Ann Oper Res (2017) 258:719–746
DOI 10.1007/s10479-015-1955-9

A heuristic approach to minimize the number of saw cycles in small-scale furniture factories

Alyne Toscano[1] · **Socorro Rangel**[2] · **Horacio Hideki Yanasse**[3]

Published online: 8 August 2015

Abstract This paper addresses a two-dimensional cutting stock problem arising in furniture factories. The problem involves the simultaneous optimization of two, usually conflicting, objectives: minimizing the total number of objects and maximizing the cutting machine productivity in terms of the number of objects that are simultaneously cut. A heuristic algorithm to solve the problem is proposed based on variables and constraints generation. The main idea is to add, in a dynamic way, bounds to the frequency of some chosen cutting patterns. At each iteration a solution is generated and at the end we have a set of non-dominated solutions. A computational study was conducted using real data from a small-scale furniture factory. The results show that the proposed algorithm finds solutions that are as good as or better than the ones used in practice in the furniture factory.

Keywords Two-dimensional cutting stock · Saw cycles · Machine productivity · Column generation · Heuristic · Furniture production

✉ Socorro Rangel
socorro@ibilce.unesp.br

Alyne Toscano
alyne@icte.uftm.edu.br

Horacio Hideki Yanasse
horacio.yanasse@unifesp.br

[1] UFTM - Universidade Federal do Triângulo Mineiro, Av. Dr. Randolfo Borges Júnior, 1250, Uberaba CEP: 38025-180, Brazil

[2] UNESP - Univ Estadual Paulista, R. Cristóvão Colombo, 2265, São José do Rio Preto CEP: 15054-000, Brazil

[3] Instituto de Ciência e Tecnologia, Universidade Federal de São Paulo, Av. Cesare Mansueto Giulio Lattes, 1201, Parque Tecnológico, Jd. Sta. Inês II, São José dos Campos CEP: 12247-014, Brazil

1 Introduction

In some industries the production process involves a stage in which large objects have to be cut into smaller pieces according to a pre-specified demand. An important aspect of the production planning at this particular stage relates to the costs of the material waste as well as the cutting equipment productivity. Such waste consists of good-quality material scraps which are useless for practical purposes. The equipment productivity is affected by the type and number of different patterns used to cut the objects and also by the number of objects that can be cut simultaneously. In this paper we consider these two aspects in the context of furniture industries. In particular, companies that produce furniture using rectangular plates of different sizes and types as their main raw material.

The problem of defining how to cut a set of objects into smaller pieces (items) so that a criterion is optimized is known in the literature as the cutting stock problem (CSP). This problem has attracted the attention of the scientific community in the search for new, solution-wise efficient models, and effective solution methods (e.g. Cherri et al. 2014; Kallrath et al. 2014; Silva et al. 2014; Morabito et al. 2009; Wang and Waescher 2002). A typology for the organization and categorization of the cutting and packing problems can be found in Dyckhoff (1990) and in Waescher et al. (2007).

The minimization of the total waste or the total number objects cut is the most common criterion used when solving the CSP. Another important criterion, observed in visits to a furniture factory in Brazil, is to maximize the cutting machine productivity (Rangel and Figueiredo 2008). In times of high demand, the cutting stage can become a bottleneck in the production process. Furthermore, a better use of the cutting machine capacity implies a reduction in the cutting time and consequently in energy savings and reduction in operational costs.

The cutting machine productivity can be improved by reducing the number of different cutting patterns. This strategy has been discussed in the literature by several authors (e.g. Haessler 1975; Foerster and Waescher 2000; Vanderbeck 2000; Vasko et al. 2000; Diegel et al. 2006; Yanasse and Limeira 2006; Moretti and Salles Neto 2008; Alves et al. 2009; Mobasher and Ekici 2013). The CSP considering the minimization of the total number of cutting patterns (RCP) is strongly NP-hard, even for the special one-dimensional case in which any two ordered items fit into an object, but not three (McDiarmid 1999). This might explain why most of the solution proposals given in the literature for this problem are heuristic methods.

Another procedure, less explored in the literature, takes into account that some cutting machines allow the objects to be stacked so that they can be cut simultaneously according to the same cutting pattern. The number of objects in a stack depends on its thickness as well as on the machine maximum load. The time necessary to adjust the cutting machine and to cut a stack of objects according to a given cutting pattern is named *cutting cycle* or *saw cycle* (Yanasse et al. 1993; Yanasse 2008). For many users, it is desirable that the number of objects cut according to a given cutting pattern is a multiple of the maximum stack height. This implies that, if the maximum stack is four, then the cutting pattern frequency should be 4, 8, 12 and so on. If the frequency of a given cutting pattern is 25, then the user might prefer to reduce it to 24 (under production) or to increase it to 28 (over production) so that a cycle that does not use the full machine capacity is avoided.

If the diversity of the cutting patterns in a solution of the CSP is reduced, the frequency of a given cutting pattern might increase (on average) given that the overall number of objects will remain the same or increase (Yanasse and Limeira 2006). So, solving the RCP

might contribute to reducing the total number of saw cycles, however this is not guaranteed. Therefore, if the cost of the machine time is dominant compared to other production costs, the CSP taking into account the minimization of saw cycles should be addressed in order to reduce operational costs. This aspect is particularly important in the furniture industry.

In the present study, we propose a heuristic algorithm to find good solutions for the CSP with saw-cycle minimization in the context of small-scale furniture factories. In Sect. 2, we briefly review the two-dimensional cutting stock problem, specify the cutting machine constraints, and the class of cutting patterns used in a representative factory of the furniture sector in Brazil. In Sect. 3 we review the literature that considers saw cycles and present the heuristic algorithm developed. A numerical study comparing the proposed solution method with the practice of a small-scale factory in Brazil (Factory V) is described in Sect. 4. In Sect. 5 we present concluding remarks.

2 The two dimensional cutting stock problem and saw cycles

The two-dimensional cutting stock problem (2D-CSP) takes into account that two dimensions will be relevant to the cut operations (e.g. Morabito and Arenales 2000). In this paper, we consider the 2D-CSP in the context of furniture production that uses rectangular wooden plates (e.g. OSB—Oriented Strand Board, MDF—Medium Density Fiberboard) as the main raw material. The problem can be stated as to define how to cut a set of rectangular, large objects, $((L, W))$, of length L and width W to produce a set of m rectangular, smaller items with specified lengths, widths and demands $[(l_i, w_i), b_i, i = 1, \ldots, m$, respectively] according to some given criteria. The objects are available in stock in a large number so as to consider them unlimited, and the items are parts of the final products (e.g. wardrobes, chest of drawers and beds).

To formulate a mathematical model to represent the problem, suppose that there are n two-dimensional (2D) possible cutting patterns and that they have been generated a priori. Let A_j $(j = 1, \ldots, n)$ be an m-dimensional column vector representing the cutting pattern j (each element a_{ij} is the number of items i in pattern j); and let the decision variables x_j, $j = 1, \ldots, n$, represent the number of objects cut according to the cutting pattern A_j. The 2D-CSP problem can be formulated as (Gilmore and Gomory 1963, 1965):

$$\text{minimize} \quad z = \sum_{j=1}^{n} c_j x_j \tag{1}$$

subject to:

$$\sum_{j=1}^{n} A_j x_j \geq b \tag{2}$$

$$x_j \in Z_+, \quad j = 1, \ldots, n. \tag{3}$$

If the optimization criterion is to minimize the total waste, then $c_j = \left(LW - \sum_{i=1}^{m} l_i w_i a_{ij}\right)$ computes the non used area in the cutting pattern A_j. Another usual criterion is to minimize the total number of objects cut, in this case $c_j = 1$. Constraints (2) guarantee that the demand is satisfied and that overproduction is allowed. Constraints (3) require the decision variables to be non-negative integers.

A common practice is to solve problem (1)–(3) considering that only a subset of m feasible cutting patterns are known a priori. A linear relaxation of the 2D-CSP (problem RL) obtained replacing constraint (3) by $x_j \in R_+$ is solved by column generation as proposed by Gilmore

and Gomory (1965). If the item demand is high, simple rounding procedures can be used to obtain a good feasible integer solution, otherwise specialized methods have to be employed (e.g. residual heuristics as proposed in Poldi and Arenales 2009). The optimal integer solution for the 2D-CSP problem can be found using the branch and price method (e.g. Munari and Gondzio 2013; Barnhart et al. 1998).

2.1 2D cutting pattern generation

Several aspects should be considered in the generation of 2D-cutting patterns (e.g. Queiroz and Miyazawa 2014; Glass and Oostrum 2010; Morabito and Arenales 2000). The majority of cutting machines found in the furniture industry impose that only orthogonal guillotine cuts can be made. A cut is an orthogonal guillotine type if, when applied to a rectangle, it produces two other rectangles. Another important consideration is the number of times the object must be rotated 90° in order to cut all the items. This is called the number of stages. In the first stage, parallel longitudinal guillotine cuts divide the object in a set of strips. In the second stage, these strips are divided by parallel transversal guillotine cuts. If there is no need for additional trimming (i.e. all pieces have the same width in each strip), the pattern is called an exact 2-stage guillotine pattern; otherwise, it is called non-exact. If there are additional cutting stages on the pattern, for example, three, four, etc., the pattern is called 3-stage, 4-stage, etc., respectively. The trimming in a non-exact cutting pattern is usually done on a secondary cutting machine and therefore it is not counted as an additional stage. Figure 1 shows a non-exact two dimensional 3-stage cutting pattern. The areas in grey represent waste. Note that in stage 2 (Fig. 1b) there is an item that needs trimming.

