RAIRO-Oper. Res. 57 (2023) 2067–2085
https://doi.org/10.1051/ro/2023079

RAIRO Operations Research
www.rairo-ro.org

MATHEMATICAL MODELS FOR THE CUTTING STOCK WITH LIMITED OPEN STACKS PROBLEM

Gabriel Gazzinelli Guimarães and Kelly Cristina Poldi*

Abstract. This research is focused on solving the Cutting Stock with Limited Open Stacks Problem (CS-LOSP). The CS-LOSP is an optimization problem which consists of the classical Cutting Stock Problem (CSP) paired with the additional constraint that the maximum number of open stacks from the sequencing of the cutting patterns obtained from the CSP solution is equal or lower than a preset limit. Despite being a problem with great practical importance, the literature lacks models for this problem, and only one-dimensional problems are addressed. In this paper, we propose two integer linear programming formulations for the CS-LOSP that are valid for solving instances of the CSP of any dimension. In order to eliminate symmetrical solutions to the problem, the proposed formulations sequence sets of cutting patterns instead of sequencing the cutting patterns individually, thus, the search space for solutions is reduced. A set of randomly generated instances for the two-dimensional problem is used to perform computational experiments in order to validate the proposed mathematical formulations.

Mathematics Subject Classification. 90-08, 90B30, 90C10.

Received January 20, 2023. Accepted May 31, 2023.

1. Introduction

The Cutting Stock Problem (CSP) is an optimization problem that aims to determine the best way to cut a set of larger objects in smaller items in order to fulfill the demand for items of a specific size. The objective function can minimize either the waste of raw material or the cost associated with the cutting process, among others. The CSP can be classified according to the number of relevant dimensions of the object to be cut, which can be one dimensional, two dimensional or three dimensional. An example of one-dimensional CSP is the cutting of paper reels, while as a two-dimensional example we have the cutting of metal sheets.

Due to its practical importance, the Cutting Stock Problem (CSP) is a well researched topic. In the literature, there are some mathematical formulations for such problem. Among the models proposed for the CSP, we can cite the formulations proposed in [6, 7, 9–12, 14, 20, 22]. To the best of our knowledge, the first time CSP was covered was in 1939, published by Kantorovich [12], in 1960. In this article, a formulation for the one-dimensional CSP is proposed, in which the cutting of each object is considered individually.

Keywords. Cutting stock problem, open stack, pattern sequencing, mathematical formulation, integer linear programming, setup cost.

Instituto de Matematica, Estatistica e Computacao Cientifica (IMECC), Universidade Estadual de Campinas (UNICAMP), R. Sergio Buarque de Holanda, 651, 13083-859 Campinas, SP, Brazil.
*Corresponding author: kelly@ime.unicamp.br

The mostly used approach for the CSP is the classical formulation proposed by Gilmore and Gomory in [9,10]. Such formulation, together with the proposed column generation approach for solving the linear programming relaxation of the CSP combined with an integralization technique, is widely applied to solve the CSP [18].

Later on, based on an arc-flow approach, Wolsey [22] and Valério de Carvalho [20] propose a mathematical formulation for the one-dimensional CSP. The model proposed in [20] is extended for the two-dimensional case, *i.e.*, to deal with objects regarding two dimensions, such as plates or boards, in [14].

In [6], the authors develop a procedure to reduce the number of variables and constraints of the arc-flow model proposed in [20]. Such procedure is named as Meet-In-the-Middle (MIM). Another contribution presented in this paper is the proposal of a novel arc-flow formulation which is more compact and presents an improved overall performance.

The current state-of-the-art formulation for the one dimensional CSP is the one proposed in [7]. The authors take advantage of the meet-in-the-middle procedure proposed in [6] and the classical arc-flow formulation proposed in [20] to build a new model in which only half of the bin capacity is considered. The resulting formulation has fewer variables and constraints, and therefore, performs better.

In a two-dimensional Cutting Stock Problem (2D-CSP), two dimensions of the object and the items are relevant, typically length and width. The construction of two-dimensional cutting patterns is considerably more labor-intensive when compared to the one-dimensional CSP. A guillotine cut on a board is a straight cut from one edge to the opposite edge; in other words, the cut is of guillotine type when applied to a rectangle it produces new rectangles. In some practical applications, such as in furniture industry, the cutting equipment is able to perform only guillotines cuts.

In many practical applications, a guillotine cutting pattern is performed in stages. At each stage, cuts are done in the same direction over the whole object or on pieces (strips) of it obtained from cuts performed in a previous stage. The direction in which the cuts are performed is changed at each stage. Thus, if at any stage the cuts are horizontal, at the next stage the cuts will be vertical and *vice versa*. Changing the direction of the cuts can be expensive, as you need to rotate the machine or the object, which is time-consuming. Therefore, the strategy of limiting the number of stages can be used to optimize the process. On the one hand, this restriction reduces the number of feasible cutting patterns, on the other hand, the amount of material waste can increase. When the number of stages is limited to k, the guillotine cutting pattern is called k-staged.

In the one-dimensional CSP, a cutting pattern can be obtained by solving a knapsack problem [9,10]. The generation of a two-dimensional cutting pattern is significantly more complex and, depending on the characteristics of the types of allowed cuts, we can find several heuristic and exact approaches to determining two-dimensional cutting patterns in the literature. In this research, we consider the two-staged guillotine cutting pattern, as proposed in [11].

The order in which the cutting patterns are carried out can be relevant to manage the storage space around the cutting machine. In many practical cases, such as cutting wooden boards for furniture making, the machines have automatic unloading stations and the number of such stations is limited. The greater the number of stations, the more expensive the cutting equipment is, so it is quite important for the company to have cutting plans that require a limited and small number of open stacks. In this case, the Minimization of Open Stacks Problem (MOSP) must be taken into account. The MOSP acts in the context in which the allocation of items obtained from the CSP solution are stored in stacks, so that different items are arranged in different stacks. When all cutting patterns containing a specific item type have already been performed, the stack referring to such item is called a closed stack, otherwise, the stack of this item is considered open. The goal of the MOSP is to minimize the maximum number of stacks that are simultaneously open.

A closely related problem to the MOSP is the Minimization of Tool Switches Problem (MTSP). The MTSP aims to determine the order in which a set of tasks must be performed on a single flexible machine. Each of the tasks needs a specific set of tools for its execution, and the in-use machine has a limitation on the maximum number of tools that can be simultaneously allocated to the machine. Thus, it is necessary to switch the tools allocated to the machine to process all tasks. The goal of the MTSP is to minimize the number of tool switches performed.

Yanasse [23] demonstrates that an optimal MTSP solution represents a sequencing of cutting patterns that opens at most C stacks if and only if the value of the solution to MTSP is equal to $|I| - C$, considering I as the set of all item types. With this proposition, in [23] a formulation for the MOSP based on the model proposed in [19] for the MTSP is developed.

The strategy used in [23] to build the model for the MOSP is to change the interpretation of variables and parameters, so that, instead of considering tools and tasks, the problem considers items and cutting patterns, respectively. Once this is done, a constraint is added to the formulation proposed in [19], in order to ensure that at most $|I| - C$ tool exchanges are performed in the context of the MTSP, thus ensuring that the sequencing is equivalent to a cutting pattern sequencing that opens at most C stacks. Finally, the model's objective function is changed to minimize the maximum number of open stacks.

In [15], the authors propose a constraint programming formulation and two new integer linear formulations for the MOSP. The first one seeks to sequence the stacks to be closed, and the second is based on the open stack counting strategy proposed in [8].

A formulation for the MTSP, inspired by the traveling salesman problem, is proposed in [13]. In this formulation, the tasks to be performed are represented by the vertices in a graph and the sequence in which the tasks are carried out is equivalent to the order in which the vertices are traversed in an Euler tour. In this model, a dummy task is established, which represents the beginning and end of the Euler tour. In [5], the formulation proposed in [13] is used as the basis for the construction of three new formulations for the MTSP.

In practical situations, the CSP is often associated with the MOSP. Despite this, in the literature, the CSP and the MOSP are usually independently treated. The independent resolution of the problems consists in solving the CSP and then sequencing the set of cutting patterns to minimize the maximum number of open stacks. In some cases, solving the CSP and the MOSP independently may yield unsatisfactory results. Two disadvantages of the independent resolution approach are discussed in [2], namely:

- Solving the two problems independently is not the same as finding a solution that minimizes the maximum number of open stacks while reducing material waste. The quality of the solution obtained from the MOSP resolution strongly depends on the set of cutting patterns determined when solving the CSP. It is possible that there are several sets of cutting patterns that have the same objective function value for the CSP, but present different results for the MOSP. In this case, solving the problems independently does not make it possible to determine the set of cutting patterns that optimizes the two intended objectives.
- If there is a limit on the maximum number of stacks that can be opened, then the solution to the cutting stock problem may not present feasible sequencing.

Therefore, the independent approach may be less efficient in optimizing the two intended goals. Instead of solving the CSP and the MOSP independently, it is possible to combine the two problems into a single formulation. In [24], the authors build a formulation for the integrated problem in which the CSP is solved while subjected to the constraints that the maximum number of open stacks during the sequencing of the cutting patterns is equal to or less than a previously defined limit. This problem is known in the literature as the Cutting Stock with Limited Open Stacks Problem (CS-LOSP).

