Exact approaches for the pattern minimization problem

Gabriel Gazzinelli Guimaraes, Kelly Cristina Poldi*

Instituto de Matemática, Estatística e Computação Científica (IMECC), Universidade Estadual de Campinas (UNICAMP), Rua Sérgio Buarque de Holanda, 651, 13083-859, Campinas, SP, Brasil.

Abstract

Realistic applications of the Cutting Stock Problem (CSP) often include additional objectives and constraints; among the extensions of the CSP, one of the most important is the Pattern Minimization Problem (PMP). The PMP seeks a cutting plan that minimizes the number of distinct cutting patterns employed and respects a threshold on the trim loss. In this research, we propose new upper bounds on the frequency of execution of the cutting patterns and develop two approaches for the PMP. The first approach consists of an iterative framework in which, at each iteration, the PMP is solved subject to a limitation on the number of distinct cutting patterns used until an optimal solution is obtained. To solve the constrained PMP, we also propose a new Integer Linear Programming (ILP) formulation. As a second approach, we develop a pattern-set generation procedure, in which a set of cutting patterns is generated first, and then an ILP model developed in this work is used to solve the PMP. We perform computational experiments to evaluate the quality of the linear relaxation and the size of the ILP models developed in this research. We also assess the performance of the proposed approaches using two sets of instances. The results show that our methods are efficient in solving the PMP, improving previous benchmarks from the literature.

Keywords: Combinatorial optimization, Cutting Stock Problem, Pattern Minimization, Setup Minimization, Integer Linear Programming, Mathematical formulation

1. Introduction

The one-dimensional Cutting Stock Problem (1D-CSP) seeks to meet the demand for a set of items by cutting larger standard objects. The main objective of the CSP is trim loss minimization, which corresponds to minimizing the amount of wasted material. The specific way a larger object is cut is known as a cutting pattern, and the set of cutting patterns used in a solution, along with the number of times each of them is performed, is called a cutting plan, which is equivalent to a CSP solution.

To accurately address the practical needs of industries, additional objectives and constraints can be added to the CSP, as demonstrated in prior studies (Yanasse and Lamosa (2007); Arbib et al. (2012); Bang et al. (2025); Leao et al. (2017); Arbib et al. (2016); Pierini and Poldi (2021); de Lara Andrade et al. (2021); Martin et al. (2022b); Guimarães and Poldi (2023); Guimarães et al. (2025)). In this sense, minimizing distinct cutting patterns is an important secondary objective. The amount of different cutting

*Corresponding author. ORCID 0000-0002-1649-6843
Email address: ggazzinelli9@gmail.com, poldi@unicamp.br (Kelly Cristina Poldi)

patterns employed directly correlates with setup costs, since whenever a new cutting pattern is performed, it is necessary to readjust the position of the cutting tools in the cutting machine, consuming time and increasing production costs (Cui et al. (2015); Martin et al. (2022a); Silva et al. (2023)).

In the literature, this additional objective is typically incorporated into the classical CSP through two approaches. The first approach employs a lexicographic strategy: a limit is imposed on the number of objects, typically set to the minimum required in a CSP solution, and the search then focuses on minimizing the number of cutting patterns, thereby defining the Pattern Minimization Problem (PMP). The second approach introduces the minimization of distinct cutting patterns as an explicit objective, resulting in the bi-objective Cutting Stock Problem with Setup Cost (CSP-S). Both the PMP and the CSP-S are NP-hard (Alves and de Carvalho (2009); Mobasher and Ekici (2013)) and have applications in various industries, such as paper, metal, wood, and furniture.

1.1. Contributions and literature review

To the best of our knowledge, the first paper to address the Pattern Minimization Problem (PMP) is McDiarmid (1999), which examines a special case where each cut of the larger object can always produce up to two items, regardless of their types. Furthermore, no combination of three items can be obtained from a single object being cut, as the total length of any three items exceeds the length of the larger object. The authors show that, in this specific case, finding a solution with minimal trim loss is straightforward. While this represents a highly restricted case, it is significant as it exemplifies the simplest non-trivial case of the PMP. It may also occur in practical applications, particularly when iteratively refining solutions.

Similarly, in Aloisio et al. (2011a), the authors explore the case where at most two items fit within the large object, with an additional focus on scheduling the cutting patterns to minimize the number of open stacks.

The PMP can be addressed heuristically by recombining cutting patterns from an optimal or near-optimal solution to minimize their total count while maintaining raw material usage. For example, Diegel et al. (1993) introduces a technique where two cutting patterns can be merged into one, or three patterns can be combined into two. Likewise, Foerster and Wascher (2000) presents the well-known KOMBI method, which transforms a set of cutting patterns with cardinality K_1 into a reduced set with cardinality K_2, where $K_1 > K_2$.

In terms of exact methods for solving the PMP, we highlight the branch-price-and-cut algorithms developed in Vanderbeck (2000), Belov and Scheithauer (2003), and Alves and de Carvalho (2009). The pioneering exact approach to the PMP, presented in Vanderbeck (2000), significantly influenced subsequent research. In their algorithm, the columns of the master problem represent cutting patterns with fixed execution frequencies, while the pricing subproblem is formulated as a quadratic integer programming model. The authors show that the subproblem can be linearized and solved as a series of bounded knapsack problems, one for each pattern and frequency pair. This model is also employed in Belov and Scheithauer (2003) as the master problem; however, despite reducing the number of variables, the linear relaxation of this formulation proves weak, which ultimately affects the algorithm's performance.

In Alves and de Carvalho (2009), a branch-price-and-cut algorithm was proposed for the PMP; this algorithm is based on the development of new upper bounds on the execution frequency of cutting patterns, together with an arc flow formulation, also proposed in the paper, to improve the approach presented in Vanderbeck (2000). A set of 16 real-life instances is used in these three papers to evaluate the performance

of the approach. In total, the algorithms proposed in Vanderbeck (2000), Belov and Scheithauer (2003), and Alves and de Carvalho (2009) were able to find an optimal solution to 12, 8, and 13 instances, respectively.

There is significant interest in the literature regarding the identification of accurate lower bounds for the PMP, as precise bounds are essential for evaluating the effectiveness of proposed methods and guiding the search for optimal or near-optimal solutions. Studies such as Aloisio et al. (2011b) and Alves et al. (2009) specifically focus on establishing these accurate lower bounds.

It is worth noting that in Vanderbeck (2000), Belov and Scheithauer (2003), Aloisio et al. (2011b), Alves et al. (2009), and Alves and de Carvalho (2009), demand requirements are precisely met, with no overproduction. This assumption plays a critical role in the upper bounds developed in these papers. As highlighted by Martin et al. (2022a), strictly meeting demand without excess can result in feasible solutions that require a larger number of distinct cutting patterns. Nevertheless, the potential increase in unique cutting patterns may be offset by the benefits of avoiding overproduction.

As for the CSP-S, extensive research has been conducted on genetic algorithms in recent years. Applications of the evolutionary approaches cover both one- and two-dimensional cases. Moreover, some papers also consider additional objectives and constraints. Among the literature, we can cite the following papers: Muñoz et al. (2007), Golfeto et al. (2009), Brandão et al. (2011), de Araujo et al. (2014), Bonnevay et al. (2015), Mellouli et al. (2019). Aside from genetic algorithms, the CSP-S is also commonly tackled by Sequential Heuristic Procedures (SHP) in the literature. The SHP consists of sequentially generating a small number of cutting patterns and then applying these cutting patterns with high frequency until the demand for all types of items is met. Different variations of the SHP have been reported in several papers as a means to find promising solutions for the CSP-S, such as Haessler (1975), Yanasse and Limeira (2006), Cui et al. (2008), Mobasher and Ekici (2013), Cui and Liu (2011), Cui et al. (2013), and Cui et al. (2015).

As an alternative to evolutionary or SHP approaches, Umetani et al. (2003) proposed an Iterated Local Search (ILS) algorithm to solve a variant of the 1D-CSP where the number of distinct cutting patterns is fixed as a preset constant. By varying the value of this constant, the authors determine several solutions for the CSP-S and analyze the trade-off between trim loss and the number of different cutting patterns. This ILS is improved in Umetani et al. (2006), where two types of local search are implemented. The first is known as the 1-add neighborhood. It corresponds to the set of solutions resulting from increasing the amount of an item type at the expense of decreasing the number of other item types in a cutting pattern. The second is known as the shift neighborhood and corresponds to the set of solutions resulting from exchanging one item type in a cutting pattern for one or more item types in another cutting pattern.

A recent contribution to the PMP is the OPTICUT algorithm by Kayhan and Tekez (2025), which introduces an effective three-stage heuristic. The method begins with an initial phase of column generation to produce a foundational set of cutting patterns. In the second stage, it iteratively enriches this set by generating and evaluating new candidate cutting patterns. Finally, in the third stage, it selects the optimal combination of cutting patterns from the current collection via integer linear programming.

As exact approaches for the bi-objective CSP-S, we can cite the papers Martin et al. (2022a) and Hadj Salem et al. (2023). In Martin et al. (2022a), two Integer Linear Programming (ILP) formulations are implemented within an exact framework to find the Pareto frontier for the bi-objective CSP-S. In Hadj Salem et al. (2023), the authors use the ε-constraint method together with an ILP formulation proposed in the same paper to solve a real application of the two-dimensional Variable-Sized Cutting Stock Problem

(2D-VSCSP) in the home textile industry, where both the usage of raw material and the number of distinct cutting patterns are minimized simultaneously.

Another interesting contribution to the CSP-S in the literature is the work by Guimarães et al. (2025), which addresses a multi-objective variant of the CSP. The objectives are to minimize the number of different cutting patterns, minimize the maximum number of simultaneously open stacks, and minimize trim loss. The approaches proposed in this paper were effective in determining the entire Pareto front for small problem instances, as well as generating several solutions for medium-sized instances with minimal trim loss, a reduced maximum number of simultaneously open stacks, and a small number of different cutting patterns.

Building on the contributions of Vanderbeck (2000), Belov and Scheithauer (2003), Aloisio et al. (2011b), Alves et al. (2009), and Alves and de Carvalho (2009), we develop bounds on cutting pattern execution frequencies specifically designed for cases that allow for overproduction. These bounds support the development of two novel approaches for solving the Pattern Minimization Problem (PMP), where the number of objects used is minimized to the lowest feasible level in a CSP solution. Both approaches can be readily adapted to scenarios without overproduction with minimal adjustments.

The first approach is an iterative algorithm that searches for the best solution given a fixed number of cutting patterns. It begins at a lower bound and progressively increases this value until it finds a solution with minimal waste. This algorithm embeds an integer linear programming (ILP) model, which is proposed in this work. The bounds developed for this study strengthen the new ILP by introducing valid inequalities, providing tighter upper bounds on cutting pattern frequencies, and refining bounds on a subset of variables. Two preprocessing procedures further tighten these limits. As a result, the ILP includes fewer variables, has a stronger linear relaxation, and delivers better computational performance.

The second approach is a two-stage method. First, it generates the set of feasible cutting patterns. Then, an ILP model proposed in this paper selects an optimal solution. We consider three pattern-reduction criteria during the cutting pattern generation to improve performance.

The proposed approaches are theoretically exact: given unlimited computational resources, they would converge to an optimal solution of the PMP. In practice, however, computational limitations may prevent them from finding optimal solutions. Both approaches require solving the corresponding CSP to optimality to obtain a valid lower bound on the number of objects used. The second approach additionally requires the generation of the complete set of cutting patterns to certify that the solution is truly optimal.

Since both the CSP and the PMP are NP-hard problems, we impose time limits in the computational experiments. Consequently, although the methods remain exact in principle, the solutions produced within those time limits cannot be guaranteed optimal for every instance.

An essential advantage of the proposed approaches is their exact nature: unlike heuristics and metaheuristics, they can, under verifiable conditions (i.e., the complete set of cutting patterns has been generated, the optimal CSP value is known, and the time limit has not been reached), provide a formal proof of optimality for a given instance.

Computational experiments are carried out to evaluate the quality of the proposed formulations and the performance of the approaches developed in this research using two sets of instances. The first set comprises 40 instances of a fiber company in Japan that was first presented in Umetani et al. (2003). The second one is generated using the one-dimensional CSP generator proposed in Gau and Wäscher (1995). The proposed models are evaluated regarding the quality of the linear programming relaxation and their

number of variables and constraints, yielding excellent results in both aspects. Furthermore, in terms of performance, the proposed approaches consistently yield superior solutions in minimizing material usage and setup costs compared to existing benchmarks from the literature across numerous instances.

1.2. Organization of the paper

The remainder of this paper is organized as follows: Section 2 provides a detailed description of the PMP and an illustrative example. Sections 3 and 4 present the two proposed approaches for the PMP. Section 5 presents the computational experiments and their results, followed by a sensitivity analysis discussed in Section 6. Finally, Section 7 offers concluding remarks and suggests directions for future research.

In the supplementary material `https://github.com/ggazzinelli/Supplementary-Material`, we provide (i) a structured presentation of the proposed upper and lower bounds; (ii) additional implementation details for both proposed approaches and those of Martin et al. (2022a) and Cui et al. (2015); (iii) an assessment of applicability when overproduction is prohibited—summarizing computation times, solution quality (for instances where optimality was not achieved), pattern counts, and surplus under allowed overproduction; (iv) adaptations of the proposed approaches for alternative strategies that trade off the number of distinct cutting patterns against overproduction; (v) formal proofs for all propositions presented in the paper; and (vi) additional experiments analyzing model sizes and linear relaxation quality.

2. Problem description

This section provides the formal definition of the Pattern Minimization Problem (PMP) as supported by the methodology used in this paper. Consistent with established literature (Cui et al. (2015)), we model the PMP as a lexicographic optimization problem: first minimizing material waste (by minimizing the number of objects used), then minimizing the number of distinct cutting patterns.

In this context, the PMP is characterized by a set of objects of length L, a limit on the number of used objects $\underline{\beta}$, and a set of item types I, where each type of item $i \in I$ has a length ℓ_i and demand d_i. A solution for the PMP consists of a cutting plan composed of a set of cutting patterns, given by $K = \{1, \ldots, \mathcal{K}\}$ and the frequency of execution $\zeta = \{\zeta_1 \ldots, \zeta_{\mathcal{K}}\}$ of each one. Moreover, each cutting pattern $k \in K$ can be represented by a $|I|$-dimensional vector $\alpha^k = \{\alpha^k_1, \ldots, \alpha^k_{|I|}\}$, where α^k_i represents the number of item type i that are produced by executing the cutting pattern k. The objective of the PMP is to determine a cutting plan (K, ζ) such that:

(i) The number of distinct cutting patterns employed ($\mathcal{K}$) is minimized.

(ii) At most $\underline{\beta}$ objects are used in the cutting plan, i.e. $\sum_{k \in K} \zeta_k \leq \underline{\beta}$.

(iii) The sum of the length of the items obtained by a cutting pattern does not exceed the length of the large object, i.e., $\sum_{i=1}^{|I|} \ell_i \alpha^k_i \leq L$, for $k \in K$.

(iv) The demand for every item type in I is met, either exactly ($\sum_{k \in K} \alpha^k_i = d_i$) or with surplus ($\sum_{k \in K} \alpha^k_i \geq d_i$).

Criterion (iv) defines which scenario is addressed: the scenario where overproduction is allowed and the scenario where there is no overproduction of items. The scenario with overproduction refers to cases where surplus production of items is allowed, although this does not necessarily imply that overproduction will

occur—solutions may meet the demand exactly. Conversely, the scenario without overproduction refers to cases where surplus production of items is strictly prohibited.

In the lexicographic approach to the Pattern Minimization Problem, $\underline{\beta}$ represents the lower bound for the minimum number of objects required, corresponding to an optimal solution for the Cutting Stock Problem. This ensures that material waste minimization is strictly prioritized, since all feasible solutions use exactly $\underline{\beta}$ objects. We employ the Gilmore-Gomory column generation method to compute this bound efficiently, given that solving the integer CSP to optimality is computationally prohibitive for larger instances.

The value of $\underline{\beta}$ is obtained by solving the continuous relaxation of the CSP via column generation and rounding up to the nearest integer. This approach yields a tight lower bound $\underline{\beta} = \lceil z_{LP} \rceil$. Due to the well-known Integer Round-Up Property (IRUP), which holds for the vast majority of practical instances (Scheithauer and Terno (1995)), this $\underline{\beta}$ typically equals the optimal integer solution value.

While the Modified Integer Round-Up Property (MIRUP) guarantees that in rare cases an optimal integer solution might require $\lceil z_{LP} \rceil + 1$ objects, the computational experiments performed in this paper across extensive test instances align with literature findings that such cases are exceptionally uncommon. In the event such a rare instance occurs, the proposed framework can easily accommodate it by incrementing $\underline{\beta}$. The computational efficiency gained by using column generation rather than solving the integer CSP to optimality justifies this approach, especially given the rarity of non-IRUP instances.

As illustrative examples, we present two cutting plans corresponding to optimal PMP and CSP solutions, respectively. These cutting plans are related to the fiber10_5180 problem instance from the dataset provided in Umetani et al. (2003). We consider that items of the same length are aggregated and that overproduction of items is permitted.

In the fiber10_5180 problem instance, the length of the objects to be cut (L) is equal to 5180, the set of item types I is composed of 6 item types of lengths ℓ_i equal to $\{750, 915, 920, 985, 1000, 1250\}$ and demand d_i equal to $\{10, 30, 5, 21, 264, 19\}$. Moreover, the limit on the number of used objects $\underline{\beta}$ equals 69.