A third aspect that should be considered when generating a 2D-cutting pattern is the possibility of rotating the items. There are applications in which the items are free of any orientation restrictions, allowing rotations in a continuous orientation angle (e.g. Kallrath et al. 2014; Rebennack et al. 2009). In others, rotations may be limited, for instance, to 90°. In case of rectangles with rotations limited to 90°, an item with size (l_i, w_i) will have length equal to w_i and width equal to l_i after rotation. Although the total number of items might increase (up to 2 m), allowing item rotation when generating a 2D-cutting pattern might help to reduce the total waste.

In some cutting machines, the saw thickness (σ) cannot be disregarded. Therefore, to avoid infeasible cutting patterns, the dimensions of all the items and of the object is increased by σ. That is, $(l_i, w_i) = (l_i + \sigma, w_i + \sigma), \forall i$ and $(L, W) = (L + \sigma, W + \sigma)$ (e.g. Morabito and Arenales 2000).

A trade-off between using simple cutting patterns and using patterns that yield less waste of material but that require longer processing times is usually considered in the furniture plants. In fact, the production managers of Factory V consider that a waste of 6 % of the

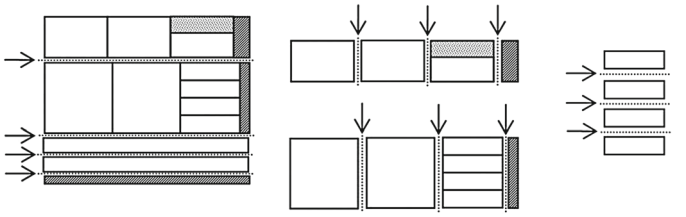


Fig. 1 A 3-stage cutting pattern

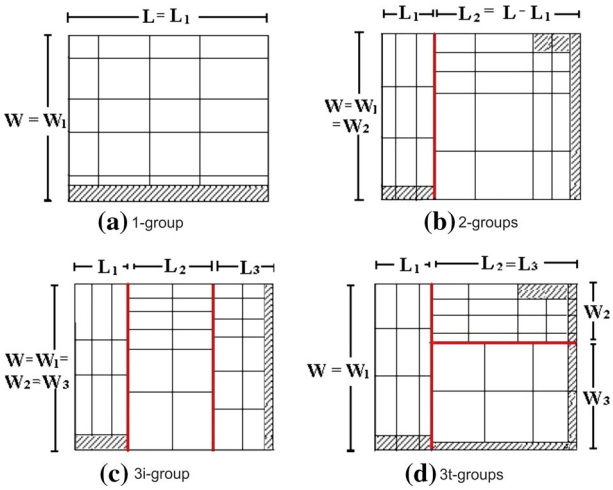


Fig. 2 *n*-Group cutting patterns (adapted from Faccio and Rangel 2009)

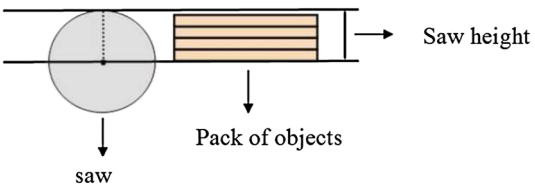


Fig. 3 Frontal sketch of the cutting operation—the maximum load in a saw cycle

object area is acceptable. They favour cutting patterns that include few items and with at most three stages (not counting trimming). In particular, they prefer maximal homogeneous cutting patterns, that is patterns composed of a single type of item, with the maximum possible number of them that can be cut from the object. They also take into account the trade-off between the benefits of combining several products in a production run and the difficulties in the production control and storage space.

An important class of cutting patterns with high productivity of the cutting machine is the *n*-group cutting patterns. An *n*-group cutting pattern is formed by *n* parts of 1-group patterns. A 1-group pattern is a two-stage cutting pattern formed by a set of strips that can be simultaneously cut in the second stage (Gilmore and Gomory 1965). The homogenous cutting pattern belongs to the 1-group class. The case of $n = 3$, the 3-group cutting patterns, can be further classified as $3i$-group and $3t$-group. The notations $3i$ and $3t$ are used to differentiate how the three 1-group parts are arranged in the patterns. In the case of the 3i-group, the three parts are obtained by two parallel guillotine cuts (in the form of a capital letter "I"). In the case of the $3t$-group the three parts are obtained by two perpendicular guillotine cuts (in the form of the capital letter "T"). Figure 2 shows four types of *n*-groups patterns: (a) 1-group, (b) 2-groups, (c) 3i-groups and (d) 3t-groups. Since most of the cutting patterns used in Factory V are *n*-group, Faccio and Rangel (2009) conducted a computational study of the mixed integer programming models presented by Yanasse and Morabito (2008) to generate *n*-group patterns ($n = 1, 2, 3$) with data from Factory V. The results showed that the generated cutting patterns have waste within the requirements of the industry (average waste between 1.1 and 2.5 %). The best results were obtained with the $3t$-group cutting patterns.

2.2 Saw cycles

In furniture factories, the cutting machines allow the objects to be stacked and simultaneously cut according to the same cutting pattern. Figure 3 shows a sketch of the cut operation. In this figure it is possible to see the saw height (h) which is used to compute the machine maximum load. The maximum number of objects that the machine can cut at the same time (cap) can be calculated considering h and the thickness of the object to be cut (t) as stated in (4):

$$cap = \left\lfloor \frac{h}{t} \right\rfloor. \tag{4}$$

Using cap it is then possible to calculate the number of saw cycles (y_j) necessary to cut x_j objects according to a cutting pattern j, that is:

$$y_j = \left\lceil \frac{x_j}{cap} \right\rceil, \quad \forall j. \tag{5}$$

Assuming that the time required to carry out one saw cycle does not vary with the number of objects being simultaneously cut, cutting a set of objects instead of just one at a time decreases the processing time of the cutting machine. If the machine's full capacity is used in a given saw cycle we say it is a complete cycle, otherwise the cycle is said to be incomplete. So, to reduce the total number of saw cycles it would be good to have complete cycles while maintaining or also reducing the total number of objects cut. In the next section we discuss in more details the CSP considering the reduction of the total number of saw cycles and the total number of objects (CSP-C) and present a heuristic algorithm to solve it.

3 A heuristic method to minimize the number of objects and saw cycles

In comparison with the RCP defined in Sect. 1, the problem of reducing the number of saw cycles has received less attention in the literature. For the one dimensional case, we are aware of the works of Degraeve and Vandebroeke (1998), Chu and Antonio (1999), Degraeve et al. (2002) and Li et al. (2008). Ranck (2008) proposes a mathematical model for the (CSP-C) that can be used in both the one-dimensional and two-dimensional case and allows the use of incomplete cycles. The model of Mosquera and Rangel (2007) only allows complete cycles. Yanasse (2008) shows that solving the CSP considering the reduction of saw cycles in a situation of high demand is equivalent to solving the CSP with the demands scaled by cap. He presents a mathematical model for the CSP-C and shows that if the item demand is low, then it reduces to the RCP.

Malaguti et al. (2014) study the CSP-C for the two-dimensional case in the context of wooden board cutting industry taking into account waste minimization and machine productivity maximization. Three heuristics are proposed to solve the two-dimensional CSP. A heuristic to reduce the number of saw cycles, similar to the procedure discussed in Yanasse et al. (1993) and in Yanasse and Limeira (2006), is applied to the CSP heuristic solutions. As done by Yanasse (2008) and Ranck (2008), Malaguti et al. (2014) also extend the classical formulation of the CSP by Gilmore and Gomory (1963) to consider the minimization of saw cycles. To cope with the high number of possible cutting patterns, they use the cutting patterns of the heuristic solutions as input data to the two proposed models.

The objective of minimizing the total number of saw cycles can conflict with the objective of minimizing the total number of objects used to attend the demand. If excess of demand is accepted, a better use of the saw capacity (complete saw cycles) might imply in an increase

of the number of objects cut. So there is interest in finding a compromise between these two objectives. The optimization of two or more conflicting objectives generates a set of efficient (Pareto optimal) solutions such that it is not possible to state whether one solution is better than another (Deb 2004). The heuristic proposed in this work aims to find a balance between the total number of objects and the total number of saw cycles in order to find a variety of non-dominated solutions while maintaining a compromise between these two objectives.

One way to control the number of saw cycles is by imposing constraints on the number of objects cut according to a given cutting pattern. However, the number of feasible cutting patterns is too high making it impossible to generate all of them a priori. Even if just a limited number of them were generated, imposing constraints to their frequency might also make the size of the mathematical formulation impractical. The idea is then to generate the cutting patterns and to impose constraints to their frequency in a dynamic way. The motivation behind this idea is that an alteration on the cutting stock solution might allow a new and better solution in terms of the total number of saw cycles. At each iteration of the proposed procedure, a solution (x, y) to the CSP-C is generated. After *it* iterations, the set of *it* solutions is analysed and the non-dominated ones are selected.

The heuristic method proposed was inspired by the works related to RCP of Diegel et al. (2006) and Vasko et al. (2000). It supposes that a feasible solution to the 2D-CSP, problem (1)–(3), is known. Two strategies are then considered to control the cutting pattern frequency. In the first strategy, an attempt to reduce the number of cycles is made by increasing the frequency of patterns in which frequencies are below a pre-defined value. In the second strategy, the control is made by imposing that, in each saw cycle, at least a pre-defined number of objects are cut. We call the procedure *price and cut heuristic (PCH)* since the linear relaxation of the problems involved are solved by column generation (as stated in Sect. 2), constraints to exclude an undesired solution are added to the formulation in a dynamical way (cuts), and a feasible solution to the CSP-C is obtained by rounding the associated linear relaxations solutions.