The idea of the formulation proposed in [24] is to build a model for the CS-LOSP that joins the formulation proposed in [9–11] for the CSP and the one proposed in [23] for the MOSP. The coupling of both models is done by a set of binary variables. The results obtained from the direct application of the formulation proposed in [24] are limited. The difficulty of solving the model for large instances is mainly due to the huge number of variables and constraints. The size of the formulation under consideration depends on the number of different types of items and the total number of possible cutting patterns, since, even for small problems, the total number of cutting patterns is considerably large, which compromises using the formulation for solving the problem. Throughout this text, we refer to the model proposed in [24] as the Integrated Yanasse and Lamosa (IYL).

Concomitantly, in 2007, in [4] a sequential heuristic for solving the CS-LOSP is proposed. In this heuristic, the cutting patterns are generated one after the other and, then, there are two approaches to deal with the limitation on the maximum number of open stacks. The first approach penalizes the pseudo-prices of some

items in order to reduce the number of items in each generated cutting pattern. The second approach explicitly restricts the maximum number of open stacks. To do so, during the generation of each cutting pattern, the current number of open stacks is evaluated and we let $\overline{C}$ be this value. Then, the new cutting pattern must contain at most $C-\overline{C}$ new item types. Thus, the limitation on the maximum number of open stacks is respected. More recently, in 2012, a tabu search heuristic combined with a constrained version of the model proposed in [9,10] to solve the CS-LOSP was developed in [1].

In [2], another formulation is proposed for the CS-LOSP. The authors manage to significantly reduce the number of constraints in the model. While in the formulation proposed in [24] the number of constraints grew exponentially as a function of the number of distinct types of items, in the formulation proposed in [2], the number of constraints grows quadratically. Despite the success in reducing the number of constraints in the formulation, the model still has a large number of variables. Another important contribution of the paper is the proposition of a column generation method to solve the linear programming relaxation of the CS-LOSP.

In [24], although the proposed model is valid for cutting stock problems of any dimensions, the focus of the computational experiments is on the one-dimensional problem. In [4], the authors propose the use of a sequential heuristic to solve the one-dimensional CS-LOSP.

To the best of our knowledge, the paper [16] is the only one in the literature to address the two-dimensional CS-LOSP. In such paper, three mathematical models are proposed, focused on the two and three staged two-dimensional guillotine cutting stock with limited open stacks problem. Despite the scarcity of papers addressing the two-dimensional CS-LOSP, the two-dimensional case has many real life applications such as in the glass, metal or furniture industries, among others.

Therefore, the main contribution of this research is the proposal of two integer linear programming formulations for the CS-LOSP for CSP-type problems of any dimension. Furthermore, such formulations are validated through computational experiments on randomly generated instances for the two-dimensional CSP.

This paper is organized as follows. In Section 2, we present a study on the elimination of symmetries in the solution of the CS-LOSP, which leads to a reduction of the search space. Next, in Section 3, we describe the two proposed formulations for the CS-LOSP. Such formulations aim to sequence sets of cutting patterns instead of sequence the cutting patterns individually. Moreover, both proposed formulations take into account the symmetry elimination stated in Section 2. Computational experiments regarding the two-dimensional CS-LOSP were carried out on randomly generated instances and are detailed in Section 4. The data set and the procedure for generating the cutting patterns are detailed in Sections 4.1 and 4.2, respectively. In order to evaluate the quality of the proposed formulations, in Section 4.3 the sizes of the formulations are analyzed. In Section 4.4 the strength of the models is shown *via* the quality of the linear programming relaxation of the three formulations addressed in this work. In Section 4.5 we evaluate the computational effort for the proposed formulations and the one from the literature. Finally, the conclusions and future research proposal are provided in Section 5.

2. Symmetries in the CS-LOSP solution

One of the challenges of solving the CS-LOSP is the large amount of symmetrical solutions. Given any set of cutting patterns, there is a variety of possible cutting patterns sequences that respect the limitation on the maximum number of open stacks, resulting in multiple solutions with the same objective function value.

It is possible to reduce the number of feasible cutting patterns sequences and, consequently, the search space for solutions. Dominance relationships among the cutting patterns are presented in [3,8] in the MOSP context. In those papers, the authors show that given a cutting pattern j_1, if there is any cutting pattern j_2, such that all items generated by the cutting pattern j_1 are also generated by the execution of the cutting pattern j_2, then the cutting pattern j_1 is eliminated from the sequence.

In the context of CS-LOSP, in order to guarantee an optimal solution, it is necessary to consider all the cutting patterns, without exception. Therefore, the strategy used in [3,8] needs to be adapted. Instead of eliminating the

dominated cutting patterns, the approach developed in this research consists of grouping the cutting patterns into sets and then determine an optimal sequence of such sets.

Definition 1. Any two cutting patterns are called to be of the same type if when they are executed they produce the same types of items, regardless of the quantities of each item. Otherwise, they are called cutting patterns of different types.

Consider a set of arbitrary feasible cutting patterns P, with cardinality equal to N, containing K distinct types of cutting patterns. The cutting patterns in P can be grouped into K proper subsets of P, in order to ensure that each of the K subsets contains only cutting patterns of the same type. Let the subsets with such properties be given by $C_k, k = 1, \ldots, K$. So, the following properties apply.

Proposition 1. *Given a sequence of cutting patterns in which n cutting patterns in C_k, for any $k \in \{1, 2, \ldots, K\}$, are executed one immediately after the other, the order in which these n cutting patterns are executed can be swapped without changing the maximum number of open stacks by this sequence.*

Proof. Let s be an arbitrary sequence of cutting patterns in which n cutting patterns in C_k are executed one immediately after the other. Then we divide s in three subsequences, s_1, s_2 and s_3, where s_1 is equivalent to the part of the sequence s prior to the execution of the cutting patterns in C_k; s_2 is the order in which the cutting patterns in C_k are executed, and the subsequence s_3 is equivalent to the part of the sequence s which is related to the cutting patterns that are executed after the cutting patterns in C_k.

Let $\overline{K}$ be the maximum number of stacks opened by the subsequence s_1 and K be the number of stacks that are opened right after the completion of the last element of subsequence s_1. When a cutting pattern is performed, for each distinct type of item generated by the cutting pattern, a new stack is opened if and only if the stack for such item has not been previously opened. Since all cutting patterns in C_k generate the same item types, then, after the execution of the first cutting pattern in s_2, all stacks related to items which were generated by cutting patterns in C_k are already opened. Thus, when executing the remaining part of the sequence s_2, no new stack will be opened. Thus, let ℓ be the amount of items of different types generated by cutting patterns in C_k and m be the number of stacks related to items generated by cutting patterns in C_k which were previously opened, that is, during execution of the sequence s_1. Then, the number of stacks opened by the sequence s_2 is given by $K + 1 - m$.

During the execution of the sequence s_3, stacks can be opened or closed. Let $\overline{K_2}$ be the maximum number of stacks opened by the sequence s_3. Therefore, the maximum number of stacks opened by the sequence s is given by:

$$\max\left\{\overline{K}, K + \ell - m, \overline{K_2}\right\}.$$

Since, by hypothesis, s_2 is an arbitrary sequence, we conclude that the maximum number of open stacks during the execution of s is independent of the sequence s_2. Therefore, the order in which the n cutting patterns in s_2 are executed can be swapped without changing the maximum number of open stacks. □

Proposition 2. *Given a set of cutting patterns P, let S be a set of all feasible sequences of cutting patterns in P, where $\overline{S} \subset S$ is the set of all feasible sequence of cutting patterns in P in which all cutting patterns that produce the same item types are performed one right after the other. Moreover, let $\overline{S}^c$ be the complement of $\overline{S}$. So, for every sequence $s^* \in \overline{S}^c$, it is possible to find a sequence $\overline{s}^* \in \overline{S}$ such that the maximum number of open stacks from the execution of $\overline{s}^*$ is equal to or less than the maximum number of open stacks from the execution of s^*.*

Proof. Let p_j be an arbitrary feasible cutting pattern $\forall j = 1, \ldots, N$; P a set containing those N cutting patterns and $s^* \in \overline{S}^c$ an arbitrary sequence of the cutting patterns in P, given by:

$$s^* = p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_N.$$

Let C_1 be the set of all the cutting patterns in P that produce the same types of items as the cutting pattern p_1 does; and let the cardinality of C_1 be equal to n_1. In the sequence s^*, at most, the first n_1 terms belong to C_1. Let n_1^* be the largest index value such that, $\forall\ 1 \leq j \leq n_1^*$, $p_j \in C_1$. By construction, $p_{n_1^*+1} \notin C_1$.

