p, & (60)\\
& s_\ell & \ge & 0, \qquad \ell = 1,\dots,p. & (61)
\end{array}
$$

3.1 MIP reformulations

A technique commonly applied to solve bilevel mathematical programming models is, when possible, to reformulate the model as an equivalent one-level model. In the following we discuss a simple way to reformulate the bilevel models of previous section as MIP models. For instance, for model $\mathcal{M}_1$, considering that the objective function (33) of the inferior level problem assumes only integer values (variables u_ℓ, $\ell = 1,\dots,p$, are binary and the objects unit costs c_ℓ, $\ell = 1,\dots p$, are integer numbers by hypothesis) and that it is independent of the variables of the superior level problem, the bilevel model based on model $\mathcal{M}_1$ and given by (26–41) can be reformulated as the following MIP:

$$
\begin{array}{llcll}
\underset{z,T,u,x,s}{\text{Min}} & F(T,u) \equiv \sum_{\ell=1}^{p} c_\ell W_\ell H_\ell u_\ell - \big(\sum_{\ell=1}^{p} \bar{c}_\ell T_\ell\big)/\big(\sum_{\ell=1}^{p} \bar{c}_\ell W_\ell H_\ell\big) & & & (62) \\
\text{s.t.} & d_{\min} & \leq & s_\ell + M_\ell z_\ell, \quad \ell = 1,\dots,p, & (63) \\
& s_\ell & \leq & d_{\min} + M_\ell(1 - z_\ell), \quad \ell = 1,\dots,p, & (64) \\
& T_\ell & \leq & s_\ell W_\ell + \bar{M}_\ell z_\ell, \quad \ell = 1,\dots,p, & (65) \\
& T_\ell & \leq & \bar{M}_\ell(1 - z_\ell), \quad \ell = 1,\dots,p, & (66) \\
& T_\ell & \geq & 0, \quad \ell = 1,\dots,p, & (67) \\
& z_\ell & \in & \{0,1\}, \quad \ell = 1,\dots,p, & (68) \\
& \sum_{k=1}^{n} h_k x_{kk\ell} + s_\ell & = & H_\ell u_\ell, \quad \ell = 1,\dots,p, & (69) \\
& \sum_{i=k+1}^{n} w_i x_{ik\ell} & \leq & (W_\ell - w_k) x_{kk\ell}, \quad k = 1,\dots,n,\ \ell = 1,\dots,p, & (70) \\
& \sum_{\ell=1}^{p} \sum_{k=1}^{i} x_{ik\ell} & = & 1, \quad i = 1,\dots,n, & (71) \\
& \sum_{\ell=1}^{p} x_{k+1,k+1,\ell} & \leq & \sum_{\ell=1}^{p} x_{kk\ell}, \quad j = 1,\dots,m,\ k \in [\alpha_{j-1}+1, \alpha_j - 1], & (72) \\
& \sum_{\ell=1}^{p} \sum_{i=k+2}^{\alpha_j} x_{i,k+1,\ell} & \leq & \sum_{\ell=1}^{p} \sum_{i=k+1}^{\alpha_j} x_{ik\ell}, \quad j = 1,\dots,m,\ k \in [\alpha_{j-1}+1, \alpha_j - 1], & (73) \\
& u_\ell & \in & \{0,1\}, \quad \ell = 1,\dots,p, & (74) \\
& x_{ik\ell} & \in & \{0,1\}, \quad k = 1,\dots,n,\ i = k,\dots,n,\ \ell = 1,\dots,p, & (75) \\
& s_\ell & \geq & 0, \quad \ell = 1,\dots,p. & (76)
\end{array}
$$

Note that the objective function $F(T,u)$ in (62) is composed by the objective functions of the superior and inferior levels (26) and (33), respectively, where the superior level objective function (26) was normalized in such a way as to assume only values in the interval $[0,1)$. The value of $F(T,u)$ is integer at feasible points with no leftovers, while it is rational at feasible points with leftovers. Using the value of $F(T,u)$ at a feasible point, it is easy to obtain the corresponding values of (26) and (33) as $\lceil F(T,u) \rceil$ and $(\lceil F(T,u) \rceil - F(T,u)) \sum_{\ell=1}^{p} \bar{c}_\ell W_\ell H_\ell$, respectively. Note that constraints (63–76) correspond exactly to constraints (27–32) and (34–41). From now on, we call the model given by (62–76) as model $\mathcal{M}_1^L$. Regarding the size of the model, it has $2p$ continuous variables, $2p + np/2 + n^2/2$ binary variables, and $7p + np + 3n - 2m$ constraints.

Following the same steps to define model $\mathcal{M}_1^L$, we combine (62) with constraints (43–48,50–61) and define model $\mathcal{M}_2^L$ given by:

$$
\begin{array}{llcll}
\underset{z,T,u,x,s}{\text{Min}} & F(T,u) \equiv \sum_{\ell=1}^{p} c_\ell W_\ell H_\ell u_\ell - \big(\sum_{\ell=1}^{p} \bar{c}_\ell T_\ell\big)/\big(\sum_{\ell=1}^{p} \bar{c}_\ell W_\ell H_\ell\big) & & & (77) \\
\text{s.t.} & d_{\min} & \leq & s_\ell + M_\ell z_\ell, \quad \ell = 1,\dots,p, & (78) \\
& s_\ell & \leq & d_{\min} + M_\ell(1 - z_\ell), \quad \ell = 1,\dots,p, & (79) \\
& T_\ell & \leq & s_\ell W_\ell + \bar{M}_\ell z_\ell, \quad \ell = 1,\dots,p, & (80) \\
& T_\ell & \leq & \bar{M}_\ell(1 - z_\ell), \quad \ell = 1,\dots,p, & (81) \\
& T_\ell & \geq & 0, \quad \ell = 1,\dots,p, & (82) \\
& z_\ell & \in & \{0,1\}, \quad \ell = 1,\dots,p, & (83) \\
& \sum_{k=1}^{n} \bar{h}_{\beta_k} q_{k\ell} + s_\ell & = & H_\ell u_\ell, \quad \ell = 1,\dots,p, & (84) \\
& \sum_{i=\beta_k}^{m} \bar{w}_i x_{ik\ell} & \leq & (W_\ell - \bar{w}_{\beta_k}) q_{k\ell}, \quad k = 1,\dots,n,\ \ell = 1,\dots,p, & (85) \\
& \sum_{\ell=1}^{p} (\sum_{k=1}^{\alpha_i} x_{ik\ell} + \sum_{k=\alpha_{i-1}+1}^{\alpha_i} q_{k\ell}) & = & b_i, \quad i = 1,\dots,m, & (86) \\
& \sum_{\ell=1}^{p} q_{k\ell} & \leq & 1, \quad k = 1,\dots,n, & (87) \\
& \sum_{\ell=1}^{p} q_{k+1,\ell} & \leq & \sum_{\ell=1}^{p} q_{k\ell}, \quad i = 1,\dots,m,\ k \in [\alpha_{i-1}+1, \alpha_i - 1], & (88) \\
& \sum_{\ell=1}^{p} x_{i,k+1,\ell} & \leq & \sum_{\ell=1}^{p} x_{ik\ell}, \quad i = 1,\dots,m,\ k \in [\alpha_{i-1}+1, \alpha_i - 1], & (89) \\
& \sum_{\ell=1}^{p} \sum_{s=k}^{\alpha_i} x_{is\ell} & \leq & b_i - (k - \alpha_{i-1}), i = 1,\dots,m,\ k \in [\alpha_{i-1}+1, \alpha_i], & (90) \\
& x_{ik\ell} & \leq & b_i, \quad i = 1,\dots,m,\ k \in [1, \alpha_i],\ \ell = 1,\dots,p, & (91) \\
& u_\ell & \in & \{0,1\}, \quad \ell = 1,\dots,p, & (92) \\
& x_{ik\ell} & \in & \mathbb{N}_{\geq 0}, \quad i = 1,\dots,m,\ k \in [1, \alpha_i],\ \ell = 1,\dots,p, & (93) \\
& q_{k\ell} & \in & \{0,1\}, \quad k = 1,\dots,n,\ \ell = 1,\dots,p, & (94) \\
& s_\ell & \geq & 0, \quad \ell = 1,\dots,p. & (95)
\end{array}
$$

Model $\mathcal{M}_2^L$ has $2p$ continuous variables, $2p + np$ binary variables, and $7p + 4n + np - m + p \sum_{i=1}^{m} \alpha_i$ constraints.

Models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$ deal with the the non-exact two-stage two-dimensional guillotine cutting problem. It should be noted that these models can be easily adapted to treat the exact case of this problem (i.e. without trimming; see Figure 1a) by imposing, for example, that $x_{ik\ell} = 0$ if $w_i \neq w_k$ in model $\mathcal{M}_1^L$, and that $x_{ik\ell} = 0$ if $w_i \neq \bar{w}_{\beta_k}$ in model $\mathcal{M}_2^L$, or simply removing these variables from the models. Moreover, both models can also be modified to deal with cases in which the items can be rotated by 90 degrees in the cutting/packing patterns. This can be done in model $\mathcal{M}_1^L$ by introducing for each item i a counterpart item i' for which its width and height are defined as $w_{i'} = h_i$ and $h_{i'} = w_i$ (i.e. the dimensions of items i and i' are swapped) and by adding to the formulation constraints to avoid that items i and i' are both produced from the objects cut. A similar reasoning can also be applied to the the item types of model $\mathcal{M}_2^L$.