It is assumed that a feasible solution to the 2D-CSP (1)–(3) that minimizes the total number of objects ($c_j = 1, \forall j$) is known. It can be found as follows. Let x^L be the solution to problem RL (obtained by the column generation method described in Sect. 2), and N_1 be the index set of all the cutting patterns generated to obtain it. An integer solution to the 2D-CSP (x^0) is obtained by taking, for each variable $x_j^L \neq 0$, $j \in N_1$, the smallest integer greater than or equal to x_j^L, i.e. $x_j^0 = \lceil x_j^L \rceil$. The associated number of saw cycles, y^0, is calculated according to (5). The solution (x^0, y^0) to the CSP-C and the associated total number of objects $z^0 = \sum_{j \in N_1} x_j^0$ is stored.

A parameter β will be used to diversify the set of solutions obtained with the procedure. It represents the minimum desired percentage usage of the cutting machine capacity in a saw cycle. For a matter of explanation and further analysis we will differentiate the index set of the columns generated to obtain x^0, defined as N_1, and the index set of the columns generated to obtain new solutions, defined as N_2.

In what follows an attempt to improve the solution (x^0, y^0) is proposed by dynamically reformulating the model 2D-CSP. We differentiate the two procedures in terms of the strategy used to generate cuts and call them PCH-F and PCH-C as presented in Sects. 3.1 and 3.2, respectively.

3.1 The PCH-F algorithm: cuts by frequencies

The strategy employed in the PCH-F algorithm for improving a feasible solution for the CSP-C is to control the number of cycles by imposing that, if a cutting pattern is used, then its frequency must be greater than or equal to a pre-specified value. The lower bound considered is based on β.

Let x^{it} be the current feasible solution to the CSP-C. The frequency of each cutting pattern is analysed in turn. If a cutting pattern is used and its frequency (x_j^{it}) satisfies condition (6) then the index j is included in the set $N_3 \subseteq N_1 \cup N_2$. Initially $N_2 = \emptyset$.

$$0 < x_j^{it} < f_{min} = \beta cap \tag{6}$$

The algorithm forces the variables in N_3 to have the value 0 or some value grater or equal to f_{min}. To model this disjunctive constraint, let λ_j be a binary variable associated with the cutting pattern j that is one if the cutting pattern can be used and zero otherwise; and define a parameter M such that its value is large enough to avoid that a feasible cutting pattern frequency becomes infeasible. The disjunction is modelled as in (7) (e.g. Williams 1990).

$$f_{min}\lambda_j \leq x_j \leq M\lambda_j, \quad j \in N_3 \tag{7}$$

The 2D-CSP is then reformulated by adding the constraints (7) and we obtain the Reformulated 2D-CSP (CSP-R1) stated by (8)–(12).

$$min \quad \sum_{j \in N_1 \cup N_2} x_j \tag{8}$$

$$s.a. \quad \sum_{j \in N_1 \cup N_2} A_j x_j \geq b \tag{9}$$

$$f_{min}\lambda_j \leq x_j \leq M\lambda_j, \quad j \in N_3 \tag{10}$$

$$x_j \in Z_+, \quad j \in N_1 \cup N_2 \tag{11}$$

$$\lambda_j \in \{0, 1\}, \quad j \in N_3. \tag{12}$$

The problem CSP-R1 is a two dimensional cutting stock problem with additional constraints and binary variables. To solve it, we adapt the methodology discussed in Sect. 2 for the 2D-CSP. At each iteration it, the constraints (11) are relaxed to $x_j \in R_+$, $j \in N_1 \cup N_2$, yielding a mixed integer problem (MIP1). The MIP1 problem is then solved by a commercial solver and the binary variable values, λ^{it}, are recovered. The values of λ in (MIP1) are fixed to λ^{it} and constraints (12) removed from it, yielding a linear relaxation of CSP-R1 (RL1).

The constraints (10) can be restated as (10-1) and (10-2). We observe that, if $\lambda_j^{it} = 0$ then the cutting pattern j is not used, constraints (10-1) and (10-2) together impose that x_j is set to zero. Otherwise, constraint (10-2) impose a lower bound to the cutting pattern frequency, and (10-1) becomes redundant. The RL1 problem is then defined by (8), (9), (10-1), (10-2) and (13). Note that most linear programming solvers include a preprocessing phase that can identify and remove redundant constraints from the problem. Therefore it does not matter keeping them in the problem formulation.

$$x_j \leq M\lambda_j^{it}, \quad j \in N_3 \tag{10-1}$$

$$x_j \geq f_{min}\lambda_j^{it}, \quad j \in N_3. \tag{10-2}$$

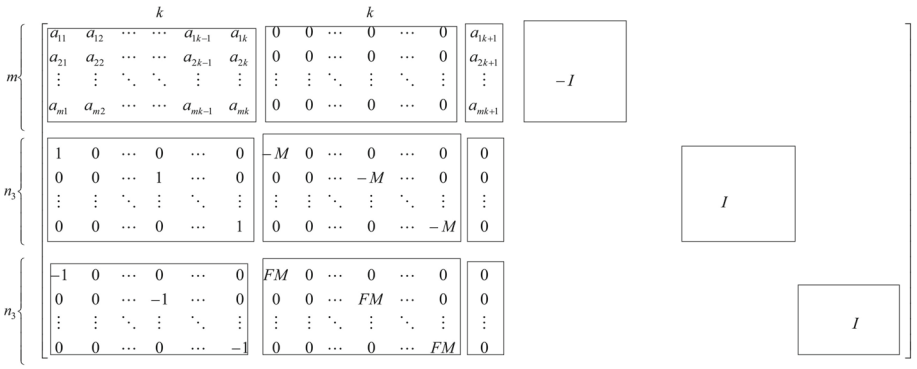


Fig. 4 Matrix structure of RL1

$$x_j \in R_+, \quad j \in N_1 \cup N_2 \tag{13}$$

The RL1 problem is then solved and the dual values associated to constraints (9), (10-1) and (10-2) are recovered (π_1, π_2, and π_3, respectively). To generate new cutting patterns, only the dual values associated to constraints (9), π_1, are used.

Suppose that $|N_1 \cup N_2| = k$ cutting patterns have been generated so far. The new cutting pattern, A_{k+1}, is accepted if $(1 - \pi_1 A_{k+1}) < 0$. For the associated variable x_{k+1} there are no constraints (10). Thus, if condition (14) is satisfied, a new column [composed by the cutting pattern A_{k+1} plus the zeros associated to constraints (10)] and the variable x_{k+1} are added to RL1 (set $N_2 = N_2 \cup \{k+1\}$). The process is repeated until no more good cutting patterns can be found, that is condition (14) is no longer satisfied.

$$z_{subp} > 1, \tag{14}$$

(z_{subp} is the optimal value of the sub-problem used to generate columns).

The structure of the RL1 constraint matrix, after adding the column associated to the variable x_{k+1} and columns associated to the slack variables, can be seen in Fig. 4, where $n_3 = |N_3|$, $FM = f_{min}$, and the capital letter "I" stands for the identity matrix of appropriated dimensions associated to the slack variables of constraints (9), (10-1) and (10-2). All the columns generated to RL1 are also included in CSP-R1.

When no more columns are generated to RL1, the linear relaxation solution x^{L1} is retrieved, and an integer feasible solution to CSP-R1 with the new cutting patterns added, $\bar{x}$, is computed by the same rounding procedure used to obtain x^0. The BRURED method (Waescher and Gau 1996) is then applied to this solution attempting to reduce the items excess caused by rounding the solution to a larger value. The BRURED method checks which variables $\bar{x}_j > 0$ can be reduced by one unit, while still meeting the demand. So, for every $\bar{x}_k > 0, k \in N_1 \cup N_2$, if the solution $\bar{x}_k$, satisfies the inequality (15):

$$a_{ik}(\bar{x}_k - 1) + \sum_{j \in N_1 \cup N_2;\, j \neq k} a_{ij} \bar{x}_j \geq b_i, \quad i = 1, \ldots, m, \tag{15}$$

then $x_k^{it} = \bar{x}_k - 1$, otherwise $x_k^{it} = \bar{x}_k$. The associated number of saw cycles is calculated according to (5), and the current solution (x^{it}, y^{it}) is stored. The solution x^{it} is analysed to check if there are patterns with frequency below f_{min} and if that is the case, the set N_3 is adapted and the process is repeated. The algorithm is interrupted if all the patterns have frequency above f_{min} or if the maximum number of iterations, *it_tot*, is reached.