Optimal solutions for the PMP and CSP are depicted in Figures 1 and 2, respectively. The colored rectangles identify the item types, the hatched areas represent the trim loss, and the frequency of each cutting pattern is indicated in parentheses on the right-hand side.

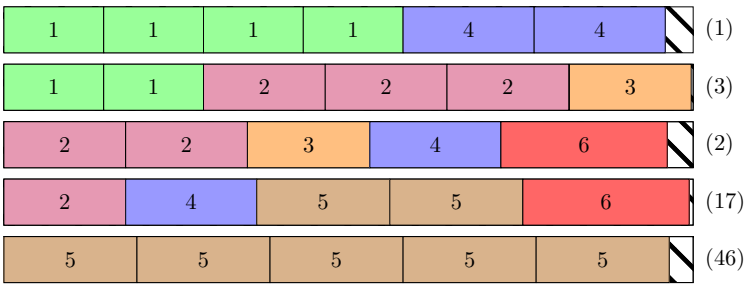


Figure 1: Optimal PMP solution for the fiber10_5180 instance ($L = 5180$, $\underline{\beta} = 69$). Colors indicate item types, hatched areas show trim loss, and cutting patterns frequencies are given in parentheses.

It is important to note that, although both solutions consume the same amount of material, the number of cutting patterns is considerably reduced, from 8 to 5, when considering the objective of minimizing the number of cutting patterns.

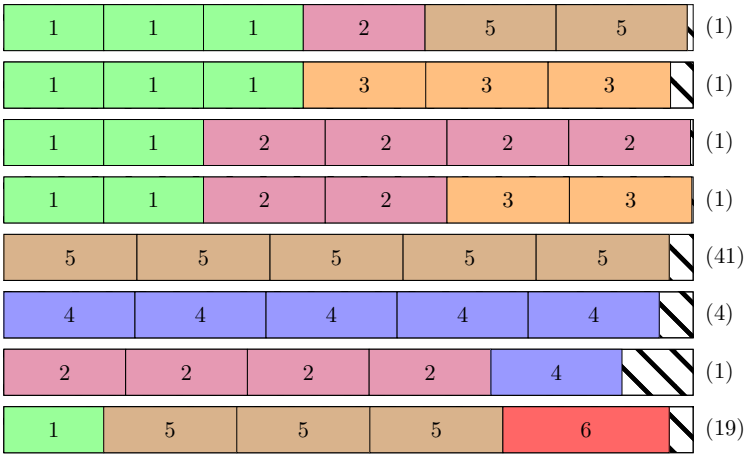


Figure 2: Optimal CSP solution for the fiber10_5180 instance ($L = 5180$, $\underline{\beta} = 69$). Colors indicate item types, hatched areas show trim loss, and cutting patterns frequencies are given in parentheses.

3. First approach to the PMP

This section details the first proposed approach for solving the Pattern Minimization Problem, referred to as Approach 1. In summary, Approach 1 is an exact approach where, at each iteration, we solve the Pattern Minimization Problem (PMP) subject to using at most $\mathcal{K}$ distinct cutting patterns. This iterative process starts with a minimal $\mathcal{K}$ value and increments it until feasibility is achieved. The algorithm relies on an Integer Linear Programming (ILP) formulation to solve each PMP($\mathcal{K}$) instance, and for this purpose, we propose a new formulation.

The remainder of this section is organized as follows. We begin by reviewing the formulation from Martin et al. (2022a), which serves as the foundation for the proposed approach. In Sections 3.2 and 3.3, we present improved upper bounds for the cutting pattern frequencies and the ILP model variables. These bounds are critical for enhancing model performance, as tighter bounds allow for the elimination of redundant variables and constraints. Building on this, Section 3.4 introduces two preprocessing procedures to strengthen these bounds further. Subsequently, Section 3.5 presents new valid inequalities to improve the model's linear relaxation. Finally, Section 3.6 outlines the complete solution framework that integrates the proposed formulation to solve the PMP.

3.1. Review of the Martin et al. (2022a) model

Anchored on the master problem developed in Vanderbeck (2000), the authors proposed an ILP for the CSP-S in which a binary variable is considered for each cutting pattern and possible frequencies of execution. The model is incorporated into a framework to determine the Pareto frontier of the bi-objective CSP-S.

Let L, ℓ_i, d_i, and K be as previously defined and F_k be the set of possible frequencies of cutting patterns, for $k \in K$; and y_{kf} be a binary variable that is equal to 1 if the cutting pattern k has a frequency equal to f, otherwise, 0. Also, let x_{kfi} be an integer variable representing the number of item types i obtained when cutting an object using the cutting pattern k with frequency f. The model proposed in Martin et al. (2022a) is given by:

$$\min \sum_{k \in K} \sum_{f \in F_k} f y_{kf} \tag{1}$$

$$\min \sum_{k \in K} \sum_{f \in F_k \setminus \{0\}} y_{kf} \tag{2}$$

$$\sum_{i=1}^{|I|} \ell_i x_{kfi} \leq L y_{kf}, \qquad k \in K, f \in F_k \setminus \{0\}, \tag{3}$$

$$\sum_{k \in K} \sum_{f \in F_k \setminus \{0\}} f x_{kfi} \geq d_i, \qquad i \in I, \tag{4}$$

$$\sum_{f \in F_k} y_{kf} = 1, \qquad k \in K, \tag{5}$$

$$\sum_{f \in F_k} f y_{kf} \geq \sum_{f \in F_{k+1}} f y_{(k+1)f}, \qquad k \in K \setminus \{\mathcal{K}\}, \tag{6}$$

$$y_{kf} \in \{0, 1\}, \qquad k \in K, f \in F_k, \tag{7}$$

$$x_{kfi} \in Z^+, \qquad k \in K, f \in F_k \setminus \{0\}, i \in I. \tag{8}$$

Objective functions (1) and (2) correspond to minimizing the trim loss and the number of distinct cutting patterns used in the solution, respectively. In the original paper Martin et al. (2022a), there is a typo in the definition of the objective function (2), as it does not exclude cutting patterns performed with a frequency of 0 when counting the number of distinct cutting patterns. Constraints (3) are related to (iii); in this sense, the constraints ensure the feasibility of one-dimensional cutting patterns. Moreover, constraints (3) imply that item types can only be obtained from cutting patterns executed with a frequency greater than zero. The set of constraints (4) is related to (iv) and ensures that the demand for all item types is met. The set of constraints in Equation (5) ensures that the frequency of execution for each cutting pattern is unique and greater than or equal to 0. Constraints (6) are a set of valid inequalities that ensure that the cutting patterns are ordered in non-increasing order according to their frequencies. The other restrictions are related to the domain of the variables.

3.2. Improved upper and lower bounds on the frequency of cutting patterns

The size of model (1)-(8) directly correlates to the bounds on the frequency of cutting pattern execution. In the PMP addressed in this research, we seek only solutions in which the number of used objects is bounded by $\underline{\beta}$, as a consequence, considering d_{max} as the maximum value of the demand and $\underline{\mathcal{K}}$ as a lower bound on the number of distinct cutting patterns, then, $\Phi^k = \min\{\lfloor(\underline{\beta} - \max\{0, \underline{\mathcal{K}} - k\})/k\rfloor, d_{\max}\}$ is a valid upper bound for the frequency at which the k-th cutting pattern should be performed and, consequently, F_k is equal to $\{0, 1, \ldots, \Phi^k\}$. This set of upper bounds was proposed in Martin et al. (2022a) in the context of the bi-objective CSP-S.

In what follows, we present five propositions that establish new bounds on the frequencies of the cutting patterns. We begin by introducing preliminary notation and definitions necessary to clearly explain these propositions, followed by the propositions themselves.

Thus, let s_i denote the indices referring to the demand vector in non-increasing order, meaning if $i_1 \geq i_2$, then $d_{s_{i_1}} \geq d_{s_{i_2}}$. Additionally, let $\overline{d}$ represent a vector of demand values in non-increasing order, i.e., $\overline{d} = [d_{s_1}, d_{s_2}, \ldots, d_{s_{|I|}}]$.

Moreover, let I^k be the set of item types generated by the execution of cutting pattern k, f_k^* as the frequency of the cutting pattern k, for $k \in K$ and $\overline{d}^k = [d^k_{s_1}, d^k_{s_2}, \ldots, d^k_{s_{|I|}}]$ denote the remaining demand immediately before executing the k-th cutting pattern. Note that the frequency of execution of the cutting pattern k is bounded by $\max_{i \in I^k}\{d_i^k\}$.

An unexplored possibility in Martin et al. (2022a) is the imposition of lower bounds on the frequency of the cutting patterns. Setting such limits enables further reduction in the model's size. In this context, we present a proposition that establishes lower bounds on the frequency of the cutting patterns; this proposition can be incorporated into the model proposed in Martin et al. (2022a) with minimal adjustments.

Proposition 1: Consider K, $\underline{\beta}$, and Φ^k previously defined. Moreover, for an arbitrary $\overline{k}$, for $\overline{k} \in K$, let $\overline{\Phi}^{\overline{k}}$ represent the cumulative sum of the maximum frequencies of the first k cutting patterns to be executed. Specifically, $\overline{\Phi}^{\overline{k}}$ is given by:

$$\overline{\Phi}^{\overline{k}} = \sum_{k \in K, k \leq \overline{k}} \Phi^k.$$

Assume that the cutting patterns are sorted in non-increasing order of their frequencies. Let Θ^k represent a lower bound on the frequency for an arbitrary cutting pattern $\overline{k}$, where $\overline{k} \in K$. The following inequality must hold:

$$\Theta^{\overline{k}} \geq \left\lceil \frac{\underline{\beta} - \overline{\Phi}^{\overline{k}-1}}{\mathcal{K} - \overline{k} + 1} \right\rceil.$$

When incorporating the lower bounds into the model proposed in Martin et al. (2022a), it is necessary to modify constraint (5). In the original model from Martin et al. (2022a), each cutting pattern can be executed with any frequency, including zero. Therefore, if we maintain inequality (5) as it is while incorporating the lower bounds, the model would use exactly K cutting patterns with a non-zero frequency. However, the objective is to allow, at most, K cutting patterns to have a frequency greater than zero.

Thus, we substitute the constraints by $\sum_{f \in F_k} y_{kf} \leq 1$, $k \in K$. This new set of constraints, paired with Proposition 1, ensures that, for all $k \in K$, if a cutting pattern is used ($\sum_{f \in F_k} y_{kf} = 1$), its frequency of execution is unique and equal to or greater than $\Theta^k = \left\lceil \frac{\underline{\beta} - \overline{\Phi}^{k-1}}{\mathcal{K} - k + 1} \right\rceil$.

Another interesting possibility is to take advantage of the non-increasing order of the cutting pattern frequencies to develop upper bounds based on their execution order. In this context, Propositions 2 to 5, presented below, establish such bounds. Proposition 2 provides an effective upper bound for the frequency of the second cutting pattern. Propositions 3 and 4 establish bounds for the third cutting pattern under different relationships between the highest, the second-highest, and the third-highest demand values. Proposition 5 addresses scenarios where the highest demand value significantly exceeds the second-highest demand value, which can be particularly useful in specific cases such as the example illustrated in Figure 2, although

it may not be applicable to all instances.

Proposition 2. The frequency of the second cutting pattern f_2^* is bounded by $\max\{d_{s_2}, \lfloor d_{s_1}/2 \rfloor\}$.

Proposition 3. If $2d_{s_2} \leq d_{s_1}$ and $3d_{s_2} > d_{s_1}$, then the frequency of the third cutting pattern f_3^* is bounded by d_{s_2}.

Proposition 4. If $2d_{s_2} > d_{s_1}$, then the frequency of the third cutting pattern f_3^* is bounded by $\max\{d_{s_3}, \lfloor d_{s_1}/2 \rfloor\}$.

Proposition 5. Let c be an integer greater than 1. If $d_{s_1} > cd_{s_2}$, then the frequency of any cutting pattern $k' \leq c$ is bounded by $\lfloor d_{s_1}/k' \rfloor$.

The supplementary material provides a detailed explanation of how the upper and lower bounds introduced in this section are used, along with the proofs of all propositions presented in the paper.

3.3. Improved upper bounds on the variables of the ILP model

Providing tight bounds for the variables of an ILP model is essential to reduce the search space for the problem solution and improve the formulation's linear relaxation. In Martin et al. (2022a), the authors show that the value of the variables x_{kfi} must be equal to or lower than $\min(\lfloor L/\ell_i \rfloor, d_i)$. In this section, we propose a new upper bound for the variables x_{kfi} related to the classical objective of the CSP, the minimization of trim loss.

In the PMP addressed in this research, the number of used objects is bounded by $\underline{\beta}$; as a consequence, the trim loss must be equal to or lower than $L\underline{\beta} - \sum_{i \in I} \ell_i d_i = T_{\underline{\beta}}$. Considering the variables and parameters of model (1)-(8), the trim loss of an arbitrary solution can be calculated in the following manner. Let $\overline{\ell}_k$ be the waste of the cutting pattern k, i.e., $\overline{\ell}_k = L - \sum_{f \in F_k \setminus \{0\}} \sum_{i \in I} \ell_i x_{kfi}$, and d'_i be the quantity of the item type i that is produced in the cutting plan. Then, the trim loss of a PMP solution is equal to $\sum_{k=1}^{K} f_k^* \overline{\ell}_k + \sum_{i \in I} \ell_i (d'_i - d_i)$. Therefore, a solution for the PMP is only feasible if the following inequality holds:

$$\sum_{k=1}^{K} f_k^* \overline{\ell}_k + \sum_{i \in I} \ell_i (d'_i - d_i) \leq L\underline{\beta} - \sum_{i \in I} \ell_i d_i = T_{\underline{\beta}}. \tag{9}$$

Inequality (9) is related to the upper bound proposed in Alves and de Carvalho (2009). The main difference lies in the fact that, as surplus is not permitted in Alves and de Carvalho (2009), the authors do not account for the trim loss resulting from overproduction. Inequality (9) is the basis of many of the propositions demonstrated in this work.

Proposition 6: In a PMP in which the number of used objects is bounded by $\underline{\beta}$, the value of the variables x_{kfi} must be equal to or lower than $\left\lfloor \dfrac{T_{\underline{\beta}} + l_i d_i}{l_i f} \right\rfloor$.

Proposition 6 provides a tight upper bound on the variables x_{kfi}. In the case in which the upper bound provided by Proposition 6 is equal to 0, we may remove the variable from model (1)-(8); consequently, we have to redefine the domain of the variables x_{kfi}. For an arbitrary item type i, let $\overline{f}_i$ be the smallest value such that $\left\lfloor \dfrac{T_{\underline{\beta}} + \ell_i d_i}{\ell_i \overline{f}_i} \right\rfloor > 0$, Φ_i^k be equal to $\min\{\Phi^k, \overline{f}_i\}$ and $F_i^k = \{\Theta^k, \ldots, \Phi_i^k\}$. Thus, regarding the lower bound on the frequency of execution of the cutting patterns presented in Section 3.2 and Proposition 6, the variables x_{kfi} are defined for $k \in K, i \in I, f \in F_i^k$.

3.4. Preprocessing strategies

According to Atamturk and Savelsbergh (2005), preprocessing is typically used to modify MIP formulations by eliminating or substituting variables and constraints in a way that the resulting formulation has a smaller feasible space and at least one optimal solution remains feasible in the altered search space.

In this section, we develop two preprocessing algorithms; the first one is anchored in inequality (9) and aims to obtain an improved upper bound on the frequency of the cutting patterns. The second one relies on the standard column generation method for the CSP to refine the upper bounds presented in Section 3.3. Both algorithms are given in Sections 3.4.1 and 3.4.2.

3.4.1. Preprocessing algorithm for the upper bound on the frequency of the cutting patterns

The first proposed preprocessing strategy builds upon the probing procedure introduced in Aloisio et al. (2011b). It involves an iterative process to evaluate whether a cutting pattern can be executed with its upper bound frequency. The key distinction of the proposed approach lies in its application to a model without prior generation of the cutting patterns. This allows for assessing whether any arbitrary cutting pattern can be executed at the upper bound frequency, rather than evaluating pre-generated patterns individually.

The preprocessing algorithm requires the following inputs: an initial upper bound on the frequency of the first cutting pattern, denoted by Φ^1, as well as the parameters L, I, $T_{\underline{\beta}}$, and the item-specific values ℓ_i, d_i for $i \in I$, as previously defined.

We introduce an ILP formulation to determine the minimum trim loss resulting from executing an arbitrary cutting pattern with a frequency of Φ^1. At each iteration, the algorithm solves the ILP to assess whether a cutting pattern can be executed with such a frequency without exceeding the trim loss limit $T_{\underline{\beta}}$. If such a cutting pattern is found, the algorithm terminates. Otherwise, the upper bound for Φ^1 is reduced by one (i.e., $\Phi^1 = \Phi^1 - 1$), and the process is repeated.

The ILP formulation utilized in this strategy introduces three integer decision variables: x_i, v_i, and s. Here, x_i represents the quantity of item type i produced by the cutting pattern; v_i accounts for the excess production of item type i beyond its demand; and s denotes the trim loss associated with the cutting pattern. The specific ILP model used for the preprocessing strategy is given by:

$$\min \sum_{i \in I} \ell_i v_i + \Phi^1 s \tag{10}$$

$$L - \sum_{i=1}^{|I|} \ell_i x_i = s, \tag{11}$$

$$v_i \geq \Phi^1 x_i - d_i, \qquad i \in I, \tag{12}$$

$$v_i \in Z^+, \qquad i \in I, \tag{13}$$

$$x_i \in Z^+, \qquad i \in I, \tag{14}$$

$$s \in Z^+. \tag{15}$$

The objective function (10) corresponds to minimizing the trim loss related to executing a feasible cutting pattern with frequency equal to Φ^1, the constraint (11) ensures that s is equal to the trim loss of the cutting pattern and that such a cutting pattern is feasible. The set of constraints (12) ensures that v_i is greater

than or equal to the surplus of the item type i. Since the objective function aims to minimize the trim loss, v_i will always assume the lowest possible value, thus equaling the surplus. The remaining constraints are related to the domain of the variables.