4 Numerical experiments

In this section we present and analyze some numerical experiments with models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$. Twenty arbitrary randomly-generated instances were considered to illustrate both models. The first ten instances have strongly heterogeneous items, being representatives of the residual bin-packing problem, while the second set of ten instances corresponds to weakly heterogeneous items and represents the residual cutting-stock problem. Table 1 describes the objects and items that compose each instance. In the table, p is the number of objects and n is the number of items. From the data in the table, it is straightforward to obtain the data to build the instances for models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$ (that consider the items individually and grouped by type, respectively). In all instances, we considered $c_\ell = \bar{c}_\ell = 1$ for all ℓ. For each instance, we also considered $d_{\min}$ equal to the smallest height of the instance' demanded items. Table 2 displays the number of continuous, binary, and integer variables and the number of constraints of the twenty considered instances of models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$. In the models, we always considered big-M constants $M_\ell \equiv H_\ell$ and $\bar{M}_\ell \equiv H_\ell W_\ell$ for $\ell = 1, \dots, p$.

MIP models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$ were implemented in C/C++ using the ILOG Concert Technology 2.9 and compiled with g++ from gcc version 4.6.1 (GNU compiler collection). Numerical experiments were conducted on a machine with two 2.67GHz Intel Xeon CPU X5650 processors, 8GB of RAM memory, and running GNU/Linux operating system (Ubuntu 12.04 LTS, kernel 3.2.0-33). Instances were solved using the branch-and-cut method in IBM ILOG CPLEX 12.1.0. By default, a solution is reported as optimal by the solver when

$$\text{absolute gap} = \text{best feasible solution} - \text{best lower bound} \leq \varepsilon_{\text{abs}}$$

or

$$\text{relative gap} = \frac{|\text{ best feasible solution} - \text{best lower bound }|}{1\text{e}{-10} + |\text{ best feasible solution }|} \leq \varepsilon_{\text{rel}},$$

with $\varepsilon_{\text{abs}} = 10^{-6}$ and $\varepsilon_{\text{rel}} = 10^{-4}$, where "best feasible solution" means the smallest value of the objective function related to a feasible solution generated by the method. Since decimal places of the objective functions (62) and (77) of models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$, respectively, represent the value of the leftovers, we inhibited the relative gap stopping criterion setting $\varepsilon_{\text{rel}} = 0$, to avoid stopping the method prematurely. All other parameters of the solver were used with their default values unless otherwise stated.

Table 3 describes the solutions to the twenty instances associated with models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$. In the table, "Optimal value" corresponds to the value of the objective function at the solution reported by the solver as optimal. "Objects cost" and "Leftovers value" correspond to the cost of the used objects at the optimal solution and the value of the leftovers, respectively. Those

Inst.	Objects		Items	
	p	width $\times$ height	n	width $\times$ height
1	3	$3(52 \times 53)$	32	19×21, 7×20, 4×20, 15×20, 14×20, 14×19, 17×19, 10×17, 13×17, 5×17, 16×17, 20×16, 5×16, 3×15, 5×14, 18×14, 10×13, 14×12, 11×11, 2×10, 7×9, 14×8, 13×8, 9×7, 7×7, 16×6, 20×5, 2×5, 9×3, 17×3, 5×3, 4×2
2	2	$2(63 \times 60)$	23	22×23, 17×23, 13×22, 7×22, 9×21, 5×20, 20×20, 6×19, 7×17, 14×16, 12×14, 15×14, 20×14, 9×10, 16×9, 20×9, 18×8, 3×8, 12×7, 18×6, 20×4, 9×3, 6×3
3	7	24×14, $2(18\times 10)$, 24×13, $3(13\times 10)$	17	3×5, 2×5, 4×3, 6×3, 3×3, 7×3, 2×2, 6×2, 9×2, 4×2, 1×2, 7×2, 4×1, 6×1, 2×1, 9×1, 7×1
4	11	$2(24\times 14)$, $3(18\times 10)$, $2(24\times 13)$, $4(13\times 10)$	27	3×5, 1×5, 7×5, 2×5, 3×4, 7×4, 4×4, 4×3, 1×3, 6×3, 8×3, 3×3, 7×3, 5×2, 6×2, 2×2, 9×2, 4×2, 1×2, 7×2, 8×1, 2×1, 6×1, 7×1, 9×1, 4×1, 5×1
5	16	$2(24\times 14)$, $5(18\times 10)$, $3(24\times 13)$, $6(13\times 10)$	37	3×5, 1×5, 7×5, 2×5, 8×5, 6×4, 5×4, 3×4, 7×4, 4×4, 1×4, 2×4, 4×3, 5×3, 1×3, 2×3, 6×3, 8×3, 3×3, 7×3, 6×2, 9×2, 7×2, 4×2, 5×2, 1×2, 2×2, 3×2, 6×1, 2×1, 4×1, 8×1, 5×1, 9×1, 3×1, 1×1, 7×1
6	8	$2(24\times 14)$, $4(18\times 10)$, $2(24\times 13)$	34	3×5, 1×5, 7×5, 2×5, 8×5, 6×4, 3×4, 7×4, 4×4, 1×4, 2×4, 4×3, 5×3, 1×3, 6×3, 8×3, 3×3, 7×3, 6×2, 7×2, 9×2, 5×2, 4×2, 2×2, 1×2, 3×2, 9×1, 7×1, 8×1, 5×1, 2×1, 6×1, 4×1, 1×1
7	3	24×14, 18×10, 24×13	11	$2(2\times 2)$, 6×2, 4×2, 1×2, $2(7\times 1)$, 4×1, 9×1, 6×1, 2×1
8	10	$3(28\times 17)$, $3(16\times 27)$, $4(13\times 23)$	19	1×10, 6×10, 4×9, 8×9, 9×9, 5×8, 2×7, 4×6, 10×6, 6×6, 10×5, 5×5, 8×5, 4×4, 7×4, 10×4, 6×4, 7×3, 10×2
9	4	$3(19\times 10)$, 19×26	17	2×9, 2×8, 5×8, 3×8, 4×7, 5×7, 4×6, 6×4, 2×4, 3×4, 4×3, 2×3, 6×2, 5×2, 1×1, 2×1, 3×1
10	6	$2(290\times 106)$, $2(148\times 183)$, $2(194\times 132)$	13	$2(63\times 59)$, 63×55, 48×48, 17×43, 98×40, 38×35, $2(114\times 33)$, 24×23, 62×19, $2(110\times 11)$
11	13	$5(25\times 21)$, $5(27\times 19)$, $3(30\times 24)$	37	$8(11\times 7)$, $5(7\times 5)$, $9(9\times 5)$, $5(10\times 5)$, $10(3\times 2)$
12	1	14×19	12	$12(2\times 4)$
13	7	$2(21\times 24)$, $5(10\times 18)$	32	$15(4\times 7)$, $17(1\times 4)$
14	4	24×14, $2(18\times 10)$, 24×13	28	$12(7\times 1)$, $7(6\times 1)$, $9(4\times 1)$
15	12	$4(26\times 19)$, $4(22\times 23)$, $4(30\times 17)$	34	$13(7\times 6)$, $10(9\times 4)$, $11(11\times 3)$
16	19	$6(30\times 11)$, $6(27\times 13)$, $7(12\times 26)$	21	$10(8\times 10)$, $9(10\times 3)$, $2(11\times 2)$,
17	1	14×19	17	$7(2\times 4)$, $10(1\times 3)$
18	5	$3(22\times 17)$, $2(14\times 30)$	24	$14(2\times 11)$, $10(5\times 5)$
19	16	$4(30\times 22)$, $4(30\times 24)$, $8(10\times 21)$	41	$15(9\times 7)$, $11(11\times 6)$, $15(1\times 5)$
20	9	$5(22\times 17)$, $4(14\times 30)$	21	$8(2\times 11)$, $7(8\times 9)$, $6(5\times 5)$

Table 1: Description of the considered instances. Dimensions are given in the format width $\times$ height. Notation $a(b \times c)$ means that there are a objects or items with dimension $b \times c$. When a is omitted it means that there is a single copy of the described object or item.

Inst.	Model $\mathcal{M}_1^L$			Model $\mathcal{M}_2^L$			
	Continuous variables	Binary variables	Constraints	Continuous variables	Binary variables	Integer variables	Constraints
1	6	1,590	149	6	102	1,584	1,797
2	4	556	83	4	50	552	681
3	14	1,084	185	14	133	1,071	1,290
4	22	4,179	401	22	319	4,158	4,613
5	32	11,280	741	32	624	11,248	12,063
6	16	4,776	362	16	288	4,760	5,190
7	6	203	69	6	39	177	266
8	20	1,920	279	20	210	1,900	2,217
9	8	620	113	8	76	612	759
10	12	558	139	12	90	420	582
11	26	9,164	673	26	507	1,391	2,106
12	2	80	53	2	14	12	78
13	14	3,710	365	14	238	329	728
14	8	1,632	218	8	120	236	485
15	24	7,164	588	24	432	840	1,465
16	38	4,426	589	38	437	950	1,563
17	2	154	71	2	19	24	114
18	10	1,510	223	10	130	190	439
19	32	13,808	885	32	688	1,312	2,241
20	18	2,096	309	18	207	396	729

Table 2: Number of continuous, binary, and integer variables and number of constraints of the twenty considered instances of models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$.

values are extracted from the optimal value according to (62) and (77) for models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$, respectively. Remaining columns "MIP iterations", "B&B Nodes", and "CPU time" (in seconds) are self-explanatory and state the effort required by the solver to obtain the reported solution. Figures 15–23 show the corresponding graphical representation (cutting/packing patterns) of the obtained solutions considering model $\mathcal{M}_2^L$. In the figures, dashed regions represent leftovers while blank spaces correspond to waste.