Pseudocode 1 - PCH-F

1. BEGIN
2. **Input data**: x^L, the solution of RL (the linear relaxation of the 2D-CSP (1)-(3)) solved by column generation (as stated in Section 2)); N_1, the index set of the generated columns; β, a desired minimum percentage of the cutting machine capacity used in each saw cycle; cap; b.
3. Set $x_j^0 = \lceil x_j^L \rceil$, calculate $y_j^0 = \left\lceil \frac{x_j^0}{cap} \right\rceil$ for $j \in N_1$ and $z^0 = \sum_{j \in N_1} x_j^0$.
4. Store z^0, and (x^0, y^0) in S_F.
5. Set:
6. i) $f\_min = \beta cap$;
7. ii) $M = \sum_{i=1}^{m} b_i$.
8. iii) $it = 0, N_2 = \emptyset; N_3 = \emptyset$.
9. WHILE ($it \leq it\_tot$) AND ($\exists \quad 0 < x_j^{it} < f_{min}, j \in (N_1 \cup N_2)$) DO
10. FOR $j \in N_1 \cup N_2$ DO if $(0 < x_j^{it} < f_{min})$ then $N_3 = N_3 \cup \{j\}$
11. Solve MIP1 (The CSP-R1 with (11) replaced by $x_j \in R_+, j \in (N_1 \cup N_2)$)
12. Retrieve λ^{it} to obtain RL1 (given by (8), (9), (10-1), (10-2), and (13))
13. REPEAT
14. Solve RL1 and retrieve x^{L1} and π_1.
15. Generate a new cutting pattern.
16. If condition (14) is verified update N_2 with the new column.
17. UNTIL ($z_{subp} \leq 1$).
18. Let $\bar{x} = \lceil x^L \rceil$.
19. set $it = it + 1$.
20. Compute x^{it} by the BRURED Method and y_j^{it} according to (5) for $j \in N_1 \cup N_2$.
21. Store the solution (x^{it}, y^{it}) in S_F
22. END_ WHILE
23. Call Dominance
24. **Output data**: a set of undominated solutions (x, y)
25. END

Fig. 5 Pseudocode of the PCH-F

At the end of this process, there is a set S_F of solutions ($|S_F| \leq it\_tot$), $(x^i, y^i), i = 1, \ldots, |S_F|$, that were kept at each iteration. A simple procedure (Algorithm Dominance) is applied to the solutions in set S_F to determine the non-dominated ones (Deb 2004). The pseudocode of algorithm PCH-F is presented in Fig. 5.

The rounding procedures used to obtain these solutions (the ceiling function and the BRURED method) take into account only the demand constraints. So, most of them (if not all) might not satisfy the desired minimum percentage use of the saw capacity. However, they might lead to good solutions in terms of the total number of objects and the total number of saw cycles. In fact, most of them improve the feasible solution (x^0, y^0) obtained for the unrestricted 2D-CSP as can be seen in the results presented in Sect. 4.

3.2 The PCH-C strategy: cuts by cycles

The algorithm PCH-F was proposed as an attempt to improve a feasible solution of the (CSP-C). The strategy used was to control the total number of saw cycles by imposing that, if a cutting pattern is used, its frequency should be above a certain threshold value. The algorithm PCH-C uses a different strategy. It was developed by noting that in the solutions obtained with the PCH-F algorithm it is possible that the frequency of a given cutting pattern is above f_{min} but it includes an incomplete saw cycle with the number of objects below the required value (βcap).

Consider, for example, an instance of the CSP-R1 with $m = 3$, $cap = 20$ and $\beta = 80\,\%$. Let $x = (25, 18, 80)$ and $y = (2, 1, 4)$ be a solution obtained with the PCH-F algorithm. So, to cut the 25 objects according to cutting pattern 1, 1 complete cycle (20 objects) and an incomplete cycle with five objects (25 % of cap) are necessary. Likewise, for cutting pattern

2, an incomplete cycle with 18 objects (90 % of *cap*) is required, and for cutting pattern 3, 4 complete cycles (80 objects) are necessary. All the three cutting patterns frequencies are above f_{min}. However, one of the seven saw cycles necessary to cut all the 123 objects is an incomplete cycle that uses only 25 % of the machine maximum load and not the 80 % defined a priori.

The procedure PCH-C also aims to improve a given feasible solution to the CSP-C. It differs from the one described in Sect. 3.1 by imposing a new set of disjunctive constraints to cut off the non-desirable solutions.

At each step of the algorithm, a feasible solution to the CSP-C is analysed. If it involves an incomplete saw cycle with the number of objects below $f_{min} = \beta cap$, that is, if there is an x_j such that:

$$0 < \left(\frac{x_j}{cap} - \left\lfloor \frac{x_j}{cap} \right\rfloor \right) < \beta, \tag{16}$$

then the frequency of this cutting pattern has to be changed (either reduced or increased) in order to have incomplete cycles with at least f_{min} objects. This can be achieved by excluding the current value of x_j from the feasible set. Consider the following union of three intervals, $I_1 \cup I_2 \cup I_3$, defined in terms of the current number of saw cycles, y_j:

$$I_1 = [0, max\{0, (y_j - 2)cap\}], \tag{17}$$

$$I_2 = [max\{0, (y_j - 2)cap + f_{min}\}, (y_j - 1)cap], \tag{18}$$

and

$$I_3 = [(y_j - 1)cap + f_{min}, \infty[. \tag{19}$$

The set of values in the union $I_1 \cup I_2 \cup I_3$ exclude the current value of x_j, and some other values. More precisely, the set $I_1 \cup I_2 \cup I_3$ excludes all the values belonging to the open intervals:

$$](y_j - 2)cap, (y_j - 2)cap + f_{min}[\tag{20}$$

and

$$](y_j - 2)cap + f_{min}, (y_j - 1)cap + f_{min}[, \tag{21}$$

that are certain to have incomplete cycles with less than f_{min} objects. The intervals I_1 and I_3 includes some undesirable solutions that can be excluded with further subdivisions of them. If we impose that $x_j \in I_1 \cup I_2 \cup I_3$, then the current value of x_j becomes infeasible.

The disjunctive constraints: $x_j \in I_1$ or $x_j \in I_2$ or $x_j \in I_3$, can be modeled by linear constraints as follows (see e.g. Williams 1990). Let:

$$l_0 = max\{0, (y_j - 2)cap\} \tag{22}$$

$$l_1 = max\{0, (y_j - 2)cap + f_{min}\} \tag{23}$$

$$l_2 = (y_j - 1)cap \tag{24}$$

$$l_3 = (y_j - 1)cap + f_{min}. \tag{25}$$

If x_j does not satisfy condition (16), the constraints (26)–(29) are used to cut the undesirable solution.

$$\lambda_j^1 + \lambda_j^2 + \lambda_j^3 = 1, \quad \lambda_j^1, \lambda_j^2, \lambda_j^3 \in \{0, 1\} \tag{26}$$

$$x_j \leq l_0 + M\left(1 - \lambda_j^1\right) + M\lambda_j^2 + M\lambda_j^3 \tag{27}$$

$$-M\lambda_j^3 - M\lambda_j^1 - M\left(1 - \lambda_j^2\right) + l_1 \leq x_j \leq l_2 + M\left(1 - \lambda_j^2\right) + M\lambda_j^1 + M\lambda_j^3 \tag{28}$$

$$x_j \geq -M\lambda_j^1 - M\lambda_j^2 - M\left(1 - \lambda_j^3\right) + l_3. \tag{29}$$

Note that $\lambda_j^1 = 1$ implies $\lambda_j^2 = 0$ and $\lambda_j^3 = 0$. Thus, constraint (27) becomes active while constraints (28) and (29) are redundant. Similarly, when $\lambda_j^2 = 1$ or $\lambda_j^3 = 1$ the active constraints are (28) or (29), respectively.

Considering again that $x_1 = 25$, $y_1 = 2$, $cap = 20$, $\beta = 80\,\%$, and $f_{min} = 16$, according to (16) we have that:

$$\left(\frac{x_1}{cap} - \left\lfloor \frac{x_1}{cap} \right\rfloor\right) = \left(\frac{25}{20} - \left\lfloor \frac{25}{20} \right\rfloor\right) = (1.25 - 1) = 0.25 < 0.8 = \beta.$$

So the frequency of cutting pattern 1, although greater than f_{min}, allows an incomplete cycle that uses only 25 % of cap, that is less than the required minimum of 80 %. To cut off this solution, the values of I_1, I_2 and I_3 are set according to (17), (18) and (19): $I_1 = [0; \max\{0, (2-2)20\}] = [0; 0]$, $I_2 = [\max\{0, (2-2)4 + 16\}, (2-1)20] = [16; 20]$ e $I_3 = [(2-1)20 + 16; \infty[= [36; \infty[$, i.e.

$$x_1 = 0 \quad \text{or} \quad 16 \leq x_1 \leq 20 \quad \text{or} \quad x_1 \geq 36.$$

The following constraints cut off the solution $x_1 = 25$, as well as the solutions with x_1 in the interval]0, 16[and]20, 36[:

$$\lambda_1^1 + \lambda_1^2 + \lambda_1^3 = 1, \quad \lambda_1^1, \lambda_1^2, \lambda_1^3 \in \{0, 1\},$$
$$x_1 \leq 0 + M\left(1 - \lambda_1^1\right) + M\lambda_1^2 + M\lambda_1^3,$$
$$-M\lambda_1^3 - M\lambda_1^1 - M\left(1 - \lambda_1^2\right) + 16 \leq x_1 \leq 20 + M\left(1 - \lambda_1^2\right) + M\lambda_1^1 + M\lambda_1^3,$$
$$x_1 \geq -M\lambda_1^1 - M\lambda_1^2 - M\left(1 - \lambda_1^3\right) + 36.$$