Let T_L be the value of the objective function in an optimal solution for model (10)-(15). It is evident that T_L corresponds to the minimum trim loss related to a feasible cutting pattern with a frequency equal to Φ^1. Thus, Algorithm 1 provides an upper bound on the frequency of the cutting patterns.

Algorithm 1 A preprocessing algorithm to determine upper bounds on the frequency of the cutting patterns

1: Set $\Phi^1 = \min\{\underline{\beta} - (\mathcal{K} - 1), d_{max}\}$
2: **repeat**
3: Solve the model (10)-(15) and determine T_L
4: **if** $T_L > T_{\underline{\beta}}$ **then**
5: $\Phi^1 = \Phi^1 - 1$
6: **end if**
7: **until** $T_L \leq T_{\underline{\beta}}$

Since $\Phi^{k+1} \geq \Phi^k$, for all $k \in K$, the algorithm may also improve the bounds on the frequency of the other cutting patterns. As stated in Andersen and Andersen (1995), for any presolve procedure, there is a trade-off between the amount of redundancy found, and the time spent in the procedure. An advantage of Algorithm 1 is that the time consumed in the algorithm is proportional to the reduction in the upper bound Φ^1. In fact, the number of times the model (10)-(15) is solved is equal to the reduction in Φ^1 plus 1. The model (10)-(15) is very simple and hardly requires any time to be solved for most instances. Unfortunately, Algorithm 1 is only effective for instances with small trim loss; for instances with big and medium values of trim loss, the algorithm fails to improve the upper bound found via the methods explored in previous sections.

3.4.2. Preprocessing algorithm for the upper bounds on the variables of the ILP model

Due to inequality (9), in an arbitrary cutting plan in which the number of item type i produced is equal to d'_i, the production excess of item type i, given by $(d'_i - d_i)$, must be equal to or lower than $\left\lfloor \frac{T_{\underline{\beta}}}{\ell_i} \right\rfloor$. As a consequence, an upper bound on the maximum number of units of item type i that can be produced in a feasible cutting plan for the PMP is equal to $d^*_i = d_i + \left\lfloor \frac{T_{\underline{\beta}}}{\ell_i} \right\rfloor$.

The second proposed preprocessing strategy has the purpose of refining the upper bounds d^*_i, for $i \in I$. To that end, we devise an algorithm that relies on the standard column generation method for the CSP. As follows, we present a proposition in which the proposed algorithm is anchored.

Proposition 7: Consider an arbitrary cutting plan for the PMP where the number of item type i produced is given by the vector $\boldsymbol{d'} = [d'_1, d'_2, \ldots, d'_{|I|}]$. This plan is feasible only if the objective function value of the linear relaxation of the CSP, with the required demand set to $\boldsymbol{d'}$, is less than or equal to $\underline{\beta}$.

The second preprocessing algorithm, given by Algorithm 2, consists of using Proposition 7 repeatedly in order to check if there is a feasible solution in which the number of units of item type i produced is equal to d^*_i. If not, we do $d^*_i = d^*_i - 1$ and repeat the process. The algorithm ends when every item type has been addressed.

More specifically, the algorithm begins by initializing the upper bounds on the demand values for each

item type, given by $d_i^* = d_i + \left\lfloor \frac{T_{\underline{\beta}}}{\ell_i} \right\rfloor$, $\forall i \in I$. For each item type $i \in I$, the algorithm iteratively refines these bounds as follows. First, we create an auxiliary demand vector $\boldsymbol{d'} = (d'_1, d'_2, \ldots, d'_{|I|})$, where $d'_i = d_i^*$, and for all $j \in I$, $j \neq i$, we set $d'_j = d_j$, with d_j being the original demand for item type j as defined in Section 2. Using this updated demand vector $\boldsymbol{d'}$, we solve the linear relaxation of the CSP to compute $\underline{\beta}^{\boldsymbol{d'}}$. If $\underline{\beta}^{\boldsymbol{d'}} > \underline{\beta}$, where $\underline{\beta}$, we reduce d_i^* by one unit and repeat the process for the same item type i. This iterative procedure continues until $\underline{\beta}^{\boldsymbol{d'}} \leq \underline{\beta}$. By applying this process to all item types in I, the algorithm refines the upper bounds on the demand values to ensure they are as tight as possible while preserving the feasibility of the PMP. As follows, we present the second proposed preprocessing algorithm:

Algorithm 2 A preprocessing algorithm to determine upper bounds on the variables x_{kfi}

Set $d_i^* = d_i + \left\lfloor \frac{T_{\underline{\beta}}}{\ell_i} \right\rfloor$, $\forall i \in I$
for $i \in I$ **do**
  repeat
    Set $d'_i = d_i^*$ and $d'_j = d_j$ for all $j \in I$, $j \neq i$
    Solve the CSP with demand equal to d' and find $\underline{\beta}^{d'}$
    if $\underline{\beta}^{d'} > \underline{\beta}$ **then**
      $d_i^* = d_i^* - 1$
    end if
  until $\underline{\beta}^{d'} \leq \underline{\beta}$
end for
return d_i^*, $\forall i \in I$

Since in a feasible solution for the PMP, the value of the variables x_{kfi} must be equal to or lower than $\left\lfloor \frac{d_i^*}{f} \right\rfloor$, then, the bounds d_i^*, $i \in I$, obtained via Algorithm 2, can be used to improve the upper bounds on the variables x_{kfi}. Once again, if $\left\lfloor \frac{d_i^*}{f} \right\rfloor = 0$, for some (k, f, i), then the variable can be removed from the model.

3.5. Valid inequalities

In order to reduce the search space size and to improve the linear programming relaxation of formulation (1)-(8), we propose three valid inequalities for the model.

According to inequality (9), in an optimal solution, we must have $\sum_{k=1}^{K} f_k \overline{\ell}_k \leq T_{\underline{\beta}}$, thus, the trim loss of an arbitrary k' cutting pattern must be less than or equal to $\frac{T_{\underline{\beta}}}{f_{k'}}$. Thus, the following set of constraints can be added to the model:

$$\sum_{i=1}^{|I|} \ell_i x_{kfi} \geq (L - \frac{T_{\underline{\beta}}}{f_{k'}}) y_{kf}, k \in K, f \in F_k \backslash \{0\}. \tag{16}$$

The second set of valid inequalities proposed in this section is based on the lower bounds Θ^k presented in Section 3.2. Since the number of objects used after the execution of the first $k-1$ cutting patterns is

given by $\sum_{k'=1}^{k}\sum_{f\in F_k} y_{kf}$ then the cutting pattern k must be executed with a frequency equal to or greater than $\dfrac{\underline{\beta}-\sum_{k'=1}^{k}\sum_{f\in F_k} y_{kf}}{\mathcal{K}+1-k}$. Otherwise, the solution would consume less than $\underline{\beta}$ objects, which is impossible, as $\underline{\beta}$ is a lower bound. Therefore, we can add the following set of valid inequalities to model (1)-(8):

$$\sum_{f\in F_k} y_{kf} \geq \frac{\underline{\beta}-\sum_{k'=1}^{k-1}\sum_{f\in F_k} y_{kf}}{\mathcal{K}+1-k}, k=2,\ldots,\mathcal{K}. \tag{17}$$

Lastly, we may strengthen the constraints (4). Observe that if, for some $i \in I$, we have $\ell_i > T_{\underline{\beta}}$, then we cannot exceed the demand for the item type. Therefore, considering I' as the set of item types such that $\ell_i > T_{\underline{\beta}}$, we can divide the set of constraints (4) into two sets:

$$\sum_{k\in K}\sum_{f\in F_k\setminus\{0\}} j x_{kfi} = d_i, \qquad i \in I', \tag{18}$$

$$\sum_{k\in K}\sum_{f\in F_k\setminus\{0\}} j x_{kfi} \geq d_i, \qquad i \in I\setminus I'. \tag{19}$$

A new ILP formulation can be obtained by adding valid inequalities (16)-(19) to model (1)-(8), considering the modified constraint (5) and the upper and lower bounds on the variables proposed in the previous sections. In this research, we refer to this model as Formulation 1.

3.6. Exact algorithm for the PMP with minimum trim loss

As previously defined, let PMP($\mathcal{K}$) be the PMP subject to the constraint that at most $\mathcal{K}$ distinct cutting patterns are used. At the beginning of the proposed algorithm, we determine a lower bound on the number of distinct patterns used in the PMP solution. We solve the associated Bin Packing Problem (BPP) to do so. Then, we determine $\underline{\beta}$ via the solution of the linear relaxation of the associated CSP.

Equipped with those two bounds, the next steps of the proposed algorithm are to determine Φ^k and Θ^k, for $k = 1,\ldots,\mathcal{K}$ and then to solve the PMP($\mathcal{K}$) through Formulation 1 or model (1)-(8).

At each iteration of Algorithm 3, the value of $\mathcal{K}$ is incremented by one unit, and the final two steps are repeated until a feasible solution is obtained. Since this solution has the smallest number of cutting patterns for which the PMP($\mathcal{K}$) is feasible, it corresponds to an optimal solution for the PMP. The algorithm is summarized in the flowchart presented in Figure 3. All algorithm steps, including the termination criteria and final convergence, are represented by blue rectangles numbered with yellow squares in the upper-left corner. The workflow demonstrates the iterative process for solving the Pattern Minimization Problem with overproduction.

(Algorithm 3)

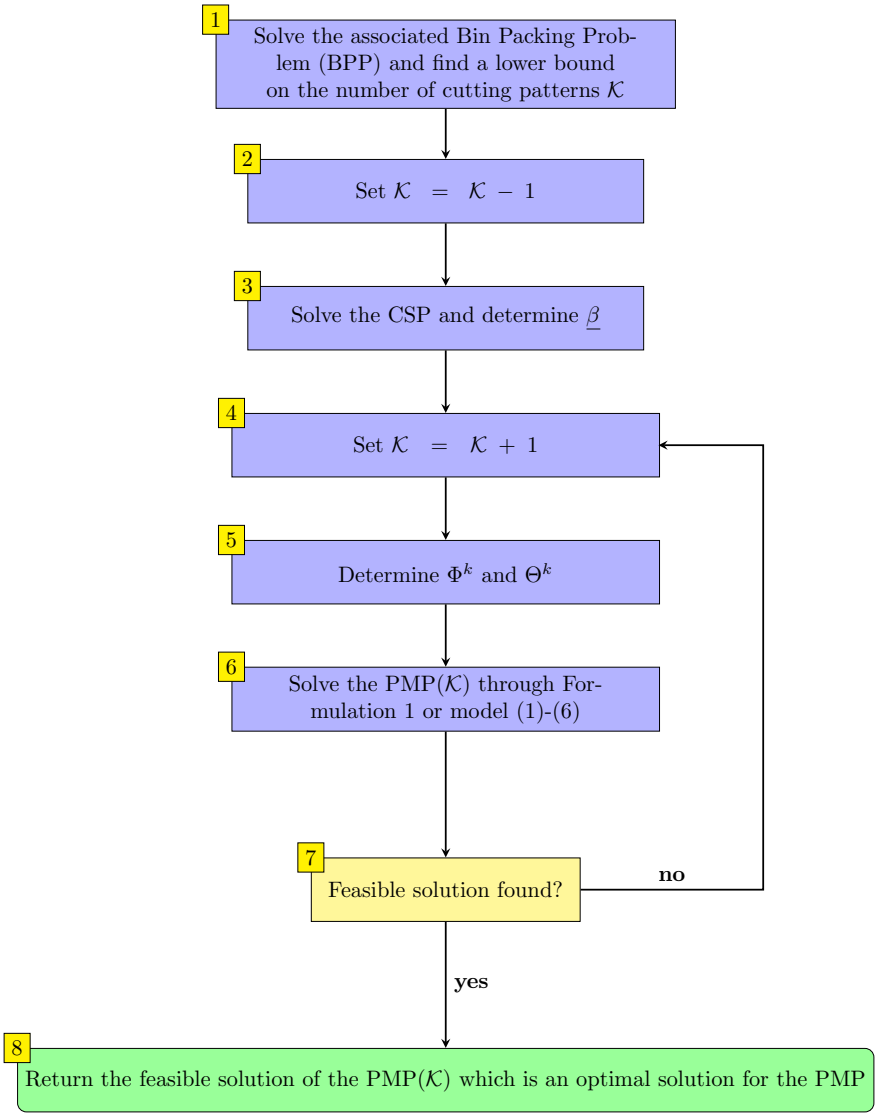


Figure 3: Flowchart of Algorithm 3: An exact algorithm for the Pattern Minimization Problem (PMP) with minimum trim loss. The iterative procedure incrementally increases the number of cutting patterns ($\mathcal{K}$) until feasibility is achieved. For clarity, all the steps are numbered and color-coded by function, i.e., blue for process steps, yellow for termination criteria, and green for final convergence.

3.7. Summary of contributions and comparison with Martin et al. (2022a)

In summary, the main contribution of Section 3 is the development of **Approach 1**, described in Algorithm 3. This approach integrates a lexicographic optimization scheme with the newly proposed ILP model (Formulation 1) to solve the PMP. Table 1 summarizes the main theoretical differences between Approach 1 and the method proposed in Martin et al. (2022a), emphasizing the distinct optimization strategies and modeling components adopted in each framework.

Table 1: Compact theoretical comparison between the proposed Approach 1 and Martin et al. (2022a).

Aspect	**Martin et al. (2022a)**	**Proposed Approach 1**	**Remark**
Optimization scheme	Bi-objective optimization	Lexicographic optimization (material minimization prioritized)	Different strategies; both capture the same trade-off under distinct hierarchies.
ILP formulation	Original model from Martin et al. (2022a)	New ILP model (Formulation 1)	Introduces tighter bounds and additional valid inequalities.
Cutting pattern frequency bounds	Upper bounds only	**Tighter** upper and **explicit** lower bounds	Reduces feasible space and improves model compactness.
Valid inequalities	One ordering constraint	Four families: — Ordering (as in Martin et al. (2022a)) — Trim-loss bounds — Cumulative bounds — Demand tightening	Stronger LP relaxation and improved integrality gap.
Bound strength	Original bounds	**Strengthened** via preprocessing	Enhances branch-and-bound efficiency.
Model size	Original variable set	**Reduced** variable set	Lower memory and computation time.

4. Second approach to the PMP

This section introduces the second proposed approach to the Pattern Minimization Problem, referred to as Approach 2. It is a two-stage method that first generates a set of feasible cutting patterns and then applies an ILP model to determine an optimal solution over this predefined set. To improve computational efficiency, we introduce three pattern-reduction criteria and derive upper bounds for the ILP variables. Since a full enumeration of feasible cutting patterns becomes computationally intractable for larger instances, we limit the generation phase to at most 1,000,000 cutting patterns. This limit ensures that the cutting pattern generation remains computationally trivial, taking less than two seconds even for the most demanding instances. Therefore, instances for which the complete set of cutting patterns could not be generated were excluded from the computational analysis. This ensures that all presented results and analyses are based on instances with guaranteed optimality.

The remainder of this section is organized as follows. Section 4.1 describes the cutting pattern generation algorithm and the proposed reduction criteria, while Section 4.2 presents the ILP formulation developed for the PMP.

4.1. Stage 1: Cutting pattern generation

In a one-dimensional CSP, the only restriction placed on the cutting patterns is that the sum of the lengths of items generated by running a cutting pattern does not exceed the length L of the object to be cut. Consider α_i as the item type i quantity obtained when an object is cut according to the cutting pattern,

and P as the set of feasible cutting patterns. The set P corresponds to the feasible solutions of a knapsack problem, presented as follows:

$$\sum_{i \in I} \ell_i \alpha_i \leq L, \tag{20}$$

$$\alpha_i \in \mathbb{Z}^+, \qquad i \in I. \tag{21}$$

A complete enumeration of the set of cutting patterns can be costly, as the number of cutting patterns grows rapidly as a function of the number of different item types and the ratio $\frac{L}{\min\{\ell_i\}}$, for $i \in I$ (Filho et al. (2018)). Furthermore, the number of variables and constraints in the proposed formulation is related to the cardinality of P. As mentioned in Section (3.5), we can disregard cutting patterns such that $\sum_{i=1}^{|I|} \ell_i \alpha_i < L - T_{\underline{\beta}}$.

Moreover, two criteria for cutting pattern elimination can be applied to a PMP, in a scenario where overproduction of items is allowed. These criteria were selected for two main reasons. First, they are the only ones we are aware of that are directly applicable in this context. Second, both contribute significantly to reducing the number of cutting patterns, which improves the approach's overall efficiency by reducing the size of the proposed formulation.

(v) All generated cutting patterns must produce at most d_i units of each item type i, for $i \in I$.

(vi) All generated cutting patterns are such that the sum of the length of the items is greater than or equal to $L - \eta$, for $\eta = \min_{i \in I} \ell_i$.

