

Pattern-based ILP models for the one-dimensional cutting stock problem with setup cost

Mateus Martin^{a,b,*}, Horacio Hideki Yanasse^b, Luiz Leduino Salles Neto^b

^a*Departamento de Engenharia de Produção, Universidade Federal Fluminense*

^b*Instituto de Ciência e Tecnologia, Universidade Federal de São Paulo*

Abstract

The One-Dimensional Cutting Stock Problem with Setup Cost (CSP-S) is a cutting problem that seeks a cutting plan with a minimum number of objects and a minimum number of different patterns. This problem gains relevance in manufacturing settings, where time consuming operations to set up the knives of the cutting machine for the new patterns increases production costs. In this paper, we aim at solving the bi-objective CSP-S that analyzes the trade-offs between the number of objects and the number of patterns. We first derive an upper bound on the maximum frequency of a pattern in the cutting plan. Then, we propose a pattern-based pseudo-polynomial Integer Linear Programming (ILP) formulation for the CSP-S. To obtain the Pareto optimal frontier, this formulation is embedded into a straightforward framework which solves the problem of minimizing the number of objects subject to a limited number of patterns in an iterative manner. Since we are not aware of other approaches in the literature that have solved the bi-objective CSP-S exactly, we derive an ILP formulation based on Harjunkoski et al. (1996) into this framework to provide an alternative exact approach. The results of the computational experiments using a general-purpose ILP solver indicated that the approaches are proper for instances with solutions characterized by a moderate number of objects and a few patterns in the Pareto optimal frontier.

Keywords: Cutting & Packing, Cutting stock problem, Setup cost, Integer linear programming, Multi-objective optimization.

1. Introduction

The One-Dimensional Cutting Stock Problem with Setup Cost (CSP-S) addresses a set of objects of length L and a set $I = \{1, \dots, M\}$ of item types. Each item type $i \in I$ has length l_i and demand d_i . The objective is to minimize both the number of objects used to manufacture all the demanded items and the number of (different) patterns. It is known that the CSP-S is strongly NP-Hard (Garey & Johnson, 1979; McDiarmid, 1999). According to the typology of Wäscher et al. (2007) for Cutting & Packing problems, the CSP-S is classified as a variant of the One-dimensional Single Stock Size Cutting Stock Problem (CSP), since it additionally considers how to reduce the

*Corresponding author. Departamento de Engenharia de Produção, Escola de Engenharia de Petrópolis, Universidade Federal Fluminense, Rua Domingos Silvério 135, 25650-050, Petrópolis-RJ, Brazil.

Email addresses: `mpmartin@id.uff.br` (Mateus Martin), `horacio.yanasse@unifesp.br` (Horacio Hideki Yanasse), `luiz.leduino@unifesp.br` (Luiz Leduino Salles Neto)

number of patterns. A feasible solution to the CSP-S can be seen as a cutting plan, that is, a set of patterns and their frequencies, where:

- (i) a pattern is a M -dimensional non-negative integer vector $x = (x_1, x_2, \dots, x_M)^\top$, where x_i represents how many items of type $i \in I$ are cut from a single object, that satisfies the inequality $\sum_{i \in I} l_i x_i \leq L$ (containment constraint);
- (ii) the frequency of a pattern represents the number of objects that has to be applied with the pattern;
- (iii) the demand of each item type $i \in I$ should be fulfilled exactly or in excess.

Real applications of the CSP-S include the cutting of objects in the paper, steel, glass and plastic film industries to manufacture items. In this context, the production is stopped to set up the knives of the cutting machine for each new pattern (Haessler, 1975). The stages of production following the cutting machine tend to be impacted as well. Thus, in addition to the main objective of minimum waste solutions, solutions with a minimum number of patterns are also desired. A typical solution to the problem usually has a small number of patterns with high frequencies. From a practical perspective, a satisfactory approach should tackle the bi-objective CSP-S to provide a set of solutions that find the best trade-off between the number of objects and the number of patterns (Umetani et al., 2003). This approach could be embedded into an expert system to assist decision-making, and it would avoid determining the setup cost explicitly.

1.1. Related work

The CSP-S is mainly addressed by heuristics, commonly based on pattern generation or meta-heuristics with local search (Henn & Wäscher, 2013). The Sequential Heuristic Procedure (SHP) of Haessler (1975) was one of the first approaches for the CSP-S. A SHP is a greedy algorithm that seeks to generate a few patterns with high frequencies sequentially to satisfy the demands of some item types until complete exhaustion of the demand. It usually requires short processing times, and has parameters to guide the search in the trade-off between the number of objects and the number of patterns. Yanasse & Limeira (2006), Cui & Liu (2011), Mobasher & Ekici (2013), Cui et al. (2015) and Martin et al. (2018) proposed other SHP-based approaches for the CSP-S. These approaches differ mainly regarding how to generate feasible patterns, and the use of (post-)optimization techniques to reduce the number of patterns further. For instance, one could combine two patterns into a single new pattern heuristically, where the frequency of the new pattern is equal to the sum of the frequencies of the two previous patterns (Diegel et al., 1993). Foerster & Wäscher (2000) developed the KOMBI method by combining p patterns into new $p - 1$ patterns, where $p \leq 4$.

As for meta-heuristics algorithms, Umetani et al. (2003) developed an Iterated Local Search (ILS) algorithm for the CSP-S that searches for solutions with a constant number of p patterns. Constant p is varied in the algorithm to obtain the more advantageous solution within an adaptive pattern generation technique used in the local search. Later Umetani et al. (2006) further improved the previous work by proposing two types of local search based on the 1-add neighborhood and the shift neighborhood, and an auxiliary linear program to reduce the size of the neighborhood. Lee (2007) developed the Crawla algorithm for the bi-objective CSP-S which can be seen as a large large neighborhood search (Henn & Wäscher, 2013). This iterative algorithm uses an Integer Linear Programming (ILP) framework for generating and fixing new patterns in the solution. Golfeto et al.

(2009) and Araujo et al. (2014) also developed meta-heuristics based on evolutionary algorithms with local search for the bi-objective CSP-S. More recently, Aliano Filho et al. (2018) presented an interesting study on four scalarization methods for the bi-objective CSP-S in which the patterns are given as input. They applied the weighted sum method, the classical Chebyshev metric, a novel modified Chebyshev metric and the ε -Constraint method.

A problem related to the CSP-S is the Pattern Minimization Problem (PMP) that consists of minimizing the number of patterns subject to a limited number of objects. We are aware of three exact approaches that addressed the PMP by proposing branch-price-and-cut algorithms that fulfilled the demand of items without surplus (Vanderbeck, 2000; Belov & Scheithauer, 2003; Alves & Valério De Carvalho, 2008). Vanderbeck (2000) proposed a master problem, where each column represents a feasible pattern associated with a possible frequency. His column generation procedure relied on solving a quadratic integer sub-problem that was copied by solving a set of bounded knapsacks for each possible value of frequency for each pattern. The author proposed superadditive functions to strengthen the model. Alves & Valério De Carvalho (2008) further improved the previous algorithm with the use of an upper bound on the number of maximum frequencies of a pattern and of dual feasible functions. To derive a robust branching scheme and stronger cuts, the authors also proposed an arc flow formulation with integer variables x_{rs}^n that represent the flow over arcs (r, s) (discrete positions within the object) in a pattern of frequency n . They also showed how the use of an upper bound on trim loss can reduce the maximum number of frequencies of a pattern. Belov & Scheithauer (2003) treated an extension of the Gilmore & Gomory (1961)'s model that was discussed in Vanderbeck (2000) as a master problem, which leads to fewer number of variables, however with weaker LP-bounds. The author seek to overcome this behavior with a branching scheme based on variables and enforcing branching constraints on single fractional variables. In a short, better results was obtained by the algorithm of Alves & Valério De Carvalho (2008) considering a set of 16 real-life instances. All these approaches had to tackle weaker lower bounds in the column generation procedure due to the structure of the PMP, which differs consistently from the tight bounds obtained by the approach of Gilmore & Gomory (1961) for the CSP. In this context, Alves et al. (2009) developed more tight lower bounds for the PMP which were obtained by solving a column generation framework. They strengthened these lower bounds with a constraint programming approach and valid inequalities.

Lastly, we note the ILP formulation of Johnston & Sadinlija (2004) for the CSP that can easily address the CSP-S. The authors proposed binary variables x_{ijk} that are equal to 1, if k copies of item type i appear in the pattern j , and 0 otherwise. To our knowledge, this model has not been specifically assessed with computational experiments in the context of the bi-objective CSP-S.