The logic of algorithm PCH-C is similar to the one used in PCH-F. Let (x^{it}, y^{it}) be a feasible solution to CSP-C at iteration it. Each cutting pattern j is analysed in turn. If it satisfies (16), that is, the cutting pattern j is used and its frequency generates an incomplete saw cycle with a number of objects below β, then its index is included in the set $N_3 \subseteq N_1 \cup N_2$ (initially $N_2 = \emptyset$). The 2D-CSP problem is reformulated by adding constraints (26)–(29) to it. The reformulated cutting stock problem (CSP-R2) in this case is given by (30)–(38).

$$min \quad \sum_{j \in N_1 \cup N_2} x_j \tag{30}$$

$$s.a. \quad \sum_{j \in N_1 \cup N_2} A_j x_j \geq b \tag{31}$$

$$x_j \leq l_0 + M\left(1 - \lambda_j^1\right) + M\lambda_j^2 + M\lambda_j^3, \quad j \in N_3 \tag{32}$$

$$x_j \geq -M\lambda_j^3 - M\lambda_j^1 - M\left(1 - \lambda_j^2\right) + l_1, \quad j \in N_3 \tag{33}$$

$$x_j \leq l_2 + M\left(1 - \lambda_j^2\right) + M\lambda_j^1 + M\lambda_j^3, \quad j \in N_3 \tag{34}$$

$$x_j \geq -M\lambda_j^1 - M\lambda_j^2 - M\left(1 - \lambda_j^3\right) + l_3, \quad j \in N_3 \tag{35}$$

$$\lambda_j^1 + \lambda_j^2 + \lambda_j^3 = 1, \quad j \in N_3 \tag{36}$$

$$x_j \in Z_+, \quad j \in N_1 \cup N_2 \tag{37}$$

$$\lambda_j^1, \lambda_j^2, \lambda_j^3 \in \{0, 1\}, \quad j \in N_3. \tag{38}$$

The problem CSP-R2 is also a 2D-cutting stock problem with additional constraints and binary variables. To solve it, we use the same methodology employed to solve the CSP-R1 (described in Sect. 3.1). Let MIP2 be a relaxation of CSP-R2 obtained by replacing constraint (37) by $x_j \in R_+$, $j \in N_1 \cup N_2$. The MIP2 problem is then solved by a commercial solver and the binary variable values, λ^{it}, recovered. The values of λ in (MIP2) are fixed to λ^{it} and constraints (38) removed from it, yielding a linear relaxation of CSP-R2 (RL2). With the values of λ fixed, constraints (36) become redundant and removed. Constraints (32)–(35) can be restated as (39)–(42). The RL2 problem is then defined by (30), (31), (39)–(42), and (13).

$$x_j \leq l_0 + M\left(1 - \left(\lambda_j^1\right)^{it}\right) + M\left(\lambda_j^2\right)^{it} + M\left(\lambda_j^3\right)^{it}, \quad j \in N_3 \tag{39}$$

$$x_j \geq -M\left(\lambda_j^3\right)^{it} - M\left(\lambda_j^1\right)^{it} - M\left(1 - \left(\lambda_j^2\right)^{it}\right) + l_1, \quad j \in N_3 \tag{40}$$

$$x_j \leq l_2 + M\left(1 - \left(\lambda_j^2\right)^{it}\right) + M\left(\lambda_j^1\right)^{it} + M\left(\lambda_j^3\right)^{it}, \quad j \in N_3 \tag{41}$$

$$x_j \geq -M\left(\lambda_j^1\right)^{it} - M\left(\lambda_j^2\right)^{it} - M\left(1 - \left(\lambda_j^3\right)^{it}\right) + l_3, \quad j \in N_3 \tag{42}$$

The RL2 problem is then solved and the dual values associated to constraints (31) are recovered (π_1).

New cutting patterns are generated using only the dual values associated to constraints (31), π_1. Suppose that $|N_1 \cup N_2| = k$ cutting patterns have been generated so far. The new cutting pattern, A_{k+1}, is accepted if the condition (43) is satisfied since for the associated variable x_{k+1} there are no constraints (39)–(42) (the associated row values in the new column are zero). The column A_{k+1} and the variable x_{k+1} are added to RL2 (set $N_2 = N_2 \cup \{k+1\}$) and the process is repeated until no more good cutting patterns can be found, that is condition (43) is no longer satisfied. All the columns generated to RL2 are also included in CSP-R2.

$$z_{subp} > 1, \tag{43}$$

(z_{subp} is the optimal value of the sub-problem used to generate columns).

When no more columns are generated in RL2, the linear relaxation solution x^{L2} is retrieved, and an integer feasible solution to CSP-R2 is obtained using the same rounding procedures used in the PCH-F (the ceiling function and the BRURED method). The associated number of saw cycles is calculated according to (5), and the solution (x^{it}, y^{it}) is stored in S_C. The new solution x^{it} is analysed to check if there are patterns satisfying (16) and if that is the case, the set N_3 is adapted and the process is repeated. The process of adding cuts is interrupted if, in the current solution, the number of objects in an incomplete saw cycle is above β or if the maximum number of iterations, *it_tot*, is reached.

Algorithm Dominance is also applied to the solutions in set S_C for determining the non-dominated ones. As observed in the end of Sect. 3.1, the solutions obtained using PCH-C might not satisfy the desired minimum percentage use of the saw capacity in some of the saw

Pseudocode 2 - PCH-C

1. BEGIN
2. Change the following lines of Pseudocode 1 - PCH-F (Figure 5)
3. **Line 4**: Store z^0, and (x^0, y^0) in S_C.
4. **Line 6**: i) Define β as the minimum percentage of the cutting machine full capacity used in each saw cycle.
5. **Line 9:** WHILE ($it \leq it\_tot$) AND ($\exists\, x_j^{it}$ that satisfies constraint (16)) DO:
6. **Line 10:** FOR all the patterns $j \in N_1 \cup N_2$ that satisfies (16) DO $N_3 = N_3 \cup \{j\}$
7. **Line 11:** Solve MIP2 (The CSP-R2 with (38) replaced by $x_j \in R_+, j \in (N_1 \cup N_2)$
8. **Line 12:** Retrieve λ^{it} to obtain RL2 (given by (30), (31), (39)-(42), and (13))
9. **Line 14:** Solve RL2 and retrieve x^{L2} and π_1.
10. **Line 16:** If condition (43) is verified update N_2 with the new column.
11. **Line 21:** Store the solution (x^{it},y^{it}) in S_C and set $it = it + 1$
12. END

Fig. 6 Changes in pseudocode PCH-F to obtain PCH-C

cycles, although they might lead to good solutions in terms of the total number of objects and the total number of saw cycles. Figure 6 highlights the lines of the PCH-F pseudocode that must be modified in order to get the PCH-C algorithm. The two algorithms discussed in this section were proposed having in mind the two-dimensional case. However to use the PCH-F and the PCH-C heuristics in the one-dimensional case just change the pricing sub-problem in line 15 of both pseudocodes to generate feasible one-dimensional cutting patterns.

4 Numerical study

This section presents the results obtained in the numerical study conducted to analyse the algorithms PCM-F and PCM-C described in Sect. 3, and to compare their results with the practice of a typical small scale furniture plant. The algorithms were implemented using the FICO Xpress Optimization Suite (FICO 2015) which includes a mathematical programming language (Xpress-Mosel) and a solver (Xpress-optimizer). All the runs were executed on an Intel Core 2 Quad 2.33 GHZ/3.23 Gb RAM.

The algorithms PCH-F and PCH-C were tested using four different values to diversify the solutions, β (the minimum percentage of capacity utilization of the cutting machine in each cycle): 30, 50, 80 and 100 %. The parameter M was taken as: $M = \sum_{i=1}^{m} b_i$. For all the instances used in this study, the constraint matrix associated to the 2D-CSP was initialized with maximal homogenous cutting patterns. The dominance criterion used to analyse the set of solutions obtained with each algorithm is the total number of objects and the total number of cycles.

To avoid infeasible cutting patterns, the dimensions of all the items and of the object are increased by σ as discussed in Sect. 2.1. The cutting patterns were generated using the 3t-group mixed integer programming model proposed in Yanasse and Morabito (2008), allowing item rotation. The maximum execution time of the Xpress-optmizer for solving the linear optimization problems 2D-CSP, MIP1, RL1, MIP2, and RL2 in PCH-F and PCH-C was limited to 600 s. The maximum execution time for solving the pricing sub-problem to generate cutting patterns was set to 60 s and the maximum number of iterations was set to $it\_tot = 15$. These parameters were defined based on some preliminary computational experiments (see details in Toscano et al. 2010). Given the computational difficulties to generate 2D cutting patterns, in the numerical study described in this section, the column generation method is

Correction in Pseudocodes 1 and 2

1. BEGIN
2. Change the following line of Pseudocode 1 - PCH-F (Figure 5) and of Pseudocode 2- PCH-C (Figure 6)
3. **Line 17:** UNTIL ($z_{subp} \leq 1$) OR (a column is regenerated) .
4. END

Fig. 7 Change in pseudocodes PCH-F and PCH-C—column regeneration

employed as a heuristic since the MIP optimizer used to solve the pricing sub-problem is halted before optimality is proved.

In the preliminary tests, it was noticed that, sometimes, during the column generation procedure employed to solve RL1 and RL2, the solution of the pricing sub-problem repeated a previously generated one. That is, the cutting pattern just generated is already in the cutting patterns sub-matrix A. Step 17 in algorithm PCH-F and PCH-C has been modified to halt the column generation process when this happen (Fig. 7).

