Exact approaches for the pattern minimization problem - Supplementary material

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.

1. Introduction

The supplementary material is divided into five sections, where we detail the structured usage of the proposed upper and lower bounds, discuss the application of our approaches to the bi-objective cutting stock problem with setup costs, provide additional implementation details for the approaches in our paper as well as those in Martin et al. (2022) and Cui et al. (2015), present the computational experiment results regarding the scenario without overproduction of items and explore adaptations of our approaches for alternative strategies to balance the number of different cutting patterns and overproduction.

2. A detailed explanation of the structured usage of the upper and lower bounds proposed

As follows, we present a general algorithm to find upper and lower bounds on the frequency of the cutting patterns for Formulation 1. The algorithm relies on Propositions 2 to 5, presented in Section 3.2, and on Algorithm 1: A preprocessing algorithm to determine upper bounds on the frequency of the cutting patterns, presented in Section 3.4.1.

The algorithm takes as input the number of cutting patterns used, $\mathcal{K}$, and an upper bound on the frequency of execution of the first cutting pattern, Φ^1, which can be determined using the algorithm presented in Section 3.4.1. Additionally, the algorithm also has as input the vector $\overline{\mathbf{d}} = [d_{s_1}, d_{s_2}, \ldots, d_{s_{|I|}}]$, where s_i represents the indices of the demand vector arranged in non-increasing order, such that if $i_1 \geq i_2$, then $d_{s_{i_1}} \geq d_{s_{i_2}}$, as defined in Section 3.2. As outputs, the algorithm returns the upper and lower bounds on the frequency of the cutting patterns. The following algorithm is used to determine the upper and lower bounds proposed in the paper:

The first four conditional operators in Algorithm 1 are related to Propositions 2 to 4, while the fifth conditional operator is related to the general case in which the propositions cannot be applied. Furthermore, the values of Θ^k are obtained via the closed-form expression presented in Section 3.2.

3. Remarks on the Cutting Stock Problem with Setup Cost (CSP-S)

We highlight that the contributions proposed in the previous sections are not limited to the context of the Pattern Minimization Problem (PMP). In fact, the improvements developed in this work can be reframed to suit the CSP-S with minimal changes.

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

Algorithm 1 An algorithm to determine upper and lower bounds on the frequency of the cutting patterns

for k from 2 to $\mathcal{K}$ **do**
  if $d_{s1} > kd_{s2}$ **then**
    $\Phi^k = \min\{\lfloor(\underline{\beta} - max\{0, \mathcal{K} - k\})/k\rfloor, \lfloor d_{s_1}/k'\rfloor, \Phi^{k-1}\}$
  else
    if $k = 2$ **then**
      $\Phi^k = \min\{\lfloor(\underline{\beta} - max\{0, \mathcal{K} - k\})/k\rfloor, max\{d_{s_2}, \lfloor d_{s_1}/2\rfloor\}, \Phi^{k-1}\}$
    end if
    if $k = 3$ and $2d_{s_2} \leq d_{s_1}$ and $3d_{s_2} > d_{s_1}$ **then**
      $\Phi^k = \min\{\lfloor(\underline{\beta} - max\{0, \mathcal{K} - k\})/k\rfloor, d_{s_2}, \Phi^{k-1}\}$
    end if
    if $k = 3$ and $2d_{s_2} > d_{s_1}$ **then**
      $\Phi^k = \min\{\lfloor(\underline{\beta} - max\{0, \mathcal{K} - k\})/k\rfloor, max\{d_{s_3}, \lfloor d_{s_1}/2\rfloor\}, \Phi^{k-1}\}$
    end if
    if $k > 3$ **then**
      $\Phi^k = \min\{\lfloor(\underline{\beta} - max\{0, \mathcal{K} - k\})/k\rfloor, \Phi^{k-1}\}$
    end if
  end if
end for
for k from 1 to $\mathcal{K}$ **do**
  $\Theta^k = \left\lceil \frac{\underline{\beta} - \overline{\Phi}^{k-1}}{\mathcal{K}-k+1} \right\rceil$
end for
return Θ^k and Φ^k, for $k = 1, \ldots, \mathcal{K}$.