The idea behind criterion (v) is that any cutting pattern $\alpha^k = \alpha^k_1, \ldots, \alpha^k_{|I|}$, in which $\alpha^k_{i'} > d_{i'}$ for arbitrary $i' \in I$, can be replaced by a cutting pattern that adheres to the first criterion. To do so, simply set $\alpha^k_{i'} = d_{i'}$. Such a reduction criterion is used by Delorme and Iori (2020).

The criterion (vi) is based on the fact that any feasible cutting pattern $\alpha^k = \alpha^k_1, \ldots, \alpha^k_{|I|}$, such that $\sum_{i=1}^{|I|} \ell_i \alpha_i < L - \eta$, where $\eta = \min_{i \in I} \ell_i$, can be replaced by another feasible cutting pattern. To do so, increase the quantity of generated item units until the criterion is satisfied. This reduction criterion is used, for instance, in the definition of a feasible cutting pattern in Filho et al. (2018) and is only valid for problems in which overproduction is allowed.

We observe that, although the two cutting pattern reduction criteria can be applied independently during the cutting pattern generation stage, considering both criteria simultaneously may result in a set of cutting patterns that do not guarantee optimality.

To illustrate, consider a PMP with two item types, and the lengths are given by $\ell_1 = 1$ and $\ell_2 = 3$, and the demands are $d_1 = 1$ and $d_2 = 1$. The standard object has a length of $L = 6$. Clearly, the problem is feasible and requires only a single cutting pattern to meet the demand. However, no cutting pattern satisfies both reduction criteria simultaneously.

For instance, any cutting pattern that obeys criterion (v) must produce at most one unit of each item type, which consequently violates criterion (vi) since $6 - (\alpha^k_1 \cdot 1 + \alpha^k_2 \cdot 3) \geq 2 > \ell_1$. On the other hand, any cutting pattern that obeys criterion (vi) must satisfy $6 - (\alpha^k_1 \cdot 1 + \alpha^k_2 \cdot 3) \leq \ell_1$, which implies that either α^k_1 or α^k_2 must be greater than 1, violating criterion (v). Thus, enforcing both criteria leads to a situation

where no feasible cutting pattern exists, demonstrating the incompatibility of these criteria when applied jointly.

To simultaneously implement criteria (v) and (vi), we modify criterion (v) to permit cutting patterns that produce more than d_{s_1} units of item type s_1, where s_1 is defined in Section 3.2. This modification preserves the problem's optimality because any cutting pattern $\alpha^k = \alpha_1^k, \ldots, \alpha_{|I|}^k$ with $\alpha_i^k > d_i$, for $i \in I \backslash \{s_1\}$ can be replaced by a cutting pattern satisfying $\alpha_i^k \leq d_i$, for $i \in I \backslash \{s_1\}$ as guaranteed by criterion (v).

Furthermore, any cutting pattern that violates criterion (vi)—where the total length of items is less than $L - \eta$—can be adjusted by increasing $\alpha_{s_1}^k$ until the total length meets or exceeds $L - \eta$. This adjustment ensures the feasibility of the cutting pattern while maintaining adherence to the problem constraints.

Additionally, determining tight upper bounds on the frequency of each cutting pattern in P is crucial to the proposed model. The inequality (9) implies that the frequency of the cutting pattern $p_j \in P$ must be equal to or lower than f^*, where f^* is the maximum integer number such as the inequality $f^*(L - \sum_{i=1}^{|I|} \ell_i \alpha_i) + \sum_{i \in I} \ell_i (\max\{0, f^* \alpha_i - d_i\}) \leq T_{\underline{\beta}}$ is verified. Moreover, considering $I^j \subseteq I$ as previously defined, i.e., the set of item types generated by the cutting pattern j, then, $\overline{f} = \left\lceil \max_{i \in I^j} \left\{ \frac{d_i}{\alpha_i^j} \right\} \right\rceil$ is another valid upper bound since $\overline{f}$ is the smallest integer number for which the inequality $\overline{f} \alpha_i^{(j)} \geq d_i$, $\forall i \in I^j$ is verified. This idea was implicitly used in Guimarães et al. (2025) to determine upper bounds on the frequency of execution of the cutting patterns.

Considering p_j as the j-th cutting pattern in P; α_i^j as the quantity of item type i that is obtained when the object is cut according to the cutting pattern p_j; T_L as the waste of material of the cutting pattern p_j, i.e. $T_L = L - \sum_{i=1}^{|I|} \ell_i \alpha_i^j$ and B_j as an upper bound on the frequency of the cutting pattern $p_j \in P$.

Algorithm 4 presents a systematic procedure for determining upper bounds on the number of cutting patterns in the Pattern Minimization Problem. The algorithm's workflow, depicted in Figure 4, follows the same visual conventions as Algorithm 3, with numbered steps and color-coded elements for enhanced readability. This approach guarantees that the obtained upper bounds are both tight and computationally efficient, serving as critical input parameters for the exact solution procedure.

(Algorithm 4)

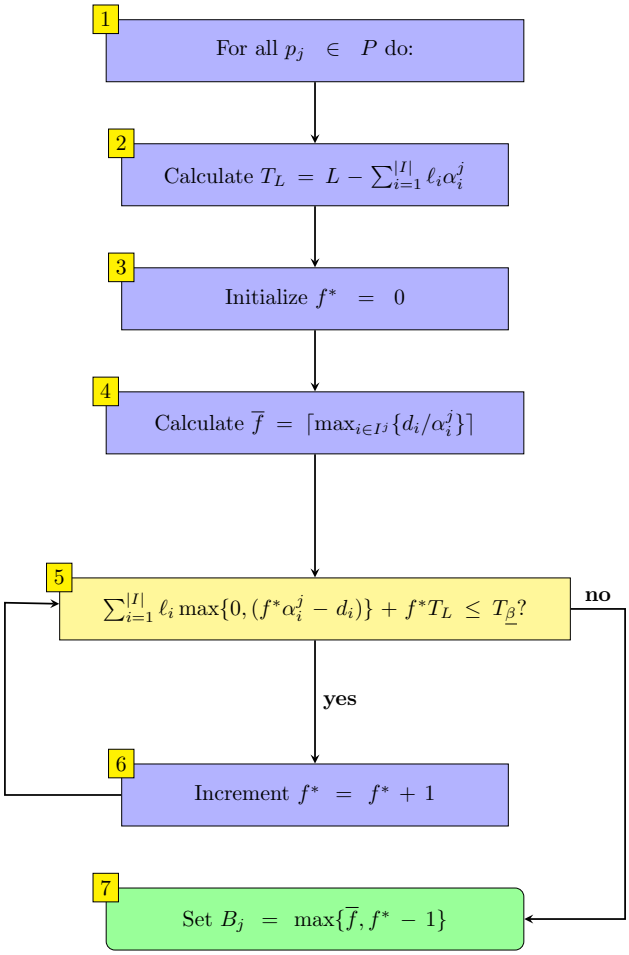


Figure 4: Flowchart of Algorithm Algorithm 4: Upper bounds on the number of cutting patterns. The iterative procedure refines the upper bounds through successive iterations. For clarity, all the steps are numbered and color-coded by function, i.e., blue for process steps, yellow for termination criteria, and green for final convergence.

4.2. Stage 2: Integer linear programming model for the CSP-S

As the model proposed by Cui et al. (2015), the second formulation proposed in this research employs a set of binary variables to evaluate the number of distinct cutting patterns. In the proposed model, the set of cutting patterns P, generated via the algorithm described in Section (4.1), is divided into two subsets: R and Q. The subset R includes all cutting patterns that may be executed with a frequency greater than one, whereas Q comprises cutting patterns which frequency of execution is limited to one. This partitioning is particularly important because when a cutting pattern can only be executed at most once, the corresponding binary variable is no longer necessary to determine whether such a cutting pattern is used.

More formally, let r_s, for $s \in R$, represent an integer variable indicating the frequency of execution of cutting pattern s in R; let v_s, for $s \in R$, be a binary variable that equals 1 if cutting pattern s in R is used, and 0 otherwise; and let B_s be as previously defined. The relationship between r_s and v_s is governed by the following constraints:

$$r_s \leq B_s v_s, \qquad s \in R. \tag{22}$$

The constraint set (22) implies that if v_s equals 0, then r_s must also equal 0. Otherwise, the constraints are trivially satisfied, as B_s is sufficiently large. Consequently, this set of constraints ensures that the cutting pattern s is used only if v_s equals 1. Now note that for the patterns in Q it is not necessary to define binary variables to count the number of cutting patterns, as their execution frequency, if used, is always equal to one. Thus, considering q_u, for $u \in Q$, as a binary variable equal to one if the cutting pattern u in Q is used and 0 otherwise, the number of distinct cutting patterns is equal to $\sum_{u \in Q} q_u + \sum_{s \in R} v_s$.

In the proposed model, the objective function is defined as the number of cutting patterns, and we ensure that the number of objects used is equal to the minimum possible through a set of constraints implemented in the model. We define the second formulation as follows:

(Formulation 2)

$$\min \quad \sum_{u \in Q} q_u + \sum_{s \in R} v_s \tag{23}$$

$$\text{subject to:} \quad \sum_{s \in R} \alpha_i^s r_s + \sum_{u \in Q} \alpha_i^u q_u \geq d_i, \qquad i = 1, \ldots, |I|, \tag{24}$$

$$\sum_{s \in R} r_s + \sum_{u \in Q} q_u = \underline{\beta}, \tag{25}$$

$$r_s \leq B_s v_s, \qquad s \in R, \tag{26}$$

$$r_s \in \mathbb{Z}^+, \qquad s \in R, \tag{27}$$

$$q_u \in \{0, 1\}, \qquad u \in Q, \tag{28}$$

$$v_s \in \{0, 1\}, \qquad s \in R. \tag{29}$$

The objective function (23) corresponds to minimizing the number of distinct cutting patterns. Constraints (24) ensure that the demand for all item types is satisfied. As discussed in Section 2, the number of objects used in a solution equates to the sum of the frequency of execution of all cutting patterns; in Formulation 2 this value is given by $\sum_{s \in R} r_s + \sum_{u \in Q} q_u$. Consequently, the constraint set (26) makes it so that the number of objects used equals $\underline{\beta}$. The constraint set (26) ensures that the cutting pattern r_s is only used if v_s is equal to 1, as previously stated. The remaining constraints concern the domains of the variables.

4.3. Summary of contributions and comparison with Cui et al. (2015) and Kayhan and Tekez (2025)

In summary, the main contribution of Section 4 is the development of **Approach 2**, an exact solution method for the PMP. Approach 2 requires exhaustive generation of cutting patterns (see Section 4.1) and applies three pattern-elimination criteria before optimization. The generated set P is partitioned into R and Q, where R contains cutting patterns that may be used with frequency greater than one, while the cutting patterns in Q can be used at most once. This partitioning allows omitting the use of binary variables for

Table 2: Compact comparison: heuristic approaches by Cui et al. (2015) and by Kayhan and Tekez (2025) versus the proposed Approach 2.

Aspect	Cui et al. (2015) and Kayhan and Tekez (2025)	**Proposed Approach 2 (exact)**
Nature	Heuristic: select a subset of cutting patterns or aggregate objectives (weighted sum); lower upfront cost but no optimality guarantee.	Exact: full enumeration of cutting patterns and ILP solution; guarantees optimality.
Cutting pattern generation	Subset selection (cheaper); reduces candidate cutting patterns at runtime.	Full cutting pattern enumeration is required (mandatory), incurring a higher preprocessing cost.
Cutting pattern reduction	Implicit via selection / heuristic filters.	Three explicit elimination criteria to prune dominated/irrelevant cutting patterns prior to solving.
Modeling device (cutting pattern variables)	Only selected cutting patterns are modeled; binary usage variables retained for modeled cutting patterns.	Partition $P = R \cup Q$: cutting patterns in Q (used at most once) do not require binary usage variables, reducing variable count.
Objective	Often, heuristic loss minimization or weighted aggregation of objectives (depends on the implementation).	Minimize the number of distinct cutting patterns; constraints enforce a minimal number of objects (ensures minimal waste).
Computational trade-off	Lower preprocessing and runtime; lacks optimality guarantees and may depend on tuning (e.g., weights).	Higher preprocessing (CSP + enumeration) but optimal and waste-optimal; three elimination criteria and R/Q partition mitigate cost.

cutting patterns in Q, thereby reducing model size. The model objective minimizes the number of distinct cutting patterns, while constraints guarantee the minimum number of objects used (hence minimal waste). Table 2 contrasts Approach 2 with the heuristic strategies in Cui et al. (2015) and Kayhan and Tekez (2025).

5. Computational experiments

This section presents the computational evaluation of the two proposed approaches for the Pattern Minimization Problem. We assess their performance against the state-of-the-art benchmarks and analyze the effectiveness of the proposed formulations through comparative implementations.

The experimental setup employs dual implementations for each approach: Approach 1 uses both the model from Martin et al. (2022a) and Formulation 1, while Approach 2 employs both the model from Cui et al. (2015) and Formulation 2. This design allows for a direct evaluation of the proposed formulations against the existing ones.

We set a time limit of 1200 seconds per iteration for Approach 1 (with termination after four consecutive timeouts) and 1800 seconds for Approach 2.

All experiments were conducted on a Dell Precision 3591 workstation equipped with an Intel Core Ultra 9 185H processor (16 cores, 22 threads) and 64 GB of RAM, running Windows 11 Pro. The computational

environment used Julia v1.11.5 with Gurobi v12.0.1 as the optimization solver, utilizing Gurobi's default parameters for all computations.

For benchmark comparisons, we focus on the state-of-the-art methods from Martin et al. (2022a), Cui et al. (2015) and Kayhan and Tekez (2025), as they consistently outperform other approaches in the literature Haessler (1975); Cui and Liu (2011); Cui et al. (2008); Umetani et al. (2003); Lee (2007). In these comparisons, Approach 1 and Approach 2 refer specifically to the versions using the proposed formulations, which consistently yielded solutions requiring a smaller average number of distinct cutting patterns.

The section is organized as follows: Section 5.1 describes the test instances, Section 5.1.1 presents results for the Fiber instances, Section 5.1.2 analyzes selected instances from the CUTGEN1 generator, and Section 5.3 synthesizes the main contributions and findings from the computational experiments, emphasizing both the theoretical advances and the practical performance of the proposed approaches compared with the state-of-the-art methods.

5.1. Instances description

The computational experiments use two sets of instances commonly referenced in the literature. The first set consists of 40 instances derived from a Japanese chemical fiber company, initially presented in Umetani et al. (2003). The second set of problem instances was generated using the CSP instance generator developed in Gau and Wäscher (1995). These instances are frequently used in research on the CSP with the auxiliary objective of minimizing cutting patterns, enabling us to evaluate the performance of the approaches proposed in this paper and compare them with established literature benchmarks.

We consider that item types of the same length are aggregated, i.e., if two item types i_1 and i_2 are such that if $\ell_{i_1} = \ell_{i_2}$ we set $d_{i_1} := d_{i_1} + d_{i_2}$ and remove i_2 from the set I. This is a well-known preprocessing procedure that has been employed in several papers in the literature, such as Lee (2007), Golfeto et al. (2009), de Araujo et al. (2014), Martin et al. (2022a), and Guimarães et al. (2025).

The first set, with 40 instances, is divided into two groups based on the length of the cut object. In the first 20 instances, the object length is 5180, and in the remaining instances, it is 9080. Additionally, demand values range from 2 to 264, item lengths (ℓ_i) range from 500 to 2000, and the number of distinct item types, represented by $|I|$, ranges from 4 to 20 after item type aggregation.

The second set of problem instances includes 18 classes, each one containing 100 instances. For the computational experiments, we consider the same subset of instances as Martin et al. (2022a), which corresponds to the first five instances from each class. The classes differ based on the ranges of item lengths ($\bar{\ell}$), average demand values ($\bar{d}$), and the number of distinct item types (M). The object length (L) is set to 1000 for all classes.

The parameter M is set to 10 for Classes 1, 2, 7, 8, 13, and 14; 20 for Classes 3, 4, 9, 10, 15, and 16; and 40 for Classes 5, 6, 11, 12, 17, and 18. The average demand $\bar{d}$ is 10 for odd classes and 100 for even classes. The item length range $\bar{\ell}$ varies as follows: $[10, 200]$ for Classes 1 to 6; $[10, 800]$ for Classes 7 to 12; and $[200, 800]$ for Classes 13 to 18.

Preliminary computational tests revealed that *Approach 1 paired with Formulation 1* is computationally impracticable for Classes 12 and 18 of the CUTGEN1 dataset. For Class 12, the first three tested instances required an average running time of approximately 25,500 seconds, yet still failed to produce any feasible solution. A similar behavior was observed for Class 18, where the average running time exceeded 10,000 seconds and again no feasible solution was identified. Given that none of the tested instances in these two

classes yielded feasible solutions—and that the required computation times were prohibitive—we classified this approach as unsuitable for Classes 12 and 18. Consequently, these two classes are not considered in the computational analyses involving Approach 1 paired with Formulation 1 in the remainder of this paper.

5.1.1. Results for the Fiber instances

OBS: Na revisão eu notei uma coisa, estamos usando italico para denotar que o limite de padrões foi alcançado, mas o dash em italico fica quase imperceptivel para quem está lendo, talvez mudar a notação?

Nossa, eu não tinha percebido dash em itálico... melhor mudar mesmo

Substitui por: the character "*" denote... Outra coisa, o certo é "is reached" ou "was reached"? acho que estou usando cada hora um...

Então, ambos estão certos, mas depende se vc quer. Acho que is reached está ok. Corrigido, coloquei "is reached" em todos.