From the results of Table 3, it is possible to see that there is no clear winner in the first half of the instances set, while model $\mathcal{M}_2^L$ is clearly easier to be solved in the second half of the instances set, as expected. The stopping criterion was satisfied in 19 cases (all but instance 11 for which the solver ran out of memory) for model $\mathcal{M}_1^L$ and in all cases for model $\mathcal{M}_2^L$. The best feasible solution reported for instance 11 by model $\mathcal{M}_1^L$ coincides with the optimal one found when solving model $\mathcal{M}_2^L$. Instances that combine a large number p of objects with a large number n of items (like instances 4, 5, 11, and 15; see Table 1) appear to be the hardest ones when a branch-and-cut exact method like CPLEX is used. This may be considered an expected consequence of the number of binary variables of the models, which depends on p and n (model $\mathcal{M}_1^L$ has $2p + np/2 + n2/2$ binary variables, while model $\mathcal{M}_2^L$ has $2p + np$ binary variables; see Section 3.1). The bottleneck for considering instances larger than the ones presented in this experiments lies on the usage of memory. On larger instances the solver ran out of memory (as it was the case of instance 11 of model $\mathcal{M}_1^L$) if only primary (RAM) memory is used. When considering secondary memory (files on disk), the time for swapping is unaffordable. This observation supports the necessity for developing dedicated exact and heuristic method to tackle the introduced problems.

Model $\mathcal{M}_1^L$							
Inst.	Optimal value	Solutions description		Effort measurements		
		Objects cost	Leftovers value	MIP Iterations	B&B Nodes	CPU Time
1	5,511.937107	5,512	520	10,778,938	649,747	171.06
2	7,559.616667	7,560	2,898	42,798	2,079	1.23
3	259.962804	260	52	146,570	7,040	1.88
4	359.999998	360	0	55,955,833	1701,516	1,286.04
5	466.000000	466	0	174,832,220	1508,414	8,752.14
6	491.976187	492	48	22,395,534	632,600	451.53
7	179.869565	180	108	317	9	0.07
8	863.983673	864	64	1,349,127	28,641	30.07
9	380.000000	380	0	25,848	1,227	1.50
10	51,215.922104	51,216	12,998	5,522	189	0.29
11	1,745.9918*	1,746	60	40,810,461	335,600	2,319.25
12	265.421053	266	154	44	0	0.00
13	683.947589	684	100	132,993	1,605	7.43
14	179.982143	180	18	17,937	383	0.67
15	1,506.000000	1,506	0	93,466,859	2033,696	2,981.30
16	1,364.994258	1,365	36	203,713	4,285	7.46
17	265.368421	266	168	92	0	0.03
18	748.000000	748	0	4,559	58	0.38
19	2,010.000000	2,010	0	789,515	3,411	43.92
20	1,167.962817	1,168	132	320,459	5,688	7.95

Model $\mathcal{M}_2^L$						
Inst.	Optimal value	Solutions description		Effort measurements		
		Objects cost	Leftovers value	MIP Iterations	B&B Nodes	CPU Time
1	5,511.937107	5,512	520	10,833,135	659,436	171.35
2	7,559.616667	7,560	2,898	78,562	3,088	1.51
3	259.962804	260	52	146,570	7,040	1.90
4	360.000000	360	0	13,905,664	215,651	247.56
5	466.000000	466	0	167,435,459	1,573,589	10,138.93
6	491.976189	492	48	24,503,629	713,002	514.95
7	179.869565	180	108	356	0	0.04
8	863.983673	864	64	873,289	16,808	12.68
9	380.000000	380	0	25,848	1,227	1.50
10	51,215.922104	51,216	12,998	4,332	165	0.26
11	1,745.991836	1,746	60	24,461,056	794,785	602.00
12	265.421053	266	154	29	0	0.03
13	683.947589	684	100	4,163	173	0.28
14	179.982143	180	18	17,766	951	0.38
15	1,505.999996	1,506	0	52,485,990	1105,225	2,298.26
16	1,364.994258	1,365	36	72,094	3,491	3.55
17	265.368421	266	168	78	0	0.00
18	748.000000	748	0	1,963	71	0.17
19	2,010.000000	2,010	0	116,686	3,161	5.79
20	1,167.962817	1,168	132	78,712	2,974	2.57

Table 3: Numerical results for the twenty instances of models $\mathcal{M}_1^L$ and $\mathcal{M}_2^L$. *Value reported for instance 11 of model $\mathcal{M}_1^L$ in the column "Optimal value" correspond to the best feasible solution found, since the solver ran out of memory and the absolute gap stopping criterion was not satisfied. The reported best lower bound was $1,745.9799$.

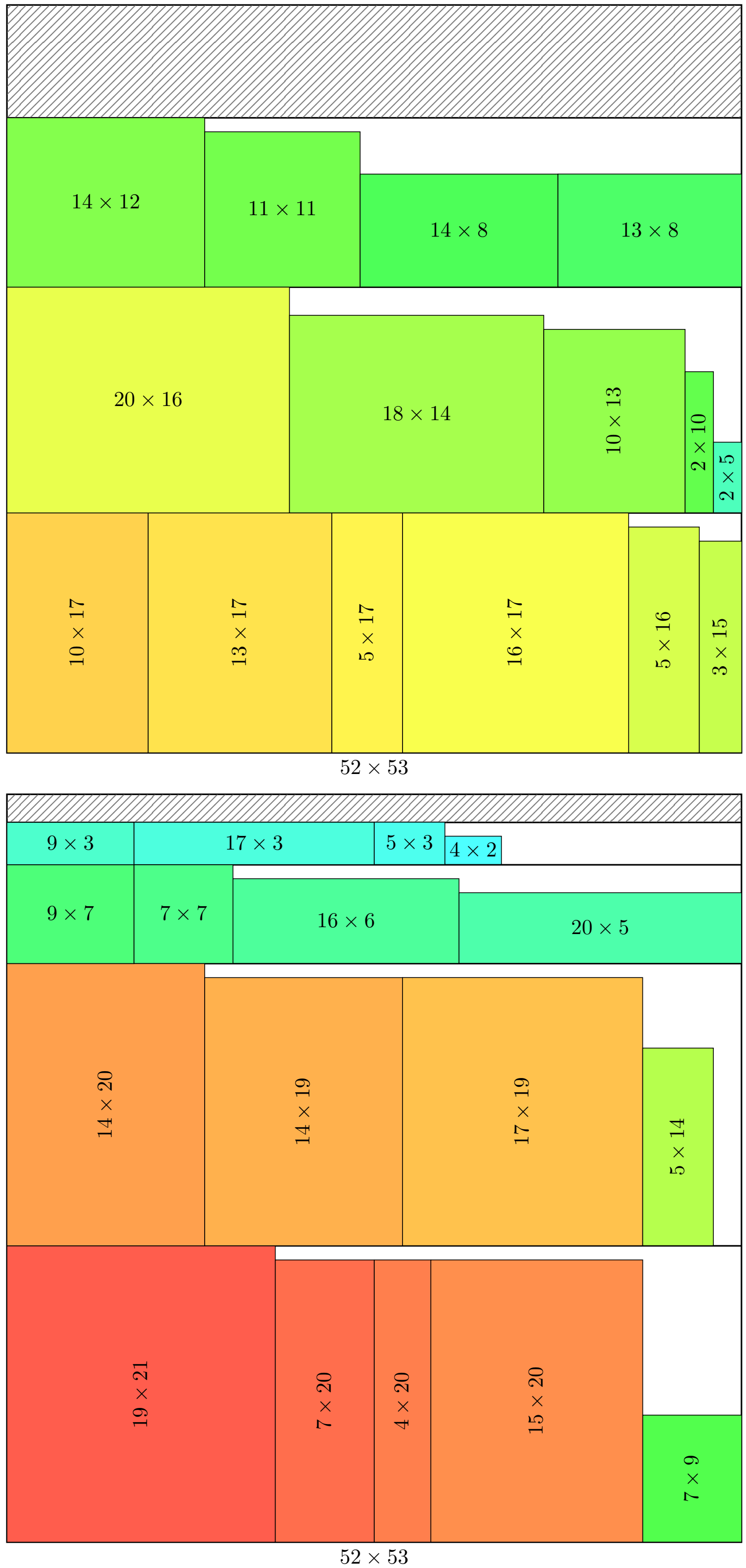

Figure 4: Graphical representation of the solution to instance 1.

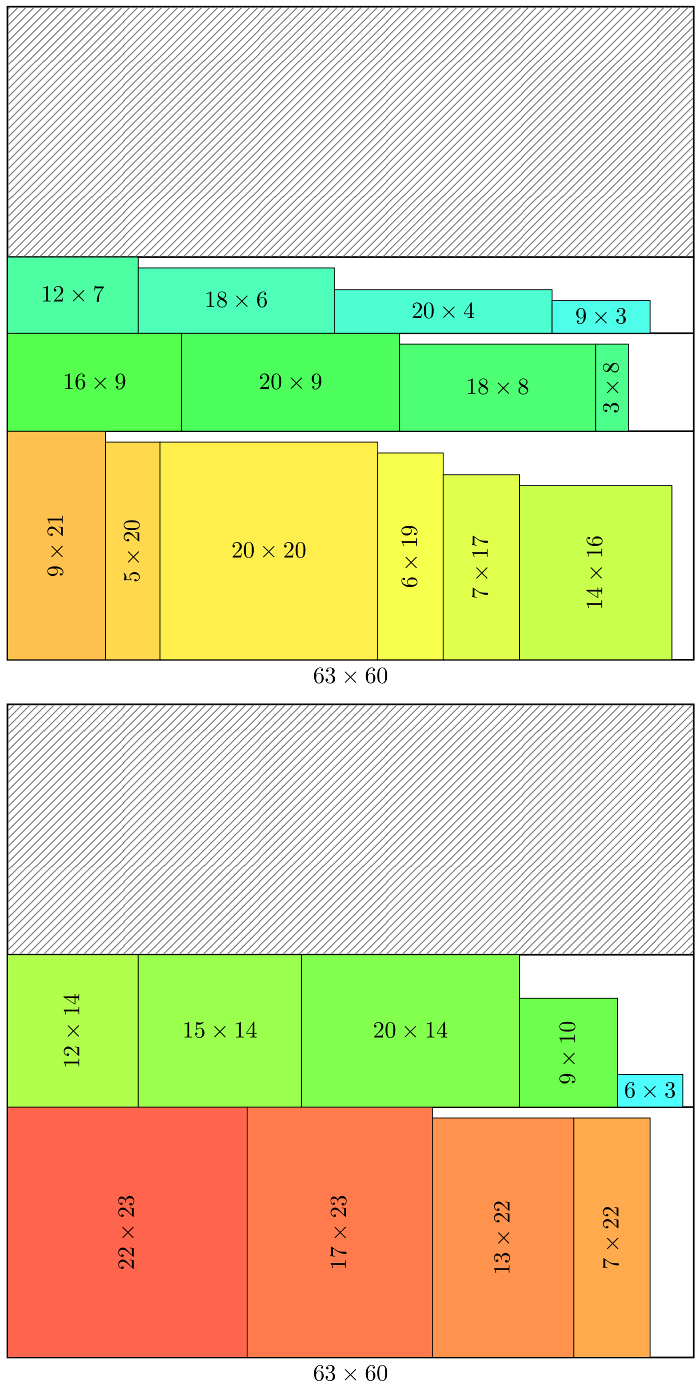

Figure 5: Graphical representation of the solution to instance 2.