An intrinsic part of the PMP definition is a predefined upper bound on the number of used objects, given by $\underline{\beta}$. Such a limitation is fundamental to the development of most of the improvements proposed in Section 3. Even though the CSP-S has no pre-established limitations in the problem definition, a valid upper bound on the number of used objects, given by $\overline{\beta}$, can be determined via a simple two-stage method in the CSP-S context.

The first stage consists of solving the associated Bin Packing Problem (BPP) and obtaining a set of cutting patterns corresponding to an optimal solution to the problem. Then, in the second stage, the frequency of the cutting patterns previously determined is post-processed to be equal to the highest value of demand of the items generated by the execution of the cutting patterns. This method was proposed in Martin et al. (2022) as a means to determine an initial value for $\overline{\beta}$.

Naturally, the initial bound may be too loose to produce significant improvements during the early stages of the process. However, it is worth emphasizing that this bound can be iteratively refined using the solution for the CSP-S obtained in the previous iteration, as demonstrated in Martin et al. (2022). Within a bi-objective approach, we expect the bounds to progressively improve as the algorithm advances, ultimately enhancing its overall performance.

Therefore, inequality (9) can be applied in the context of the CSP-S by switching $\underline{\beta}$ with $\overline{\beta}$. As a consequence, the propositions and valid inequalities, as well as the preprocessing procedure, can be applied to the CSP-S by changing $T_{\underline{\beta}}$ to $T_{\overline{\beta}}$ whenever necessary.

4. Additional details regarding the implementation specifics in each paper

The approaches in Martin et al. (2022) were implemented in C++ and utilized GUROBI v.9.1.1 as the ILP solver. The experiments were conducted on a PC equipped with an Intel Xeon E5-2680 processor (2.7 GHz), limited to 4 threads, 32 GB of RAM, and running Ubuntu 16.04 LTS.

The sequential heuristic from Cui et al. (2015) was implemented in C++, utilizing the CPLEX solver (v 12.5), and executed on a computer with an Intel Core i7-3632QM CPU (2.20 GHz) and 8 GB of RAM.

As mentioned in our paper, our approaches were implemented in the Julia programming language (v 1.9.4) using Gurobi solver (version 11.0.3) as the ILP solver. The experiments were conducted on a computer with an Intel i7-8700 processor (3.20 GHz) and 16 GB of RAM.

To solve the instances generated by the CUTGEN1 generator, the authors Martin et al. (2022) adopted a weighted sum approach. They assigned a weight 10 times greater to minimizing the number of objects than to minimizing the number of different cutting patterns, following the same strategy as Cui et al. (2015). We also employed this strategy when implementing the model proposed in Cui et al. (2015).

It is worth noting that this strategy may produce solutions that do not minimize total trim loss, although it did not occur in our experiments. To prevent such outcomes, one could assign a higher weight to the objective of minimizing trim loss—for example, a weight equal to the total demand of all item types. While we tested higher weights, they negatively impacted the computational performance of the model proposed in Cui et al. (2015). Therefore, we opted to use the exact same weights as the original paper to better reflect the model's performance.

In Martin et al. (2022), for the Fiber instances, the authors adopted a bi-objective approach, aiming to identify all efficient solutions. This contrasts with our work, which focuses solely on solutions that achieve minimum trim loss.

5. Additional discussion on item overproduction

Ensuring exact demand fulfillment or allowing surplus in the production of items is an important aspect of the problem, which can lead to different solutions. Notably, a solution that minimizes the number of cutting patterns may produce significantly different quantities of items compared to one that uses a small, though not minimal, number of cutting patterns. Moreover, overproduction generally entails additional costs, such as storage, handling, and potential waste.