1.2. Our contributions

In this paper, we propose a pattern-based pseudo-polynomial ILP formulation for the CSP-S. It relies on the development of an upper bound on the maximum frequency of each pattern in the cutting plan. The formulation is related to the master problem proposed in Vanderbeck (2000) in the sense that it considers a binary variable for each pair of a pattern and a possible frequency. Then, we incorporate it into a straightforward framework to solve the bi-objective CSP-S that minimizes the number of objects subject to a limited number of patterns. Since we are not aware of other exact approaches for the problem, we derive an ILP formulation based on Harjunkoski et al. (1996) into this framework to provide an alternative approach. We performed computational experiments with two well-known sets of benchmark instances from the literature to evaluate the advantages/disadvantages of the approaches. The results indicated that the proposed

models using a general-purpose ILP solver are able to obtain Pareto (near-)optimal frontier for small and moderate-sized problem instances. They are appropriate for instances with solutions characterized by a moderate number of objects and a few patterns in the Pareto optimal frontier. We presume that these models can benefit OR practitioners interested in solving the CSP-S with a general-purpose ILP solver as a black box.

1.3. Organization of the paper

The rest of the paper is organized as follows. In Section 2, we describe the CSP-S with a bi-objective non-linear model. We discuss how to eliminate symmetries through an upper bound on the maximum frequency of each pattern in the cutting plan, as well as an upper bound on the number of items per type in a pattern. In Section 3, we introduce an ILP formulation for the CSP-S that is based on the non-linear model, which consists in modeling the frequency of each pattern as a binary expansion. Then, we propose a novel pseudo-polynomial ILP formulation for the CSP-S that considers each pair of a pattern and a possible frequency as a binary variable. In Section 4, we develop a straightforward framework that can use either of the two proposed models to solve the bi-objective CSP-S that minimizes the number of objects subject to a limited number of patterns. In Section 5, the performance of the proposed models derived from this framework is analyzed through computational experiments using two sets of benchmark instances from the literature. We also present a brief discussion of the ILP formulations of Johnston & Sadinlija (2004) and Alves & Valério De Carvalho (2008) with some additional results. Final remarks and perspectives of future research are discussed in Section 6.

2. A non-linear model for the CSP-S

The CSP-S is a cutting problem that seeks a cutting plan with a minimum number of objects used to manufacture all the demanded items and a minimum number of patterns. The problem arises in manufacturing settings with a limited capacity in the stage of cutting operations. That is the case of production systems that improve the productivity of the cutting machine (e.g. meters cut per hour) with fewer setups (i.e. cutting plans with fewer patterns) at the expense of increasing the trim loss. Thus, it is commonly assumed that the number of setups is equal to the number of different patterns. Note that limited capacity is not an issue for most of the cutting problems, like in the approach of Cerqueira et al. (2021) for the CSP, since they seek mainly solutions with minimum waste for input minimization problems or maximum profit for output maximization problems (Wäscher et al., 2007). However, these solutions could even not be feasible in the context mentioned above. We define the following parameters and sets to be used in the presentation of the models:

- L is the length of the objects;
- $I = \{1, \dots, M\}$ is the set of item types;
- l_i and d_i are the length and demand of item type $i \in I$, respectively; and
- $K = \{1, \dots, N\}$ is the set of patterns.

We assume that $l_i \leq L$ and d_i , $i \in I$, are positive integers. Parameter N of the maximum number of patterns can be obtained by solving a heuristic for the CSP without considering the setup cost or by a previous definition of the decision-maker. We next formulate the CSP-S as a Bi-objective Integer Non-linear Programming model, inspired in a pattern-based approach (Kantorovich, 1960; Scheithauer, 2018). The formulation is given by Model (1), where the patterns are not given *a priori*.

$$\text{Minimize } \sum_{k \in K} z_k. \quad (1a)$$

$$\text{Minimize } \sum_{k \in K} y_k. \quad (1b)$$

subject to

$$\sum_{i \in I} l_i x_{ki} \leq L y_k, \quad k \in K. \quad (1c)$$

$$\sum_{i \in I} z_k x_{ki} \geq d_i, \quad k \in K. \quad (1d)$$

$$z_k \leq M^k y_k, \quad k \in K. \quad (1e)$$

$$x_{ki} \in \mathbb{Z}_+, \quad k \in K, i \in I. \quad (1f)$$

$$z_k \in \mathbb{Z}_+, y_k \in \{0, 1\}, \quad k \in K, \quad (1g)$$

where

x_{ki} is an integer variable that represents how many items of type $i \in I$ are produced when cutting one object with pattern $k \in K$;

z_k is an integer variable that represents the frequency of pattern $k \in K$, i.e., the number of objects that have to be cut with this pattern; and

y_k is a binary variable that equals (1), if pattern $k \in K$ is used, and (0) otherwise.

Objective function (1a) minimizes the number of objects, and objective function (1b) minimizes the number of patterns. These are conflicting objectives, since improving one usually is to the detriment of the other (Aliano Filho et al., 2018). Constraints (1c) ensure that the sum of the length l_i of the items does not exceed the length L of the object, for each pattern $k \in K$. These constraints also enforce that $x_{ki} = 0$ for each $i \in I$, when $y_k = 0$. Constraints (1d) ensure that the demand of each item type $i \in I$ is fulfilled exactly or in excess. Constraints (1e) enforce that $y_k = 1$, when $z_k > 0$ for each $k \in K$. Constraints (1f) and (1g) define the domain of the variables. Model (1) is non-linear and non-convex due to the bilinear terms in the left-hand side of constraints (1d). Note that we can reformulate Model (1) as a mono-objective ILP formulation by replacing only objective functions (1a) and (1b) with the following objective function:

$$\text{Minimize } c_o \sum_{k \in K} z_k + c_p \sum_{k \in K} y_k,$$

that minimizes the total cost of objects and patterns, where c_o and c_p are the coefficients that represent the unitary cost of a object and of a pattern, respectively.

2.1. Eliminating symmetries or equivalent solutions

The pattern-based formulations for the CSP and its variants present the disadvantage of the symmetries on the patterns, that is, the interchangeability of indices $k \in K$, which may lead to difficulties in a branch-and-bound procedure (Vanderbeck, 2000). One strategy to partially circumvent the disadvantage is to introduce the valid inequalities $z_k \geq z_{k+1}$, $k \in K \setminus \{N\}$. These valid inequalities ensure that the patterns in the cutting plan are sorted in a non-increasing order

of the frequencies of the patterns (Johnston & Sadinlija, 2004). To attenuate other equivalent solutions, we define the following parameters to focus on the domain of variables z_k and x_{ki} next:

\bar{z}	is the upper bound on the number of objects in the solution;
\underline{y}	is the lower bound on the number of patterns in the solution;
$d_{max} = \max_{i \in I} \{d_i\}$	is the maximum demand of the item types;
M^k	is the upper bound on the frequency of pattern $k \in K$; and
O^i	is the upper bound on the number of items of type $i \in I$ in any pattern.

In Theorem 1 we state an upper bound on each variable z_k , $k \in K$.

Theorem 1. *An upper bound on each variable z_k , $k \in K$, in the cutting plan is given by $M^k = \min\{\lfloor (\bar{z} - \max\{0, \underline{y} - k\})/k \rfloor, d_{max}\}$.*

Proof. We assume that valid inequalities $z_k \geq z_{k+1}$, $k \in K \setminus \{N\}$, hold. It is easy to see that $z_1 \leq \min\{\bar{z}, d_{max}\}$, since the frequency of a pattern in an optimal solution cannot be greater than the total number of objects in the solution, nor the maximum demand of any item type. Recall that $z_1 \geq z_2$ and $z_1 + z_2 \leq \bar{z}$. Therefore, $z_2 + z_2 \leq z_1 + z_2 \leq \bar{z}$. Then $2z_2 \leq \bar{z}$, and so $z_2 \leq \lfloor \bar{z}/2 \rfloor$. More precisely, $z_2 \leq \min\{\lfloor \bar{z}/2 \rfloor, d_{max}\}$. Using a similar reasoning by induction, we can derive $z_3 \leq \min\{\lfloor \bar{z}/3 \rfloor, d_{max}\}$. In a general form, $z_k \leq \min\{\lfloor \bar{z}/k \rfloor, d_{max}\}$ for each $k \in K$. Considering that any solution has at least \underline{y} patterns, then $z_k \leq \min\{\lfloor (\bar{z} - \max\{0, \underline{y} - k\})/k \rfloor, d_{max}\}$ for each $k \in K$. \square

An upper bound on each variable x_{ki} , $k \in K$ and $i \in I$, is given by $O^i = \min\{\lfloor L/l_i \rfloor, d_i\}$. Alternatively stated, it is the minimum value between the geometric bound $\lfloor L/l_i \rfloor$ and the demand d_i of item type $i \in I$ (Nitsche et al., 1999).