Nas legendas das tabelas estava:A dash (—) represents infeasible cases. Achei ruin porque esse "cases" não explica bem o que é e o infeasible cases dá impressão que a instancia não tem solução factível oque está errado. Corrigi para: A dash (—) represents instances for which the approach failed to find a feasible solution within the time limit. Nas tabelas Fiber. E A dash (—) represents classes for which the approach failed to find a feasible solution for any instance within the time limit. Nas Classes do CUTGEN1.

Table 3 reports the results for the Fiber instances with lengths 5180. Column 1 lists the instance; Column 2 (Obj) gives the number of objects in the PMP solution (β). Columns 3–4 (P_M, T_M) show the number of distinct cutting patterns and the running time of Algorithm 3 using the model in Martin et al. (2022a). Columns 5–6 (P_{F1}, T_{F1}) give the same metrics for Algorithm 3 applied to Formulation 1. Columns 7–8 (P_C, T_C) report the number of distinct cutting patterns and the running time to solve the PMP with the model by Cui et al. (2015). Columns 9–10 (P_{F2}, T_{F2}) present the results for the PMP solved with Formulation 2.

The "–" symbol indicates that no feasible solution was found within the imposed time limit for that instance, while "tl" denotes that the time limit is reached for Approach 2 (1800s). Additionally, the number of cutting patterns is reported in **bold** when the solver certifies the optimal solution, and the character "*" to represent when the limit on cutting patterns is reached in the second approach.

Regarding the quality of the solutions, the second approach presented the best results for the instances with $L = 5180$. This approach was not only able to find solutions with minimal trim loss for all instances but also with a very small number of distinct cutting patterns. In terms of formulations, there was a sufficiently good improvement in the number of cutting patterns employed when using Formulation 2 compared to when using the model proposed in Cui et al. (2015). The average values of distinct cutting patterns used were 7.25 and 6.95 for the model proposed in Cui et al. (2015) and Formulation 2, respectively.

In general, the first approach proved to be ineffective for determining solutions with minimum total trim loss for this set of instances. In total, Algorithm 3 with Formulation 1 was able to find a solution with a minimum trim loss for 17 out of 20 instances and 15 out of 20 using the model proposed in Martin et al. (2022a).

Table 3: Computational results for Fiber instances with object length $L = 5180$. Columns show: Obj: number of objects used ($\underline{\beta}$); P_M, T_M: number of cutting patterns and running time for Algorithm 3 with Martin et al. (2022a) model; P_{F1}, T_{F1}: number of cutting patterns and running time for Algorithm 3 with Formulation 1; P_C, T_C: number of cutting patterns and running time for Cui et al. (2015) model; P_{F2}, T_{F2}: number of cutting patterns and running time for Formulation 2. Bold indicates certified optimality, the character "*" denote that the cutting pattern limit (1,000,000) is reached, a dash (—) represents instances for which the approach failed to find a feasible solution within the time limit, and "tl" indicates that the time limit (1800s) is reached for Approach 2.

Instance	Obj	P_M	T_M	P_{F1}	T_{F1}	P_C	T_C	P_{F2}	T_{F2}
Fiber06_5180	33	**5**	14.05	**5**	17.5	**5**	2.09	**5**	0.8
Fiber07_5180	33	**3**	1.66	**3**	2.39	**3**	44.58	**3**	0.06
Fiber08_5180	86	**4**	3.92	**4**	10.29	**4**	4.74	**4**	0.18
Fiber09_5180	53	**6**	222.18	**6**	25.73	**6**	2.04	**6**	0.17
Fiber10_5180	69	**5**	17.11	**5**	13.59	**5**	4.9	**5**	0.7
Fiber11_5180	67	**5**	13.71	**5**	11.04	5	tl	**5**	1.29
Fiber13a_5180	56	**4**	5.95	**4**	7.79	**4**	18.41	**4**	0.91
Fiber13b_5180	28	**4**	1.64	**4**	16.97	**4**	779.58	**4**	25.67
Fiber14_5180	47	9	2607.56	8	1566.97	**8**	6.79	**8**	12.74
Fiber15_5180	57	**5**	7.3	**5**	8.85	**5**	7.22	**5**	2.36
Fiber16_5180	82	-	-	11	4119.59	**8**	54.57	**8**	142.79
Fiber17_5180	83	7	2898.34	7	1912.63	7	tl	7	tl
Fiber18_5180	96	8	3021.88	7	1901.51	7	tl	**7**	306.58
Fiber19_5180	133	6	2208.59	6	2271.85	7	tl	**6**	99.99
Fiber20_5180	32	**7**	408.58	**7**	157.22	10	tl	8	tl
Fiber23_5180	142	-	-	-	-	11	62.48	11	16.63
Fiber26_5180	190	-	-	10	5066.57	9	tl	8	tl
Fiber28a_5180	83	-	-	-	-	11	tl	11	tl
Fiber28b_5180	117	-	-	-	-	15	tl	15	271.01
Fiber29_5180	62	9	5008.98	**7**	609.2	11	tl	9	tl

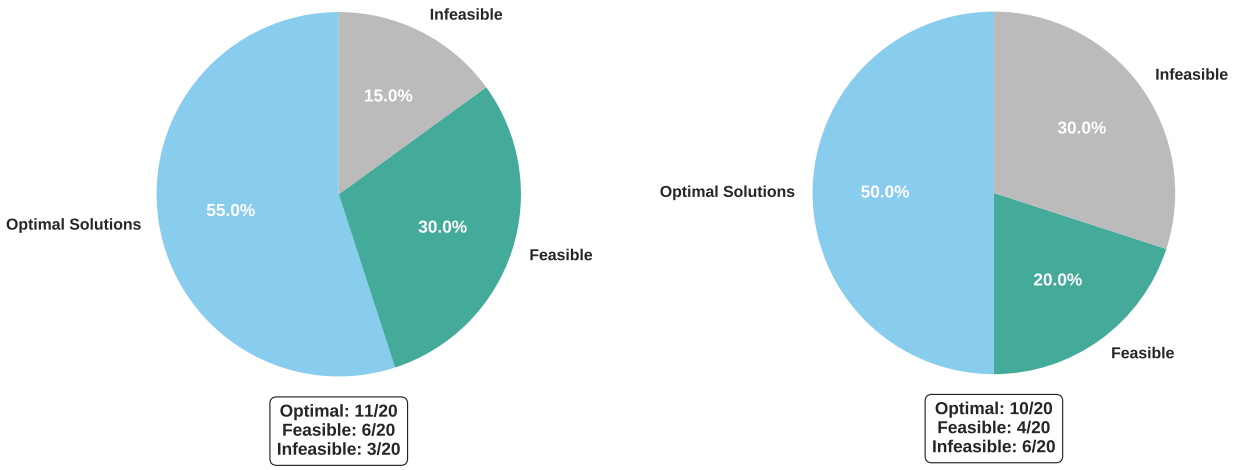
In terms of computational time, considering only the instances for which at least a feasible solution was found, the analysis reveals that Approach 1 consumes significantly more time. Algorithm 3 with Formulation 1 required an average of 1042.33 seconds (ranging from 7.79 to 5066.57), while with the model proposed in Martin et al. (2022a) it averaged 1096.10 seconds (ranging from 1.64 to 5008.98). In comparison, Approach 2 achieved better performance, with averages of 859.37 seconds for model Cui et al. (2015) (ranging from 2.09 to the solver's time limit) and 494.09 seconds for Formulation 2 (ranging from 0.7 to the solver's time limit). The auxiliary components of Approach 2—namely, the column generation procedure and the cutting pattern generation step—were computationally negligible, never exceeding 5 seconds even in the worst case.

Benchmark comparisons against results reported in the literature highlight distinct performance levels for each proposed approach. We classify outcomes into three categories: *Optimal* for instances that the solver reached optimality within time limits; *Feasible* for instances that a minimum trim loss solution was found but optimality wasn't certified by the solver; and *Infeasible* for instances that no feasible PMP solution was found within time limits.

Approach 1 demonstrates incremental improvements over the exact framework byMartin et al. (2022a), achieving 17 feasible solutions (85% feasibility rate) with 11 optimal (55% optimality rate) and 3 infeasible instances. This compares favorably to Martin et al. (2022a)'s 14 feasible (70% feasibility rate), 10 optimal

(50% optimality rate), and 6 infeasible results for the same 20 instances, representing a 21.4% improvement in solution coverage. However, its performance remains limited for this particular instance set.

These comparative results are summarized in Figure 5, which compares Approach 1 with the results in Martin et al. (2022a), evaluating three aspects: the number of feasible PMP solutions (i.e., with minimum trim loss), the number of solutions with proven optimality, and the number of infeasible instances.



(a) Approach 1.

(b) Martin et al. (2022a).

Figure 5: Solution quality comparison between (a) Approach 1 and (b) Martin et al. (2022a) for the 20 Fiber 5180 instances. Pie charts show the distribution of optimal solutions , the number of feasible non-optimal solutions, and the number of infeasible instances.

Approach 2 achieves substantial improvements over existing methods, outperforming both the heuristic of Cui et al. (2015) and the recent OPTICUT algorithm developed by Kayhan and Tekez (2025) on the Fiber 5180 benchmark set. It reduces the average number of distinct cutting patterns from 9.70 (as reported by Cui et al. (2015)) and 7.6 (as reported by OPTICUT) to 6.95, which corresponds to significant reductions of 28.4% and 8.6%, respectively. Compared directly to the state-of-the-art OPTICUT, the per-instance relative improvement averages 7.9% (std: 16.6%) with gains reaching up to 33.3% reduction in pattern count.

While the average number of objects used decreases only marginally, from 77.45 to 77.40, Approach 2 consistently attains the minimum trim loss across all instances, indicating that the gains in cutting pattern reduction do not come at the expense of material efficiency.

We summarize the results of Approach 2 in Figures 6 and 7, which compare its performance against the best benchmarks in the literature. The figures focus on two key metrics: the number of distinct cutting patterns and the number of objects used.

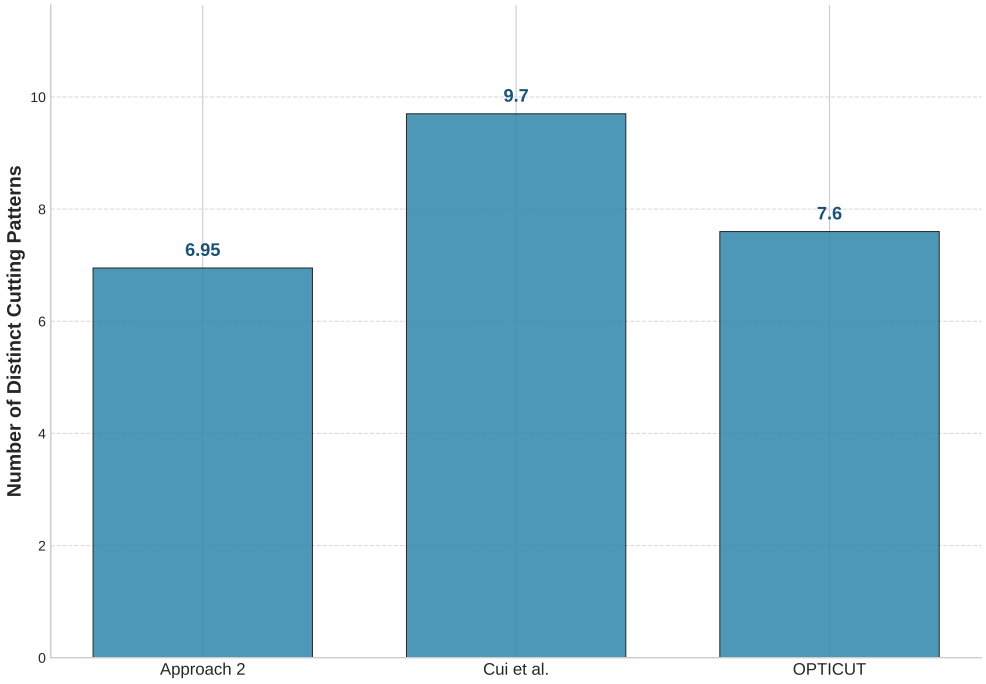


Figure 6: Comparison of the average number of distinct cutting patterns required to solve the 20 Fiber 5180 benchmark instances, assessing the performance of Approach 2 relative to the heuristics proposed in Cui et al. (2015) and Kayhan and Tekez (2025).

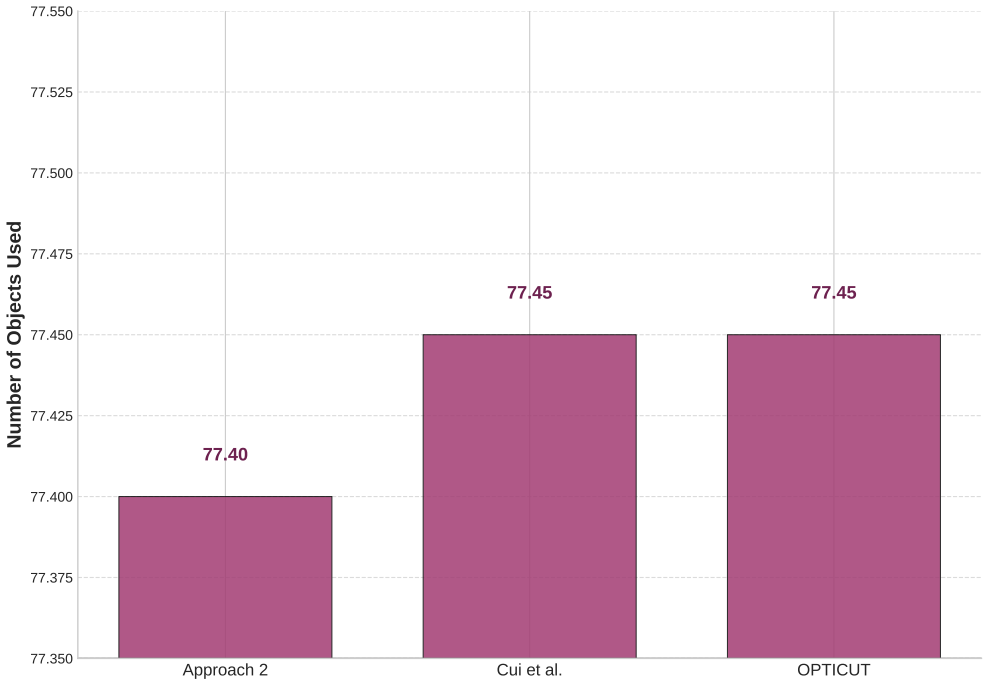


Figure 7: Average number of objects used across the 20 Fiber 5180 benchmark instances, comparing the results of Approach 2 with those obtained by the heuristics in Cui et al. (2015) and Kayhan and Tekez (2025).

The columns in Table 4 are organized in the same manner as those in Table 3; however, Table 4 reports the results for the Fiber 9080 instances, produced by the methods: Martin et al. (2022a), Formulation 1, Cui et al. (2015), and Formulation 2.

Table 4: Computational results for Fiber instances with object length $L = 9080$. Columns follow the same structure as Table 3: Obj - number of objects used (β); P_M, T_M - number of cutting patterns and running time for Martin et al. (2022a) model; P_{F1}, T_{F1} - number of cutting patterns and running time for Formulation 1; P_C, T_C - number of cutting patterns and running time for Cui et al. (2015) model; P_{F2}, T_{F2} - number of cutting patterns and running time for Formulation 2. Bold indicates certified optimality, the character "*" denote that the cutting pattern limit (1,000,000) is reached, a dash (—) represents instances for which the approach failed to find a feasible solution within the time limit, and "tl" indicates that the time limit (1800s) is reached for Approach 2..

Instance	Obj	P_M	T_M	P_{F1}	T_{F1}	P_C	T_C	P_{F2}	T_{F2}
Fiber06_9080	19	**3**	3.85	**3**	14.45	3	tl	**3**	0.92
Fiber07_9080	19	**2**	0.53	**2**	3.03	2	tl	**2**	0.05
Fiber08_9080	48	**3**	0.86	**3**	8.38	**3**	7.97	**3**	0.27
Fiber09_9080	29	**3**	1.0	**3**	4.3	4	tl	**3**	1.12
Fiber10_9080	39	**3**	2.17	**3**	42.66	3	tl	**3**	3.58
Fiber11_9080	38	**4**	3.65	**4**	59.8	4	tl	**4**	14.07
Fiber13a_9080	32	**4**	5.66	**4**	55.66	5	tl	**4**	540.65
Fiber13b_9080	16	**3**	2.09	**3**	39.99	4	tl	**3**	5.04
Fiber14_9080	27	**3**	4.54	**3**	76.9	5	tl	**3**	21.72
Fiber15_9080	32	**4**	3.01	**4**	59.24	4	tl	**4**	37.44
Fiber16_9080	47	**5**	18.23	**5**	354.54	7	tl	6	tl
Fiber17_9080	47	**4**	14.68	**4**	218.19	7	tl	6	tl
Fiber18_9080	54	**5**	17.97	**5**	213.21	6	tl	6	tl
Fiber19_9080	73	-	-	-	-	8	tl	**6**	468.91
Fiber20_9080	19	**4**	4.67	**4**	103.86	*	*	*	*
Fiber23_9080	80	8	2549.06	7	1547.5	9	tl	7	tl
Fiber26_9080	107	**5**	146.01	**5**	721.27	10	tl	7	tl
Fiber28a_9080	48	**5**	19.32	**5**	212.42	*	*	*	*
Fiber28b_9080	67	7	1888.51	7	2151.74	*	*	*	*
Fiber29_9080	35	**5**	23.72	**5**	120.99	*	*	*	*

For instances with larger object lengths ($L = 9080$), the trend was reversed, with Approach 1 exhibiting greater reliability. It identified 19 feasible solutions (i.e., solutions with minimum trim loss) and proved optimality for 17 of them. In one specific case, Approach 1 obtained a better solution using Formulation 1 than the model proposed in Martin et al. (2022a), while for all remaining instances, both approaches achieved solutions of equal quality. The average number of distinct cutting patterns among these feasible solutions was 4.17 when using Formulation 1 and 4.21 with the model by Martin et al. (2022a).