However, allowing overproduction of items can be advantageous in practical settings, as it may reduce the number of different cutting patterns required. The number of different cutting patterns is directly related to setup costs, as each new cutting pattern requires readjusting the position of the cutting tools on the machine, which consumes time and raises production costs. Reducing the number of different cutting patterns simplifies production planning and leads to operational efficiencies, including shorter setup times and more streamlined processes.

Both scenarios—restricting overproduction and allowing it—are relevant and should be analyzed. By solving both cases, we can provide a more comprehensive understanding of the trade-offs involved and offer flexible solutions for different production environments. In this section, we present computational experiments to evaluate the effectiveness of our proposed approaches in a scenario that prohibits item overproduction. Furthermore, we compare the solutions obtained in these experiments with those reported

in the main paper, focusing on two key aspects: the number of distinct cutting patterns used and the total amount of overproduced items.

5.1. Results for the Fiber and the CUTGEN1 instances in a no-overproduction scenario

In this section, we conduct computational experiments using Approach 1 and Approach 2 paired with Formulation 1 and Formulation 2, respectively. The modifications presented in Section 5 of the main paper (Remarks on Overproduction) are applied to adapt both approaches for the non-overproduction scenario. In Table 1, we present the results for the Fiber instances considering both approaches in a no-overproduction scenario. The table is divided vertically into instances with lower object lengths and instances with higher object lengths.

We utilize the same notation as in the first part of our computational experiments: *Obj* represents the number of objects used in the PMP solution ($\underline{\beta}$); P_{F_1} and T_{F_1} denote the number of different patterns used and the time required to run Algorithm 3 (an exact algorithm for the PMP with minimum trim loss), presented in the main paper, with Formulation 1, while P_{F_2} and T_{F_2} represent the number of different patterns used and the time required for Approach 2 paired with Formulation 2. Cases where no feasible solution was found are represented by the " $-$ " character, and the number of cutting patterns is formatted in bold or italics to indicate optimality or that the limit of 150,000 cutting patterns is reached in the second approach, respectively. As follows, we present Table 1 and a brief discussion on the results of the computational experiments:

Instance	*Obj*	P_{F1}	T_{F1}	P_{F2}	T_{F2}	Instance	*Obj*	P_{F1}	T_{F1}	P_{F2}	T_{F2}
06_5180	33	**5**	1.40	**5**	1.21	06_9080	19	**3**	0.33	**3**	4.12
07_5180	33	**4**	0.47	**4**	1.48	07_9080	19	**3**	0.08	**3**	2.58
08_5180	86	**4**	0.46	**4**	1.49	08_9080	48	**3**	0.44	**3**	0.95
09_5180	53	**6**	3.00	**6**	0.09	09_9080	29	**4**	0.83	**4**	0.73
10_5180	69	**5**	1.21	**5**	3.21	10_9080	39	**4**	1.06	**4**	5.27
11_5180	67	**5**	0.49	**5**	2.48	11_9080	38	**4**	1.03	**4**	4.06
13a_5180	56	**5**	0.80	**5**	2.79	13a_9080	32	**4**	1.15	**4**	160.47
13b_5180	28	**4**	0.25	**4**	4.50	13b_9080	16	**3**	0.54	-	-
14_5180	47	**8**	176.99	**8**	8.81	14_9080	27	**4**	0.94	*5*	tl
15_5180	57	**5**	0.98	**5**	3.20	15_9080	32	**4**	0.95	**4**	9.79
16_5180	82	**8**	385.37	**8**	218.35	16_9080	47	**5**	9.45	-	-
17_5180	83	**7**	129.22	7	1800.00	17_9080	47	**4**	3.08	*6*	tl
18_5180	96	**7**	67.61	**7**	517.51	18_9080	54	**5**	4.87	-	-
19_5180	133	**7**	85.98	7	1800.00	19_9080	73	7	3114.83	-	-
20_5180	32	**7**	17.94	8	1800.00	20_9080	19	**4**	4.05	-	-
23_5180	141	-	-	**11**	47.41	23_9080	80	**7**	396.90	-	-
26_5180	190	**8**	1515.23	9	1800.00	26_9080	107	**5**	7.38	-	-
28a_5180	83	-	-	11	1800.00	28a_9080	48	**5**	5.17	-	-
28b_5180	117	-	-	**15**	687.77	28b_9080	67	**6**	8.35	-	-
29_5180	62	**7**	5.54	11	1800.00	29_9080	35	**5**	5.76	-	-