3. Two pattern-based ILP formulations for the CSP-S

In this section, we introduce two pattern-based ILP formulations for the CSP-S. The first is based on the non-linear model of the previous section, and it consists of representing variables z_k as a binary expansion. The second is a novel pseudo-polynomial formulation that considers each possible combination of pattern and frequency as a binary variable. These two formulations make use of the upper bounds discussed in the previous section, especially upper bound M^k on the frequency of each pattern $k \in K$, since it determines their number of variables and constraints.

3.1. Representing the frequency of each pattern as a binary expansion

The bilinear terms in the left-hand side of constraints (1d) can be circumvented in the following way: let x_{ki} and y_k be as defined in the previous section, and let $z_k = \sum_{j \in J_k} 2^j z_{kj}$, where:

$J_k = \{0, 1, \dots, 1 + \lfloor \log_2(M^k) \rfloor\}$	is the set of bits to represent the frequency of pattern $k \in K$; and
$z_{kj} \in \{0, 1\}$	is the binary variable for bit $j \in J_k$ of pattern $k \in K$.

The bilinear constraints (1d) are reformulated as:

$$\sum_{k \in K} \sum_{j \in J_k} 2^j z_{kj} x_{ki} \geq d_i, \quad i \in I,$$

which can be replaced by a set of disjunctive inequalities that expresses the terms $w_{kij} = z_{kj}x_{ki}$, $k \in K$, $i \in I$, and $j \in J_k$ in a linear form (Harjunkoski et al., 1996). Having defined all the parameters and variables, we formulate a Bi-objective Integer Linear Programming formulation for the CSP-S in Model (2).

$$\text{Minimize } \sum_{k \in K} \sum_{j \in J_k} 2^j z_{kj}. \quad (2a)$$

$$\text{Minimize } \sum_{k \in K} y_k. \quad (2b)$$

subject to

$$\sum_{i \in I} l_i x_{ki} \leq Ly_k, \quad k \in K. \quad (2c)$$

$$\sum_{k \in K} \sum_{j \in J_k} 2^j w_{kji} \geq d_i, \quad i \in I. \quad (2d)$$

$$\sum_{j \in J_k} 2^j z_{kj} \leq M^k y_k, \quad k \in K. \quad (2e)$$

$$\sum_{j \in J_k} 2^j z_{kj} \geq \sum_{j \in J_{k+1}} 2^j z_{(k+1)j}, \quad k \in K \setminus \{N\}. \quad (2f)$$

$$w_{kij} \leq x_{ki}, \quad k \in K, i \in I, j \in J_k. \quad (2g)$$

$$\sum_{i \in I} w_{kij} \leq \sum_{i \in I} O^i z_{kj}, \quad k \in K, i \in I. \quad (2h)$$

$$w_{kij} \geq x_{ki} - O^i(1 - z_{kj}), \quad k \in K, i \in I, j \in J_k. \quad (2i)$$

$$x_{ki} \in \mathbb{Z}_+, x_{ki} \leq O^i, \quad k \in K, i \in I. \quad (2j)$$

$$z_{kj} \in \{0, 1\}, \quad k \in K, j \in J_k. \quad (2k)$$

$$y_k \in \{0, 1\}, \quad k \in K. \quad (2l)$$

$$0 \leq w_{kij} \leq O^i, \quad k \in K, i \in I, j \in J_k. \quad (2m)$$

Objective function (2a) minimizes the number of objects, and objective function (2b) minimizes the number of patterns. Constraints (2c) ensure that the sum of the length l_i of the items does not exceed the length L of the object, for each pattern $k \in K$. Constraints (2d) ensure that the demand of each item type $i \in I$ is fulfilled exactly or in excess. Constraints (2e) enforce that $y_k = 1$ when the frequency of pattern $k \in K$ is not zero (i.e., $\sum_{j \in J_k} 2^j z_{kj} > 0$). Valid inequalities (2f) ensure that the patterns are sorted in a non-increasing order of their frequencies. Constraints (2g), (2h), and (2i) are the set of disjunctive linear inequalities that formulates the expression $w_{kij} = z_{kj}x_{ki}$, $k \in K$, $i \in I$, and $j \in J_k$. In other words, these constraints enforce that $w_{kij} = 0$ when $z_{kj} = 0$, and $w_{kij} = x_{ki}$ when $z_{kj} = 1$. Constraints (2j) to (2m) define the domain of the variables.

3.2. A novel pseudo-polynomial ILP formulation

In this section, we present an alternative modeling approach that links the pattern and its frequency in a single decision variable. Then we make use of several knapsacks represented by disjunctive inequalities to guarantee the feasibility of the patterns. Thus, we now redefine sets J_k to be:

$J_k = \{0, 1, \dots, M^k\}$ is the set of possible frequencies of pattern $k \in K$.

The two sets of variables of the proposed model are as follows:

λ_{kj} binary variable that equals (1), if pattern $k \in K$ has frequency $j \in J_k$, and (0), otherwise;

α_{kji} integer variable that represents the number of items of type $i \in I$ that are produced when cutting one object with pattern $k \in K$ that has frequency $j \in J_k \setminus \{0\}$.

Having defined all the parameters and variables, we formulate a novel Bi-objective Integer Linear Programming formulation for the CSP-S in Model (3).

$$\text{Minimize } \sum_{k \in K} \sum_{j \in J_k} j \lambda_{kj}. \quad (3a)$$

$$\text{Minimize } \sum_{k \in K} \sum_{j \in J_k} \lambda_{kj}. \quad (3b)$$

subject to

$$\sum_{i \in I} l_i \alpha_{kji} \leq L \lambda_{kj}, \quad k \in K, j \in J_k \setminus \{0\}. \quad (3c)$$

$$\sum_{k \in K} \sum_{j \in J_k \setminus \{0\}} j \alpha_{kji} \geq d_i, \quad i \in I. \quad (3d)$$

$$\sum_{j \in J_k} \lambda_{kj} = 1, \quad k \in K. \quad (3e)$$

$$\sum_{j \in J_k} j \lambda_{kj} \geq \sum_{j \in J_{k+1}} j \lambda_{(k+1)j}, \quad k \in K \setminus \{N\}. \quad (3f)$$

$$\lambda_{kj} \in \{0, 1\}, \quad k \in K, j \in J_k. \quad (3g)$$

$$\alpha_{kji} \in \mathbb{Z}_+, \alpha_{kji} \leq O^i, \quad k \in K, j \in J_k \setminus \{0\}, i \in I. \quad (3h)$$

Objective function (3a) minimizes the number of objects, and objective function (3b) minimizes the number of patterns. Constraints (3c) ensure that the sum of the length l_i of the items does not exceed the length L of the object for each pattern $k \in K$ of frequency $j \in J_k$. Note that these constraints are also disjunctive inequalities that enforce $\sum_{i \in I} \alpha_{kji} = 0$, when $\lambda_{kj} = 0$, for each $k \in K$ and $j \in J_k \setminus \{0\}$. Constraints (3d) ensure that the demand of each item type $i \in I$ is fulfilled exactly or in excess. Constraints (3e) ensure that each pattern $k \in K$ has only one frequency $j \in J_k$ (sets J_k include the value of zero). Valid inequalities (3f) ensure that the patterns are sorted in a non-increasing order of their frequencies. Constraints (3g) and (3h) define the domain of the variables. We chose to not define variables α_{kji} when $j = 0$, because these variables would not have a contribution in the left-hand side of constraints (3d).

In Table 1, we present the number of variables and constraints of Models (2) and (3) in terms of the problem instance and sets J_k .

We highlight the importance of the proposed upper bound M^k used to define sets J_k in order to obtain a model with fewer variables and constraints, which is usually helpful to accelerate the convergence of the branch-and-bound procedure in the context of an ILP solver. This is in contrast, for example, with Harjunkoski et al. (1996) who when addressing a mono-objective optimization problem, suggested the constant $\bar{M}^k = \sum_{i \in I} d_i$, $k \in K$. Note that the parameters M and N also

Table 1: Number of variables and constraints of the ILP models.

Model*	Number of variables	Number of constraints
Model (2)	$N \sum_{k \in K} J_k (M+1) + N(M+1)$	$2MN \sum_{k \in K} J_k + MN + 3N + M - 1$
Model (3)	$N \sum_{k \in K} J_k (M+1) - MN$	$N \sum_{k \in K} J_k + M + N - 1$

*: the definition of sets J_k is different in Models (2) and (3).

affect the number of variables and constraints of Models (2) and (3), and so they are analyzed in the next section.