Let C_2 be the set of all the cutting patterns in P that produce the same item types as the cutting pattern $p_{n_1^*+1}$, and let the cardinality of C_2 be equal to n_2. Therefore, in the sequence s^*, at most, the first $n_1 + n_2$ terms belong to $C_1 \cup C_2$; let n_2^* be the largest value such that, $p_j \in C_1 \cup C_2, \forall\ 1 \leq j \leq n_2^*$. By construction, $p_{n_2^*+1} \notin C_1 \cup C_2$. Suppose there are K distinct sets $C_k, 1 \leq k \leq K$ such that all cutting patterns in C_k produce the same types of items. This process can be repeated in order to obtain all sets $C_k, 1 \leq k \leq K$ and the values n_k^*, so that, $p_j \in \cup_{i=1}^{k} C_i, \forall\ 1 \leq j \leq n_k^*$, and $n_k^* \leq \sum_i^k n_i$. Let $\overline{S}^* \in \overline{S}$ be a sequence in which the cutting patterns in C_1 are executed, followed by the execution of all the cutting patterns in C_2, and so on, *i.e.*:

$$\overline{s}^* = C_1 \rightarrow C_2 \rightarrow \ldots \rightarrow C_K.$$

It can then be seen that the maximum number of open stacks from the execution of $\overline{s}^*$ is equal to or less than the maximum number of open stacks from the execution of s^*. The number of open stacks at time instant h is equal to the total number of open stacks minus the total number of closed stacks. Let $\overline{A_h}$ be the number of open stacks during the execution of the hth cutting pattern of the sequence $\overline{s}^*$.

Consider l_k the amount of types of items that are produced by a cutting pattern of the set C_k and l_1 the number of stacks opened by the execution of the first cutting pattern of the sequence $\overline{s}^*$. When executing the other $n_1 - 1$ cutting patterns, no new stack is opened since all cutting patterns in C_1 generate the same types of items, therefore, all the stacks related to these items are already opened. When executing the cutting pattern $n_1 + 1$, the stacks related to the cutting patterns in C_2, that were not previously opened due to the execution of the cutting patterns in C_1, must be opened. Consequently, the total number of stacks that have been opened up to the instant $h = n_1 + 1$ is equal to $l_1 + (l_2 - |J_1 \cap J_2|)$, where J_k is the set of item types cut by a cutting pattern in C_k. The completion of the next $n_2 - 1$ cutting patterns does not open any new stack, as all stacks referring to C_2 have already been opened previously. When the cutting pattern $n_2 + 1$ is executed, exactly the same process is repeated, so the total of stacks that were opened up to the instant $h = n_2 + 1$ is equal to $l_1 + (l_2 - |J_1 \cap J_2|) + (l_3 - |J_1 \cup J_2 \cap J_3|)$. The function $\overline{A_h}$ is given by:

$$\overline{A_h} = \begin{cases} l_1, & h \leq n_1 \\ l_1 + (l_2 - |J_1 \cap J_2|), & n_1 < h \leq n_2 \\ \vdots & \\ \sum_{k=1}^{K} l_k - \sum_{k=1}^{K} \left|\left(\cup_{i=1}^{k-1}(J_i) \cap J_k\right)\right|, & n_{K-1} < h \leq n_K. \end{cases}$$

Now, consider $\hat{s}^*$ a subsequence of s^* composed of the first $n_{K-1}^* + 1$ cutting patterns of the sequence s^*, in the same order of execution as in s^*. Let A_h^* be the total number of open stacks at the instant h. It is easy to show that, when executing the first cutting pattern of the sequence $\hat{s}^*$, l_1 stacks are opened, and when executing the remaining $n_1^* - 1$ cutting patterns, no new stack is opened since all cutting patterns in C_1 generate the same item types. Accordingly, all stacks related to these items are already opened. When executing the cutting pattern $n_1^* + 1$, the stacks referring to the cutting patterns in C_2 that were not previously opened. Hence, the total number of stacks that have been opened up to the instant $h = n_1^* + 1$ is equal to $l_1 + (l_2 - |J_1 \cap J_2|)$. The completion of the next $n_3^* - n_1^* - 1$ cutting patterns does not open any new stack, since only cutting patterns in $C_1 \cup C_2$ will be performed and all the stacks related to C_1 and C_2 have already been opened before. When the cutting pattern $n_3^* + 1$ is executed, exactly the same process happens. Thus, the total number of stacks that have been opened up to the instant $h = n_3^* + 1$ is equal to $l_1 + (l_2 - |J_1 \cap J_2|) + (l_3 - |J_1 \cup J_2 \cap J_3|)$. Therefore, the function A_h^* is very similar to the function $\overline{A_h}$, the only difference between them is the range of h:

$$A_h^* = \begin{cases} l_1, & h \leq n_1^* \\ l_1 + (l_2 - |J_1 \cap J_2|), & n_1^* < h \leq n_2^* \\ \vdots & \\ \sum_{k=1}^{K} l_k - \sum_{k=1}^{K} \left|\left(\cup_{i=1}^{k-1}(J_i) \cap J_k\right)\right|, & n_{K-1}^* < h \leq n_K^*. \end{cases}$$

Let $\overline{F_h}$ be the number of closed stacks after the execution of the hth cutting pattern of the sequence $\overline{s}^*$. If $h \leq n_1$ then no stack can be closed since there is no item such that all cutting patterns containing this item have already been executed. At the instant $h = n_1 + 1$, all cutting patterns in C_1 were already completed, then, the stacks related to the items in C_1, that are not present in any other cutting pattern, can be closed. Therefore, there is a total of $|J_1 \backslash \{(\cup_{k=2}^{K}(J_k)\}|$ closed stacks at this instant. Analogous to what was done before, no stack can be closed until the instant $h = n_2 + 1$. At this instant, all the cutting patterns related to the sets C_1 and C_2 have been executed. Therefore, the stacks related to the types of items present in the cutting patterns in $C_1 \cup C_2$ which are not present in any other cutting pattern in C_k, $\forall k = 1, \ldots, K$ can be closed, *i.e.*, $|J_1 \cup J_2 \backslash \{(\cup_{k=3}^{K}(J_k)\}|$ stacks are closed. In general, we have:

$$\overline{F_h} = \begin{cases} 0, & h \leq n_1 \\ \left|J_1 \backslash \left\{\left(\cup_{k=2}^{K}(J_k)\right)\right\}\right|, & n_1 < h \leq n_2 \\ \left|J_1 \cup J_2 \backslash \left\{\left(\cup_{k=3}^{K}(J_k)\right)\right\}\right|, & n_2 < h \leq n_3 \\ \qquad\qquad \vdots & \\ \left|\cup_{k=1}^{K-2}(J_k) \backslash \left\{\left(\cup_{k=K-1}^{K}(J_k)\right)\right\}\right|, & n_{K-2} < h \leq n_{K-1} \\ \left|\cup_{k=1}^{K-1}(J_k) \backslash \{J_K\}\right|, & n_{K-1} < h \leq n_K. \end{cases}$$

Let $\overline{P_h}$ be the number of stacks that are open during the completion of the hth cutting pattern of the sequence $\overline{s}^*$, then:

$$\overline{P_h} = \overline{A_h} - \overline{F_{h-1}}.$$

The maximum number of stacks that are open simultaneously during the execution of the sequence $\overline{s}^*$ is given by:

$$\max\left\{\overline{P_1}, \overline{P_2}, \ldots, \overline{P_N}\right\} = P_{\overline{h}},$$

for some $\overline{h} \in \{1, 2, \ldots, N\}$. Note that $\overline{h}$ is contained in the range $\{n_j, n_{j+1}\}$ for some j and therefore:

$$\overline{P_{\overline{h}}} = \overline{A_{\overline{h}}} - \overline{F_{\overline{h}-1}} = \left[\sum_{k=1}^{j+1} l_k - \sum_{k}^{j+1}\left|\left(\bigcup_{i=1}^{k-1}(J_i) \cap J_k\right)\right|\right] - \left[\left|\bigcup_{k=1}^{j}(J_k) \backslash \left\{\left(\bigcup_{k=j+1}^{K}(J_k)\right)\right\}\right|\right].$$

Let P_h^* be the number of stacks that are open during the completion of the hth cutting pattern of the sequence $\overline{s}^*$, then: where F_h^* is the number of close stacks after the completion of the hth cutting pattern of the sequence $\hat{s}^*$. Unless all cutting patterns of a set C_k have been executed, the stacks referring to the items generated by this set cannot be closed. At the instant $h^* \in (n_j^*, n_{j+1}^*]$, all executed cutting patterns belong to one of the sets $C_1, C_2, \ldots, C_{n_j}^*, C_{n_{j+1}^*}$. Let C be the set containing all the sets C_k, such that, all cutting patterns in C_k were executed until the instant h^* and C^C the complement of C; hence, $C \subset \{1, 2, \ldots, j\}$ and therefore:

$$F_{h^*}^* = \left(\left|\bigcup_{k \in C}(J_k) \backslash \left(\bigcup_{k \in C^c}(J_k)\right)\right|\right) \leq \left(\left|\bigcup_{k=1}^{j}(J_k) \backslash \left(\bigcup_{k=j+1}^{K}(J_k)\right)\right|\right) = \overline{F_{\overline{h}}}.$$

Concomitantly, we have:

$$A_{h^*}^* = \sum_{k=1}^{j+1} l_k - \sum_{k=1}^{j+1}\left(\bigcup_{i \leq (k-1)}(J_i) \cap J_k\right) = \overline{A_{\overline{h}}},$$

and then:

$$\overline{P_{\overline{h}}} \leq P_{h^*}^* \leq \max\left\{P_1^*, P_2^*, \ldots, P_{n_{k-1}^*+1}^*\right\} \leq P^*,$$

where P^* is the maximum number of open stacks when executing the sequence s^*. Therefore, for every sequence in $\overline{S}^c$ it is possible to find a sequence $\overline{s}^* \in \overline{S}$ such that the maximum number of open stacks from execution of $\overline{s}^*$ is equal to or less than the maximum number of open stacks from the execution of s^*. □

3. The proposed formulations for the CS-LOSP

Propositions 1 and 2 presented in Section 2 can be used to reformulate the CS-LOSP as shown in this section. Proposition 1 guarantees that, for a sequence in $\overline{S}$, the order in which the cutting patterns of the same type are carried out can be changed without changing the maximum number of stacks opened by this sequence. Proposition 2 guarantees that if there is a sequence of cutting patterns that opens at most C stacks, then there is a sequence in which cutting patterns of the same type are performed, one right after the other, that respects the limitation on the maximum number of open stacks. Thus, the search space is reduced and thereby only the sequences in $\overline{S}$ need to be considered. In this way, the CS-LOSP can be reformulated as a problem of determining the frequency at which the cutting patterns should be performed in order to minimize material waste and, at the same time, to find a sequence of sets of cutting patterns of the same type such that its fulfillment opens, at most, C stacks.