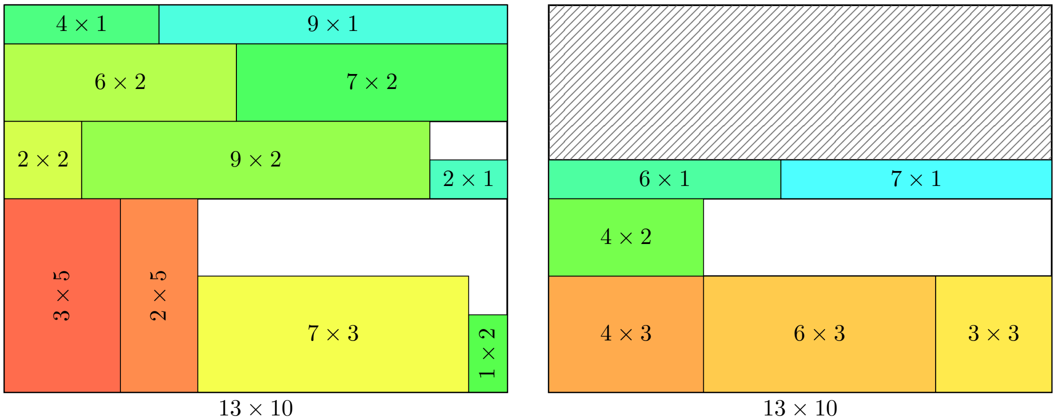

Figure 6: Graphical representation of the solution to instance 3.

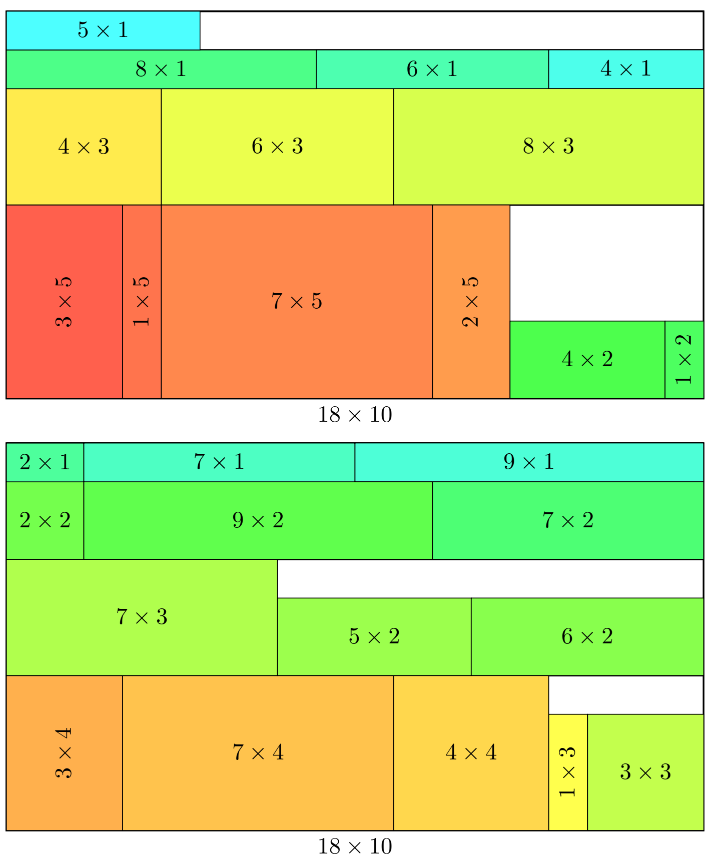

Figure 7: Graphical representation of the solution to instance 4.

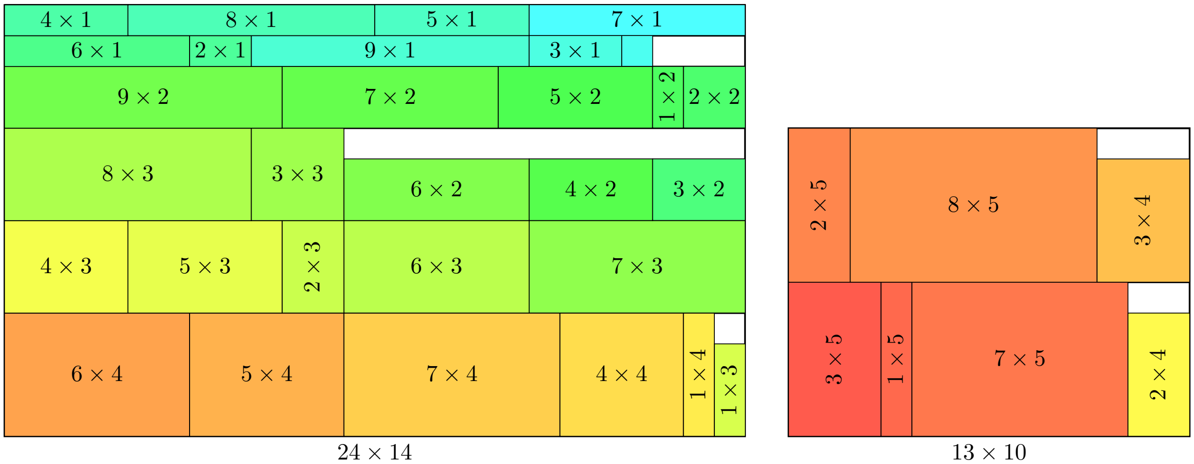

Figure 8: Graphical representation of the solution to instance 5.

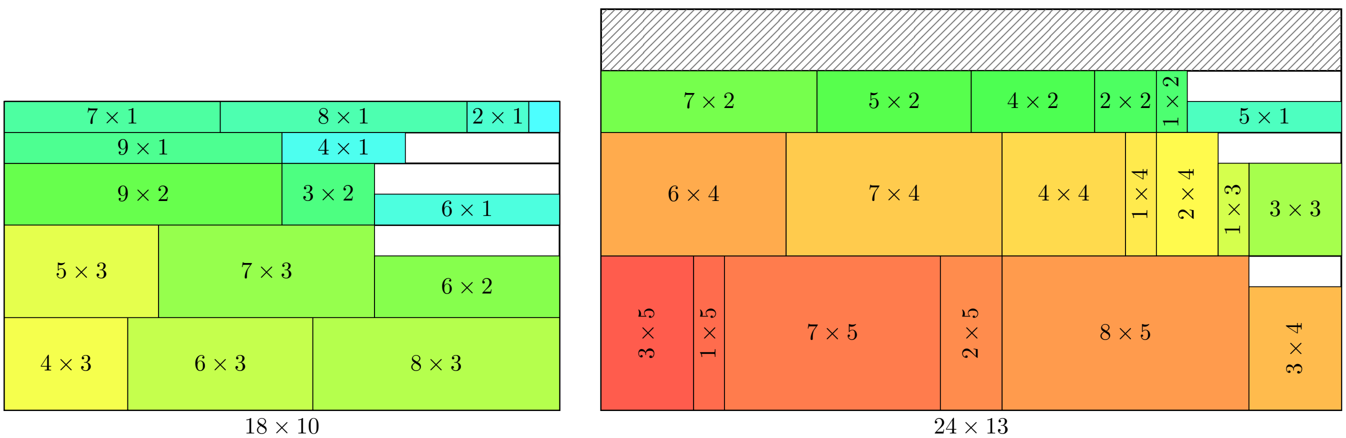

Figure 9: Graphical representation of the solution to instance 6.

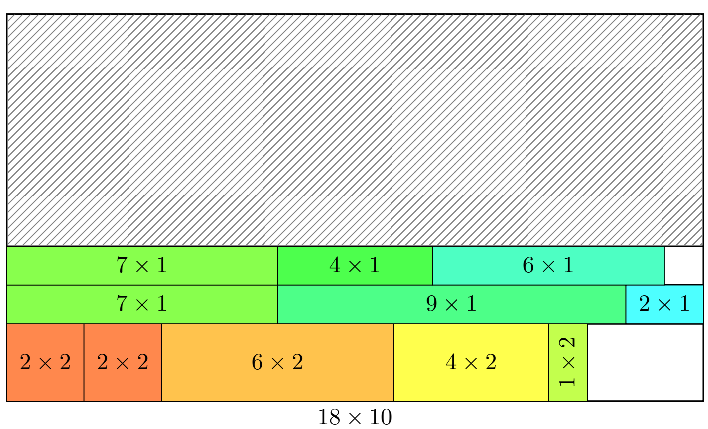

Figure 10: Graphical representation of the solution to instance 7.