4. A framework for generating the Pareto optimal frontier

We develop a straightforward framework that makes use of either of the two models proposed in the previous section to generate the Pareto optimal frontier for the CSP-S. For this purpose, we first discuss a simple preprocessing operation. Let $E = (M, l = (l_1, \dots, l_M)^\top, d = (d_1, \dots, d_M)^\top, L)$ be a problem instance of the CSP-S. We seek to obtain an equivalent instance by modifying single input data as follows:

- (P1). Two item types $a \in I$ and $b \in I$, $a < b$, with the same length (i.e., $l_a = l_b$) are aggregated. In other terms, we set $d_a := d_a + d_b$, and then item type b is removed from set I .

Other authors also considered using (P1) before solving the problem instance (Lee, 2007; Golfeto et al., 2009; Araujo et al., 2014). At first, (P1) is relevant to reduce the number M of (different) item types. In addition, we highlight its relevance in Theorem 2 to avoid loss of optimality.

Theorem 2. *A problem instance $E = (M, l = (l_1, \dots, l_M)^\top, d = (d_1, \dots, d_M)^\top, L)$ with two or more item types of the same length considered as different item types may lead to loss of optimality.*

Proof. Consider a problem instance $E = (2, l = (l_1, l_2)^\top, d = (d_1, d_2)^\top, L)$, where $L/2 < l_a = l_b \leq L$. For this non-aggregated instance, the Pareto optimal frontier has a single cutting plan with two patterns, that is, a first pattern with a single item of type 1 (frequency of d_1), and a second pattern with a single item of type 2 (frequency of d_2). However, if we apply (P1), then we obtain an equivalent instance $\bar{E} = (1, l = (l_1)^\top, d = (d_1 + d_2)^\top, L)$ that reaches an optimal cutting plan of one pattern with a single item of type 1 (frequency of $d_1 + d_2$). \square

We focus now on the two nadir vectors, which are the two solutions in the Pareto optimal frontier characterized by having the minimum value in one objective function and the maximum value for the other objective function. Thus, let:

- z^* minimum value on the number of objects, which we obtain by solving the arc-flow model of Valério de Carvalho (1999, 2002) for the CSP;
- y^* minimum value on the number of patterns, which we obtain by solving the associated Bin Packing Problem (BPP), where the demand of all item types $i \in I$ is set to $d'_i = 1$.

Notice that we also obtain parameter N as a result of solving the arc-flow model and post-processing the solution into a cutting plan. And, we also obtain parameter \bar{z} as a result of solving the associated BPP, where each bin represents a pattern whose frequency is post-processed to be equal to the highest (original) demand d_i of its packed items.

In Algorithm 1 we present a pseudocode of the developed framework for generating the Pareto optimal frontier for the CSP-S, taking into account Model (3). It is easy to adapt the algorithm to make use of Model (2). Algorithm 1 consists of solving the version of Model (3) that minimizes the number of objects subject to a limited number of patterns in an iterative manner, which starts as $\mathbf{y} := y^*$ patterns and increases by one unit in each iteration. In contrast, the number of objects \mathbf{z} tends to decrease as the algorithm iterates. Note that if a (short) time limit is considered as an additional stopping criterion on the execution of the ILP solver, then there is no guarantee to obtain efficient solutions only. We alter the (real) size of the model to be solved (i.e., the number of variables and constraints) in line 9 of Algorithm 1, since the *presolve* of the ILP solver will consider the variables with the same lower and upper bounds as constants. Thus, we expect the solver to find an optimal solution more quickly in the first iterations of the algorithm. We emphasize that one could limit the solution space of each iteration of the algorithm to exactly \mathbf{y} patterns. However, we chose to limit the solution space to up to \mathbf{y} patterns so that the solution from the previous iteration would also be feasible for the current iteration.

Algorithm 1: A framework for generating the Pareto optimal frontier for the CSP-S.

Input: Instance $E = (M, l = (l_1, \dots, l_M)^\top, d = (d_1, \dots, d_M)^\top, L)$.

Output: Pareto optimal frontier.

- 1 Do preprocessing operation (P1) on instance E ;
 - 2 Set z^* and N by solving the arc-flow model for the CSP;
 - 3 Set y^* and S by solving the associated BPP;
 - 4 Mount Model (3) without objective function (3b);
 - 5 Set $\mathbf{y} := y^*$;
 - 6 Set $\mathbf{z} := S$;
 - 7 **repeat**
 - 8 Restore the original lower and upper bounds of all variables in Model (3);
 - 9 Disable all variables λ_{kj} and α_{kji} , where $k \in K \mid k > \mathbf{y}$ or $j \in J_k \setminus \{0\} \mid j > S$; // i.e.,
 set their lower and upper bounds as zero.
 - 10 Set \mathbf{z} by solving Model (3) without objective function (3b); // and with patterns
 $k \in K \mid k \leq \mathbf{y}$.
 - 11 Save this cutting plan of \mathbf{y} patterns and \mathbf{z} objects;
 - 12 **if** $S > \mathbf{z}$ **then**
 - 13 └ Set $S := \mathbf{z}$;
 - 14 Set $\mathbf{y} := \mathbf{y} + 1$;
 - 15 **until** $\mathbf{z} > z^*$ and $\mathbf{y} \leq N$.
-

Additional preprocessing operations could be done in line 1 of Algorithm 1. For instance, removing an item type $i \in I$ that $l_i + \min_{i \in I} \{l_i\} > L$, or the reduction method to decrease the length L of the object and increase the length l_i of the item types (Scheithauer, 2018). We chose to not consider these preprocessing operations because we also report results on trim loss in Section 5.

4.1. An illustrative example

In Table 2, we provide an illustrative problem instance of the CSP-S, initially proposed in Haessler (1975). In Fig. 1 we illustrate the Pareto optimal frontier of this instance obtained with

Algorithm 1 considering Model (3). In Table 3, we present details on the cutting plan of 6 patterns and 25 objects, where each item type is represented by its length and its number of copies is in parenthesis. For this example, we note that Vahrenkamp (1996) found a cutting plan of 20 patterns and 25 objects, which is essentially different from the solution presented in this table.

Table 2: An illustrative example of the CSP-S with $L = 141,000$ and $m = 27$ item types (Haessler, 1975).

i	l_i	d_i	i	l_i	d_i	i	l_i	d_i
1	54,000	4	10	19,500	10	19	11,625	7
2	52,500	2	11	18,250	2	20	11,250	3
3	47,500	3	12	17,500	16	21	10,125	3
4	26,500	4	13	15,250	6	22	10,000	37
5	25,000	8	14	13,875	2	23	9,250	4
6	24,750	10	15	13,750	2	24	9,125	3
7	23,250	13	16	12,500	28	25	8,750	1
8	22,500	8	17	12,250	5	26	8,500	14
9	20,000	5	18	12,000	10	27	7,000	2

*: this instance originally had the lengths of the objects and item types as fractional numbers; thus, we multiplied them by a factor of one thousand to get integer numbers.

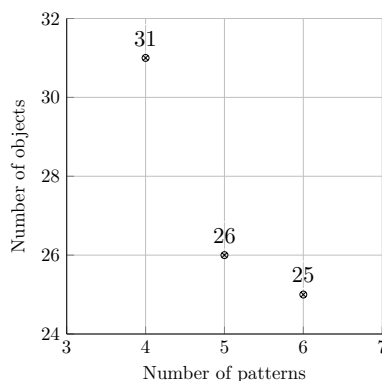


Figure 1: Pareto optimal frontier of the illustrative example defined in Table 2.

Table 3: A cutting plan of 6 patterns and 25 objects for the illustrative example defined in Table 2.

Patterns	Frequencies	Trim loss
24750(1), 23250(1), 19500(1), 17500(1), 12500(2), 10000(3)	10	10000
25000(1), 22500(1), 20000(1), 15250(1), 12250(1), 12000(2), 11625(1), 10000(1)	5	1875
54000(1), 26500(1), 17500(1), 12500(2), 9250(1), 8500(1)	4	1000
47500(1), 23250(1), 22500(1), 11250(1), 10125(1), 9125(1), 8500(2)	3	750
52500(1), 25000(1), 17500(1), 11625(1), 10000(1), 8500(2), 7000(1)	2	750
25000(1), 18250(2), 15250(1), 13875(2), 13750(2), 8750(1)	1	250
Sum	25	14625

We also generated the Pareto optimal frontier of this instance for the case in which the demand of the items should be fulfilled exactly (without surplus), i.e., we changed the sign of constraints (3d) from \geq to $=$. For this new case, the Pareto optimal frontier has a single cutting plan of 6 patterns and 25 objects. In other words, there is no cutting plan with less than 6 patterns that fulfills exactly the demand of the items for this instance. Note that additional patterns could be

required in this new case in order to reach a solution with the minimum number of objects, but this was not the case for this instance.