According to Definition 1 (Sect. 2), any set of P cutting patterns containing K different types of cutting patterns can be divided into K subsets of cutting patterns of the same type. Let C_k, $k = 1, \ldots, K$, be the disjoint subsets of P such that every cutting pattern in C_k is of the same type.

Next, we state the indices, parameters and variables that are used in the proposed formulations for the CS-LOSP which are presented in Sections 3.1 and 3.2.

Indices, parameters and variables

I Set of all types of items;
i Index related to the item type, $i = 1, \ldots, |I|$;
N Number of cutting patterns;
j Index related to the cutting pattern, $j = 1, \ldots, N$;
C Maximum limit on the number of open stacks;
K Number of distinct sets C_k;
c_j Cost associated to the cutting pattern j, $j = 1, \ldots, N$;
d_i Demand of item type i, $i = 1, \ldots, |I|$;
N_2 Number of distinct sets C_k plus the dummy set C_0;
t Index related to the time instant, $t = 1, \ldots, K$;
R_k Set of the indices of the cutting patterns in the set C_k;
J_2' Set of the sets $C_k, k = 0, \ldots, K$, *i.e.*, including the dummy set;
J_2 Set of the sets $C_k, k = 1, \ldots, K$, *i.e.*, excluding the dummy set;
P_2^i Set which contains sets of the cutting patterns of the same type that produce item type i;
J' Set of all cutting patterns, including the dummy cutting pattern;
J Set of all cutting patterns, excluding the dummy cutting pattern;
I_k Set of items types produced by a cutting pattern in C_k;
B constant given by $\sum_{i=1}^{|I|} d_i$;
$\boldsymbol{\alpha}^{(j)} = \left(\alpha_1^{(j)}, \alpha_2^{(j)}, \ldots, \alpha_{|I|}^{(j)}\right)^T$ $|I|$-dimensional vector such that $\alpha_i^{(j)}$ indicates the number of times that an item type i is allocated in the jth cutting pattern;
p_j Integer decision variable that determines how many times the cutting pattern j, $j = 1, \ldots, N$ is performed;
v_j Binary decision variable that is equal to 1 if the the cutting pattern j is used on the solution and 0 otherwise.

Considering the indices, parameters and sets defined above, in the following Sections 3.1 and 3.2, we present two new linear integer programming models for the CS-LOSP that aims to sequence sets of cutting patterns of the same type instead of sequencing the cutting patterns individually.

3.1. Formulation I

While in the formulation proposed in [24] the cutting patterns were individually sequenced, the Formulation I, proposed here, seeks to determine a sequence of sets of cutting patterns of the same type that respects the limitation on the maximum number of open stacks. This approach significantly reduces the number of variables and constraints of the model. Firstly, consider the following variables:

$$w_{kt} = \begin{cases} 1, & \text{if the set of cutting patterns of the same type } C_k \text{ is the } t\text{th to be processed;} \\ 0, & \text{otherwise.} \end{cases}$$

$$z_{ti} = \begin{cases} 1, & \text{if the stack referring to the item type } i \text{ is opened from instant } t-1 \text{ to } t; \\ 0, & \text{otherwise.} \end{cases}$$

$$y_{ti} = \begin{cases} 1, & \text{if the stack referring to the item type } i \text{ is open at the instant } t; \\ 0, & \text{otherwise.} \end{cases}$$

Note that, a set of cutting patterns of the same type C_k must be performed at some point in the sequence of a solution if, and only if, at least one cutting pattern in C_k is used. Thus, consider the following constraints:

$$B\left(\sum_{t=1}^{K} w_{kt}\right) \geq \left(\sum_{j \in R_k} v_j\right), \qquad k = 1, \ldots, K \tag{1}$$

$$\sum_{t=1}^{K} w_{kt} \leq \left(\sum_{j \in R_k} v_j\right), \qquad k = 1, \ldots, K \tag{2}$$

$$\sum_{t=1}^{K} w_{kt} \leq 1, \qquad k = 1, \ldots, K \tag{3}$$

$$B\left(\sum_{k=1}^{K} w_{kt}\right) \geq \left(\sum_{j \in R_t} v_j\right), \qquad t = 1, \ldots, K \tag{4}$$

$$\sum_{k=1}^{K} w_{kt} \leq \left(\sum_{j \in R_t} v_j\right), \qquad t = 1, \ldots, K \tag{5}$$

$$\sum_{k=1}^{K} w_{kt} \leq 1, \qquad t = 1, \ldots, K. \tag{6}$$

Constraint sets (1) and (2) cause $\sum_{t=1}^{K} w_{kt} = 0$ if and only if $\sum_{j \in R_k} v_j = 0$, as long as the value of the constant B is large enough. Thus, the set of cutting patterns C_k is sequenced if and only if any cutting pattern in C_k is used. Since w_{kt} is a binary variable for every $k, t \in \{1, \ldots, K\}$, the set of constraints (3) guarantees that, if any pattern in the set of cutting patterns of the same type C_k is used, this set of cutting patterns of the same type is the tth to be processed, for a unique $t \in \{1, \ldots, K\}$. Similarly, the sets of constraints (4) and (5) guarantee that $\sum_{k=1}^{K} w_{kt} = 0$ if and only if $\sum_{j \in R_t} v_j = 0$, as long as the value of the constant B is large enough. Therefore, if no cutting pattern belonging to the cutting pattern set of the same type C_t is used, then no cutting pattern is executed at the instant t. This way, this instant can be ignored in sequencing. Since w_{kt} is a binary variable for every $k, t \in \{1, \ldots, K\}$, the set of constraints (6) guarantees that at most one set of cutting patterns of the same type is executed at the instant t, for every $t \in \{1, \ldots, K\}$. Note that, since there are cutting patterns that may not be used in the final solution, it is allowed that there are instants of time in which no cutting pattern is executed, thus, in the optimal sequencing it is necessary to exclude the instants in which no set of cutting patterns is used.

Moreover, if the set of cutting patterns C_k is carried out at instant t, then all the stacks related to the item types in P_2^i must be opened at the instant t, so, we introduce a new set of constraints, given by:

$$w_{kt} \leq y_{ti}, \qquad k \in P_2^i, t = 1, \ldots, K, i \in I. \tag{7}$$

Note that if none of the cutting patterns in the set of cutting patterns of the same type C_k is used then $w_{kt} = 0$, $t \in \{1, \ldots, K\}$, and, therefore, the set of constraints (7) becomes trivial. The remaining constraints of Formulation I are identical to those of the proposed model in [24]. The Formulation I is given by:

(Formulation I)

$$\min \sum_{j=1}^{N} c_j p_j \tag{8}$$

$$\text{subject to: } \sum_{j=1}^{N} \alpha_i^{(j)} p_j \geq d_i, \qquad i \in I \tag{9}$$

$$\sum_{t=2}^{K} \sum_{i \in I} z_{ti} \leq |I| - C, \tag{10}$$

$$v_j \leq p_j, \qquad j = 1, \ldots, N \tag{11}$$

$$Bv_j \geq p_j, \qquad j = 1, \ldots, N \tag{12}$$

$$w_{kt} \leq y_{ti}, \qquad k \in P_2^i, t = 1, \ldots, K, i \in I \tag{13}$$

$$\sum_{i \in I} y_{ti} \leq C, \qquad t = 1, \ldots, K \tag{14}$$

$$y_{ti} - y_{(t-1)i} \leq z_{ti}, \qquad t = 2, \ldots, K, i \in I \tag{15}$$

$$p_j \in \mathbb{Z}^+, \qquad j = 1, \ldots, N \tag{16}$$

$$v_j \in \{0, 1\}, \qquad j = 1, \ldots, N \tag{17}$$

$$w_{kt} \in \{0, 1\}, \qquad k = 1, \ldots, K, t = 1, \ldots, K \tag{18}$$

$$y_{ti} \in \{0, 1\}, \qquad t = 1, \ldots, K, i \in I \tag{19}$$

$$z_{ti} \in \{0, 1\}, \qquad t = 1, \ldots, K, i \in I, \tag{20}$$

and constraints (1)–(6).