Table 1: Number of objects used, number of different cutting patterns employed, and time required to solve the Fiber instances considering the Approaches 1 and 2 together with Formulation 1 and Formulation 2 and the models proposed in Martin et al. (2022) and Cui et al. (2015).

From the results presented in Table 1, it is evident that Approach 1 can be adapted to the no-

overproduction scenario while maintaining strong performance. For the Fiber instances with an object length of 5180, the approach yielded better results in the no-overproduction scenario, likely benefiting from the stronger bound discussed in Section 5 of the main paper (Remarks on Overproduction). In these instances, the approach identified 17 optimal solutions, compared to 11 optimal solutions in the scenario where item overproduction is allowed.

For instances with an object length of 9080, a total of 19 and 18 optimal solutions were identified using Approach 1 in the scenarios without and with overproduction, respectively. However, a minimum trim loss solution was achieved only for instance Fiber29_9080 in the scenario with overproduction.

The strong performance of Approach 2 for the instances with an object length of 5180 is maintained in the no-overproduction scenario. Once again, the approach successfully determined feasible solutions for all instances. Furthermore, it identified 14 optimal solutions in the no-overproduction scenario compared to 13 in the overproduction scenario.

Conversely, Approach 2 struggled with instances where the object length was 9080. For these instances, the approach achieved minimum trim loss for only 9 instances in the no-overproduction scenario, whereas it succeeded for 15 instances in the overproduction scenario.

Following the same notation as in the previous tables, Table 2 presents the results regarding the average values for the number of cutting patterns and the time required by Approach 1 and Approach 2 for the instances generated by the CUTGEN1 generator.

Class	*Obj*	P_{F1}	T_{F1}	P_{F2}	T_{F2}	Class	*Obj*	P_{F1}	T_{F1}	P_{F2}	T_{F2}
1	11.4	**3.2**	1.41	*4.0(2)*	931.64	2	109.4	5.0	590.36	*6.0(2)*	tl
3	23.2	**4.8**	4.23	-	-	4	225.8	7.0(4)	2193.36	-	-
5	42.4	**7.0**	30.46	-	-	6	420.8	-	-	-	-
7	51.2	**7.4**	3.72	**7.4**	0.47	8	510.0	9.25(4)	2426.74	**8.6**	91.17
9	100.4	14.0(4)	1455.40	**13.6**	262.78	10	1000.8	-	-	16.4	770.07
11	175.4	-	-	*24(1)*	tl	12	1756.0	-	-	*34(1)*	tl
13	62.8	**8.8**	2.49	**8.8**	0.04	14	625.4	9.33(3)	986.249	**9.8**	0.10
15	125.2	15.6	1067.18	**15.6**	0.34	16	1251.2	-	-	**18.40**	34.43
17	221.4	29(1)	3364.58	**27.8**	12.06	18	2213.6	-	-	33.0	1578.60

Table 2: Number of objects used, number of different cutting patterns employed, and time required to solve the instances generated via the CUTGEN1 generator considering the Approaches 1 and 2 together with Formulation 1 and Formulation 2 and the models proposed in Martin et al. (2022) and Cui et al. (2015).