5. Computational experiments

In this section we present the results of computational experiments performed with the developed framework with Models (2) and (3). This section is divided into three parts. In Section 5.1, we consider 90 problem instances (out of 1800) randomly generated by problem generator CutGen1 (Gau & Wäscher, 1995). We used them to assess the relative performance of the mono-objective versions of the proposed models in order to report information about the number of variables and constraints, the value of the linear relaxation, and, especially the scenarios in which the models perform better or worse. In Section 5.2, we consider 40 problem instances taken from a real fiber company in Japan (Umetani et al., 2003), which were used to assess the performance of the developed framework with the models to generate the Pareto optimal frontier. We considered the Pareto frontiers provided in Umetani et al. (2003) and Lee (2007) as benchmark approaches. In Section 5.3, we present a brief discussion of the ILP formulations of Johnston & Sadinlija (2004) and Alves & Valério De Carvalho (2008) with some additional results. All the instances are available at <https://sites.google.com/site/shunjiumetani/benchmark>, and they are further detailed at the beginning of each section. Our approaches were coded in C++, using the GUROBI v.9.1.1 as the general-purpose ILP solver. The experiments were carried out on a PC with an Intel Xeon E5-2680 (2.7 GHz), limited to 4 threads, 32 GB of RAM, running the Ubuntu 16.04 LTS Operating System.

5.1. Results for CutGen1 instances

Foerster & Wäscher (2000) used the problem generator CutGen1 to randomly generate a set of instances with 18 different classes for the CSP-S. A class is characterized by the number of item types (M), the length of the object $L = 1,000$, the values v_1 and v_2 that determine the length of the items in the interval $[v_1L, v_2L]$, and the average of the demands (d_{av}) for each item type. These classes are divided into three groups: (G1) six classes of small items ($v_1 = 0.01$ and $v_2 = 0.20$), (G2) six classes of assorted items ($v_1 = 0.01$ and $v_2 = 0.80$), and (G3) six classes of large items ($v_1 = 0.20$ and $v_2 = 0.80$). Parameter M assumes the values of 10, 20, and 40 item types, while parameter d_{av} assumes the values of 10 and 100 copies of items. For each class, they generated 100 instances, but we considered here just the first 5 instances generated for each class in a total of 90 ($=18 \times 5$) instances.

We considered the weight sum method with unitary costs c_o and c_p for adapting Models (2) and (3) to obtain mono-objective ILP formulations, which are denoted as M3 and M4, respectively, similar to what was discussed in Section 2. As in Cui et al. (2015), we used $c_o = 1,000$ and $c_p = 100$, which tend to cause a lexicographic hierarchy in the two objective functions. Thus, we carried out lines 1 to 4 in Algorithm 1 with the adaptation in the objective function, and then executed the solver with a time limit of 3,600 seconds. For each class of instances, we report the class identification, the number of item types (M), the values of v_1 and v_2 and the average demand (d_{av}) in Table 4. For M3 and M4 and each class of instances, we also report the number of variables (var), number of constraints (cons), relative optimality gap in percentage (gap[%]), processing time in seconds (time[s]), value of the linear relaxation (LR), and number of instances in the class with certificate of optimality (opt). The relative optimality gap is calculated as $100 \times (z_p - z_d) / z_p$, where z_p is the incumbent objective value and z_d is the lower bound.

The results in Table 4 show that the solver with M3 obtained average gaps of 1.69% (2,203.59 s), 10.00% (2,304.26 s) and 14.80% (2,544.21 s) for instances of groups G1, G2 and G3, respectively, while the solver with M4 obtained average gaps of 1.04% (1,838.50 s), 5.59% (2,344.94 s) and 11.29% (2,567.69 s). The optimality was proven by the solver in 34 (out of 90) instances with M3 and in 40 (out of 90) instances with M4. As expected, the models performed better (resp. worse) in those classes with small (resp. large) values of the number of item types (M) and/or (resp. large) small value of the average demand (d_{av}). For instance, the optimality was proven by the solver with M4 in all instances of classes 1, 3 and 5 of group G1, where these classes have $d_{av} = 10$. In contrast, the average gaps are higher for both models in classes 6, 12 and 18, which have $M = 40$ and $d_{av} = 100$. We ran additional computational experiments with M4 considering all 100 instances in classes 1, 3, 7, and 13. The optimality was proven by the solver in all these 400 ($=4 \times 100$) instances. The average number of objects (setups) obtained by the solver with M4 was 11.48 (2.90), 22.13 (4.19), 50.24 (6.32), and 63.47 (7.51) in classes 1, 3, 7, and 13, respectively. As a comparison, for these classes, the state-of-the-art heuristic of Cui et al. (2015) reported 11.49 (3.73), 22.13 (5.42), 50.24 (6.40), and 63.47 (7.52). For both models, we believe that a longer time limit would contribute to obtaining more certificates of optimality only in classes with up to $M = 20$ item types or $d_{av} = 10$, given that most of the time spent by the solver is to prove the optimality of the solution.

We note the quality of the linear relaxation of M4 in comparison to M3 reported in Table 4. For instance, the average values of the linear relaxation of M3 and M4 in class 1 was 1,465.33 and 10,984.99, respectively, while the average value of the optimal solutions was 11,720.00. However, the quality of the linear relaxation of M4 tends to decrease as the number of variables and constraints also increase. For example, the average values of the optimal solutions in classes 7 and 13 were 51,820.00 and 63,580.00, respectively. Indeed, the number of variables and constraints of both models increase as the number of item types also increases; they increase quickly as the average demands also increase, but notably in M4. In summary, M4 outperformed M3 in instances of group G1, while both models performed similarly in groups G2 and G3, with better average gaps for M4, and slightly better average times for M3.

Table 4: Results for the CutGen1 instances.

Class	M	v_1	v_2	d_{av}	Model M3						Model M4					
					var	cons	gap[%]	time[s]	LR	opt	var	cons	gap[%]	time[s]	LR	opt
1	10	0.01	0.20	10	297.00	467.80	0.00	0.46	1,465.33	5	284.00	48.80	0.00	0.18	10,984.99	5
2	10	0.01	0.20	100	808.40	1,363.40	0.18	2,339.93	7,644.07	3	3,511.00	355.40	0.16	1,764.59	108,945.14	4
3	20	0.01	0.20	10	1,246.00	1,945.40	0.00	81.13	1,662.53	5	1,341.80	114.20	0.00	12.95	22,640.64	5
4	20	0.01	0.20	100	3,088.40	5,265.20	0.90	3,600.00	8,181.54	0	15,916.20	856.40	1.22	3,600.00	225,257.93	0
5	40	0.01	0.20	10	4,989.20	7,666.60	2.01	3,600.00	1,546.62	0	5,340.00	236.00	0.00	2,053.28	42,148.81	5
6	40	0.01	0.20	100	15,958.40	26,924.40	7.07	3,600.00	7,174.91	0	69,205.60	1,947.20	4.88	3,600.00	420,700.35	0
Avg./Sum					4,397.90	7,272.13	1.69	2,203.59	4,612.50	13	15,933.10	3,558	1.04	1,838.50	138,446.31	19
7	10	0.01	0.80	10	503.80	832.40	0.00	24.24	4,096.52	5	967.00	113.80	0.00	40.84	42,325.97	5
8	10	0.01	0.80	100	807.40	1,408.40	0.12	1,009.81	23,127.70	4	9,872.80	923.80	0.13	1,496.75	421,145.39	3
9	20	0.01	0.80	10	2,175.60	3,607.40	0.91	2,885.23	4,162.29	1	4,259.40	257.80	0.40	2,883.70	87,795.33	1
10	20	0.01	0.80	100	3,322.20	5,853.60	1.70	2,922.31	21,984.11	1	43,432.60	2,122.80	1.70	3,036.38	873,767.07	1
11	40	0.01	0.80	10	8,011.20	13,300.80	5.51	3,600.00	3,431.09	0	15,506.00	490.40	1.17	3,011.99	162,725.93	1
12	40	0.01	0.80	100	12,831.40	22,712.80	51.72	3,600.00	21,348.93	0	163,345.40	4,152.80	30.12	3,600.00	1,623,418.72	0
Avg./Sum					4,608.60	7,952.57	10.00	2,340.26	13,025.11	11	39,563.87	8,061	5.59	2,344.94	535,196.40	11
13	10	0.20	0.80	10	622.60	1,026.80	0.04	761.52	5,949.27	4	1,213.00	139.60	0.00	216.02	51,494.52	5
14	10	0.20	0.80	100	910.80	1,591.40	0.06	1,607.51	56,573.41	3	12,126.40	1,130.20	0.15	2,826.86	512,385.34	2
15	20	0.20	0.80	10	2,234.40	3,737.40	0.36	2,575.06	6,349.71	2	4,889.00	287.00	0.32	2,199.80	105,346.45	2
16	20	0.20	0.80	100	3,561.60	6,290.60	0.36	3,121.18	57,076.57	1	51,064.80	2,487.80	1.17	3,600.00	1,048,556.50	0
17	40	0.20	0.80	10	8,808.20	14,707.60	23.73	3,600.00	6,017.75	0	18,455.20	565.40	0.40	2,963.49	200,958.75	1
18	40	0.20	0.80	100	14,181.00	25,165.00	64.24	3,600.00	55,867.73	0	195,024.20	4,919.80	65.73	3,600.00	2,003,135.91	0
Avg./Sum					5,053.10	8,753.13	14.80	2,544.21	31,305.74	10	47,128.77	9,530	11.29	2,567.69	653,646.24	10