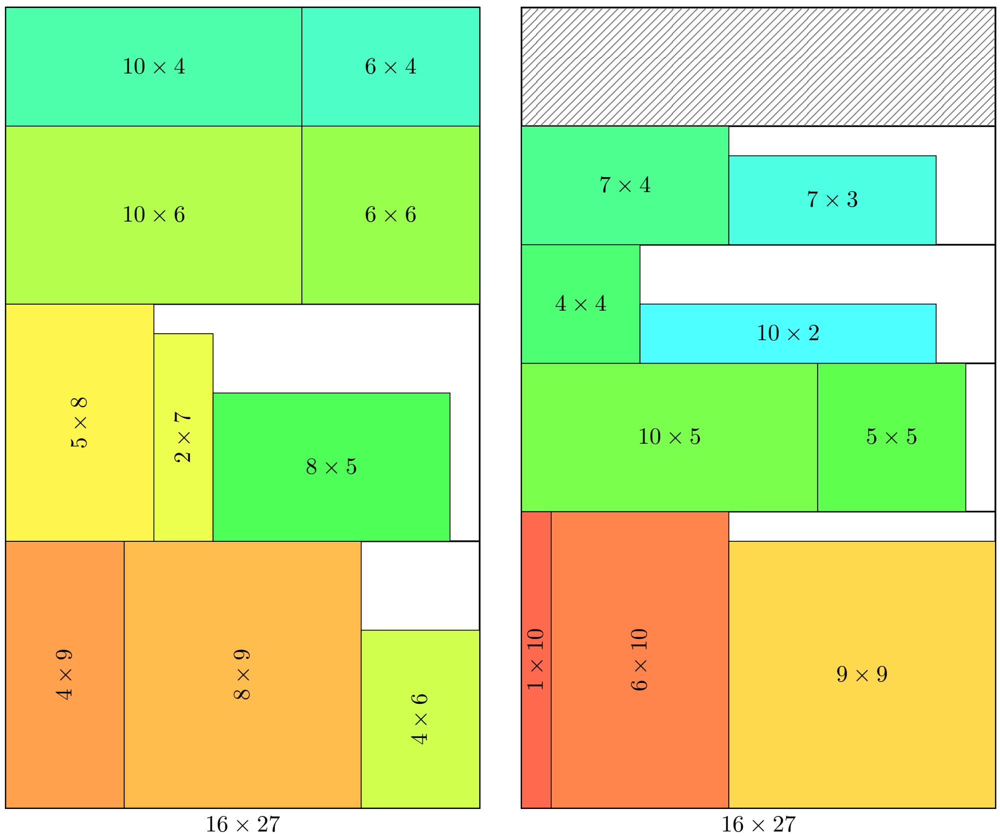

Figure 11: Graphical representation of the solution to instance 8.

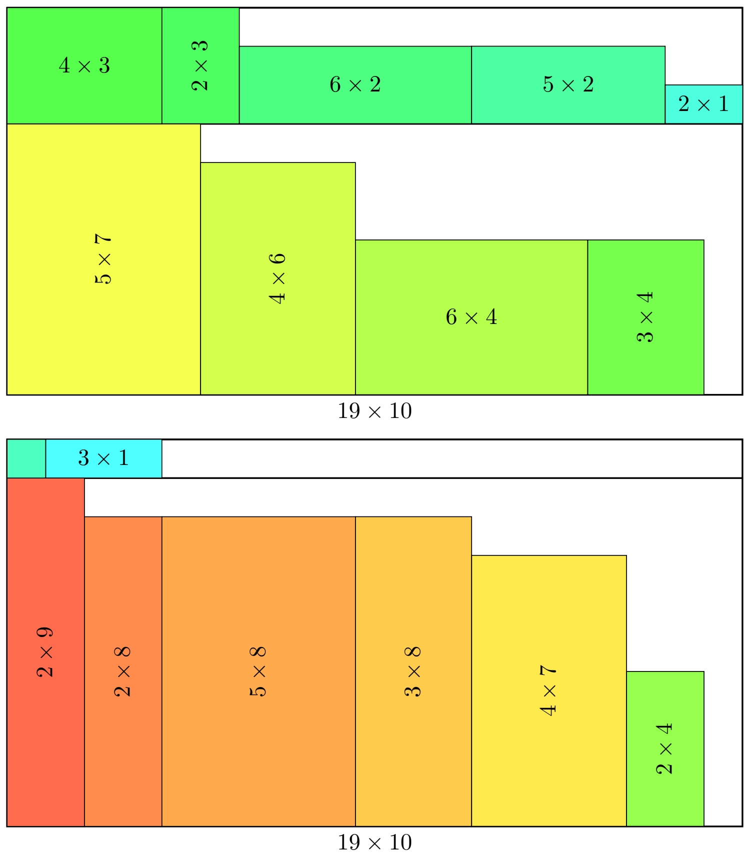

Figure 12: Graphical representation of the solution to instance 9.

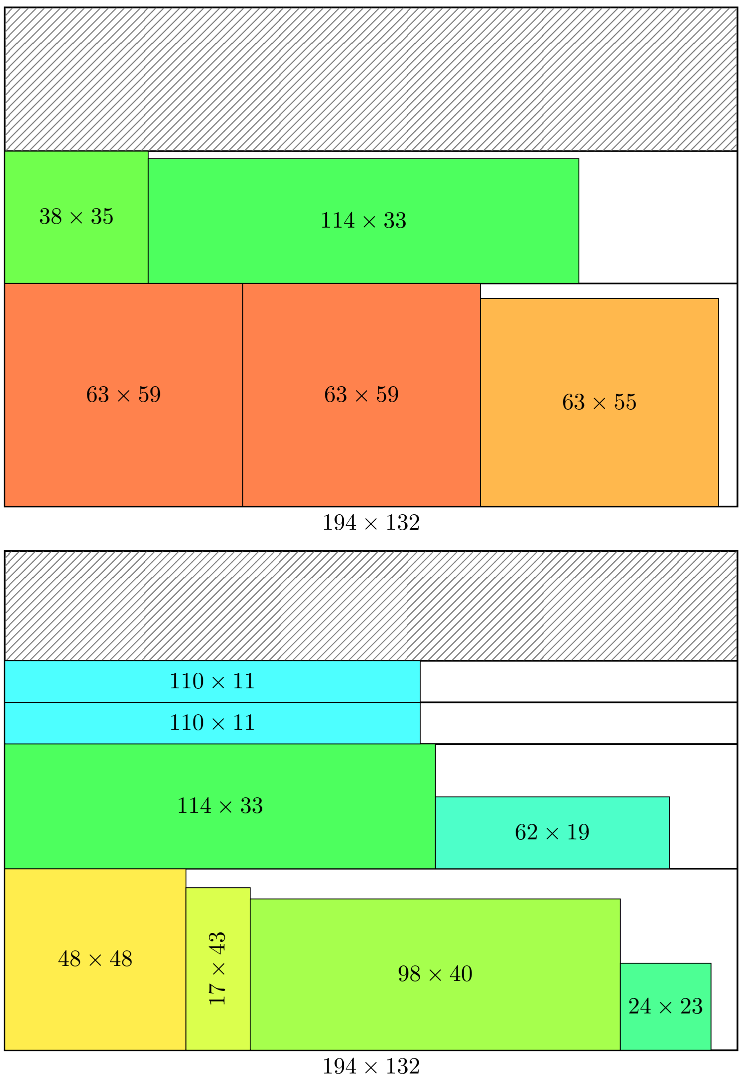

Figure 13: Graphical representation of the solution to instance 10.

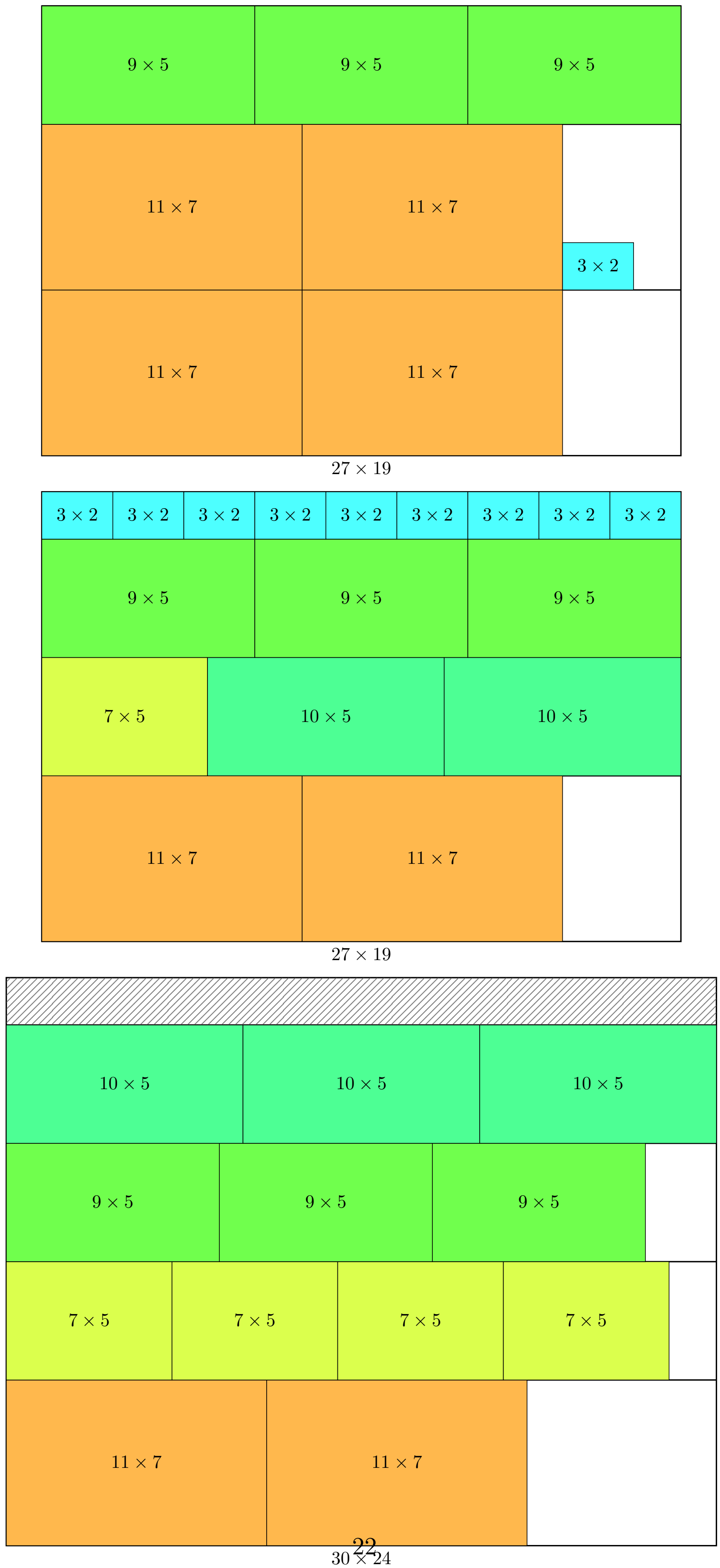

Figure 14: Graphical representation of the solution to instance 11.