The objective function (8) aims to minimize the overall cost associated to the cutting patterns. The constraints (9) ensures that the demand is fulfilled. The constraint (10) guarantees that the maximum number of simultaneously open stacks equals C. This constraint derives from the fact that a solution for the MTSP in which $|I| - C$ tool changes are performed represents, in the context of the MOSP, a sequence of cutting patterns that opens at most C stacks as demonstrated in [23]. The constraints (11) and (12) guarantee that $v_j = 0$ if and only if $p_j = 0$. The constraints (1)–(6) were explained earlier in this section. The constraint set (13) guarantees that if a cutting pattern in C_k is used, then all stacks open due to the execution of this set must be taken into account; otherwise, the constraint becomes trivial. The set of constraints (14) prevents the open stack capacity limit from being exceeded. The set of constraints (15) ensure that if the stack i was not open at the instant of time $t - 1$ and it is open at the time instant t, then, the stack must be open from the instant $t - 1$ to t. The constraints (16)–(18) define the domain of variables.

3.2. Formulation II

In this section, we propose a second formulation for the CS-LOSP based on the models proposed in [5,9]. In this formulation, the sets of cutting patterns to be performed are represented by the vertices of a graph and the

sequence in which the sets of cutting patterns are carried out is equivalent to the order in which the vertices are traversed in an Euler circuit. In this model, a dummy set of cutting patterns is established, which represents the beginning and the end of the Euler circuit. Let k, f and o be the indices related to the sets of cutting patterns of the same type, $\forall k, f, o \in J_2'$. For all $k \in J_2'$, $f \in J_2'$ and $k \neq f$, let x_{kf} be equal to 1, if the execution of the set of cutting patterns of the same type C_f is done immediately after the execution of the set C_k; and 0, otherwise. Similarly, for all $k \in J_2'$, $f \in J_2'$, $k \neq f$ and $i \in (I \backslash I_k) \cup (I_k \backslash I_f)$, let y_{kf}^i be a binary variable equal to 1 if the execution of the set of cutting patterns of the same type C_f is done immediately after the execution of the set C_k and if the stack related to the item type i is open after the execution of the set of cutting patterns of the same type C_k and before the execution of the set of cutting patterns of the same type C_f; and 0, otherwise. Note that the variable y_{kf}^i is not defined for $i \in I_k \cap I_f$, once the stack i must be open after the execution of the set C_k which implies $y_{kf}^i = 1$, so there is no need to consider the set of variables for $i \in I_k \cap I_f$. Finally, for all $k \in J_2$, let u_k be an auxiliary variable used to prevent the formation of sub-cycles, as proposed in [17]. In this section, we chose to present the constraints of Formulation II in blocks to facilitate the understanding of the modeling. Firstly, we present the constraints regarding the sequence of the sets of cutting patterns of the same type, then we present another block of constraints that ensures that the sequence of the sets of cutting patterns of the same type respects the limitation on the maximum number of simultaneously open stacks. This latter block of constraints is based on some of the constraints proposed in [5]. Finally, we present the objective function and some additional constraints.

For all $k \in J'$, the set of cutting patterns of the same type C_k must be present in the sequence of sets of cutting patterns of the same type if, and only if, at least one cutting pattern in C_k is executed in the CS-LOSP solution. Therefore, we propose the following block of constraints:

$$\sum_{k \in J_2' \backslash \{0\}} x_{k0} = 1, \tag{21}$$

$$\sum_{f \in J_2' \backslash \{0\}} x_{0f} = 1, \tag{22}$$

$$B\left(\sum_{f \in J_2' \backslash \{k\}} x_{kf}\right) \geq \left(\sum_{j \in R_k} v_j\right), \qquad k \in J_2 \tag{23}$$

$$\sum_{f \in J_2' \backslash \{k\}} x_{kf} \leq \left(\sum_{j \in R_k} v_j\right), \qquad k \in J_2 \tag{24}$$

$$\sum_{f \in J_2' \backslash \{k\}} x_{kf} \leq 1, \qquad k \in J_2 \tag{25}$$

$$B\left(\sum_{k \in J_2' \backslash \{f\}} x_{kf}\right) \geq \left(\sum_{j \in R_f} v_j\right), \qquad f \in J_2 \tag{26}$$

$$\sum_{k \in J_2' \backslash \{f\}} x_{kf} \leq \left(\sum_{j \in R_f} v_j\right), \qquad f \in J_2 \tag{27}$$

$$\sum_{k \in J_2' \backslash \{f\}} x_{kf} \leq 1, \qquad f \in J_2 \tag{28}$$

$$v_0 = 1. \tag{29}$$

The constraints (21), (22) and (29) guarantee that the set of dummy cutting patterns is used. The sets of constraints (23) ensure that, considering B as a sufficiently large constant, if a cutting pattern in C_k is used

($\sum_{j\in R_k} v_j > 0$) then at least one set C_f must be performed immediately after the execution of the cutting patterns in C_k. The constraints (24) guarantee that if no cutting pattern in C_k is used ($\sum_{j\in R_k} v_j = 0$) then no cutting patterns set C_f is executed immediately after the execution of the cutting patterns in C_k. Lastly, the constraints (25) ensure that if any cutting pattern in C_k is used, then exactly one set of cutting patterns C_f is executed immediately after the execution of the cutting patterns set C_k. Similarly, the sets of constraints (26), (27) and (28) assure that if any cutting pattern in C_f is used, then exactly one set of cutting patterns C_k is executed prior to the execution of the set of cutting patterns C_k. If no cutting pattern in C_f is used, then the execution of the set of cutting patterns C_f can not be preceded by the execution of any set of cutting patterns of the same type.

Following, we present and describe, in the context of the CS-LOSP, the constraints proposed in [5] that are used in the Formulation II.

$$\sum_{k\in J_2'\setminus\{f\}} y_{kf}^i - \sum_{o\in J_2'\setminus\{f\}} y_{fo}^i \geq 0, \qquad f\in J_2', i\in I\setminus I_f \tag{30}$$

$$\sum_{\substack{i\in I:\\ i\in I_k\\ i\notin I_f}} y_{kf}^i + \sum_{\substack{i\in I:\\ i\notin I_k\\ i\notin I_f}} y_{kf}^i \leq (C-|I_f|)x_{kf}, \qquad k,f\in J_2', k\neq f \tag{31}$$

$$y_{kf}^i \leq x_{kf}, \qquad k\in J_2, i\in I\setminus I_f. \tag{32}$$

The constraints (30) ensure that a stack related to an item type in $I\setminus I_f$ can only be open after the execution of the set of cutting patterns of same type C_f if the stack was already open before the execution of the set of cutting patterns of same type C_f. The left-hand side of the set of constraints (31) counts the number of open stacks after the execution of the set C_k referring to items that are not generated when executing a cutting pattern in C_f. Thus, the constraints (31) ensure that this quantity, plus the number of different item types generated by the execution of a cutting pattern in C_f is less than or equal to C, ensuring that the limit on the maximum number of open stacks is respected. Finally, the constraints (32) ensure that y_{kf}^i can only be equal to one if the set C_f is executed immediately after C_k. If a set of cutting patterns o is not used, then we have $x_{of} = x_{ko} = 0$, for every $f,k\in J_2'$, the set of constraints (32), proposed in [5], does not account for this particular case, therefore, we add the following constraints to the model:

$$\sum_{\substack{i\in I:\\ i\notin I_k\\ i\in I_f}} (x_{kf} - y_{kf}^i) \geq 0, \qquad k,f\in J_2', k\neq f. \tag{33}$$

The set of constraints (33) together with the constraints (32) ensure that if a set of cutting patterns o is not used, then $y_{of}^i = y_{ko}^i = 0$ for $k,f\in J_2'$. The remaining constraints as well as the objective function of the proposed Formulation II is given by:

(Formulation II)

$$\min \sum_{j=1}^{N} c_j p_j \tag{34}$$

$$\text{subject to: } \sum_{j=1}^{N} \alpha_i^{(j)} p_j \geq d_i, \qquad i\in I \tag{35}$$

$$\sum_{\substack{k,f\in J\\ k\neq f}} \sum_{\substack{i\in I:\\ i\notin I_k\\ i\in I_f}} \left(x_{kf} - y_{kf}^i\right) \leq |I| - C, \tag{36}$$

$$v_j \leq p_j, \qquad j \in J \tag{37}$$
$$Bv_j \geq p_j, \qquad j \in J \tag{38}$$
$$u_k - u_f + N_2 x_{kf} \leq N_2 - 1, \qquad k, f \in J_2, k \neq f \tag{39}$$
$$x_{kf} \in \{0,1\}, \qquad k, f \in J_2', k \neq f \tag{40}$$
$$y^i_{kf} \in \{0,1\}, \qquad k, f \in J_2', k \neq f, i \in (I \backslash I_k) \cup (I_k \backslash I_f) \tag{41}$$
$$v_j \in \{0,1\}, \qquad j \in J' \tag{42}$$
$$p_j \in \mathbb{Z}^+, \qquad j \in J' \tag{43}$$
$$u_k \in \{1,2,\ldots,N_2\} \qquad k \in J_2 \tag{44}$$

and constraints (21)–(33).