*: the letters "tl" indicate when the time limit was reached in all 5 instances of the class.

5.2. Results for Fiber instances

In this section, we present the results performed to assess the quality of the Pareto frontiers obtained by the developed framework of Section 4 with Models (2) and (3), which are denoted as F+M3 and F+M4, respectively. We limited each execution of the solver to 900 seconds and stopped the algorithm if there was no improvement in reducing the number of objects for three consecutive iterations. We considered the set of 40 practical instances for the CSP-S provided by Umetani et al. (2003) which were taken from a chemical fiber industry. In these instances, the number of item types M varied from 6 to 29, the length of the items varied from 500 to 2,000, and the demand of each item type varied from 2 to 264. They were divided into two groups: the first twenty instances had $L = 5,180$ and the remaining twenty instances had $L = 9,080$. After applying preprocessing (P1) of Algorithm 1, the number of item types M varied from 4 to 20. The heuristic approaches of Umetani et al. (2003), Lee (2007), Golfeto et al. (2009) and Araujo et al. (2014) for the CSP-S used these instances to provide solutions with minimum trim loss for different fixed numbers of patterns. They calculated the trim loss as the percentage ratio of the total trim loss to the total length of the items, i.e., $100 \times (L\hat{z} - \sum_{i \in I} l_i d_i) / \sum_{i \in I} l_i d_i$, where \hat{z} is the number of objects required in the cutting plan. As we are not aware of other approaches in the literature that have solved the CSP-S exactly, we considered the approaches of Umetani et al. (2003) and Lee (2007) as benchmark approaches, denoted as ILS03 and Crawla, respectively. We note that the performance of the algorithms of Lee (2007) and Golfeto et al. (2009) was similar, while the genetic algorithm of Araujo et al. (2014) found more solutions with minimum trim loss, but required a significant additional number of patterns.

In Tables 5 and 6, we report the Pareto frontiers obtained by ILS03, Crawla, F+M3 and F+M4 for this set of instances where $L = 5,180$ and $L = 9,080$, respectively. For each of the approaches, we present the percentage of trim loss obtained by the approach when considering the respective number of patterns (column y). For F+M3 and F+M4, we report three additional pieces of information. The first is the processing time in seconds required to generate the Pareto frontier, which is reported in *italic* on the same line of the instance name (foreground of the entry in gray). Then we report their entries of percentage trim loss in **bold**, when the optimality was proven by the solver. Finally, we use the symbol * to indicate when the approach was not able to find a cutting plan with the minimum amount of trim loss, which is indicated in parentheses next to the instance name, when neither F+M3 nor F+M4 was able to achieve the minimum. Since the computational environment of the benchmark approaches is less powerful, we limit the analysis to the quality of the solution next. We note that ILS03 was executed in seconds/minutes (Umetani et al., 2003), while Crawla required from 1 to 20 minutes (Lee, 2007).

The results in Tables 5 and 6 show that the Pareto optimal frontier was obtained by F+M3 and F+M4 in 26 (out of 40) instances. In addition, F+M3 found the solution with minimum trim loss in instance fiber17_5180, while F+M4 in instance fiber20_5180. The processing times of F+M4 were better than those of F+M3 in almost all instances. We highlight the quality of the Pareto frontiers of F+M3 and F+M4 in comparison to ILS03 and Crawla. For instance, in fiber13a_5180, ILS03 ranged from 10 to 14 patterns, while Crawla and the proposed approaches required less than half that number of patterns. Note that Crawla did not report results in this instance for $y=2$, and obtained much higher trim loss values than the proposed approaches for $y=3$ and $y=4$. We did not report in **bold**, but we highlight that Crawla found a few optimal solutions, for instance, in fiber07_5180 and fiber09_5180. Considering instances with approximate Pareto frontiers as in fiber29a_5180, the trim loss values obtained by F+M3 and F+M4 were considerably lower than the

Table 5: Results for the Fiber instances of $L = 5, 180$.

y	ILS03	Crawla	F+M3	F+M4	y	ILS03	Crawla	F+M3	F+M4	y	ILS03	Crawla	F+M3	F+M4
fiber06_5180					fiber13b_5180					fiber20_5180				
7	5.19		62.05	8.36	9	6.59		1.96	0.84	8	13.41		2,896.94	1,205.81
6	5.19				8	6.59				7	19.71	7.11	*	3.96
5	5.19		2.09	2.09	7	6.59				6	26.02	10.26	3.96	3.96
4	8.28	8.28	5.19	5.19	6	6.59				5	54.37	22.86	7.11	7.11
3	8.28	12.47	8.28	8.28	5	6.59				4	57.52	54.37	19.71	19.71
2		48.50	17.56	17.56	4		6.59	2.92	2.92	fiber23_5180 (0.70)				
fiber07_5180					fiber14_5180 (1.15)					fiber26_5180 (2.31)				
10	4.98		0.15	0.28	3		10.27	6.59	6.59	15	4.98		9,000.00	4,651.96
9	4.98				2		105.83	21.29	21.29	14	2.84			
8	4.98				fiber15_5180					13	6.41			
7	4.98				11	5.45		1,389.40	1,948.02	12	9.27		*	
6	8.16				10	5.45				11	17.84		2.12	
3		8.16	4.98	4.98	9	5.45				10			4.27	
2		11.34	11.34	11.34	8	7.61				9		2.84	4.98	
1		68.61	68.61	68.61	7	9.76		*	*	8		4.27	4.98	*
fiber08_5180					fiber16_5180 (0.34)					fiber28a_5180 (0.06)				
10	4.47		1.00	2.98	6		5.45	3.30	3.30	7		5.70	5.70	2.84
9	4.47				5		9.76	3.30	3.30	6		7.84	7.84	4.27
8	4.47				4		14.06	5.45	5.45	5		19.98	10.69	7.84
7	4.47				3		63.56	9.76	9.76	4			20.69	20.69
6	4.47				2			63.56	63.56	fiber29_5180				
5					fiber17_5180					29	3.39		6,300.00	5,972.31
4					14	4.76		102.02	15.79	28	2.31			
3					13	4.76				27	3.39			
2					12	4.76				26	3.39			
1					11	4.76				25	3.93			
fiber09_5180					fiber18_5180					fiber28b_5180 (0.53)				
9	9.27		1,030.78	1,291.81	5		4.76	2.95	2.95	11	9.70		*	*
8	9.27				4		10.17	4.76	4.76	8	4.47		2.85	2.85
7	9.27				3		49.91	6.56	6.56	7	5.01		3.39	3.39
6	11.29				2			39.07	39.07	6	7.16		3.93	3.93
5	11.29				fiber19_5180					5	15.78		5.01	5.01
4					10	7.68		6,334.12	4,683.52	4	78.24		20.08	20.08
3					9	6.46				fiber29_5180				
2					8	8.91		*	*	11	10.91		*	*
1					7	8.91	4.01	1.56		10	10.91			
fiber10_5180					fiber20_5180					fiber29_5180				
14	4.20		13.76	26.04	6	27.26	5.24	2.79	1.56	9	10.91		*	*
13	4.20				5		11.35	4.01	4.01	8	18.14	4.88		
12	4.20				4		13.80	7.68	7.68	7	26.58	7.29	1.26	*
11	4.20				3		45.62	28.48	28.48	6		9.70	2.47	1.26
10	4.20				fiber17_5180					5		13.32	3.67	3.67
9					9	5.46		1,438.49	2,918.21	4		28.99	16.93	16.93
8					8	6.69				3			156.77	156.77
5					7	7.92				fiber28b_5180 (0.53)				
4					6	17.28	4.24	1.78		15	5.69		9,000.00	6,300.00
3					5	22.63	6.69	3.01	*	14	6.55			
2					4		7.92	3.01	3.01	13	6.55			
fiber11_5180					fiber18_5180					fiber29_5180				
12	6.06		71.09	25.61	3		29.99	23.86	23.86	12	8.27		*	*
11	6.06				fiber18_5180					11	6.55		2.25	
10	4.52				11	5.10		1,935.83	1,905.90	10			3.11	
9	4.52				10	6.16				9			3.11	*
8	6.06				9	5.10				8		5.69	3.97	3.11
5					8	10.41				7		8.27	8.27	4.83
4					7	11.47		1.91	1.91	6		15.14	9.99	9.13
3					6		5.10	2.97	2.97	5		34.90	22.87	22.01
2					5		6.16	4.04	4.04	4			202.46	119.97
fiber13a_5180					fiber19_5180					fiber29_5180				
14	2.41		16.54	6.89	4		7.22	6.16	6.16	fiber29_5180				
13	4.24				3		18.90	18.90	18.90	13	11.03		4,671.54	2,627.35
12	2.41				fiber19_5180					12	9.40			
11	4.24				25	4.98		1,335.62	2,733.32	11	14.30			
10	4.24				24	5.77				10	9.40			
4					23	4.98				9	12.66	4.50	1.23	
3					22	4.98				8		9.40	2.87	1.23
2					21	4.98				7		14.29	2.87	2.87
fiber13a_5180					fiber19_5180					fiber29_5180				
14	2.41				9		5.77			6		24.09	4.50	4.50
13	4.24				8		6.56			5		32.26	27.36	15.93
12	2.41				7		7.35			4			113.89	113.89
11	4.24				6		8.14	4.98	4.98					
10	4.24				5		9.72	6.56	6.56					
4					4		14.45	11.30	11.30					
3					3			32.61	32.61					
2														