Both approaches performed better in the scenario with overproduction of items. In the scenario with overproduction, Approach 1 identified 56 feasible solutions, including 44 optimal ones, compared to 51 feasible solutions and 39 optimal ones in the scenario without overproduction. Similarly, Approach 2 demonstrated weaker performance in the scenario without overproduction, identifying 56 feasible solutions, of which 44 were optimal. In contrast, it achieved 62 feasible solutions, including 52 optimal ones, when overproduction was allowed.

5.2. Analysis of the trade-off between number of different cutting patterns and overproduction of items

The results obtained via Formulation 1 indicate a reasonable trade-off between item overproduction and setup costs. For the 16 instances where a minimum total trim loss was determined for Fiber 5180, the average overproduction using Approach 1 paired with Formulation 1 was 0.28%, with values ranging from

0% to 1.64%. Similarly, for the 20 instances with object length of 9080, the average overproduction using the same approach and formulation was 1.07%, with values ranging from 0% to 5.46%.

For the 56 instances generated by CUTGEN1 where a minimum trim loss was achieved, the average overproduction using Approach 1 paired with Formulation 1 was 12.93%, with values ranging from 0.6% to 20.63%.

In terms of the number of cutting patterns, for the 11 instances where optimality was confirmed by the solver for the fiber instances with object length of 5180, there was an increase of 4.16%. Additionally, considering the 18 fiber instances with object length of 9080, there was an increase of 5.63%.

Moreover, for the CUTGEN1 classes where optimality was achieved using Formulation 1 in both scenarios (Classes 1, 3, 7, and 13), there was an increase in the number of different cutting patterns by 0%, 14.3%, 19.35%, and 12.82%, respectively, averaging 11.62%.

While Approach 1 demonstrates a reasonable balance between item overproduction and the reduction in setup costs—suggesting a fair trade-off between the two scenarios addressed—Approach 2 shows unsatisfactory results in terms of item overproduction.

As mentioned in Section 4.1 of the main paper (Stage 1: Cutting pattern generation), the reduction criterion (vi) may result in excessive levels of overproduction, which is reflected in the results obtained by Approach 2. When combined with criterion (v), these reduction criteria lead to cutting patterns that disproportionately produce the item type with the smallest length, resulting in extreme surplus levels for this item type.

For example, in Class 7, the mean overproduction value reached an average of 115.40% for items, which is clearly unacceptable. Therefore, postprocessing strategies are necessary to mitigate the overproduction caused by Approach 2.

To address this issue, we propose a postprocessing strategy that adapts the model proposed in Martin et al. (2022) to handle item overproduction. A solution for the PMP consists of a set of cutting patterns K and their corresponding execution frequencies. Thus, in a feasible solution, let f_k represent the frequency of execution of cutting pattern $k \in K$.

The core idea of the postprocessing strategy is to use the model proposed in Martin et al. (2022), considering that $\mathcal{K}$ cutting patterns are executed, each with a fixed frequency equal to f_k, for $k \in K$. Additionally, we introduce a set of variables s_i to account for the overproduction of item type i. The objective of the model is to minimize item overproduction. The model used in the postprocessing stage is presented as follows:

$$\min \sum_{i \in I} s_i \tag{1}$$

$$\sum_{i=1}^{|I|} \ell_i x_{ki} \leq L, \qquad k \in K, \tag{2}$$

$$\sum_{k \in K} f_k x_{ki} = d_i + s_i, \qquad i \in I, \tag{3}$$

$$x_{ki} \in Z^+, \qquad k \in K, i \in I, \tag{4}$$

$$s_i \in Z^+, \qquad i \in I. \tag{5}$$

The objective function (1) corresponds to minimizing the overproduction of items. Constraints (2) ensure the feasibility of the cutting patterns, noting that every cutting pattern is executed with a frequency greater than zero. For $i \in I$, constraints (3) ensure that the total production of each item type equals the sum of its demand and overproduction, thereby satisfying the demand requirements and accurately accounting for overproduction. The remaining constraints define the domain of the variables.