For the Fiber9080 instances, a different performance behavior emerges. Considering only the instances for which at least one feasible solution was obtained, Approach 1 demonstrates significantly lower computational times, with Algorithm 3 using Formulation 1 averaging 316.22 seconds (ranging from 3.03 to 2151.74) and the model by Martin et al. (2022a) averaging 247.87 seconds (ranging from 0.53 to 2549.06). In contrast, Approach 2 shows considerably longer running times, with model Cui et al. (2015) requiring 1688.0 seconds (ranging from 7.97 to the solver's time limit) and Formulation 2 taking 630.86 seconds on average (ranging from 0.05 to the solver's time limit). The auxiliary stages of Approach 2—such as column generation and cutting pattern generation—were computationally negligible and did not meaningfully contribute to the total running time. This represents a complete reversal of the previous trend, where Approach 1 now proves to be the more efficient choice for these instances.

The results further suggest that the second approach is less efficient for solving instances with a large number of cutting patterns. Nevertheless, Formulation 2 once again outperformed the model proposed in Cui et al. (2015) for this set of instances.

For the Fiber9080 instances, Approach 1 achieved minimum trim loss in 19 out of 20 cases. For these 19 feasible instances, it consistently outperformed all benchmarks from the literature. The remaining instance was not solved feasibly by the proposed approach. Specifically, Approach 1 reduced the average number of distinct cutting patterns from 5.55 (as reported by Cui et al. (2015)) and 4.84 (as reported by Kayhan and Tekez (2025)) to 4.16, representing a **14.1**% **overall reduction** compared to OPTICUT. The per-instance analysis shows even stronger performance with an average reduction of **12.9**% (std: 10.6%) across all feasible cases, with improvements ranging from 0.0% to 25.0%. Additionally, the proposed approach surpassed Martin et al. (2022a), which reported minimum trim loss solutions for only 18 out of 20 instances.

The performance improvement is illustrated in Figure 8, which compares the number of cutting patterns obtained by Approach 1 with those reported by Cui et al. (2015) and OPTICUT, considering the 19 instances where minimum trim loss was achieved.

5.1.2. Results for selected instances from the CUTGEN1 generator

The results for the selected instances from the CUTGEN1 generator are presented in Table 5. For a direct comparison with the benchmarks established in Martin et al. (2022a), we followed the same experimental design by selecting the first five instances from each class, totaling 90 instances. The table adopts the same notation as Tables 3 and 4; however, instead of providing detailed results for each instance, it reports the averages for the five instances in each class. As before, the number of cutting patterns is highlighted in bold when the solver certifies optimality. Additionally, the number of solutions found for the PMP is shown in parentheses, except for classes where all instances were solved, in which case the parentheses are omitted. Below, we present Table 5 along with a brief discussion of the computational experiment results.

Agora vi no texto, está em azul. Desculpa. Mas agora eu vi na legenda da Table 5: "Parentheses indicate partially solved classes" Acho que não fica claro essa frase. A que vc escreveu é mais clara que essa

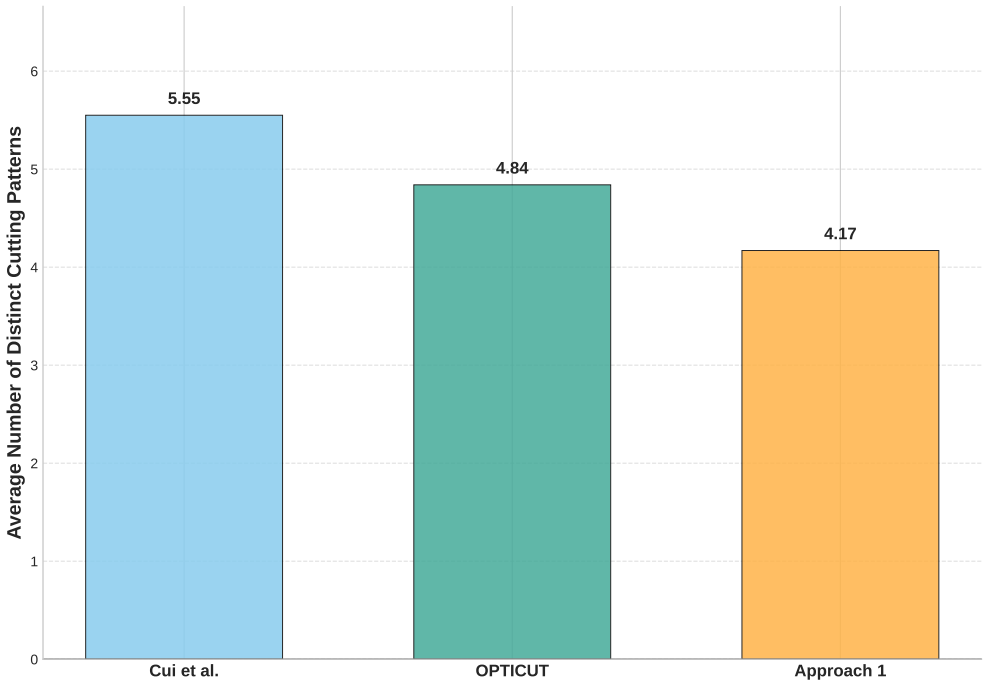


Figure 8: Comparison of the average number of distinct cutting patterns for Fiber9080 instances, based on results reported in the literature. Approach 1 achieves an average of 4.17 cutting patterns over the 19 instances where minimum trim loss was determined, representing a significant reduction compared with the 5.55 cutting patterns reported by Cui et al. (2015) and 4.84 cutting patterns reported by OPTICUT.

"parcialmente" Melhor trocar a frase da legenda. Oi Kelly sem problemas!. Ajustei nas legendas conforme solicitado

For the 90 instances generated by the CUTGEN1 generator, Approach 2 demonstrated superior performance. It identified 68 solutions with minimum trim loss regardless of the model used, while achieving 57 and 48 optimal solutions, as confirmed by the solver, when paired with Formulation Formulation 2 and the model proposed by Cui et al. (2015), respectively. The average computation times were 949.20 seconds for Approach 2 with Formulation 2 and 551.48 seconds for Approach 2 with the model by Cui et al. (2015).

In contrast, Approach 1 identified 54 and 50 minimum trim loss solutions and achieved 41 and 30 optimal solutions, as confirmed by the solver, when paired with Formulation 1 and the model by Martin et al. (2022a), respectively. The average computation times—computed only over the instances for which feasible solutions were obtained—were 813.88 seconds for Approach 1 with Formulation 1 and 1040.34 seconds for Approach 1 with the model by Martin et al. (2022a).

The results presented in Table 5 suggest that three primary instance parameters significantly affect the performance of Approach 1 and Approach 2: the average demand values of the item types, denoted by $\bar{d}$, the lengths of the item types relative to the larger object, determined by the class-specific range of the item lengths, denoted by $\bar{\ell}$, and the number of distinct item types, given by M.

As previously defined, for the instances generated by the CUTGEN1 generator, $\bar{\ell}$ is set to one of the following intervals: $[10, 200]$, $[10, 800]$, or $[200, 800]$. Specifically, $\bar{\ell}$ is set to $[10, 200]$ for Classes 1 to 6; $[10, 800]$ for Classes 7 to 12; and $[200, 800]$ for Classes 13 to 18, ensuring that the length of each generated

Table 5: Computational results for the CUTGEN1 instances (first five instances of each class). Averages reported for: *Obj* - number of objects used; P_M, T_M - number of cutting patterns and running time for Martin et al. (2022a) model; P_{F1}, T_{F1} - number of cutting patterns and running time for Formulation 1; P_C, T_C - number of cutting patterns and running time for Cui et al. (2015) model; P_{F2}, T_{F2} - number of cutting patterns and running time for Formulation 2. The numbers in parentheses represent the number of PMP solutions found (omitted when all instances were solved), bold shows certified optimality, the character "*" denote that the cutting pattern limit (1,000,000) is reached, a dash (—) represents classes for which the approach failed to find a feasible solution for any instance within the time limit, and "tl" indicates that the time limit (1800s) is reached for Approach 2.

Class	*Obj*	P_M	T_M	P_{F1}	T_{F1}	P_C	T_C	P_{F2}	T_{F2}
1	11.4	**3.2**	1.78	**3.2**	4.01	3.6	1801.47	3.2	556.64
2	109.4	4.6	856.51	4.4	384.09	6.8	1801.49	5.0	1668.74
3	23.2	**4.2**	5.56	**4.2**	22.4	*	*	*	*
4	225.8	7.0 (1)	4395.73	7.0 (2)	3357.91	*	*	*	*
5	42.4	7.4	2383.37	7.2	1406.94	*	*	*	*
6	420.8	-	-	-	-	*	*	*	*
7	51.2	**6.2**	48.19	**6.2**	24.41	**6.2**	25.47	**6.2**	0.36
8	510.0	6.0 (4)	959.86	6.0 (4)	450.61	**6.6**	720.25	**6.6**	0.72
9	100.4	12.5 (2)	2122.58	11.67 (3)	1199.56	**12.2**	74.43	**12.2**	11.59
10	1000.8	13.0 (1)	424.0	13.0 (1)	1777.0	**13.8**	4.93	**13.8**	4.16
11	175.4	20.5 (2)	1592.08	17.0 (1)	3500.38	23.0 (4)	946.04	22.75 (4)	1112.11
12	1756.0	-	-	-	-	26.25 (4)	980.94	25.75 (4)	1194.2
13	62.8	**7.8**	402.78	**7.8**	17.26	**7.8**	0.32	**7.8**	0.12
14	625.4	7.67 (3)	1565.84	8.25 (4)	819.49	**8.4**	0.37	**8.4**	0.21
15	125.2	13.67 (3)	1275.99	13.5 (4)	258.76	**13.6**	4.11	**13.6**	1.62
16	1251.2	14.5 (2)	1897.09	14.33 (3)	2576.26	15.4	720.23	**14.8**	17.78
17	221.4	24.5 (2)	2558.89	24.0 (2)	2609.02	25.0	377.77	**24.8**	45.18
18	2213.6	-	-	-	-	28.6	430.6	27.6	749.71

item is within each defined range, i.e., $\ell_i \in \bar{\ell}$ for $i \in I$. Additionally, the parameter M is set to 10 for Classes 1, 2, 7, 8, 13, and 14; 20 for Classes 3, 4, 9, 10, 15, and 16; and 40 for Classes 5, 6, 11, 12, 17, and 18. The average demand $\bar{d}$ is set to 10 for odd-numbered classes and 100 for even-numbered classes. Table 6 summarizes the results based on these parameters.

Table 6: Number of feasible solutions and number of optimal solutions found based on the instance parameters (the number of different item types, average length of the item types, and the average demand for the item types). The results include Approaches 1 and 2 combined with Formulation 1 and Formulation 2, as well as the models by Martin et al. (2022a) and Cui et al. (2015).

Model	Martin et al. (2022a)		Formulation 1		Cui et al. (2015)		Formulation 2	
	Feas.Sol.	Opt.Sol.	Feas.Sol.	Opt.Sol.	Feas.Sol.	Opt.Sol.	Feas.Sol.	Opt.Sol.
$M = 10$	27	21	28	25	30	18	30	25
$M = 20$	14	8	18	14	20	18	20	20
$M = 40$	9	1	8	2	18	12	18	12
$\bar{\ell} = [10, 200]$	21	14	22	15	10	0	10	5
$\bar{\ell} = [10, 800]$	14	9	14	11	28	22	28	24
$\bar{\ell} = [200, 800]$	15	7	18	15	30	26	30	28
$\bar{d} = 10$	34	22	35	27	34	26	34	29
$\bar{d} = 100$	16	8	19	14	34	22	34	27

The results reveal a clear trend: as the number of distinct item types increases, the performance of the

proposed approaches, particularly their ability to achieve optimal solutions, declines. This indicates that the approaches are most effective for small to moderately-sized instances. For larger instances, purely heuristic methods, such as the ones proposed by Cui et al. (2015) and Kayhan and Tekez (2025), may offer better performance.

Analyzing the impact of demand values, the results suggest that Approach 1 faces notable challenges when solving instances with higher demand. While Approach 2 is also influenced by demand size, it demonstrates significantly greater efficiency in handling high-demand scenarios, making it a better choice in such cases.

The relationship between the lengths of item types and the object's length further highlights the contrasting strengths of the approaches. Consistent with the findings from Section 5.1.1, Approach 1 performs better when the ratio between item lengths and the object length is smaller. In contrast, Approach 2 excels in scenarios where this ratio is larger or intermediate, consistently delivering superior results in such instances.

Lastly, Formulation 1 and Formulation 2 consistently outperform the models proposed in Martin et al. (2022a) and Cui et al. (2015), respectively.

To address the 90 instances generated by the CUTGEN1, the authors in the original paper Martin et al. (2022a), instead of using the exact algorithm proposed, considered a weighted sum approach, assigning a weight 10 times greater to the objective of minimizing the number of objects than to the objective of minimizing the number of distinct cutting patterns, following the same strategy as Cui et al. (2015). As reported in their original work, the authors also considered an upper bound on the number of cutting patterns to determine $\mathcal{K}$. According to the results presented in Martin et al. (2022a), their proposed algorithm was able to find optimal solutions for 40 instances using model (1)-(8) and for 34 instances using the other model proposed in the paper. In comparison, Approach 1 found optimal solutions for a total of 41 instances (2.5% improvement), while Approach 2 demonstrated substantially superior performance, solving 57 instances to optimality (42.5% improvement over Martin et al. (2022a)'s best result). The comparative performance is summarized in Figure 9.

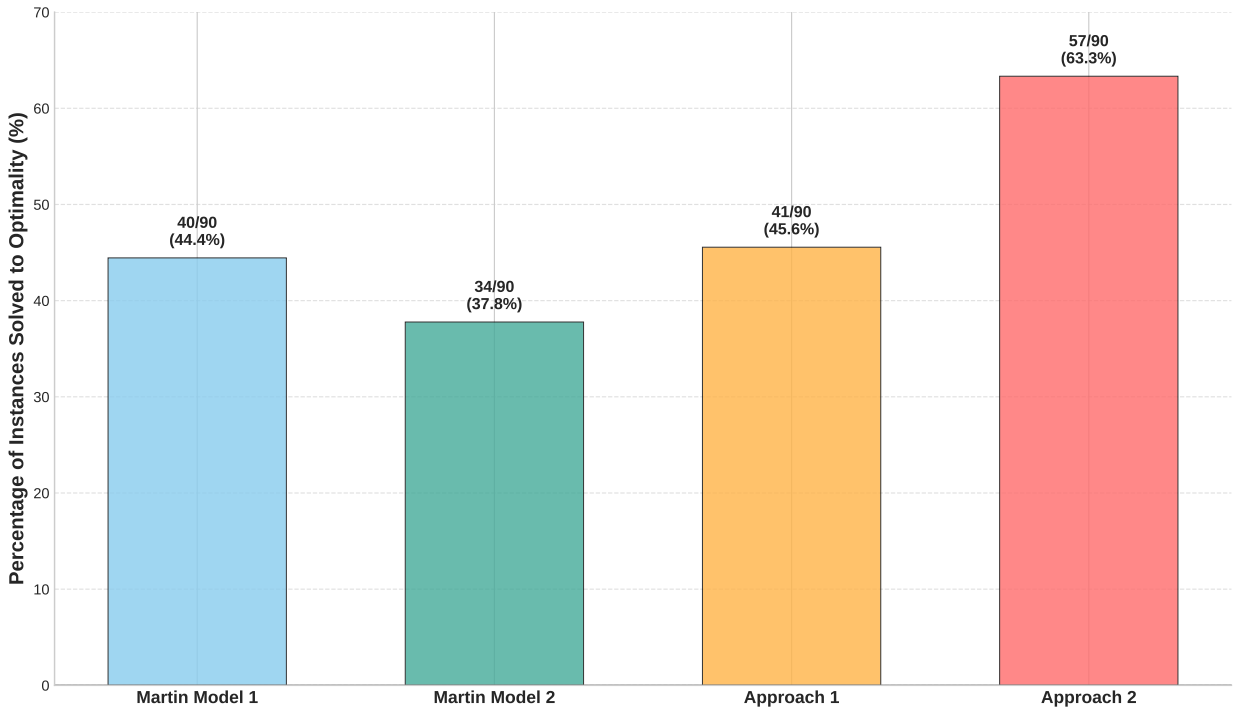


Figure 9: Comparison of optimal solutions found for the CUTGEN1 instances. Approach 2 achieves the best performance for 57 out of 90 instances solved to optimality, followed by Approach 1 (41 instances), both outperforming the results reported by Martin et al. (2022a) for their two models (40 and 34 instances, respectively).

5.2. Computational cost of auxiliary steps in Approaches 1 and 2

This section reports the computational effort associated with the auxiliary procedures used in Approaches 1 and 2, namely preprocessing procedures, column generation, and cutting pattern generation. For illustration, we measured the computational times for the first instance of each CUTGEN1 class.