Table 6: Results for the Fiber instances of $L = 9,080$.

y	ILS03	Crawla	F+M3	F+M4	y	ILS03	Crawla	F+M3	F+M4	y	ILS03	Crawla	F+M3	F+M4			
fiber06_9080				0.38	0.14	fiber14_9080				0.38	0.14	fiber23_9080 (0.15)				3,112.41	2,777.96
5	8.46				5	5.63				7	11.41						
4	8.46				4	5.63				6	12.67	6.41	*	*			
3	8.46	8.46	3.04	3.04	3	43.35	5.63	1.86	1.86	5	16.42	7.66	1.40	1.40			
2		19.30	8.46	8.46	2		24.49	13.17	13.17	4		11.41	3.90	3.90			
1		73.53	73.53	73.53	fiber15_9080				1.12	0.80	3		20.18	13.92	13.92		
fiber07_9080				0.01	0.02	5	10.81			2				126.58	126.58		
4	5.95				4	7.64	4.48	1.31	1.31	fiber26_9080				233.33	231.76		
3	5.95				3	32.97	7.64	4.48	4.48	9		6.66					
2	5.95	5.95	5.95	5.95	2		23.47	7.64	7.64	8		7.61					
1		17.10	17.10	17.10	1			200.77	200.77	7		8.55					
fiber08_9080				0.24	0.49	fiber16_9080				543.28	61.81	6		1.94			
4	3.13				6	7.25				5		3.83	1.00	1.00			
3	7.34	1.03	1.03	1.03	5	7.25	7.25	0.81	0.81	4		10.44	1.94	1.94			
2	17.87	7.34	3.13	3.13	4	13.68	9.39	2.96	2.96	3		33.09	9.50	9.50			
1		32.60	32.60	32.60	3		17.97	7.25	7.25	2			173.74	173.74			
fiber09_9080				0.34	0.77	2		45.86	28.70	28.70	fiber28a_9080				360.00	105.35	
5	9.95				fiber17_9080				46.27	17.28	8		9.88				
4	9.95	6.41			12	5.33				7		9.88					
3	9.95	9.95	2.86	2.86	11	5.33				6		18.33					
2		27.69	13.50	13.50	10	3.18				5		5.66	1.43	1.43			
1			254.69	254.69	5		3.18			4		11.99	3.54	3.54			
fiber10_9080				0.28	0.69	4		5.33	1.03	1.03	3		20.45	9.88	9.88		
5	6.98				3		7.48	5.33	5.33	2		60.60	60.60	60.60			
4	6.98	1.76			2		26.83	24.68	24.68	fiber28b_9080				3,378.21	1,361.93		
3	9.59	4.37	1.76	1.76	fiber18_9080				85.24	29.89	12		3.93				
2		14.81	6.98	6.98	6		6.07			11		6.94					
1			72.21	72.21	5	15.37	2.35	0.49	0.49	10		14.47					
fiber11_9080				2.18	1.60	4	19.09	4.21	2.35	2.35	8			0.91			
5	7.77				3		7.93	4.21	4.21	6		3.93	2.42	0.91			
4	5.08	7.77	2.38	2.38	2		33.98	19.09	19.09	5		6.94	3.93	2.42			
3	10.47	10.47	5.08	5.08	fiber19_9080 (1.00)				1,254.74	2,076.44	4		14.47	6.94	5.43		
2		34.71	13.16	13.16	8		3.77			3		29.53	23.51	23.51			
1			228.70	228.70	7		6.54			fiber29_9080				409.59	381.10		
fiber13a_9080				10.42	2.38	6		6.54			13		5.90				
6	5.79				5			2.39	*	12		8.76					
5	8.99				4		3.27	2.39	2.39	11		11.62					
4	12.20	8.99	2.58	2.58	3		6.10	5.16	5.16	6		5.90					
3		15.40	5.79	5.79	2		49.96	30.06	30.06	5		8.76	0.17	0.17			
2		31.43	12.20	12.20	fiber20_9080				16.55	1.28	4		20.21	5.90	5.90		
1			406.49	406.49	7		15.97			3		45.97	20.21	20.21			
fiber13b_9080				2.60	0.83	6		15.97									
4	15.97				5		10.45										
3	9.53		3.09	3.09	4		15.97	4.92	4.92								
2	22.42	9.53	9.53	9.53	3		27.01	10.45	10.45								
1		138.39	138.39	138.39	2		137.46	131.94	131.94								

benchmark approaches. As expected, these results show that increasing the number of patterns significantly reduces the trim loss in the first cutting plans of the Pareto curve (i.e., when y is close to y^*), as shown in Table 6. In the last cutting plans of the Pareto curve when z is close to z^* , this behavior was attenuated, as shown in Table 5. We also note the cases of fiber06_5180 (y from 3 to 4) and fiber14_5180 (y from 5 to 6), in which the trim loss remained even with the increase in the number of patterns. In summary, the better results in Table 6 than in Table 5 can be explained by the increase of L from 5180 to 9080. Since the length of the item types is maintained, this increase requires cutting planes with fewer objects, which, in turn, requires fewer variables and constraints.

5.3. Comments on other ILP formulations

In the development phase of this study, we performed preliminary computational experiments with the ILP formulations of Johnston & Sadinlija (2004) and Alves & Valério De Carvalho (2008) that can be adapted to address the CSP-S. Since the variables in these formulations are indexed in the number of items that fit in the object, they tend to have a better (resp. worse) performance when the number of items in a pattern is small (resp. large). For instance, for the Cutgen1 instances, the optimality was proven by the solver with the adapted version of the arc flow model of Alves & Valério De Carvalho (2008) in 1 out of 30 instances in group G1 and in 25 out of 30 instances in group G3. Certificates of optimality were obtained by the solver with the adapted version of the model of Johnston & Sadinlija (2004) in 9 out of 30 instances in group G1 and in 12 out of 30 instances in group G3. Note that the average number of items in a pattern of group G3 was 2 ($v_1 = 0.20$ and $v_2 = 0.80$). Another difficulty was in the instances with large values for the maximum frequency of the patterns, especially in the first cutting plans of the Pareto curves for the Fiber instances. Indeed, the framework of Section 1 reached the stopping criterion without solutions in 19 out of 40 of Fiber instances with this formulation. Therefore, we consider that the proposed pattern-based formulations were more suitable for the practical scenarios we intend to solve.