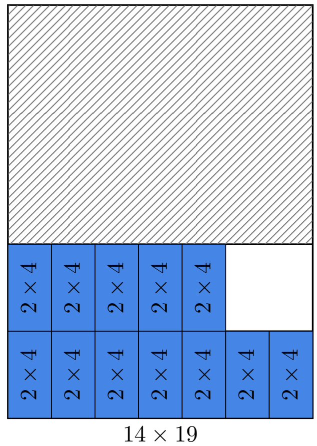

Figure 15: Graphical representation of the solution to instance 12.

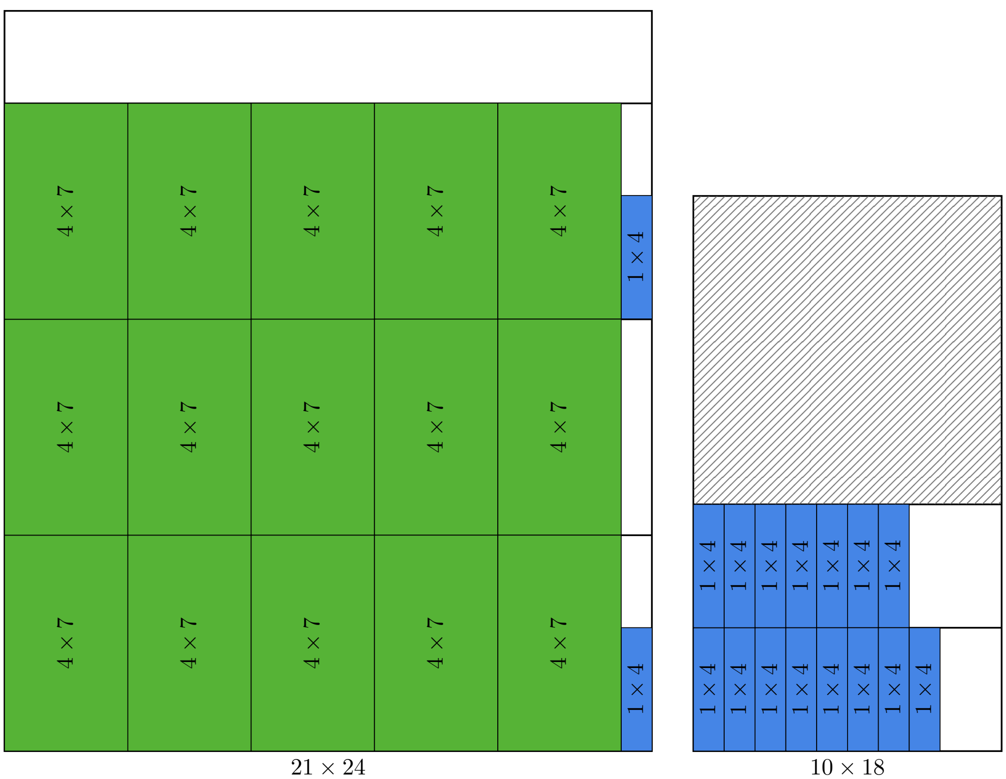

Figure 16: Graphical representation of the solution to instance 13.

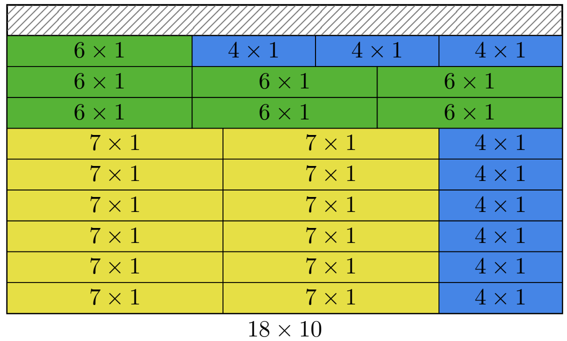

Figure 17: Graphical representation of the solution to instance 14.

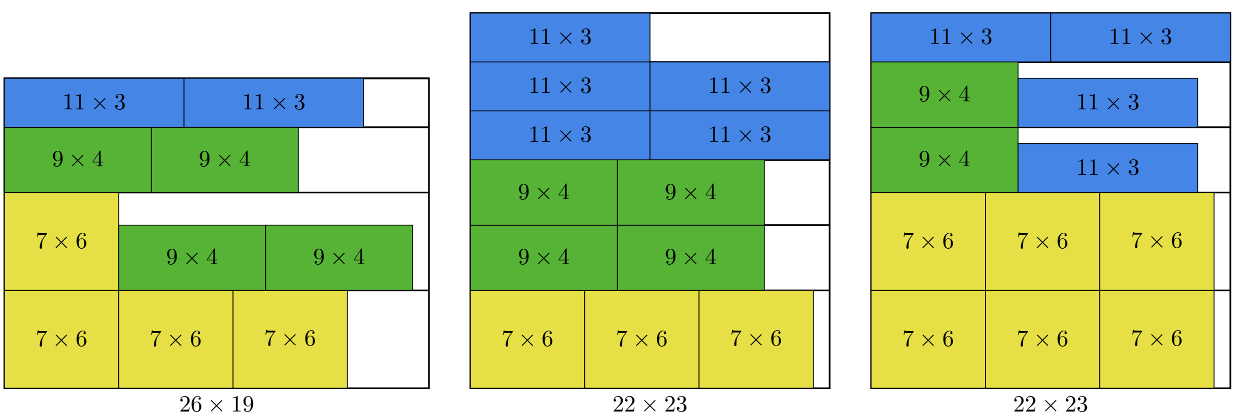

Figure 18: Graphical representation of the solution to instance 15.

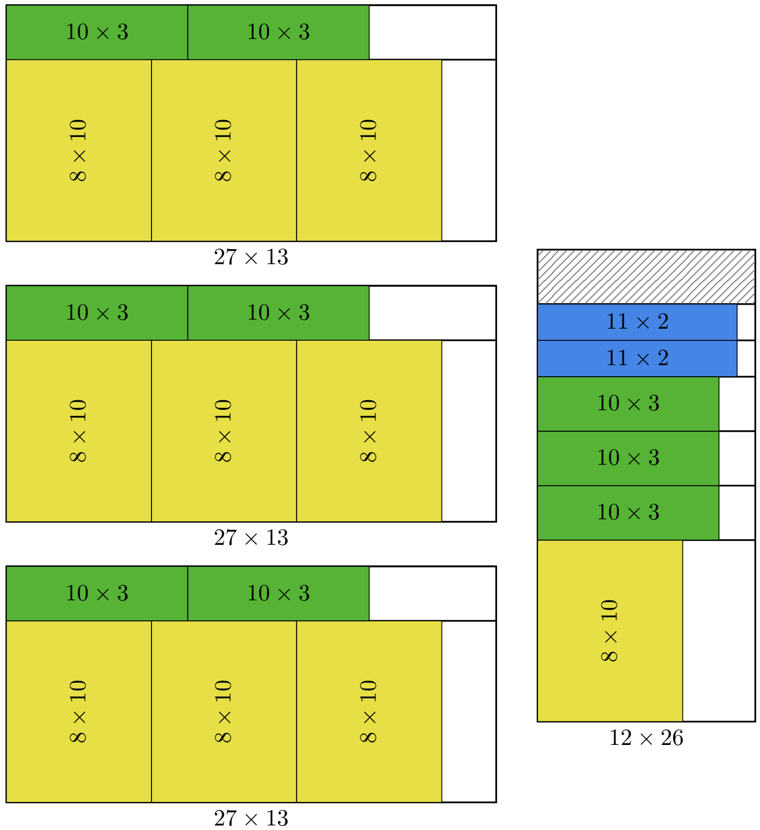

Figure 19: Graphical representation of the solution to instance 16.

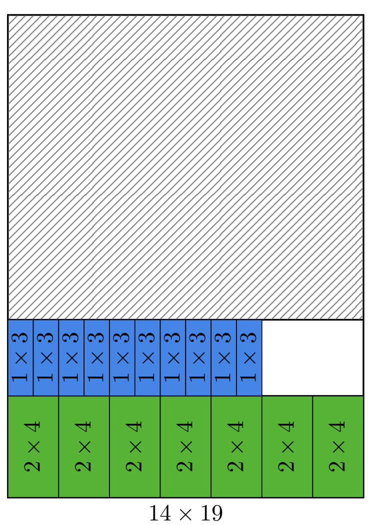

Figure 20: Graphical representation of the solution to instance 17.