4.1 The data set

As stated in Sects. 1 and 2, the focus of the present work is on small scale factories that produce furniture from rectangular wooden plates. In this study, we used data collected in a representative factory of the furniture sector in Brazil. The most important differences in the production process of small-scale factories are at a specific production stage of a given product or in the type of machines used. A common feature among these small-scale factories is to design the products taking into account the cutting stock problem. That is, when defining a new type of furniture, the cutting patterns and their frequencies are also defined taking into account that a certain fixed number of products (a single lot of products) will be produced. The number of products in a lot is defined according to several parameters, among them the type of furniture, the cost of the plates and the factory capacity in a working day. For example, in the visited factory, a single lot of a special type of wardrobe (product A5P) is defined as 60 units. More details of the production processes of furniture factories in Brazil can be found, for example, in Calderon (2013) and Rangel and Figueiredo (2008).

At the time the data was collected, each product was composed of items cut from objects of different thickness (3, 9, 15, 18, 20, and 25), all with the same size, $(L, W) = (2750, 1830)$ (units are all given in mm). Therefore, in order to get all the necessary items to produce a lot of a given product it is necessary to solve one cutting stock problem for each object thickness. Results associated to five products (A5P, A4P, A3P, Cmd and Crd) are reported. The main characteristics of the test instances are described in Table 1. The single-lot instances were named according to the product and object thickness. For example, instance A5P-15 refers to the test set associated with the items cut from the 15 mm object to produce one single-lot of the product A5P, and Cmd-09 is the set associated with the items cut from the 9 mm object to produce the product Cmd. For each instance, the total number of ordered items (m) and the total demand (TD) is shown. We also considered double and triple lots of products (lots derived by joining two or three single lots, respectively). In this case, the items with the same thickness are grouped and the demand of items with same sizes added. The products A5P, A4P and A3P are also composed of items cut from the 9 mm objects. However, there is only one item size for each product and the demand is low (less than 50 for each product), therefore the items of 9 mm are considered only in the instances associated to the double lots. The cutting machine considered has saw thickness equal to $\sigma = 4$ mm and saw height equal to $h = 60$ mm.

Table 1 Summary of the data used to generate the instances

Instance	m	TD	Instance	m	TD
(a) Single lots			*(b) Double/triple Lots*		
A5P-03	8	1080	A5P_Cmd-03	10	2680
A5P-12	2	480	A5P_Cmd-09	4	1320
A5P-15	7	880	A5P_Cmd-12	6	3360
A5P-20	9	2720	A5P_Cmd-15	10	2160
A5P-25	3	400	A5P_Crd-03	11	3780
A4P-03	7	1070	A5P_Crd-12	4	3180
A4P-12	2	540	A5P_Crd-15	9	2380
A4P-15	7	855	Cmd_Crd-03	6	4300
A4P-20	9	2610	Cmd_Crd-12	8	5580
A4P-25	3	315	Cmd_Crd-15	6	2780
A3P-03	5	1080	A5P_A4P_A3P-03	11	3230
A3P-12	2	720	A5P_A4P_A3P-12	2	1740
A3P-15	6	960	A5P_A4P_A3P-15	7	2695
A3P-20	10	2940	A5P_A4P_A3P-20	12	8270
A3P-25	3	400	A5P_A4P_A3P-25	3	1115
Cmd-03	3	1600			
Cmd-09	3	1280			
Cmd-12	6	2880			
Cmd-15	4	1280			
Crd-03	3	2700			
Crd-12	2	2700			
Crd-15	2	1500			

4.2 Results 1: Comparison PCH-F × PCH-C

At first, the instance A5P_Cmd-03 is used to illustrate the details of the computational behaviour of the heuristics PCH-F and PCH-C. Each one of the two algorithms might generate up to 15 solutions ($it\_tot = 15$) in each one of the four runs. In Tables 2 and 3 only the non-dominated solutions for each value of β are presented. For each solution, we show the total number of objects (Ob), cycles (Cy), cutting patterns (Pt), generated columns $|N_1 \cup N_2|$ (GC), cuts (Ct), the total number of iterations (It_T), the iteration number in which the non-dominated solution was found (It_S), and the value of β. There is also an interest in knowing, for each solution, the total number of incomplete cycles in which the number of objects is below f_{min}. This value is shown in the tables as (Fi). For this instance, a maximum load of the saw machine in each cycle is 20 objects ($cap = 20$). To represent the solution of the 2D-CSP ($it = 0$) the columns are filled with the symbol "–", except for the Ob, Cy and Pt columns.

For all values of β, the heuristic PCH-F terminated because all the cutting patterns frequencies were above f_{min}, with the total number of iterations ranging from 6 to 12. However, all the solutions have cycles that do not satisfy the desirable minimum use of the saw capacity ($Fi > 0$). Still, it was successful in finding solutions with the total number of cycles below the number given by the 2D_CSP solution. All the seven non-dominated solutions reduced the total number of cycles with a maximum increase in the total number of objects of 10 %.

Table 2 Non-dominated solutions for each β value for the PCH-F—instance A5P_Cmd-03

Sol	Ob	Cy	Fi	It_S	It_T	Pt	GC	Ct	β (%)
0	260	20	–	–	–	10	28	–	–
1	256	18	3	6	6	9	33	26	30
2	261	17	3	7	7	9	35	36	50
3	260	19	6	3	7	11	35	36	50
4	260	18	7	4	10	8	43	64	80
5	264	16	4	10	10	8	43	64	80
6	260	18	7	3	12	8	39	68	100
7	262	17	7	2	12	7	39	68	100

Table 3 Non-dominated solutions for each β value for the PCH-C—instance A5P_Cmd-03

Sol.	Ob	Cy	Fi	It_S	It_T	Pt	GC	Ct	β (%)
0	260	20	–	–	–	10	28	–	–
1	255	17	0	11	11	10	37	95	30
2	260	19	5	1	6	12	30	100	50
3	265	18	3	2	6	12	30	100	50
4	266	16	0	3	6	9	30	100	50
5	261	18	7	1	7	12	35	155	80
6	266	16	3	3	7	9	35	155	80
7	270	15	1	5	7	10	35	155	80
8	263	15	3	3	15	9	42	315	100

The total number of objects returned by the solution of the PCH-F with $\beta = 30\,\%$ was smaller than the Ob value associated to the 2D-CSP solution. At this point, we need to remember that the solution for the 2D-CSP used as the initial value for the PCH-F and the PCH-C procedures might not be optimal. Therefore, it is possible that the alterations in the problem formulation introduced by the cuts and the new columns guide the solver to find solutions with a smaller number of objects. This happened with both procedures and for other instances and other values of β.

The two non-dominated solutions obtained with $\beta = 100\,\%$ were found at early iterations of the total of 12 iterations (iterations 2 and 3). As the value of β increased, the number of new columns and cuts associated to the non-dominated solutions increased as well, although the number of new columns for the solutions with $\beta = 100\,\%$ is smaller than the number associated to $\beta = 80\,\%$.

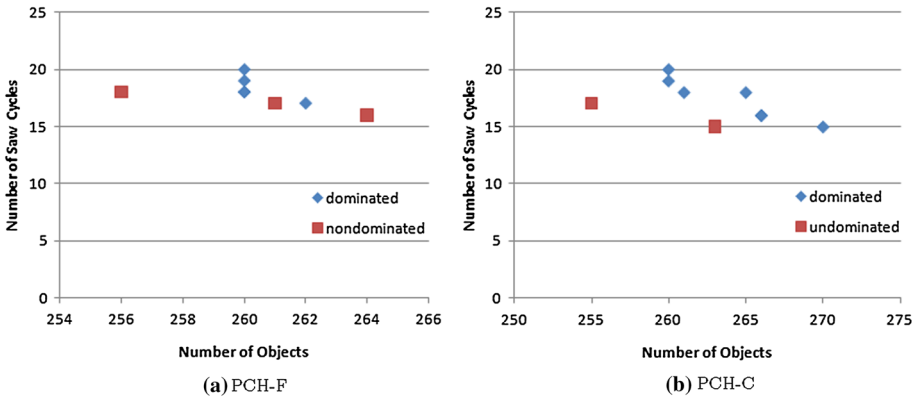
The constraint matrix associated to the A5P_Cmd-03 instance of the 2D-CSP model was initialized with 18 maximal homogenous cutting patterns. Ten $3t$-group cutting patterns were generated and included in the matrix during the column generation procedure applied to solve it, so $|N_1| = 28$. The number of new columns included in the cutting patterns matrix during the four runs of the PCH-F ranges from 5 to 15 ($5 \leq |N_2| \leq 15$). Note that it is possible to include more than one new column at each iteration of the heuristic (steps 13–17 in Fig. 5), or none at all. For example, there are 43 patterns in the matrix associated to the solutions number 4 and 5, and for this value of β, there were 10 iterations of the PCH-F algorithm, as opposed to solution number 6 in which $|N_2| = 11$, and $It\_T = 12$. So, although there were iterations in which no new columns were obtained, the new constraints (7) included in Step 10 influenced positively the solution process.