The objective function (34), as well as the constraints (35)–(38), are equivalent to the sets of constraints (8)–(12), respectively. The constraints (39) are based on the constraints proposed in [17], to avoid sub-cycles. This set of constraints have two main goals, the first is to ensure that the sequence of sets of cutting patterns starts and ends with the dummy set C_0. The second goal of the constraints is to eliminate sequences in which more than N_2 sets of cutting patterns are executed, *i.e.*, to eliminate sub-cycles in which a set of cutting patterns C_k is executed more than once. A detailed explanation of this set of constraints can be found in [17]. As stated in [17], the variables u_k are defined as non-negative integers. We can show that in this particular case their domain can be restricted to $\{1,2,\ldots,N_2\}$. Let $\bar{s} = C_{k_1} \rightarrow C_{k_1} \rightarrow \ldots \rightarrow C_{k_n}$ be a feasible sequence of sets of cutting patterns with $n \leq N_2$, then, by setting $u_{k_1} = 1, u_{k_2} = 2, \ldots, u_{k_n} = n$ we can see that the constraints (39) are satisfied. Since in all feasible sequence of sets of cutting patterns, at most N_2 sets of cutting patterns are executed, we conclude that the domain of the variables u_k can be restricted to $\{1,2,\ldots,N_2\}$. The remaining constraints are related to the domain of the variables.

4. Computational experiments

In order to validate the proposed models and to compare the time required to determine an optimal solution, computational tests on random instances are performed and analyzed. In Section 4.1 we describe the random data set generated for the two-dimensional CS-LOSP, which are available at https://github.com/ggazzinelli/2D-CS-LOSP, and in Section 4.2 we detail the cutting pattern generation used in this work as well as a proposition that guarantees that the dominated cutting patterns can be removed from the solution without loss of optimality. Analyses of the size and the quality of the linear programming relaxation of the proposed mathematical formulations are presented in Sections 4.3 and 4.4, respectively. Finally, in Section 4.5 we present the computational results for the two-dimensional CS-LOSP.

The approaches were coded in Julia Programming Language (version 1.1.1) using the IBM ILOG CPLEX (version 20.1) as the general ILP solver. The experiments were carried out on a computer with an Intel 7i-8700 (3.20 GHz), 16 GB of RAM.

4.1. Data set for the two-dimensional CS-LOSP

In this section we describe the proposed data set for the two-dimensional CS-LOSP that was considered for analyzing and comparing the Integrated Yanasse and Lamosa (IYL), Formulation I and Formulation II models. We assume that a sufficiently large number of identical two-dimensional objects is available to be cut. According to the typology proposed in [21], in the context of CSPs, this particular CSP is known as the Single Stock Size Cutting Stock Problem (SSSCSP). Furthermore, for the computational experiments, exact two-stage guillotine cutting patterns were considered. The characteristics of the instances are described as follows.

L Length of the standard object;
W Width of the standard object;
$|I|$ Number of item types;

ℓ_i Length of item type i, $i = 1, \ldots, |I|$;
w_i Width of item type i, $i = 1, \ldots, |I|$;
d_i Demand of item type i, $i = 1, \ldots, |I|$.

The dimension of the standard objects to be cut $L \times W$ is fixed at 1000×500. Two intervals were considered to generate the length of the items $[200, 500]$ and $[300, 800]$. The length of the demanded items ℓ_i was randomly generated without repetition, that is, items of different types have different lengths. The width of the required items w_i was randomly generated as one of the values of the set $\{100, 200, 300, 400\}$. The number of distinct types of items in each instance is 5 or 10. The demand of items was randomly generated in the range $[100, 400]$, *i.e.*, $d_i \in [100, 400]$, $i = 1, \ldots, |I|$. By combining the different parameters mentioned above, four classes of instances were generated:

- Class 1: $|I| = 5$; $\ell_i \in [200, 500]$;
- Class 2: $|I| = 10$; $\ell_i \in [200, 500]$;
- Class 3: $|I| = 5$; $\ell_i \in [300, 800]$;
- Class 4: $|I| = 10$; $\ell_i \in [300, 800]$.

Each class contains 10 instances. Classes 1–4 are used to compare the time each model spent to solve the instances of the CS-LOSP.

The C parameter represents the amount of stacks that can be opened, this parameter can significantly change the time required to reach an optimal solution. Since $|I|$ is the number of types of items, the value of C can vary from 1 up to $|I|$. In [2], the authors argue that the relevant values of C are typically small. Furthermore, from the point of view of practical application, this number should also be small because the greater the number of unloading stations (number of open stacks), the more expensive and spacious the equipment must be. Thus, in the computational tests, we considered three values for C, namely: $C = 2$, $C = 3$ and $C = 4$. Note that for $C = 1$ only homogeneous cutting patterns can be used and, therefore, the problem is trivial. Moreover, for the computational experiments, we consider $c_j = 1$, $j = 1, \ldots, N$.

4.2. Generation of cutting patterns and reduction strategies *via* dominance and feasibility

As described in [11], the 2-stage guillotine cutting patterns can be built in two steps. In the first step, horizontal strips are defined and, for each horizontal strip, the feasible cutting patterns. In the second step, the number of times each strip is used with its respective cutting pattern is determined.

Considering the exact 2-stage guillotine cutting pattern, an item type i can be produced by cutting a strip if and only if the width of the strip and the item type i is the same. Thus, we are interested only in strips such as its width is equal to w_i for at least one index $i \in I$. Let O be the number of distinct strips that are capable of producing at least one type of item when cut, note that since it is possible to have $w_{i_1} = w_{i_2}$ for $i_1 \neq i_2$, then $O \leq |I|$. Let $o = 1, \ldots, O$ be the indexes that identify the strips of dimensions $L \times w_o$ and E_o be the set of item types that may be produced by the strip o, *i.e.*, $E_o = \{i \mid w_i = w_o\}$.

Each o strip can be cut in several ways, the specific way in which a strip is cut corresponds to a one-dimensional cutting pattern. Considering θ_o^i as the number of times that item type i is present in the strip o, $i \in I$ we must have:

$$\sum_{i \in E_o} \ell_i \theta_o^i \leq L \qquad o = 1, \ldots, O. \tag{45}$$

Every feasible solution of (45) corresponds to a feasible one-dimensional cutting pattern for the strip o. Let N_o be the set of the distinct one-dimensional cutting patterns that can be obtained by cutting a strip of dimensions $L \times w_o$. For all $o \in O$ and $j \in N_o$, let β_o^j be the number of times the strip o is cut according to the cutting pattern j, $\forall j \in N_o$ and $o \in O$, β_o^j must satisfy:

$$\sum_{j \in N_o} \sum_{o=1}^{O} w_o \beta_o^j \leq W. \tag{46}$$

Finally, let γ_{oi}^{j} be the number of times item type i is produced by cutting the strip o according to the one-dimensional cutting pattern j. A two-dimensional guillotine 2-stage exact cutting pattern can be represented by an $|I|-$dimensional vector $\boldsymbol{\alpha^{(j)}} = (\alpha_1^{(j)}, \alpha_2^{(j)}, \ldots, \alpha_{|I|}^{(j)})^t$ where the ith coordinate corresponds to the number of times that item type i is present in the cutting pattern.

$$\alpha_i^{(j)} = \gamma_{oi}^{j}\beta_o^j \qquad i \in I. \tag{47}$$

The number of cutting patterns of a class depends on the dimension of the items (length and width) and the number of different types of items. For classes such that the size of the items is relatively small or that there are a lot of items of different types, the number of feasible cutting patterns can be excessively large, making it difficult to solve the problem up to optimality.

In this subsection, we show that there are two subsets of the cutting patterns that can be eliminated without affecting the CS-LOSP optimal solution. The first one regards the cutting patterns that become infeasible due to the limitation on the number of open stacks. Whenever a cutting pattern is performed, the stacks related to the cut items must be opened. Thus, if the number of different types of items in a single cutting pattern already exceeds the maximum limit of stacks that can be opened simultaneously, this cutting pattern cannot be part of the solution and can be removed from the problem without changing the CS-LOSP optimal solution. The second one regards the cutting patterns that are dominated by another cutting pattern. We define dominated cutting patterns, in the context of a CS-LOSP, through Proposition 3, as follows:

Proposition 3. *In a CS-LOSP in which overproduction is allowed, and the objective is to minimize the number of cut objects, consider two distinct cutting patterns:* $\boldsymbol{\alpha^{(j_1)}} = (\alpha_1^{(j_1)}, \alpha_2^{(j_1)}, \ldots, \alpha_{|I|}^{(j_1)})^T$ *and* $\boldsymbol{\alpha^{(j_2)}} = (\alpha_1^{(j_2)}, \alpha_2^{(j_2)}, \ldots, \alpha_{|I|}^{(j_2)})^T$. *If both cutting patterns produce the same types of items when cut, that is, for all* i *such that* $\alpha_i^{(j_1)} = 0$, *we have* $\alpha_i^{(j_2)} = 0$ *and vice versa, and* $\alpha_i^{(j_2)} \geq \alpha_i^{(j_1)}$ *for* $i \in I$. *Then, the cutting pattern* $\boldsymbol{\alpha^{(j_1)}}$ *is dominated by the cutting pattern* $\boldsymbol{\alpha^{(j_2)}}$. *Therefore,* $\boldsymbol{\alpha^{(j_1)}}$ *can be removed from the problem without changing the optimal CS-LOSP solution.*

Proof. Consider an optimal solution for the CS-LOSP in which the cutting pattern $\boldsymbol{\alpha^{(j_1)}}$ is part of the solution. When $\alpha_i^{(j_2)} \geq \alpha_i^{(j_1)}$ for $i \in I$, then the quantity of items obtained from the cutting pattern j_2 is equal to or greater than the quantity of items obtained from the cutting pattern j_1, for any type of item. Therefore, when replacing the cutting pattern j_1 by the cutting pattern j_2, the quantity of items remains the same or increases, this way the demand constraints are fulfilled and the quantity of used objects remains the same. Furthermore, if for all i such that $\alpha_i^{(j_1)} = 0$, we have $\alpha_i^{(j_2)} = 0$, and *vice versa*, then the number of open stacks is exactly the same, since both cutting patterns generate the same types of items. Therefore, an execution of the cutting pattern j_1 can be replaced by the execution of the cutting pattern j_2 without changing the value of the objective function. Thus, the removal of the cutting pattern j_1 does not change the value of the objective function at all in an optimal solution. In this case, the cutting pattern j_1 is dominated by the cutting pattern j_2. □

Proposition 3 guarantees that all cutting patterns dominated by some other cutting pattern can be removed without altering the optimal solution to the problem. Consequently, the overall number of cutting patterns is reduced. The number of variables and constraints of the models proposed in this research is closely linked to the number of cutting patterns. Thus, adopting the strategy of removing dominated cutting patterns considerably reduces the computational complexity of the problem and, therefore, this procedure is used in computational tests.