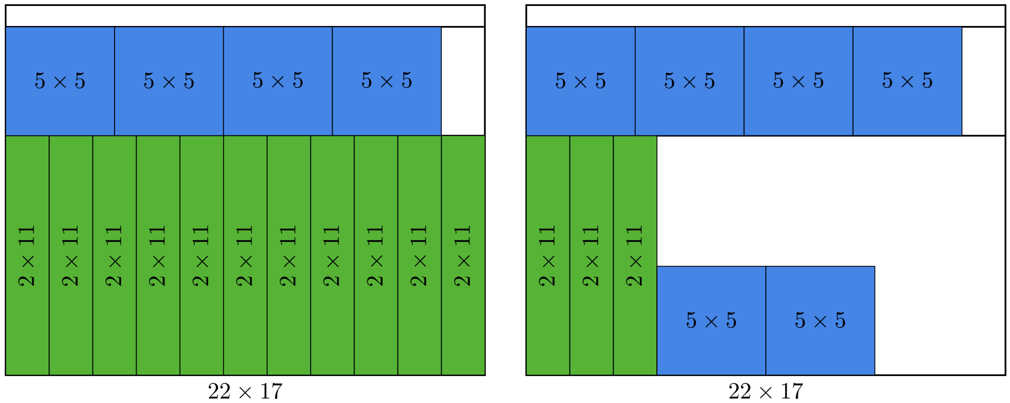

Figure 21: Graphical representation of the solution to instance 18.

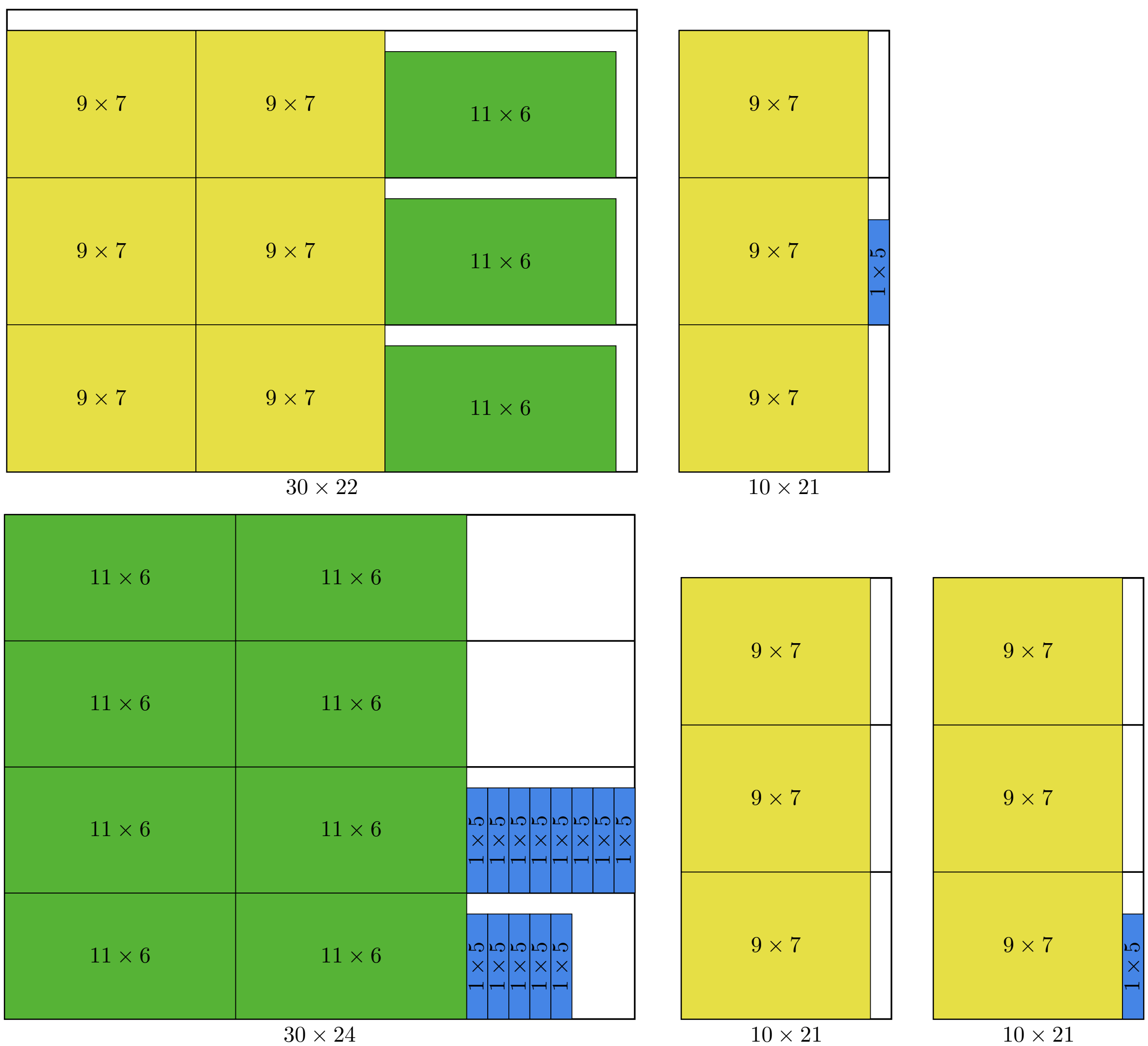

Figure 22: Graphical representation of the solution to instance 19.

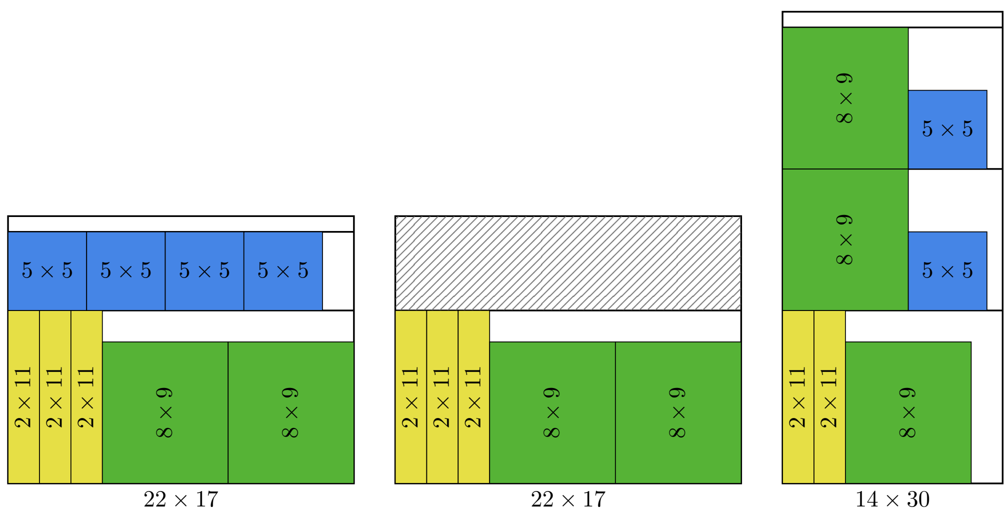

Figure 23: Graphical representation of the solution to instance 20.

5 Conclusions

In this study we presented two MIP models for the non-exact two-stage two-dimensional guillotine cutting/packing problem with usable objects remainders. Both models are based on bilevel mathematical programming formulations for the problem, but because of special characteristics of these bilevel models, they can be reformulated as one-level MIP models. The models can be modified to deal with particular cases of the tackled problem, such as the exact two-stage guillotine cutting without trimming, and with more general cases, such as the case in which the items can be rotated and the case in which the first-stage cuts on the plates can be either horizontal (parallel to the plate width) or vertical (parallel to the plate height). To the best of our knowledge, there are no other studies in the literature dealing with two-stage two-dimensional guillotine cutting/packing problems with usable leftovers. Numerical experiments illustrating the models and their scope and limitations were presented. The formal definition of these variants of two-dimensional guillotine cutting problems with residual pieces opens up interesting possibilities for the development of dedicated exact and heuristic methods for the resolution and practical application of these and other cutting/packing problems of two and more dimensions as, for example, the two-dimensional non-guillotine cutting/packing problem with usable leftovers.

References

Abuabara, A. & Morabito, R. (2009), 'Cutting optimization of structural tubes to build agricultural light aircrafts', *Annals of Operations Research* **169**(1), 149–165.

Alem, D. J. & Morabito, R. (2012), 'Production planning in furniture settings via robust optimization', *Computers & Operations Research* **39**(2), 139–150.

Arenales, M. N., Morabito, R. & Yanasse, H. H. (1999), 'Cutting and packing problems (Special Issue)', *Pesquisa Operacional* **19**(2).

Bang-Jensen, J. & Larsen, R. (2012), 'Efficient algorithms for real-life instances of the variable size bin packing problem', *Computers & Operations Research* **39**(11), 2848–2857.

Beasley, J. E. (1985), 'Algorithms for unconstrained two-dimensional guillotine cutting', *Journal of the Operational Research Society* **36**(4), 297–306.

Belov, G. & Scheithauer, G. (2006), 'A branch-and-cut-and-price algorithm for one-dimensional stock cutting and two-dimensional two-stage cutting', *European Journal of Operational Research* **171**(1), 85–106.

Bischoff, E. & Wäescher, G. (1995), 'Cutting and packing (Special Issue)', *European Journal of Operational Research* **84**(3).

Brown, A. R. (1971), *Optimum packing and depletion: the computer in space and resource usage problem*, Macdonald and Co. (London and New York), London.

Carnieri, C., Mendoza, G. A. & Gavinho, L. G. (1994), 'Solution procedures for cutting lumber into furniture parts', *European Journal of Operational Research* **73**(3), 495–501.

Cherri, A. C., Arenales, M. N. & Yanasse, H. H. (2009), 'The one-dimensional cutting stock problem with usable leftover – A heuristic approach', **196**(1), 897–908.