6. Conclusions

We proposed an exact framework to obtain the Pareto optimal frontier for the One-dimensional Cutting Stock Problem with the bi-objective function of minimizing the number of objects and the number of distinct patterns. Instead of using decomposition methods for solving the Pattern Minimization Problem as in Vanderbeck (2000), we relied on the development of ILP formulations to be solved by a general-purpose solver. The results of the computational experiments showed that the approaches are appropriate for instances with solutions characterized by a moderate number of objects and a few patterns in the Pareto optimal frontier. As future research, a challenging possibility is the development of new valid inequalities to strengthen the linear relaxation of the ILP formulations, as well as to avoid symmetries in the solutions. Since the problem is mainly addressed by heuristics, another possibility would be the development of a highly flexible algorithm based on a Sequential Heuristic Procedure. This algorithm would identify the characteristics of the problem instance, for instance, to generate patterns, by closing the demand of more than one item type and to consider patterns with trim loss, as in the illustrative example provided.

Acknowledgements

We thank Prof. Silvio Araujo and Prof. Kelly Poldi for sharing with us part of their results. The authors would like to thank the São Paulo Research Foundation (FAPESP-Brazil) [grant

numbers 2020/00747-2, 2016/01860-1] and the National Council for Scientific and Technological Development (CNPq-Brazil) [grant number 304144/2019-3] for the financial support. Research was carried out using the computational resources of the Center for Mathematical Sciences Applied to Industry (CeMEAI) funded by FAPESP-Brazil [grant number 2013/07375-0].

References

- Aliano Filho, A., Moretti, A. C., & Pato, M. V. (2018). A comparative study of exact methods for the bi-objective integer one-dimensional cutting stock problem. *Journal of the Operational Research Society*, *69*, 91–107. doi:10.1057/s41274-017-0214-7.
- Alves, C., Macedo, R., & Valério de Carvalho, J. (2009). New lower bounds based on column generation and constraint programming for the pattern minimization problem. *Computers and Operations Research*, *36*, 2944–2954. doi:10.1016/j.cor.2009.01.008.
- Alves, C., & Valério De Carvalho, J. M. (2008). A branch-and-price-and-cut algorithm for the pattern minimization problem. *RAIRO - Operations Research*, *42*, 435–453. doi:10.1051/ro:2008027.
- Araujo, S. A., Poldi, K. C., & Smith, J. (2014). A genetic algorithm for the one-dimensional cutting stock problem with setups. *Pesquisa Operacional*, *34*, 165–187. doi:10.1590/0101-7438.2014.034.02.0165.
- Belov, G., & Scheithauer, G. (2003). *The number of setups (different patterns) in one-dimensional stock cutting*. Technical Report Institute for Numerical Mathematics, Dresden University, MATH-NM-15-2003.
- Cerqueira, G. R., Aguiar, S. S., & Marques, M. (2021). Modified Greedy Heuristic for the one-dimensional cutting stock problem. *Journal of Combinatorial Optimization*, . doi:10.1007/s10878-021-00695-4.
- Cui, Y., & Liu, Z. (2011). C-Sets-based sequential heuristic procedure for the one-dimensional cutting stock problem with pattern reduction. *Optimization Methods and Software*, *26*, 155–167. doi:10.1080/10556780903420531.
- Cui, Y., Zhong, C., & Yao, Y. (2015). Pattern-set generation algorithm for the one-dimensional cutting stock problem with setup cost. *European Journal of Operational Research*, *243*, 540–546. doi:10.1016/j.ejor.2014.12.015.
- Diegel, A., Chetty, M., Van Schalkwyck, S., & Naidoo, S. (1993). *Setup combining in the trim loss problem—3-to-2 & 2-to-1*. Technical Report Business Administration, University of Natal, Durban, First Draft.
- Foerster, H., & Wäscher, G. (2000). Pattern reduction in one-dimensional cutting stock problems. *International Journal of Production Research*, *38*, 1657–1676. doi:10.1080/002075400188780.
- Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability; A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman & Co.
- Gau, T., & Wäscher, G. (1995). CUTGEN1: A problem generator for the standard one-dimensional cutting stock problem. *European Journal of Operational Research*, *84*, 572–579. doi:10.1016/0377-2217(95)00023-J.
- Gilmore, P. C., & Gomory, R. E. (1961). A Linear Programming Approach to the Cutting-Stock Problem. *Operations Research*, *9*, 849–859. doi:10.1287/opre.9.6.849.
- Golfeto, R. R., Moretti, A. C., & Neto, L. L. d. S. (2009). A genetic symbiotic algorithm applied to cutting stock problem with multiple objectives. *Advanced Modeling and Optimization*, *11*, 473–501. URL: <https://camo.ici.ro/journal/vol11/v11d9.pdf>.
- Haessler, R. W. (1975). Controlling Cutting Pattern Changes in One-Dimensional Trim Problems. *Operations Research*, *23*, 483–493. doi:10.1287/opre.23.3.483.
- Harjunkoski, I., Westerlund, T., Isaksson, J., & Skrifvars, H. (1996). Different formulations for solving trim loss problems in a paper-converting mill with ILP. *Computers and Chemical Engineering*, *20*, 121–126. doi:10.1016/0098-1354(96)00031-2.
- Henn, S., & Wäscher, G. (2013). Extensions of cutting problems: Setups. *Pesquisa Operacional*, *33*, 133–162. doi:10.1590/S0101-74382013000200001.
- Johnston, R. E., & Sadinlija, E. (2004). A new model for complete solutions to one-dimensional cutting stock problems. *European Journal of Operational Research*, *153*, 176–183. doi:10.1016/S0377-2217(02)00704-X.
- Kantorovich, L. V. (1960). Mathematical methods of organizing and planning production. *Management Science*, *6*, 366–422. doi:10.1287/mnsc.6.4.366.
- Lee, J. (2007). In situ column generation for a cutting-stock problem. *Computers and Operations Research*, *34*, 2345–2358. doi:10.1016/j.cor.2005.09.007.
- Martin, M., Moretti, A., Gomes-Ruggiero, M., & Neto, L. S. (2018). Modification of Haessler’s sequential heuristic procedure for the one-dimensional cutting stock problem with setup cost. *Producao*, *28*. doi:10.1590/0103-6513.20170105.

- McDiarmid, C. (1999). Pattern minimisation in cutting stock problems. *Discrete Applied Mathematics*, 98, 121–130. doi:10.1016/S0166-218X(99)00112-2.
- Mobasher, A., & Ekici, A. (2013). Solution approaches for the cutting stock problem with setup cost. *Computers and Operations Research*, 40, 225–235. doi:10.1016/j.cor.2012.06.007.
- Nitsche, C., Scheithauer, G., & Terno, J. (1999). Tighter relaxations for the cutting stock problem. *European Journal of Operational Research*, 112, 654–663. doi:10.1016/S0377-2217(97)00404-9.
- Scheithauer, G. (2018). *Introduction to cutting and packing optimization: problems, modeling approaches, solution methods*. International Series in Operations Research and Management Science. Springer. doi:10.1007/978-3-319-64143-0.
- Umetani, S., Yagiura, M., & Ibaraki, T. (2003). One-dimensional cutting stock problem to minimize the number of different patterns. *European Journal of Operational Research*, 146, 388–402. doi:10.1016/S0377-2217(02)00239-4.
- Umetani, S., Yagiura, M., & Ibaraki, T. (2006). 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*, 5, 43–64. doi:10.1007/s10852-005-9031-0.
- Vahrenkamp, R. (1996). Random search in the one-dimensional cutting stock problem. *European Journal of Operational Research*, 95, 191–200. doi:10.1016/0377-2217(95)00198-0.
- Valério de Carvalho, J. M. (1999). Exact solution of cutting stock problems using column generation and branch-and-bound. *Annals of Operations Research*, 86, 629–659. doi:10.1023/a:1018952112615.
- Valério de Carvalho, J. M. (2002). LP models for bin packing and cutting stock problems. *European Journal of Operational Research*, 141, 253–273. doi:10.1016/S0377-2217(02)00124-8.
- Vanderbeck, F. (2000). Exact Algorithm for Minimising the Number of Setups in the One-Dimensional Cutting Stock Problem. *Operations Research*, 48, 915–926. doi:10.1287/opre.48.6.915.12391.
- Wäscher, G., Haubner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183, 1109–1130. doi:10.1016/j.ejor.2005.12.047.
- Yanasse, H. H., & Limeira, M. S. (2006). A hybrid heuristic to reduce the number of different patterns in cutting stock problems. *Computers and Operations Research*, 33, 2744–2756. doi:10.1016/j.cor.2005.02.026.