Similar to the formulations proposed in [2, 24], the formulations proposed in this research require that the feasible cutting patterns be determined *a priori*. The amount of feasible cutting patterns can be reduced due to Proposition 3. Furthermore, observe that if a cutting pattern produces more than C types of items it is infeasible in the context of CS-LOSP, therefore we only generate cutting patterns that produce at most four different item types, since this is the biggest value of C considered in the computational experiments performed

TABLE 1. Average number of variables and constraints regarding IYL, Formulation I and Formulation II models and Classes 1–4.

Class	IYL	Formulation I	Formulation II
Number of variables			
1	317.66×10	562.70	181.17×10
2	383.669×10^3	230.87×10^2	197.49×10^3
3	154.88×10	424.60	137.14×10
4	281.39×10^2	869.18×10	729.27×10^2
Number of constraints			
1	804.15×10	955.9	224.72×10
2	123.56618×10^4	593.29×10^2	199.79×10^3
3	316.38×10	677.70	171.30×10
4	748.115×10^2	196.49×10^2	744.27×10^2

in this section. Hence, the first step for carrying out the computational experiments, is the generation of the feasible cutting patterns, ignoring the dominated patterns in accordance with Proposition 3 and those such that the number of items generated from their execution is greater than four. This set of generated cutting patterns is used in all computational experiments performed in this section.

4.3. Models size

By sequencing sets of cutting patterns of the same type rather than sequencing the cutting patterns individually, it is possible to significantly reduce the number of variables and constraints of the models. While the IYL model, proposed in [24], has $2|I|N + N^2 + 2N - |I|$ variables, the Formulation I has $2|I|N_2 + N_2^2 + 2N - |I|$ variables. Since the value of N_2 can be significantly lower than the value of N, the number of variables in the Formulation I tends to be smaller than the number of variables in the IYL model.

The same happens for the amount of constraints, while the IYL formulation has $N(5 + |I| + \sum_{i \in I} |P_i|) + 1$ constraints, the Formulation I has $2N + N_2(7 + |I| + \sum_{i \in I} |P_2^i|) + 1$ constraints, and again, we can expect a reduction in the amount of constraints.

The Formulation II has $2(N + 1) + (N_2 + 1)^2 - (N_2 + 1) + \sum_{\substack{k,f \in J_2' \\ k \neq f}} |(I \backslash I_k) \cup (I_k \backslash I_f)| + N_2$ variables and $|I| + 2N + 8N_2 + 3N_2^2 + \sum_{k \in J_2'} |I \backslash I_k| + \sum_{\substack{k,f \in J_2' \\ k \neq f}} |I \backslash I_f| + 4$ constraints. Table 1 shows the comparison between the number of variables and constraints regarding the three formulations for Classes 1–4.

We can notice that, among the three formulations, the Formulation I is the most compact. Moreover, the proposed Formulation I and Formulation II have a lower number of constraints and variables when compared to the formulation from the literature. The difference in size of the models is particularly noticeable for classes with items of small size, *i.e.*, Classes 1 and 2.

4.4. Analysis of the quality of linear programming relaxation

The linear programming relaxation of an ILP model is an important tool to assess the model strength. Moreover, the linear programming relaxation of a formulation can provide useful lower bounds. In order to compare the strength of the formulations proposed in this research and the one proposed in [24], we perform some computational experiments to compare the quality of the linear programming relaxation of the models. In this subsection, we present the results for the computational experiments performed on 10 instances of Class 1 considering $C = 2$, *i.e.*, the maximum number of open stacks allowed was fixed at 2.

For this set of instances, the average value for the integer optimal solution to the CS-LOSP the equals to 347. Regarding the linear programming relaxation of the Formulation I and the model proposed in [24], both

TABLE 2. Time spent to solve instances of Class 1 regarding the IYL, Formulation I and Formulation II models as a function of the value of C.

Id	C	IYL	F1	F2	OF	C	IYL	F1	F2	OF	C	IYL	F1	F2	OF
Class 1															
1	2	0.09	0.35	0.09	467	3	0.22	0.37	0.53	467	4	0.12	0.07	0.15	467
2	2	0.12	0.78	0.09	334	3	0.24	0.07	0.10	334	4	0.14	0.08	0.16	334
3	2	0.11	0.08	0.09	401	3	0.15	0.09	0.13	401	4	0.20	0.08	0.21	401
4	2	0.32	0.19	0.10	187	3	19.15	0.16	0.23	173	4	3.88	0.19	3.11	173
5	2	0.05	0.70	0.08	249	3	0.09	0.07	0.15	249	4	0.06	0.06	0.32	249
6	2	0.73	0.27	0.33	243	3	9.60	0.24	0.68	224	4	11.42	0.31	1.01	224
7	2	–	17.70	–	233	3	2.52	1.56	0.33	227	4	3.64	0.15	0.63	227
8	2	3.36	0.75	0.31	387	3	0.19	0.10	0.21	387	4	0.13	0.08	0.19	387
9	2	1.01	0.07	0.25	395	3	0.08	0.09	0.12	395	4	0.07	0.05	0.24	395
10	2	0.75	0.12	0.09	574	3	0.07	0.54	0.16	574	4	0.07	0.05	0.18	574
Class 2															
1	2	–	–	441.03	426	3	–	–	–	–	4	–	–	–	–
2	2	1427.34	–	8.25	520	3	–	3757.72	–	511	4	–	1855.08		511
3	2	–	–	602.25	448	3	–	3672.45	–	445	4	–	2159.90	2981.36	445
4	2	–	–	2897.06	486	3	–	–	–	–	4	–	1019.69	–	470
5	2	–	–	54.48	523	3	–	1157.67	1042.81	510	4	–	1205.03	–	501
6	2	–	23.34	4.91	537	3	–	4368.41	221.69	535	4	–	2018.19	4886.98	535
7	2	–	–	109.05	391	3	–	550.82	–	367	4	–	–	–	–
8	2	–	623.50	37.90	529	3	–	–	–	–	4	–	65.24	–	475
9	2	–	–	–	–	3	–	157.18	106.67	555	4	–	2593.08	–	555
Class 3															
1	2	0.95	0.07	0.09	369	3	0.92	0.18	0.14	369	4	0.71	0.10	0.16	369
2	2	0.80	0.12	0.16	412	3	0.13	0.07	0.15	412	4	0.09	0.05	0.21	412
3	2	4.21	0.05	0.05	534	3	0.09	0.05	0.13	534	4	0.05	0.21	0.16	534
4	2	0.08	0.07	0.06	376	3	0.11	0.06	0.08	376	4	0.05	0.06	0.13	376
5	2	0.22	0.06	0.15	245	3	0.63	0.13	0.17	245	4	0.54	0.09	0.46	245
6	2	0.07	0.05	0.08	658	3	0.09	0.02	0.21	658	4	0.07	0.05	0.15	658
7	2	0.06	0.03	0.05	637	3	0.05	0.03	0.08	637	4	0.02	0.02	0.08	637
8	2	0.06	0.02	0.07	671	3	0.03	0.02	0.06	671	4	0.02	0.02	0.08	671
9	2	2.23	0.26	0.25	323	3	1.95	0.27	0.70	323	4	0.41	0.09	1.29	323
10	2	0.18	0.02	0.04	853	3	0.25	0.02	0.05	853	4	0.02	0.02	0.05	853
Class 4															
1	2	17.36	0.97	10.22	914	3	564.94	76.18	–	913	4	13.09	54.87	–	913
2	2	106.37	262.12	7.95	1090	3	319.28	7.10	–	1090	4	165.28	14.82	–	1090
3	2	789.44	662.93	18.53	606	3	–	1572.77	–	606					
4	2	–	–	–	–	3	9.31	6.42	3670.90	1042	4	2.34	7.05	–	1042
5	2	3.42	1044.19	45.45	1312	3	26.45	16.27	19.55	1312	4	108.77	11.20	–	1312
6	2	0.42	0.69	4.64	1478	3	0.42	0.69	4.75	1478	4	0.33	9.88	185.51	1478
7	2	17.11	7.25	1.57	1093	3	29.20	23.57	–	1093	4	39.99	0.56	–	1093
8	2	677.95	56.09	5.64	1388	3	14.19	49.89	–	1388	4	8.53	6.74	–	1388
9	2	14.46	0.44	0.76	1168	3	4.81	19.96	620.05	1168	4	19.48	6.63	–	1168
10	2	9.40	13.27	2.45	711	3	4.47	3.97	87.90	711	4	7.33	1.31	1025.38	711

average values equal 342.24; while the average value for linear programming relaxation of the Formulation II is equal to 346.02. These results suggest that the Formulation II is stronger than the Formulation I and the model proposed in [24] as well. This result is somewhat expected since both the Formulation I and the model proposed in [24] are anchored in the [19], which, as stated in [5] is weaker than the formulation proposed in [5], in which the Formulation II is based. A simple way to improve the quality of the linear programming relaxation is to consider only cutting patterns in which at most C items are generated. In this case, the linear programming relaxation yields identical results for all the three formulations. Moreover, for the three of them, the average for the linear programming relaxation is equal to 346.02 for the three formulations, corresponding to a difference of only 0.28% from the integer programming problem.