Cherri, A. C., Arenales, M. N. & Yanasse, H. H. (2012), 'The usable leftover one-dimensional cutting stock problem – A priority-in-use heuristic', *International Transactions in Operational Research* . To appear.

Cintra, G. F., Miyazawa, F. K., Wakabayashi, Y. & Xavier, E. C. (2008), 'Algorithms for two-dimensional cutting stock and strip packing problems using dynamic programming and column generation', *European Journal of Operational Research* **191**(1), 61–85.

Cui, Y., Wang, Z. & Li, J. (2005), 'Exact and heuristic algorithms for staged cutting problems', *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* **219**(2), 201–207.

Cui, Y. & Yang, Y. (2010), 'A heuristic for the one-dimensional cutting stock problem with usable leftover', *European Journal of Operational Research* **204**(2), 245 – 250.

Dempe, S. (2002), *Foundations of bilevel programming*, Springer, Dordrecht.

Dimitriadis, S. & Kehris, E. (2009), 'Cutting stock process optimisation in custom door and window manufacturing industry', *International Journal of Decision Sciences, Risk and Management* **1**(1), 66–80.

Dowsland, K. A. & Dowsland, W. B. (1992), 'Packing problems', *European Journal of Operational Research* **56**(1), 2–14.

Dyckhoff, H. (1981), 'A new linear programming approach to the cutting stock problem', *Operations Research* **29**(6), 1092–1104.

Dyckhoff, H. (1990), 'A typology of cutting and packing problems', *European Journal of Operational Research* **44**(2), 145–159.

Dyckhoff, H., Scheithauer, G. & Terno, J. (1997), Cutting and Packing, *in* S. Martello, M. Dell'Amico & F. Maffioli, eds, 'Annotated Bibliographies in Combinatorial Optimization', John Wiley and Sons Inc., New York, chapter 22, pp. 393–413.

Dyckhoff, H. & Wäscher, G. (1990), 'Cutting and packing (Special Issue)', *European Journal of Operational Research* **44**(2).

Farley, A. (1983), 'Practical adaptations of the Gilmore-Gomory approach to cutting stock problems', *OR Spektrum* **10**(2), 113–123.

Foronda, S. U. & França, H. (1991), 'A heuristic approach to the lumber allocation problem in hardwood dimension and furniture manufacturing', *European Journal of Operational Research* **54**(2), 151–162.

Gilmore, P. C. & Gomory, R. E. (1965), 'Multistage Cutting Stock Problems of Two and More Dimensions', *Operations Research* **13**(1), 94–120.

Gradisar, M., Erjavec, J. & Tomat, L. (2011), 'One-Dimensional Cutting Stock Optimization with Usable Leftover: A Case of Low Stock-to-Order Ratio', *International Journal of Decision Support System Technology* **3**(1), 54–66.

Gradisar, M., Jesenko, J. & Resinovic, C. (1997), 'Optimization of roll cutting in clothing industry', *Computers & Operational Research* **24**(10), 945–953.

Gradisar, M., Kljajic, M., Resinovic, C. & Jesenko, J. (1999), 'A sequential heuristic procedure for one-dimentional cutting', *European Journal of Operational Research* **114**(3), 557–568.

Gradisar, M. & Trkman, P. (2005), 'A combined approach to the solution to the general one-dimensional cutting stock problem', *Computers & Operations Research* **32**(7), 1793–1807.

Gramani, M. C. N. & Franca, P. (2006), 'The combined cutting stock and lot-sizing problem in industrial processes', *European Journal of Operational Research* **174**(1), 509–521.

Gramani, M. C. N., Franca, P. M. & Arenales, M. N. (2009), 'A lagrangean relaxation approach to a coupled lot-sizing and cutting stock problem', *International Journal of Production Economics* **119**(2), 219–227.

Hadjiconstantinou, E. & Christofides, N. (1995), 'An exact algorithm for general, orthogonal, two-dimensional knapsack problems', *European Journal of Operational Research* **83**(1), 39–56.

Hifi, M. (1997), 'The DH/KD algorithm: A hybrid approach for unconstrained two-dimensional cutting problems', *European Journal of Operational Research* **97**(1), 41–52.

Hifi, M. (2002), 'Cutting and packing problems (Special Issue)', *Studia Informatica Universalis* **2**(1).

Hifi, M. & Roucairol, C. (2001), 'Approximate and Exact Algorithms for Constrained (Un)Weighted Two-dimensional Two-staged Cutting Stock Problems', *Journal of Combinatorial Optimization* **5**(4), 465–494.

Hinxman, A. I. (1980), 'The trim-loss and assortment problems: A survey', *European Journal of Operational Research* **5**(1), 8–18.

Kendall, G., Daniels, K. M. & Burke, E. (2010), 'Cutting, packing, layout, and space allocation (Special Issue)', *Annals of Operations Research* **179**(1).

Koch, S., König, S. & Wšcher, G. (2009), 'Integer linear programming for a cutting problem in the wood-processing industry: a case study', *International Transactions in Operational Research* **16**(6), 715–726.

Lodi, A., Martello, S. & Monaci, M. (2002), 'Two-dimensional packing problems: A survey', *European Journal of Operational Research* **141**(2), 241–252.

Lodi, A. & Monaci, M. (2003), 'Integer linear programming models for 2-staged two-dimensional Knapsack problems', *Mathematical Programming* **94**(2-3), 257–278.

Macedo, R., Alves, C. & De Carvalho, J. M. V. (2010), 'Arc-flow model for the two-dimensional guillotine cutting stock problem', *Computers & Operations Research* **37**(6), 991–1001.

Martello, S. (1994*a*), 'Knapsack, Packing and Cutting – Part I: One Dimensional Knapsack Problems (Special Issue)', *INFOR* **32**(3).

Martello, S. (1994*b*), 'Knapsack, Packing and Cutting – Part II: Multidimensional Knapsack and Cutting Stock Problems (Special Issue)', *INFOR* **32**(4).

Morabito, R. & Arenales, M. N. (1996), 'Staged and constrained two-dimensional guillotine cutting problems: An AND/OR-graph approach', *European Journal of Operational Research* **94**(3), 548–560.

Morabito, R. & Arenales, M. N. (2000), ‘Optimizing the cutting of stock plates in a furniture company’, *International Journal of Production Research* **38**(12), 2725–2742.

Morabito, R., Arenales, M. N. & Yanasse, H. H. (2009), ‘Cutting, packing and related problems (Special Issue)’, *International Transactions in Operational Research* **16**(6).

Morabito, R. & Belluzzo, L. (2005), ‘Otimizao nos padrões de corte de chapas de fibra de madeira reconstituda: um estudo de caso’, *Pesquisa Operacional* **25**(3), 391–415.

Morabito, R. & Garcia, V. (1998), ‘The cutting stock problem in a hardboard industry: A case study’, *Computers & Operations Research* **25**(6), 469–485.

Morabito, R. & Pureza, V. (2010), ‘A heuristic approach based on dynamic programming and and/or-graph search for the constrained two-dimensional guillotine cutting problem’, *Annals of Operations Research* **179**, 297–315.

Oliveira, J. F. & Wäescher, G. (2007), ‘Cutting and packing (feature cluster)’, *European Journal of Operational Research* **183**(3).

Poldi, K. C. & Arenales, M. N. (2006), ‘Heurísticas para o problema de corte de estoque unidimensional inteiro’, *Pesquisa Operacional* **26**(3), 473–492.

Poldi, K. C. & Arenales, M. N. (2009), ‘Heuristics for the one-dimensional cutting stock problem with limited multiple stock lengths’, *Computers & Operations Research* **36**(6), 2074–2081.

Riehme, J., Scheithauer, G. & Terno, J. (1996), ‘The solution of two-stage guillotine cutting stock problems having extremely varying order demands’, *European Journal of Operational Research* **91**(3), 543–552.

Roodman, G. M. (1986), ‘Near-optimal solutions to one-dimensional cutting stock problems’, *Computers and Operations Research* **13**(6), 713–719.

Scheithauer, G. (1991), ‘A note on handling residual lengths’, *Optimization* **22**(3), 461–466.

Silva, E., Alvelos, F. & Valrio de Carvalho, J. M. (2010), ‘An integer programming model for two- and three-stage two-dimensional cutting stock problems’, *European Journal of Operational Research* **205**(3), 699–708.

Sinuany-Stern, Z. & Weiner, I. (1994), ‘The one dimensional cutting stock problem using two objectives’, *Journal of Operations Research Society* **45**(2), 231–236.

Sweeney, P. E. & Paternoster, E. R. (1992), ‘Cutting and Packing Problems: A Categorized, Application-Orientated Research Bibliography’, *The Journal of the Operational Research Society* **43**(7), 691–706.

Trkman, P. & Gradisar, M. (2007), ‘One-dimensional cutting stock optimization in consecutive time periods’, *European journal of Operational Research* **179**(2), 291–301.

Wang, P. Y. & Wäscher, G. (2002), ‘Cutting and packing (Special Issue)’, *European Journal of Operational Research* **141**(2).

Wäscher, G. & Gau, T. (1996), ‘Heuristics for the integer one-dimensional cutting stock problem: A computational study’, *Operations-Research-Spektrum* **18**, 131–144.

Wäscher, G., Hauß ner, H. & Schumann, H. (2007), 'An improved typology of cutting and packing problems', *European Journal of Operational* **183**(3), 1109–1130.

Yanasse, H. H. & Katsurayama, D. M. (2005), 'Checkerboard pattern: proposals for its generation. International Transactions in Operational Research', **12**(1), 21–45.

Yanasse, H. H. & Morabito, R. (2012), 'Modelos lineares e não-lineares inteiros para problemas da mochila bidimensional restrita a dois estágios', *Produção* . To appear.

Yanasse, H. H., Zinober, A. & Harris, R. (1991), 'Two-dimensional cutting stock with multiple stock sizes', *Journal of the Operational Research Society* **42**(8), 673–683.