It is important to note that a solution for model (1)-(5) corresponds to a PMP solution that uses the same number of objects and different cutting patterns as the original input solution. However, the new solution minimizes the overproduction of items effectively.

We implemented the postprocessing strategy with a time limit of 50 seconds to solve model (1)-(5). The time limit was reached in only one instance; nevertheless, the postprocessing strategy proved to be highly effective in reducing item overproduction. For the CUTGEN1 classes where optimality was achieved by Approach 2 in both scenarios (Classes 7, 8, 9, 13, 14, 15, 16, and 17), the initial overproduction percentages were 115.40%, 127.79%, 95.60%, 20.60%, 5.42%, 25.20%, 25.00%, and 10.95%, respectively. After applying the postprocessing strategy, these values were significantly reduced to 10.00%, 11.60%, 4.30%, 5.80%, 19.90%, 4.20%, 7.45%, and 0.80%, respectively.

Regarding the number of different cutting patterns, there was an average increase of 16.52% for the CUTGEN1 classes where optimality was achieved by Approach 2 in both scenarios. Moreover, for the Fiber instances where optimality was achieved in both scenarios, the average increase in the number of different cutting patterns was 2.67% for instances with an object length of 5180 and 11.54% for instances with an object length of 9080.

The results presented highlight the relationship between the overproduction of items and the number of different cutting patterns. Overall, there is a balanced trade-off between these two factors. Therefore, it is essential to carefully assess which scenario should be implemented in practical applications, considering the specific characteristics of the situation to align with the interests of the decision-makers.

Additionally, our results suggest that cutting pattern criteria should be used with caution, as they may lead to high levels of overproduction. In such cases, postprocessing strategies become imperative to reduce the overproduction of items and should be implemented alongside the proposed approaches.

Minimizing the overproduction of items can also be viewed as another objective of the problem. In this regard, both approaches proposed in this study could be adapted to adopt a multi-objective framework, where both the number of cutting patterns and item type surplus are minimized simultaneously. This would allow for the generation of a Pareto front for each instance, offering multiple efficient solutions that address the trade-off comprehensively. Alternatively, a simpler approach could involve penalizing overproduction in the objective functions of Formulation 1 and Formulation 2, providing a solution that balances setup costs and item overproduction.

It is important to note that while these solution approaches are relatively straightforward to implement, they significantly increase the problem's complexity, making instances even more challenging to solve. For interested readers, supplementary material is provided to demonstrate how the proposed approaches can be adapted to incorporate overproduction penalties in the objective functions or to construct Pareto fronts for multi-objective optimization.

6. Alternative approaches to balancing the number of different cutting patterns and overproduction

Both approaches proposed in this paper can be adapted to either include the overproduction of item types as an additional objective or penalize overproduction within the objective function. To achieve this, we can adopt the same strategy as the postprocessing approach presented in Section 7.2, which introduces a set of variables, s_i, to account for the overproduction of items i. In this framework, inequalities (4) and (24) are modified as follows:

$$\sum_{k \in K} \sum_{f \in F_k \setminus \{0\}} j x_{kfi} = d_i + s_i, \quad \forall i \in I,$$

$$\sum_{s \in R} \alpha_i^s r_s + \sum_{u \in Q} \alpha_i^u q_u = d_i + s_i, \quad \forall i \in I.$$

To explicitly address item overproduction as an additional objective, one can augment Formulation 1 and Formulation 2 with the objective of minimizing $\sum_{i \in I} s_i$. The resulting multi-objective problem can then be solved using conventional multi-objective optimization methods.

Alternatively, to address overproduction using a penalizing strategy, the term $\sum_{i \in I} s_i$, multiplied by a small weight, can be added to the original objective functions of Formulation 1 and Formulation 2. This modification prioritizes minimizing the number of cutting patterns while simultaneously reducing item overproduction as a secondary objective.

References

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.

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 2022;44:557—582. doi:10.1007/s10878-022-00848-z.