For Approach 1, the analysis reveals that preprocessing accounts for 23.64% of the total computational time on average, while the optimization model itself dominates with 76.32% of the execution time. Column generation and bin packing procedures represent negligible fractions of the total cost (0.04% and $\approx 0.0\%$, respectively). In terms of maximum observed times, the preprocessing phase reached 4544.69 seconds in the most challenging instances, while the optimization model required up to 5333.88 seconds. The column generation step exhibited a maximum time of 3.09 seconds, whereas the bin packing procedure reached a maximum of 0.24 seconds. This distribution highlights that in Approach 1, both preprocessing routines and the main optimization model contribute substantially to the overall computational burden, with preprocessing requiring an average of 657.84 seconds and the model consuming 2123.79 seconds across the instances.

For Approach 2, the analysis reveals a significantly different computational profile compared to Approach 1. The solution time is overwhelmingly dominated by the optimization model itself. While column generation may represent a reasonable percentage of the total time in trivial instances that are solved very quickly, its absolute duration remains minimal, requiring at most 2.28 seconds even in the most challenging cases. Cutting pattern generation is even faster, never exceeding one second for those instances. Combined,

these preprocessing steps account for less than 5 seconds in all instances, representing a negligible fraction of the total computational effort, especially when compared to the model solution time, where many instances have reached the solver time limit of 1800 seconds.

5.3. Summary of contributions and comparative analysis

This section synthesizes the main contributions and findings from the computational experiments, emphasizing both the theoretical advances and the practical performance of the proposed approaches compared with the state-of-the-art methods. The comprehensive benchmarking compares Our Best Results (OBR) against the Best Known Results (BKR) from the literature. The BKR values were sourced from Martin et al. (2022a); Kayhan and Tekez (2025), which include both exact approaches by Martin et al. (2022a) and the heuristic methods by Martin et al. (2018); Kayhan and Tekez (2025); Cui et al. (2015), and Yanasse and Limeira (2006). For each Fiber instance and CUTGEN1 class, we define BKR as the solution that uses the lowest number of objects as the primary criterion. In cases of ties, we select the solution with the lowest distinct number of cutting patterns. Similarly, OBR represents the best results obtained in this study, selected from either Approach 1 or Approach 2 using the same hierarchical criteria.

To enable comparison with the benchmarks from Cui et al. (2015) and Kayhan and Tekez (2025), we conducted additional experiments using the complete CUTGEN1 instance set (100 instances per class). However, based on the results from Section 5.1.2, we excluded Classes 4, 6, 11, and 12 from this comparative analysis, as the approaches were unable to determine feasible solutions for all instances in such classes. For the remaining classes where feasible solutions (i.e., with minimum trim loss) could be obtained for all instances, we employed the most effective approach for each class: Approach 1 for Classes 1, 2, 3 and 5, and Approach 2 for Classes 7, 8, 9, 10, 13, 14, 15, 16, 17, and 18.

Table 7 provides a comprehensive comparison of our best results against the state-of-the-art benchmarks across all instance sets. The table reports both the number of objects (Obj.) and the number of distinct cutting patterns (Pat.) for Our Best Results (OBR) and the Best Known Results (BKR) from the literature. The comparison encompasses Fiber instances with object lengths 5180 and 9080, along with the CUTGEN1 classes, each evaluated over 100 instances. Results where OBR outperforms BKR are highlighted in green, while cases where BKR remains superior are marked in pink.

The central contribution of this work lies in introducing exact methods for the Pattern Minimization Problem (PMP), addressing a long standing gap in the literature where previous studies have focused primarily on heuristic or metaheuristic techniques. Beyond raw performance metrics, the proposed exact approaches provide a crucial theoretical advantage: they deliver mathematically proven optimal solutions for the PMP, offering certainty where heuristic methods can only guarantee approximate results. The proposed approaches exhibit remarkable performance, particularly for instances with up to 20 item types, consistently outperforming the best known results in this range while simultaneously providing optimality certificates.

However, we observe the classical scalability trade off in combinatorial optimization: as problem size increases beyond 20 item types, the computational complexity of the proposed exact models becomes prohibitive. This delineates the practical applicability boundary for exact methods in PMP and provides valuable guidance for method selection based on instance characteristics.

Table 7: Comprehensive comparison of our best results against the state-of-the-art benchmarks across all instance sets. The table reports both the number of objects (Obj.) and the number of distinct cutting patterns (Pat.) for Our Best Results (OBR) and the Best Known Results (BKR) from the literature. The comparison encompasses Fiber instances with object lengths 5180 and 9080, along with the CUTGEN1 classes, each evaluated over 100 instances. Results where OBR outperforms BKR are highlighted in green, while cases where BKR is superior are marked in pink.

Instance	Fiber 5180				Fiber 9080				CUTGEN1				
	BA		BL		BA		BL		Class	BA		BL	
	Obj.	Pat.	Obj.	Pat.	Obj.	Pat.	Obj.	Pat.		Obj.	Pat.	Obj.	Pat.
Fiber06	33	5	33	5	19	3	19	3	C1	11.48	2.90	11.48	2.90
Fiber07	33	3	33	3	19	2	19	2	C2	-	-	110.25	6.27
Fiber08	86	4	86	4	48	3	86	3	C3	22.13	4.19	22.13	4.19
Fiber09	53	6	53	6	29	3	29	3	C4	-	-	215.93	8.25
Fiber10	69	5	69	5	39	3	39	3	C5	-	-	42.95	7.84
Fiber11	67	5	67	5	38	4	38	4	C6	-	-	424.68	12.48
Fiber13a	56	4	56	6	32	4	32	4	C7	50.24	6.32	50.24	6.32
Fiber13b	28	4	28	4	16	3	16	3	C8	499.62	6.94	499.62	7.03
Fiber14	47	8	47	8	27	3	27	3	C9	93.65	11.74	93.65	12.26
Fiber15	57	5	57	5	32	4	32	4	C10	932.26	13.32	932.26	13.89
Fiber16	82	8	82	9	47	5	47	5	C11	-	-	176.91	24.21
Fiber17	83	7	83	7	47	4	47	4	C12	-	-	1763.46	27.99
Fiber18	96	7	96	7	54	5	54	5	C13	63.47	7.51	63.47	7.51
Fiber19	133	6	133	6	73	6	73	10	C14	632.36	8.08	632.36	8.10
Fiber20	32	7	32	7	19	4	19	4	C15	119.59	13.98	119.59	14.15
Fiber23	142	11	142	13	80	7	80	7	C16	1192	15.32	1192.00	15.62
Fiber26	190	8	190	12	107	5	107	5	C17	224.85	25.56	224.85	27.08
Fiber28a	83	11	83	12	48	5	48	5	C18	2242.59	28.25	2242.59	30.16
Fiber28b	117	15	*118*	*12*	67	7	67	6					
Fiber29	62	7	62	8	35	5	35	5					

6. Sensitivity analysis

This section presents a comprehensive sensitivity analysis to evaluate the reliability of our proposed methodology and provide empirical justification for key parameter selections. We focus our analysis on instances generated via CUTGEN1, as these exhibit substantial variation in parameters, including the number of items, the length of items relative to the larger object, and demand values, enabling a thorough investigation across diverse scenarios. All experiments in this section utilize the first five instances from each class generated by the CUTGEN1 generator, following the same experimental protocol established in Section 5.1.2. We systematically examine how three critical parameters affect solution quality and computational performance: (1) the maximum number of cutting patterns generated in Approach 2; (2) the time limits imposed on the MIP solver; and (3) the lower bound on the number of objects ($\underline{\beta}$).

6.1. Analysis of the limit on cutting pattern generation

A fundamental constraint of Approach 2 is its requirement to generate and store all feasible cutting patterns. This raises the critical question of how to select an appropriate maximum limit for pattern generation. In this work, we established a limit of 1,000,000 cutting patterns, which represents a balance between computational feasibility and solution quality. This threshold was chosen based on preliminary experiments showing that pattern generation remains computationally tractable within this bound and that it is sufficient to obtain high-quality solutions for most instances in our benchmark.

To provide a comprehensive understanding of cutting pattern generation requirements across different problem classes, Table 8 categorizes the five instances of each class into six cutting pattern count

ranges: ≤100,000; (100,000–500,000]; (500,000–1,000,000]; (1,000,000–2,000,000]; (2,000,000–5,000,000]; and >5,000,000. The data reveals substantial variation in computational complexity: Classes 1, 2, 7–10, and 13–18 contain only instances in the lowest range (≤100,000 cutting patterns), while Classes 5 and 6 demonstrate the highest complexity, with all five instances falling into the highest category (>5,000,000 cutting patterns). Classes 3, 4, 11, and 12 exhibit intermediate behavior, with their instances distributed across the medium cutting pattern count ranges.

Table 8: Distribution of pattern generation complexity across problem classes, showing how many of the five instances per class fall into different ranges of cutting patterns generated.

Class	**≤100k**	**(100k,500k]**	**(500k,1M]**	**(1M,2M]**	**(2M,5M]**	**>5M**
C1	5	0	0	0	0	0
C2	3	2	0	0	0	0
C3	0	0	0	1	1	3
C4	0	0	0	0	1	4
C5	0	0	0	0	0	5
C6	0	0	0	0	0	5
C7	5	0	0	0	0	0
C8	5	0	0	0	0	0
C9	5	0	0	0	0	0
C10	5	0	0	0	0	0
C11	1	1	2	1	0	0
C12	1	1	2	1	0	0
C13	5	0	0	0	0	0
C14	5	0	0	0	0	0
C15	5	0	0	0	0	0
C16	5	0	0	0	0	0
C17	5	0	0	0	0	0
C18	5	0	0	0	0	0
Total	**60 (66.67%)**	**4 (4.44%)**	**4 (4.44%)**	**3 (3.33%)**	**2 (2.22%)**	**17 (18.89%)**

Based on this distribution, our choice of 1,000,000 cutting patterns as the generation limit is well justified, covering 68 out of 90 instances (approximately 76%). Furthermore, increasing the cutting pattern limit to 5,000,000 enables Approach 2 to handle only 5 additional instances, while the remaining 17 instances would still reach this new limit.

To thoroughly assess whether a more permissive cutting pattern limit could enhance the applicability of Approach 2 for challenging problem classes, we conducted additional computational experiments. These experiments focus on Classes 3, 4, 11, and 12, representing cases where cutting pattern generation exceeds the initial limit, using an extended cutting pattern limit of 5,000,000 cutting patterns and an increased solver time limit of 3600 seconds.

The experimental results are presented in Table 9. The table reports average values for the five instances in each class, excluding executions that reached the cutting pattern generation limit. Classes 5 and 6 are omitted from this analysis as they consistently reached the extended cutting pattern generation limit of 5,000,000 across all instances, preventing Approach 2 from obtaining feasible solutions.The columns in Table 9 represent the following: P_{F2} for the number of cutting patterns used in Formulation 2 (with the number of PMP solutions shown in parentheses); T_{F2} for the computational time for Formulation 2 (in seconds); P_C for the cardinality of the set of feasible cutting patterns generated; and T_P for the computational time spent generating the cutting patterns (in seconds).

The key findings from this analysis reveal important limitations and opportunities for Approach 2: For the most complex classes (3, 4, 5, and 6), increasing the cutting pattern generation limits and the computational time provided limited benefits. Even when allowing substantially higher cutting pattern generation and longer solving times, Approach 2 finds only a low number of feasible solutions of low quality for Classes 3 and 4, while Classes 5 and 6 remain intractable. The results for Classes 3 and 4 show that

Table 9: Performance analysis of Approach 2 with extended limits: cutting pattern generation (up to 5,000,000) and running time (up to 3600 seconds).

Class	P_{F2}	T_{F2} **(s)**	P_C	T_P **(s)**
C3	6.5 (2)	2159.0	1,814,506.0	1.7476
C4	14.0 (1)	2173.1	2,774,826.0	1.6992
C11	22.6 (5)	2339.1	521,081.6	0.8189
C12	26.8 (5)	3118.6	637,388.2	1.3043

Approach 2 finds fewer feasible solutions while requiring a much larger number of distinct cutting patterns compared to Approach 1, despite the substantially higher computational effort.

Eu trocaria o "investment", esse termo me remete muito ao financeiro

Corrigido

For intermediate complexity classes (Classes 11 and 12), however, the extended limits enable Approach 2 to find good quality solutions for all instances. Nevertheless, the computational time required by the approach, averaging 2339.1 and 3118.6 seconds for Classes 11 and 12, respectively, remain considerable and must be carefully weighed against potential improvements in solution quality, particularly when compared with heuristic approaches available in the literature.

6.2. Analysis of the time limits

The time limits of 1200 seconds for Approach 1 and 1800 seconds for Approach 2 were established based on preliminary experiments and represent conservative choices to ensure that potential solutions are not missed due to excessively restrictive limits. To validate these choices, we conduct a sensitivity analysis that evaluates the impact of both lower and higher time limits on solution quality and computational performance.

6.2.1. Evaluation of higher time limits

We assess whether additional computational time improves solution quality for instances where optimality was not proven in the original experiments. For Approach 1, we re-evaluated Classes 2 and 5 with a 2400-second per iteration time limit. For Approach 2, we tested Class 17 with a 3600-second time limit, since Classes 3, 4, 5, 6, 11, and 12 were already examined in Section 6.1, where both the cutting pattern limit (5,000,000 cutting patterns) and the time limit (3600 seconds) were simultaneously increased.

The results show that neither Approach 1 nor Approach 2 improved solution quality under the higher time limits. Identical solutions, in terms of objective value (i.e., the number of distinct cutting patterns used), were obtained for all 15 instances tested (Classes 2 and 5 for Approach 1, and Class 17 for Approach 2).

6.2.2. Evaluation of lower time limits

To assess the impact of reduced computational resources, we conducted experiments with lower time limits on selected classes from the CUTGEN benchmark, where at least one feasible solution was obtained for the class under original time limits. We implemented a time limit of 200 seconds per iteration for Approach 1 (with termination after four consecutive timeouts) and 300 seconds per iteration for Approach 2. The computational results are presented separately in Table 10 for Approach 1 and Table 11 for Approach 2.

Tables 10 and 11 present comparative analyses of Approach 1 and Approach 2 performances, respectively, under different time constraints for the PMP. Both tables focus specifically on classes where performance differences were observed, omitting classes with identical results under the original time limits and the reduced 200/300-second limits. We employ the following notation: P_l represents the number of cutting patterns obtained with the lower time limit (200 seconds for Approach 1, and 300 seconds for Approach 2), while P_r denotes the number of cutting patterns achieved with the regular time limit (1800 seconds). Similarly, T_l and T_r indicate the computational times, in seconds, for the lower and the regular time limits, respectively. The final column displays the relative percentage degradation in solution quality for the lower time limit, calculated as $(P_l - P_r)/P_r \times 100\%$, where positive values indicate worse solutions (more cutting patterns) obtained with reduced computation time. The number of PMP solutions found is reported in parentheses, and it is omitted in cases where all instances were solved.

Table 10: Analysis of solution quality degradation under reduced time limits for Approach 1: lower time limit (200 seconds) vs regular time limit (1200 seconds).

Class	$\mathbf{P_l}$	$\mathbf{T_l(s)}$	$\mathbf{P_r}$	$\mathbf{T_r(s)}$	**Degradation (%)**
C2	4.60	108.35	4.4	384.09	4.55%
C14	8.50 (4)	453.51	8.25 (4)	819.49	3.03%
C15	13.75 (4)	345.25	13.5 (4)	258.76	1.85%

Table 11: Analysis of solution quality degradation under reduced time limits for Approach 2: lower time limit (300 seconds) vs regular time limit (1800 seconds).

Class	$\mathbf{P_l}$	$\mathbf{T_l(s)}$	$\mathbf{P_r}$	$\mathbf{T_r(s)}$	**Degradation (%)**
C2	5.20	301.70	5.00	1668.74	4.00%
C11	24.50	251.60	22.75	1112.11	7.69%
C12	26.50	255.19	25.75	1194.20	2.91%
C17	25.00	80.12	24.80	45.18	0.81%
C18	27.80	180.55	27.60	749.71	0.72%

Approach 1 demonstrates varying sensitivity to time constraints across different problem classes. The most significant degradation in solution quality occurs in Class 2 (4.55%), followed by Class 14 (3.03%) and Class 15 (1.85%). Notably, Class 16 experienced a reduction in feasible solutions, yielding only one solution under time constraints compared to three with regular limits. For the remaining classes, solution quality remained consistent between time conditions, indicating reliability for most problem instances.

Despite these variations, Approach 1 generally maintains acceptable solution quality under strict time constraints, particularly for classes showing minimal sensitivity to computational time reductions.

Approach 2 exhibits a broader range of performance impacts under reduced computation time. Class 11 shows the most pronounced degradation at 7.69%, while Classes 2 and 12 present moderate declines of 4.00% and 2.91%, respectively. Classes 17 and 18 demonstrate minimal impact, with degradations below 1%.

The majority of classes (Classes 1, 3-10, 13-16) maintained consistent performance regardless of time constraints, with Class 16 showing particular reliability. This suggests that Approach 2 remains effective under strict time limits for many problem types, despite meaningful quality reductions in sensitive classes.

6.3. Analysis of the $\underline{\beta}$ parameter

The parameter $\underline{\beta}$, which sets the lower bound on the number of objects, plays a central role in both approaches. We analyze how changes to $\underline{\beta}$ influence feasibility, number of cutting patterns, and computational effort. To assess this effect, we conducted additional experiments using three enlarged values that increase $\underline{\beta}$ by approximately 1%, 2.5%, and 5%, which are: $\beta_1 = \lceil \underline{\beta} \cdot 1.01 \rceil$, $\beta_2 = \lceil \underline{\beta} \cdot 1.025 \rceil$, and $\beta_3 = \lceil \underline{\beta} \cdot 1.05 \rceil$. The results for Approach 1 are shown in Table 12.