The iterative use of constraints (32)–(36) in the heuristic PCH-C guided the search for solutions that comply with the minimum requirement of the saw capacity in each cycle (Table 3). In fact, two of the eight non-dominated solutions obtained with this heuristic have $Fi = 0$ ($\beta = 30\,\%$ and $\beta = 50\,\%$). For the other six solutions, the Fi values ranges from 3 to 5, except for one solution obtained in the run with $\beta = 80\,\%$. All the solutions have the total number of cycles below 20 which is the number of cycles associated to the solution of the 2D-CSP. For $\beta = 100\,\%$ the total number of cycles was reduced by 25 % with an increase in the Ob value of less than 2 %. All but one of the non-dominated solutions was obtained at early iterations of the PCH-C heuristic. Three out of the four runs of the heuristic PCH-C were interrupted because condition (16) was satisfied. However, except for the run using $\beta = 30\,\%$, the solutions obtained in the last iterations were dominated by the solutions obtained at earlier iterations. As in the runs of the PCH-F heuristic, new $3t$-group cutting patterns were not generated in all the iterations. For example, for the run with $\beta = 50\,\%$ only two new columns were generated in a total of six iterations. Still, the non-dominated solutions found in this run have the total number of cycles ranging from 16 to 19. Compared with the 2D-CSP solution, the PCH-C solutions had a reduction in the Cy value of up to 25 % with an increase in the associated Ob value of less than 4 %. As the level of saw usage in each cycle increases so does the number of cuts added to the CSP-R2 formulation. As expected, the total number of cuts in the PCH-C heuristic is higher than the ones in the PCH-F.

It is interesting to note that there are several solutions with the same number of cutting patterns and different number of saw cycles in the solutions of the PCH-F and of the PCH-C, even when the percentage of the cutting machine capacity usage is the same. Also, there are solutions with the same number of objects and cycles and different number of cutting patterns. These observations show that reducing the number of cutting patterns does not guarantee a reduction in the number of saw cycles. The total number of cutting patterns used in the solutions of the PCH-F is smaller than in the solution of the 2D-CSP for all but one solution. This is not the case for the solutions of the PCH-C heuristic. In five out of the eight solutions, the total number of cutting patterns has the same value or is higher than in the 2D-CSP solution. At this point, it is good to remember that the cuts used in the PCH-C algorithm allows that the frequency of a given cutting pattern might be reduced or increased in order to satisfy the desirable level of saw usage in each cycle. This adjustment is not allowed in the PCH-F heuristic. In the latter case, either the frequency of a given cutting pattern ($x_j, j \in N_3$) is set to zero or it is set to a value above f_{min}. The PCH-C heuristic allows that if a good cutting pattern is found, it can be kept in the solution with its frequency reduced as opposed to what happens in the PCH-F algorithm in which a cutting pattern is only maintained if the increase in its frequency leads to a better solution value.

Figure 8 illustrates the dominance relationship among the solutions given by the two algorithms shown in Tables 2, 3, PCH-F in Fig. 8a) and PCH-C in Fig. 8b, respectively. Three overall non-dominated solutions were obtained with algorithm PCH-F, one uses 256 objects and 18 cycles, the second one with 261 objects and 17 cycles and the third one uses 264 objects and 16 cycles. With algorithm PCH-C only two non-dominated solutions were obtained, one with 255 objects and 17 cycles and the other one with 263 objects and 15 cycles. For this instance the non-dominated solutions obtained with PCH-C dominate all those obtained with the algorithm PCH-F. As will be seen next, this is not always the case.

The algorithms PCH-F and PCH-C were also tested with the other 36 instances described in Table 1, a total of 22 instances considering the production of only one type of furniture



(a) PCH-F **(b)** PCH-C

Fig. 8 Dominance relationship between the solutions of the PCH-F and PCH-C, respectively, for the instance A5P_Cmd-03

(single lot) and 15 instances that combine two or three lots of furniture in a run (double/triple lot) (Tables 4, 5; respectively). For these instances, we restricted the analysis to the main features of the solutions: total number of objects (Ob), cycles (Cy) and the total number of incomplete cycles in which the number of objects is below f_{min} (Fi).

The PCH-F algorithm found solutions that dominated the 2D-CSP solutions for 11 of the 22 single lot instances (Table 4). For three of the remaining instances (A3P-12, Cmd-03, Cmd-15), the cutting patterns frequencies of the 2D-CSP solution were above the minimum frequency required (did not satisfy condition (6)) and therefore the PCH-F algorithm halted at $it = 0$ in all the four runs. For the other remaining six instances, the 2D-CSP solution dominated all the PCH-F solutions. We did not find any explanation for the lack of improvement of the solutions provided by PCH-F for these nine instances. These instances have $m \in [2, 6]$ and TD $\in [315, 2940]$ with an average demand *per* item equal to 434.92. Among the 13 instances to which the PCH-F algorithm returned non-dominated solutions, there are some with similar characteristics and the PCH-F solutions dominate or are non-dominated by the associated 2D-CSP solution. For example, the instances Cmd-09 and Cmd-12 have m in the same interval and the demand *per* item equal to 426.67 and 480.00, respectively. Among the 16 non-dominated solutions found by the PCH-F heuristic, only two had $Fi = 0$. The average number of incomplete cycles that do not comply with minimum requirements of saw usage is 2.93.

The results of the PCH-C heuristic are better than the ones of the PCH-F. For all but five single lot instances, the solutions of the former dominate or are non-dominated by the solutions of the 2D-CSP. For instance A3P-12, the PCH-C algorithm halted at $it = 0$ because all the cutting patterns with non-zero frequencies in the 2D-CSP solution satisfy the minimum requirement of saw usage for all β values. The use of the diversify parameter β was more efficient in the PCH-C algorithm that returned different non-dominated solutions for six instances as opposed to the PCH-F algorithm that returned alternative solutions for only three instances. The Cy value associated to the PCH-C solutions is smaller or equal to the Cy values of PCH-F solutions. In terms of Fi values, PCH-C solutions are also superior to the PCH-F solutions. For ten instances, the former algorithm returned solutions with $Fi = 0$ and the maximum value is $Fi = 4$. These results reinforce that the iterative use of constraints (32)–(36) was efficient to reduce the total number of cycles and the number of incomplete cycles with saw usage below f_{min}.

Table 4 Results PCH-F × PCH-C—single lot instances

Instance	PCH-F				PCH-C			
	Ob	Cy	Fi	β (%)	Ob	Cy	Fi	β (%)
A5P-03	134	9	0	30	133	1	4	30
					141	9	0	50
A5P-12	10	3	–	–	10	2	0	100
A5P-15	57	17	2	30	57	16	4	100
A5P-20	31	12	3	100	31	11	2	80
A5P-25	4	2	2	100	4	2	0	80
A4P-03	122	11	–	–	122	11	–	–
	128	10	5	50	129	10	3	50
	137	9	4	80	133	9	0	50
A4P-12	11	3	–	–	11	3	–	–
A4P-15	55	16	3	100	57	15	2	80
A4P-20	28	12	4	80	28	11	0	50
A4P-25	4	2	1	100	3	2	0	80
A3P-03	118	9	–	–	118	9	–	–
					120	8	0	50
					128	7	0	80
A3P-12	15	3	–	–	15	3	–	–
A3P-15	64	18	–	–	62	18	1	100
					63	16	4	100
A3P-20	29	11	3	100	30	11	1	100
A3P-25	4	2	1	100	4	2	0	80
Cmd-03	124	7	–	–	124	7	–	–
Cmd-09	7	3	1	50	8	2	0	50
	10	2	0	80	7	3	3	100
Cmd-12	46	12	–	–	46	10		80
	48	11	2	80				
	45	14	8	100				
Cmd-15	64	16	–	–	64	16	–	–
Crd-03	72	5	–	–	72	5		–
					73	4	0	80
Crd-12	24	6	2	100	23	6	2	100
					24	5	1	100
Crd-15	46	12	–	–	46	12	–	–

It is expected that the diversity and/or the higher demands of the items in a double or triple lot might allow the generation of better cutting patterns and might improve the cutting machine productivity. From the results shown in Table 5 we can say that this strategy gave good results in terms of the reduction of the total number of saw cycles. From the 15 instances, the PCH-F found solutions that dominated the 2D-CSP solution for 60 % of the instances and the PCH-C returned solutions that dominated the 2D-CSP solutions for all but one instance. The triple lot instance A5P_A4P_A3P-12 shares two items with the single lot instances A5P-12, A4P-12 and A3P-12 (see Toscano et al. 2010). For this instance, neither the modifications

Table 5 Results PCH-F × PCH-C—double/triple lot instances

Instance	PCH-F				PCH-C			
	Ob	Cy	Fi	β (%)	Ob	Cy	Fi	β (%)
A5P_Cmd-03	256	18	3	30	255	17	0	30
	261	17	3	50	263	15	3	100
	264	16	4	80				
A5P_Cmd-09	10	3	0	50	9	3	1	50
					12	2	0	100
A5P_Cmd-12	55	13	2	80	54	12	2	80
A5P_Cmd-15	120	33	6	100	121	33	1	30
A5P_Crd-03	203	16	1	30	208	15	4	50
	204	15	0	30	209	14	2	50
	205	14	1	50	215	13	1	50
	220	13	3	80	219	12	7	100
A5P_Crd-12	34	9	–	–	34	8	0	100
A5P_Crd-15	103	31	–	–	100	28	1	50
					101	27	1	50
Cmd_Crd-03	193	12	3	80	192	14	6	100
	191	13	1	30	193	12	1	30
					196	11	1	80
Cmd_Crd-12	70	18	5	80	71	16	0	50
	72	17	5	100				
Cmd_Crd-15	110	30	–	–	107	29	5	100
					108	28	2	80
A5P_A4P_A3P-03	374	23	–	–	373	23	3	100
					374	22	1	50
					379	21	6	30
					380	20	4	100
A5P_A4P_A3P-12	35	8	–	–	35	8	–	–
A5P_A4P_A3P-15	172	45	–	–	171	44	2	100
					170	45	5	80
A5P_A4P_A3P-20	83	31	7	80	84	30	5	100
					82	31	6	100
A5P_A4P_A3P-25	10	6	2	80	10	5	0	100

to the cutting patterns used nor the changes in their frequencies made by the two heuristics were enough to improve the 2D-CSP solution.