4.5. Computational results

In this subsection, we compare the computational time spent to determine an optimal solution *via* the IYL, Formulation I and Formulation II models for Classes 1 to 4 and the obtained results are presented in Table 2.

The CPLEX absolute GAP was set at its default of 10^{-6}. Remembering that no cutting pattern with more than C item types was considered. The first column of Table 2 identifies the instance of the class; the second column presents the value of C; the third, fourth and fifth columns present the computational time spent to reach an optimal solution considering the IYL, Formulation I and Formulation II models, respectively. The sixth column presents the value of the objective function of the optimal solution to the problem. The remaining columns of the table repeat the same pattern of the second, third, fourth, fifth and sixth columns. In the Table 2 we refer to the Formulation I and Formulation II as F1 and F2, respectively. Instances in which none of the three models was able to reach an optimal solution, within the time limit of 7200 s, were omitted from the table.

Among the 120 test instances, it was possible to solve 87, 103 and 88 by the IYL, Formulation I and Formulation II models, respectively. The Formulation II model presents interesting results for the instances where $C = 2$, for these instances, while Formulation II outperforms the other two formulations by solving 36 instances up to optimality while the IYL and Formulation I models solve 29 and 31 instances, respectively. For $C = 3$ and $C = 4$ the Formulation I presents better results.

Based on the results presented in the previous tables, we can observe that the size of the items is a factor with a great impact on computational time. For classes with items of smaller size, the time required to determine an optimal solution was greater, and, therefore, the number of instances solved for these classes was smaller. This result is expected since the smaller the size of the items, the greater the number of possible cutting patterns and, consequently, the size of the model. Another parameter that influences the time required to determine an optimal solution is the number of items of different types. Classes containing a greater number of types of items require more computational time.

As discussed in Section 2, by sequencing subsets of cutting patterns of the same type instead of sequencing the cutting patterns individually, it is possible to significantly reduce the number of variables and constraints in the models. This reduction is particularly notable for classes in which the item size is shorter. The reduction in the number of variables and constraints directly reflects on the time required to determine an optimal solution, the models proposed in this research are able to solve a larger number of instances when compared to the IYL formulation, the results are particularly remarkable for Class 2 in which the IYL formulation is able to determine the optimal solution for only one instance, while the Formulation I and Formulation II solve 13 and 15 instances, respectively.

5. Conclusions and future research

In this research, we propose two new integer linear programming models for the cutting stock with limited open stacks problem (CS-LOSP), both valid for the one-dimensional and the two-dimensional CS-LOSP. The proposed models are compared with the formulation proposed in [24]. To guarantee the validity of the model, several computational tests were performed with randomly generated two-dimensional instances of the CSP. In order to eliminate symmetrical solutions of the problem and to reduce the search space for solutions, we demonstrate that the CS-LOSP can be reformulated as the problem of determining how often cutting patterns should be performed in order to minimize material waste while simultaneously finding a sequence of sets of cutting patterns of the same type whose execution respects the pre-established maximum limit on the number of open stacks. It is worth mentioning that sequencing sets of cutting patterns of the same type instead of sequencing all the cutting patterns individually enables a reduction on the number of variables and constraints in the model. This reduction is more expressive for instances where the demanded item sizes are relatively small. Furthermore, by removing symmetric solutions from the problem, it is possible to notice an increase in model performance. Considering the 120 instances used in the computational tests, it was possible to solve a total of 102 and 89 instances respecting the time limitation of 7200, using Formulation I and Formulation II, respectively, while for the model proposed in [24], a total of 87 instances were solved. Although the proposed modeling presents interesting results, the computational tests performed are limited and suggest the need to develop more efficient models for the resolution of the CS-LOSP, especially to deal with instances containing a large number of different types of items where the results obtained in this work are unsatisfactory. As further research, we recommend investigating alternative solution approaches, such as heuristics and metaheuristics, aiming to solve larger instances.

Funding informations. This research was funded by the São Paulo Research Foundation – FAPESP (grant 2016/01860-1) and by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

References

[1] C. Arbib, F. Marinelli and F. Pezzella, An LP-based tabu search for batch scheduling in a cutting process with finite buffers. *Int. J. Prod. Econ.* **136** (2012) 287–296.

[2] C. Arbib, F. Marinelli and P. Ventura, One-dimensional cutting stock with a limited number of open stacks: bounds and solutions from a new integer linear programming model. *Int. Trans. Oper. Res.* **23** (2016) 47–63.

[3] J.C. Becceneri, H.H. Yanasse and N.Y. Soma, A method for solving the minimization of the maximum number of open stacks problem within a cutting process. *Comput. Oper. Res.* **31** (2004) 2315–2332.

[4] G. Belov and G. Scheithauer, Setup and open-stacks minimization in one-dimensional stock cutting. *INFORMS J. Comput.* **19** (2007) 27–35.

[5] D. Catanzaro, L. Golveia and M. Labbé, Improved integer linear programming formulations for the job sequencing and tool switching problem. *Eur. J. Oper. Res.* **244** (2015) 766–777.

[6] J.F. Côté and M. Iori, The meet-in-the-middle principle for cutting and packing problems. *INFORMS J. Comput.* **30** (2018) 646–661.

[7] M. Delorme and M. Iori, Enhanced pseudo-polynomial formulations for bin packing and cutting stock problems. *INFORMS J. Comput.* **32** (2020) 101–119.

[8] E. Faggioli and C.A. Bentivoglio, Heuristic and exact methods for the cutting sequencing problem. *Eur. J. Oper. Res.* **110** (1998) 564–575.

[9] P.C. Gilmore and R.E. Gomory, A linear programming approach to the cutting-stock problem. *Oper. Res.* **9** (1961) 849–859.

[10] P.C. Gilmore and R.E. Gomory, A linear programming approach to the cutting-stock problem – part II. *Oper. Res.* **11** (1963) 863–888.

[11] P.C. Gilmore and R.E. Gomory, Multi-stage cutting stock problems of two and more dimensions. *Oper. Res.* **13** (1965) 94–120.

[12] L.V. Kantorovich, Mathematical methods of organizing and planning production. *Manage. Sci.* **6** (1960) 366–422.

[13] G. Laporte, J.J. Salazar-González and F. Semet, Exact algorithms for the job sequencing and tool switching problem. *IIE Trans.* **36** (2004) 37–45.

[14] R. Macedo, C. Alves and J.M. Valério de Carvalho, Arc-flow model for the two-dimensional guillotine cutting stock problem. *Comput. Oper. Res.* **37** (2010) 991–1001.

[15] M. Martin, H.H. Yanasse and M.J. Pinto, Mathematical models for the minimization of open stacks problem. *Int. Trans. Oper. Res.* **29** (2022) 2944–2967.

[16] M. Martin, H.H. Yanasse, M.O. Santos and R. Morabito, Models for two- and three-stage two-dimensional cutting stock problems with a limited number of open stacks. *Int. J. Prod. Res.* **61** (2023) 2895–2916.

[17] C.E. Miller, A.W. Tucker and R.A. Zemlin, Integer programming formulation of traveling salesman problems. *J. ACM (JACM)* **7** (1960) 326–329.

[18] K.C. Poldi and M.N. Arenales, Heuristics for the one-dimensional cutting stock problem with limited multiple stock lengths. *Comput. Oper. Res.* **36** (2009) 2074–2081.

[19] C.S. Tang and E.V. Denardo, Models arising from a flexible manufacturing machine, part I: minimization of the number of the tool switches. *Oper. Res.* **36** (1988) 767–777.

[20] J.M. Valério de Carvalho, Exact solution of bin-packing problems using column generation and branch-and-bound. *Ann. Oper. Res.* **86** (1999) 629–659.

[21] G. Wäscher, H. Haußner and H. Schumann, An improved typology of cutting and packing problems. *Eur. J. Oper. Res.* **183** (2007) 1109–1130, 12.

[22] L.A. Wolsey, Valid inequalities, covering problems and discrete dynamic programs. *Ann. Discrete Math.* **1** (1977) 527–538.

[23] H.H. Yanasse, On a pattern sequencing problem to minimize the maximum number of open stacks. *Eur. J. Oper. Res.* **100** (1997) 454–463.

[24] H.H. Yanasse and M.J.P. Lamosa, An integrated cutting stock and sequencing problem. *Eur. J. Oper. Res.* **183** (2007) 1353–1370.