Tables 12 and 13 details the impact of increasing the lower bound $\underline{\beta}$ on the performance of Approach 1 and Approach 2, respectively. For each class, three metrics are reported: a) the number of cutting patterns (Pat.); b) the objective value (Obj.); and c) the computing time (Time(s)), for the original $\underline{\beta}$ and the increased values β_1, β_2, and β_3. This structure enables a direct analysis of how tightening this constraint affects cutting pattern generation, solution quality, and computational effort.

Table 12: Impact of increasing the lower bound $\underline{\beta}$ in Approach 1. The table shows how feasibility, number of cutting patterns, and computational effort are affected when $\underline{\beta}$ is increased by approximately 1% (β_1), 2.5% (β_2), and 5% (β_3). A dash (—) represents classes for which the approach failed to find a feasible solution for any instance within the time limit, and "tl" indicates that the time limit (1800s) is reached for Approach 2.

Class	$\underline{\beta}$			β_1			β_2			β_3		
	Pat.	Obj.	Time(s)	Pat.	Obj.	Time(s)	Pat.	Obj.	Time(s)	Pat.	Obj.	Time(s)
1	3	13	5.41	2	14	5.71	2	14	5.73	2	14	8.69
2	5	123	1243.11	4	125	240.48	3	127	6.57	3	130	7.79
3	4	23	14	3	24	16.97	3	24	12.60	3	25	15.47
4	-	-	-	7	228	3986.36	5	231	2024.45	4	237	288.58
5	7	41	2521.95	6	42	1302.26	6	43	831.19	5	44	110.18
7	5	66	18.46	5	67	12.21	5	68	12.66	5	70	15.33
8	5	647	149.11	5	654	107.73	5	664	103.01	5	680	117.37
9	-	-	-	14	92	5676.41	11	94	2537.38	11	96	1534.04
10	-	-	-	-	-	5281.29	-	-	5260.48	12	963	4546.54
11	-	-	-	-	-	5461.96	-	-	5495.99	18	167	3074.41
13	7	80	22.53	7	81	18.84	7	82	26.45	7	84	22.85
14	7	790	135.67	7	798	203.88	7	810	308.75	7	830	267.34
15	13	127	108.07	13	129	139.35	13	131	200.72	13	134	212.24
16	13	1283	1206.527	13	1296	2156.45	13	1316	1938.78	13	1348	2164.12
17	-	-	-	-	-	-	24	215	5272.97	23	220	3946.79

Table 13: Impact of increasing the lower bound $\underline{\beta}$ in Approach 2. The table shows how feasibility, number of cutting patterns, and computational effort are affected when $\underline{\beta}$ is increased by approximately 1% (β_1), 2.5% (β_2), and 5% (β_3). A dash (—) represents classes for which the approach failed to find a feasible solution for any instance within the time limit, and "tl" indicates that the time limit (1800s) is reached for Approach 2.

Class	$\underline{\beta}$			β_1			β_2			β_3		
	Pat.	Obj.	Time(s)	Pat.	Obj.	Time(s)	Pat.	Obj.	Time(s)	Pat.	Obj.	Time(s)
1	3	13	54.48	2	14	1145.83	2	14	1246.02	2	14	1208.35
2	5	123	tl	4	125	tl	4	127	tl	3	130	tl
7	5	66	0.06	6	67	0.07	6	68	0.05	6	70	0.05
8	5	647	0.08	6	654	0.05	6	664	0.07	6	680	0.05
9	14	91	2.36	13	92	5.85	11	94	1.76	11	96	3.16
10	15	917	1.16	14	927	20.49	12	940	4.91	11	963	1.81
11	25	159	tl	22	161	tl	20	163	tl	17	167	140.87
12	31	1594	tl	27	1610	tl	24	1634	tl	19	1674	tl
13	7	80	0.07	7	81	0.05	7	82	0.04	7	84	0.04
14	7	790	0.13	7	798	0.02	7	810	0.05	7	830	0.04
15	13	127	0.15	13	129	0.08	13	131	0.07	13	134	0.04
16	13	1283	0.48	13	1296	0.15	13	1316	0.08	13	1348	0.1
17	26	209	4.05	24	212	2.17	23	215	3.94	23	220	0.96
18	30	2103	tl	26	2125	246.79	24	2156	24.7	23	2209	3.57

Gabriel, a linha da Classe 15 da tabela 13 está desconfigurada. Corrigido.

Eu colocaria na legenda das tabelas 12 e 13 o que é o dash e na tabela 13 p que é o tl. Corrigido.

não seria beta com a barra? Tem razão, corrigido. (esse martin aqui, trocar pelo cite com a referencia para qual artigo vc se refere.) Corrigido. Marquei em magenta

Experiments with larger values of $\underline{\beta}$ show some positive effects: both approaches achieve a lower average running time, and Approach 1, in particular, also finds more feasible solutions. However, these gains come with important drawbacks. Solution quality deteriorates as the number of objects increases, more than the reduction achieved for the number of cutting patterns. Because our approaches lexicographically prioritize minimizing the number of objects, increasing β is generally detrimental. If, instead, the goal is to explore the trade-off between the number of objects and the number of cutting patterns, methods such as those proposed in Martin et al. (2022a) and Kayhan and Tekez (2025) may be more appropriate.

7. Final remarks

This research addressed the Pattern Minimization Problem (PMP) through two novel approaches: an exact iterative framework based on ILP formulations, and a two-stage algorithm that combines pattern generation with reduction criteria followed by ILP optimization. The experimental analysis revealed that both approaches exhibit a complementary relationship: the iterative framework performs better on instances with larger object lengths, while the two-stage heuristic performs better on instances with smaller object lengths. Notably, our methods achieved optimal solutions for several benchmark instances where optimality had not previously been established in the literature, yielding concrete quantitative improvements over the state-of-the-art.

From a practical standpoint, our approaches deliver significant value for industrial cutting applications. The ability to find optimal or provably high-quality solutions for complex instances translates directly into minimizing raw material waste. Furthermore, the core objective of the PMP addresses key operational costs by reducing the number of distinct cutting patterns required, thereby decreasing production downtime, simplifying scheduling, and reducing the frequency of complex machine setups and knife changes. This combination of material efficiency and operational simplicity makes these approaches particularly suitable for real world implementation.

For future research, promising directions include enhancing the exact framework with advanced branch-and-bound strategies and decomposition techniques to improve scalability. The development of heuristics for the cutting pattern generation stage that intelligently balance the reduction criteria also warrants further investigation. Additionally, exploring hybrid methods that combine the strengths of both approaches could lead to more robust solutions across diverse instance types. Extending these methodologies to address more complex industrial constraints and multi-objective optimization scenarios would further broaden their practical applicability.

Data availability

Data will be made available on request.

Acknowledgements

The authors would like to express their gratitude to Professor Mateus Martin for generously sharing his research findings with us. The authors would like to thank the São Paulo Research Foundation (FAPESP-Brazil) [grant number 2022/05803-3], the National Council for Scientific and Technological Development (CNPq-Brazil) [grant number 402240/2023-5] and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES-Brazil) [Finance Code 001] for the financial support.

References

Aloisio A, Arbib C, Marinelli F. Cutting stock with no three parts per pattern: Work-in-process and pattern minimization. Discrete Optimization 2011a;8:315–32. doi:10.1016/j.disopt.2010.10.002.

Aloisio A, Arbib C, Marinelli F. On lp relaxations for the pattern minimization problem. Networks 2011b;57(3):247–53.

Alves C, de Carvalho JV. A branch-and-price-and-cut algorithm for the pattern minimization problem. RAIRO - Operations Research 2009;42:435–53. doi:10.1051/ro:2008027.

Alves C, Macedo R, de Carvalho JV. New lower bounds based on column generation and constraint programming for the pattern minimization problem. Computers & Operations Research 2009;36(11):2944–54.

Andersen E, Andersen K. Presolving in linear programming. Mathematical Programming 1995;71:221–45. doi:10.1007/BF01586000.

de Araujo SA, Poldi KC, Smith J. A genetic algorithm for the one-dimensional cutting stock problem with setups. Pesquisa Operacional 2014;34(2):165–87. doi:10.1590/0101-7438.2014.034.02.0165.

Arbib C, Marinelli F, Pezzella F. An lp-based tabu search for batch scheduling in a cutting process with finite buffers. International Journal of Production Economics 2012;136:287–96. doi:10.1016/j.ijpe.2011.12.003.

Arbib C, Marinelli F, Ventura P. One-dimensional cutting stock with a limited number of open stacks: bounds and solutions from a new integer linear programming model. International Transactions in Operational Research 2016;23:47—-63. doi:10.1111/itor.12134.

Atamturk A, Savelsbergh M. Integer-programming software systems. Annals of Operations Research 2005;140:67–124. doi:10.1007/s10479-005-3968-2.

Bang I, Kim BI, Park J, Kim G. Integrated cutting stock and multi-period inventory optimization considering raw material–product eligibility for steel-pipe manufacturers. Applied Mathematical Modelling 2025;142:115953. doi:https://doi.org/10.1016/j.apm.2025.115953.

Belov G, Scheithauer G. The number of setups (different patterns) in one-dimensional stock cutting. Technical Report MATH-NM-15-2003; Institute for Numerical Mathematics, Dresden University; 2003.

Bonnevay S, Aubertin P, Gavin G. A genetic algorithm to solve a real 2-d cutting stock problem with setup cost in the paper industry. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation. 2015. p. 807–814. doi:10.1145/2739480.2754660.

Brandão JS, Coelho AM, Vasconcellos JFV, Salles Neto LL, Pinto AV. Application of genetic algorithm to minimize the number of objects processed and setup in a one-dimensional cutting stock problem. International Journal of Applied Evolutionary Computation 2011;2(1):34–48. doi:10.4018/jaec.2011010103.

Cui Y, Liu Z. C-sets-based sequential heuristic procedure for the one-dimensional cutting stock problem with pattern reduction. Optimization Methods and Software 2011;26(1):155–67. doi:10.1080/10556780903420531.

Cui Y, Yang L, Zhao Z, Tang T, Yin M. Sequential grouping heuristic for the two-dimensional cutting stock problem with pattern reduction. International Journal of Production Economics 2013;144(2):432–9. doi:10.1016/j.ijpe.2013.03.011.

Cui Y, Zhao X, Yang Y, Yu P. A heuristic for the one-dimensional cutting stock problem with pattern reduction. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture 2008;222(6):677–85. doi:10.1243/09544054JEM966.

Cui Y, Zhong C, Yao Y. Pattern-set generation algorithm for the one-dimensional cutting stock problem with setup cost. European Journal of Operational Research 2015;243:540–6. doi:10.1016/j.ejor.2014.12.015.

Delorme M, Iori M. Enhanced pseudo-polynomial formulations for bin packing and cutting stock problems. INFORMS Journal on Computing 2020;32:101—-119. doi:10.1287/ijoc.2018.0880.

Diegel A, Chetty M, Schalkwyk SV, Naidoo S. Setup combining in the trim loss problem - 3-to-2 & 2-to-1. Working Paper; 1993.

Filho AA, Moretti AC, Pato MV. A comparative study of exact methods for the bi-objective integer one-dimensional cutting stock problem. Journal of the Operational Research Society 2018;69(1):91–107. doi:10.1057/s41274-017-0214-7.

Foerster H, Wascher G. Pattern reduction in one-dimensional cutting stock problems. International Journal of Production Research 2000;38(7):1657–76. doi:10.1080/002075400188780.

Gau T, Wäscher G. Cutgen1: A problem generator for the standard one-dimensional cutting stock problem. European Journal of Operational Research 1995;84:572–9. doi:10.1016/0377-2217(95)00023-J.

Golfeto RR, Moretti AC, Salles Neto LL. A genetic symbiotic algorithm applied to the one-dimensional cutting stock problem. Pesquisa Operacional 2009;29(2):365–82. doi:10.1590/S0101-74382009000200008.

Guimarães GG, Poldi KC. Mathematical models for the cutting stock with limited open stacks problem. RAIRO-Operations Research 2023;57(4):2067–85. doi:10.1051/ro/2023079.

Guimarães GG, Poldi KC, Martin M. Mathematical models for the one-dimensional cutting stock problem with setups and open stacks. Journal of Combinatorial Optimization 2025;49:43. doi:10.1007/s10878-025-01276-5.

Guimarães GG, Poldi KC, Martin M. On formulations for the one-dimensional cutting stock with a limited number of open stacks problem. International Journal of Production Research 2025;:1–28doi:10.1080/00207543.2025.2568739.

Hadj Salem K, Silva E, Oliveira JF, Carravilla MA. Mathematical models for the two-dimensional variable-sized cutting stock problem in the home textile industry. European Journal of Operational Research 2023;306(2):549–66. doi:10.1016/j.ejor.2022.08.018.

Haessler RW. Controlling cutting pattern changes in one-dimensional trim problems. Operations Research 1975;23(3):483–93. doi:10.1287/opre.23.3.483.

Kayhan N, Tekez EK. OPTICUT: a new heuristic algorithm for the one-dimensional cutting stock problem with pattern minimization. Engineering Optimization 2025;:1–23doi:10.1080/0305215X.2025.2480864.

de Lara Andrade PR, de Araujo SA, Cherri AC, Lemos FK. The integrated lot sizing and cutting stock problem in an automotive spring factory. Applied Mathematical Modelling 2021;91:1023–36. doi:https://doi.org/10.1016/j.apm.2020.10.033.

Leao AA, Furlan MM, Toledo FM. Decomposition methods for the lot-sizing and cutting-stock problems in paper industries. Applied Mathematical Modelling 2017;48:250–68. doi:https://doi.org/10.1016/j.apm.2017.04.010.

Lee J. In situ column generation for a cutting-stock problem. Computers & Operations Research 2007;34(8):2345–58. doi:10.1016/j.cor.2005.09.007.

Martin M, Moretti AC, Gomes-Ruggiero MA, Salles-Neto LL. Modification of haessler's sequential heuristic procedure for the one-dimensional cutting stock problem with setup cost. Production 2018;28:e20170105. URL: https://doi.org/10.1590/0103-6513.20170105. doi:10.1590/0103-6513.20170105.

Martin M, Yanasse HH, Salles-Neto LL. Pattern-based ilp models for the one-dimensional cutting stock problem with setup cost. Journal of Combinatorial Optimization 2022a;44:557—582. doi:10.1007/s10878-022-00848-z.

Martin M, Yanasse HH, Santos MO, Morabito R. Models for two- and three-stage two-dimensional cutting stock problems with a limited number of open stacks. International Journal of Production Research 2022b;61:2895–916. doi:10.1080/00207543.2022.2070882.

McDiarmid C. Pattern minimisation in cutting stock problems. Discrete Applied Mathematics 1999;98(1-2):121–30. doi:10.1016/S0166-218X(99)00112-2.

Mellouli A, Mellouli R, Masmoudi F. An innovative genetic algorithm for a multi-objective optimization of two-dimensional cutting-stock problem. Applied Artificial Intelligence 2019;33(6):531–47. doi:10.1080/08839514.2019.1583857.

Mobasher A, Ekici A. Solution approaches for the cutting stock problem with setup cost. Computers & Operations Research 2013;40:225–35. doi:10.1016/j.cor.2012.06.007.

Muñoz C, Sierra M, Puente J, Vela CR, Varela R. Improving cutting-stock plans with multi-objective genetic algorithms. In: Bio-inspired Modeling of Cognitive Tasks. 2007. p. 528–37. doi:10.1007/978-3-540-73053-8_53.

Pierini LM, Poldi KC. An analysis of the integrated lot-sizing and cutting-stock problem formulation. Applied Mathematical Modelling 2021;99:155–65. doi:https://doi.org/10.1016/j.apm.2021.06.009.

Scheithauer G, Terno J. The modified integer round-up property of the one-dimensional cutting stock problem. European Journal of Operational Research 1995;84(3):562–71. doi:10.1016/0377-2217(95)00022-I.

Silva EM, Melega GM, Akartunalı K, Araujo SA. Formulations and theoretical analysis of the one-dimensional multi-period cutting stock problem with setup cost. European Journal of Operational Research 2023;304(2):443–60. doi:10.1016/j.ejor.2022.04.023.

Umetani S, Yagiura M, Ibaraki T. One-dimensional cutting stock problem to minimize the number of different patterns. European Journal of Operational Research 2003;146:388–402. doi:10.1016/S0377-2217(02)00239-4.

Umetani S, Yagiura M, Ibaraki T. One-dimensional cutting stock problem with a given number of setups: A hybrid approach of metaheuristics and linear programming. Journal of Mathematical Modelling and Algorithms 2006;5:43–64. doi:10.1007/s10852-005-9031-0.

Vanderbeck F. Exact algorithm for minimising the number of setups in the one-dimensional cutting stock problem. Operations Research 2000;48:915–26. doi:10.1287/opre.48.6.915.12391.

Yanasse HH, Lamosa MJP. An integrated cutting stock and sequencing problem. European Journal of Operational Research 2007;183:1353–70. doi:10.1016/j.ejor.2005.09.054.

Yanasse HH, Limeira MS. A hybrid heuristic to reduce the number of different patterns in cutting stock problems. Computers & Operations Research 2006;33(9):2744–56. doi:10.1016/j.cor.2005.02.026.