For all instances, the PCH-C algorithm returned non-dominated solutions with the total number of cycles smaller or equal than the PCH-F algorithm, although the associated number of objects is not necessarily smaller. As for the *Fi* value, in four instances, the former algorithm returned solutions that comply with the minimum requirement of saw usage in each incomplete cycle. Considering the results in Tables 4 and 5, we can say that, in situations with a high variety of items and high demand values, the use of constraints (32)–(36) was efficient to reduce the total number of cycles and the number of incomplete cycles with less

Table 6 Comparison between the PCH-F and PCH-C algorithm—average values

Lot type	Algorithm	Iterations	Cuts	Time (s)	Time/iter
Single	PCH-F	4.52	26.14	655.05	110.88
	PCH-C	8.43	109.03	943.37	96.50
Double/triple	PCH-F	5.73	32.73	974.84	132.38
	PCH-C	9.75	192.17	1518.54	164.81

than f_{min} objects. Considering all but one of the 15 double/triple lot instances, the average total number of items is 7.64, the average total demand is 3345, and the average demand per item equal to 456.72. For the single lot instances, these numbers are 4.82, 1408.61 and 354, respectively. The PCH-C algorithm has 14.6 and 12.5 % of the incomplete cycles with less than f_{min} objects for the single and the double/triple lot instances respectively while the PCH-F algorithm has 30.5 and 18 %, respectively.

As for the computational effort necessary to obtain the results reported in the above tables, a summary is given in Table 6 in terms of the average value of the total number of iterations, cuts, solution time (in seconds), and the time spent per iteration (time/iter). The CPU time used to solve the 2D-CSP problem, whose solution is an input to the heuristics, is included in the CPU times shown.

The two algorithms have similar behaviour for both the single and the double/triple lot instances. This is expected because, although the total demand and variety of items is higher for most instances of the double/triple lot instances, some instances of the two groups have similar number of items and total demand. The computational effort of the proposed strategies depends upon the variety of items and the total demand. For all instances, the bulk of the CPU time is spent solving the pricing sub-problem either in the solution of the 2D-CSP or in step 15 of both the PCH-F and PCH-C heuristics. The time spent in the re-optimization of the CSP-R1 in PCH-F or of the CSP-R2 in the PCH-C heuristics after adding new columns and/or rows is very small compared to the effort of solving the MIP problem to generate $3t$-group cutting patterns in spite of the new binary variables associated to the cuts that are added to the formulations.

The number of cuts observed depends on the cutting pattern frequencies: in the case of the PCH-F heuristic, it is a multiple of two and for the PCH-C a multiple of 5. The high number of iterations and cuts of the PCH-C heuristic confirms that the expression (16) is more effective than the expression (6) to evaluate if the frequency of a given cutting pattern satisfies the minimum requirement of saw usage in each incomplete cycle. The number of new columns generated for the reformulated problems in each algorithm ($|N_1 \cup N_2| - |N_1|$) is small. The time spent on each iteration of the algorithms is no longer than 180 s. That is, on average, less than three columns were generated at each iteration since the solver is interrupted after 60 s when solving the pricing sub-problem. In all cases, the solver did not find its optimal solution.

The PCH-C heuristic is better than the PCH-F heuristic since, for most instances, it returns solutions with a smaller total number of cycles and a smaller total number of incomplete cycles that do not comply with the requirement. However, it requires more computational effort in terms of CPU time and memory. Therefore both algorithms might be considered when solving the CSP-C problem. In the following section we compare the solutions returned by the algorithms with the solutions employed by Factory V.

Table 7 Heuristics (PCH-F and PCH-C) × Factory V—single lot instances

Instance	Heuristics				Factory V		
	Ob	Cy	Pt	SM	Ob	Cy	Pt
A5P-03	134	9	7	PCH-F	134	11	7
	133	11	7	PCH-C			
A5P-12	10	2	2	PCH-C	11	3	2
A5P-15	57	16	9	PCH-C	57	17	6
A5P-20	31	11	6	PCH-C	32	14	8
A5P-25	4	2	2	PCH-C	5	4	3
A4P-03	122	11	6	CSP	124	10	5
	128	10	6	PCH-F			
	133	9	8	PCH-C			
A4P-12	11	3	2	PCH-C/F	12	3	2
A4P-15	57	15	7	PCH-C	55	16	6
	55	16	6	PCH-F			
A4P-20	28	11	8	PCH-C	27	12	6
A4P-25	3	2	2	PCH-C	5	4	3
A3P-03	118	9	5	2D-CSP	126	7	3
	120	8	6	PCH-C			
	128	7	4	PCH-C			
A3P-12	15	3	2	–	15	4	2
A3P-15	62	18	8	PCH-C	63	18	5
	63	16	6	PCH-C			
A3P-20	29	11	9	PCH-F	29	13	6
A3P-25	4	2	1	PCH-C	3	2	1
Cmd-03	124	7	2	2D-CSP	124	7	2
Cmd-09	7	3	3	PCH-F	7	2	1
	8	2	2	PCH-C			
Cmd-12	46	10	6	PCH-C	46	13	6
	45	14	10	PCH-F			
Cmd-15	64	16	4	2D-CSP	64	18	3
Crd-03	72	5	3	2D-CSP	74	5	2
	73	4	3	PCH-C			
Crd-12	23	6	3	PCH-C	24	6	2
	24	5	3	PCH-C			
Crd-15	46	12	2	2D-CSP	46	12	2

4.3 Results 2: Comparison heuristic solutions × factory practice

As shown in Sect. 4.2, for several instances the PCH-F and PCH-C heuristics returned a set of non-dominated solutions. In this section we compare the overall non-dominated solutions given by the two solution strategies with those used in practice at the factory. In Tables 7 and 8 the total number of objects, cuts and patterns associated with the non-dominated solutions for the single lot and double/triple lot instances respectively are shown. In the cases of a tie between the total number of objects and saw cycles the parameter used as a tiebreaker is first

Table 8 Heuristics (PCH-F and PCH-C) × Factory V—double/triple lot instances

Instance	Heuristics				Factory V		
	Ob	Cy	Pt	SM	Ob	Cy	Pt
A5P_Cmd-03	255	17	10	PCH-C	258	18	9
	263	15	9	PCH-C			
A5P_Cmd-09	9	3	3	PCH-C	9	3	2
	12	2	2	PCH-C			
A5P_Cmd-12	54	12	7	PCH-C	57	16	8
A5P_Cmd-15	120	33	9	PCH-F	121	35	9
A5P_Crd-03	203	16	12	PCH-F	208	16	9
	204	15	11	PCH-F			
	205	14	11	PCH-F			
	215	13	10	PCH-C			
	219	12	8	PCH-C			
A5P_Crd-12	34	8	5	PCH-C	35	9	4
A5P_Crd-15	100	28	9	PCH-C	103	29	8
	101	27	9	PCH-C			
Cmd_Crd-03	191	13	7	PCH-F	198	12	4
	193	12	7	PCH-C			
	196	11	7	PCH-C			
Cmd_Crd-12	70	18	10	PCH-F	70	19	8
	71	16	9	PCH-C			
	72	17	5	PCH-F			
Cmd_Crd-15	107	29	7	PCH-C	108	30	5
	108	28	9	PCH-C			
A5P_A4P_A3P-03	373	23	12	PCH-C	384	25	12
	374	22	12	PCH-C			
	379	21	11	PCH-C			
	380	20	11	PCH-C			
A5P_A4P_A3P-12	35	8	2	2D-CSP	37	9	9
A5P_A4P_A3P-15	171	44	8	PCH-C	177	49	10
	170	45	7	PCH-C			
A5P_A4P_A3P-20	82	31	14	PCH-C	88	37	18
	84	30	12	PCH-C			
A5P_A4P_A3P-25	10	5	4	PCH-C	10	7	5

the total number of patterns and second the total number of incomplete cycles with less than f_{min} objects.

Considering all the 22 single lot instances (Table 7), the two procedures returned a set of 25 non-dominated solutions. Among them, the PCH-C solutions dominate or are non-dominated by the PCH-F ones for 17 instances. Together, the solutions obtained by the two algorithms dominate the Factory V solutions for 13 instances. For all but two solutions, the total number of cycles returned by the heuristics is less than or equal to the Factory V ones.

In this work, the reduction of the total number of cutting patterns was not considered when searching for good solutions for the CSP-C problem. But, we noticed that the total number of cutting patterns in the Factory V solutions is smaller than in the heuristics solutions in 12 single lot instances. We observe that at the time the data was collected, the Factory V managers took a lot of care to find a solution with the smallest possible number of cutting patterns. A reduction of the number of cutting patterns does not necessarily reduce the number of cycles as can be verified in the solutions given for instance A5P-15. The PCH-C and the Factory V solution have the same number of objects, the former has 16 cycles and 9 cutting patterns and the latter has 17 cycles and 6 cutting patterns.

The results for the double/triple lot instances are similar, except that the PCH-C heuristic solutions are much better (Table 8). This confirms that the scope for improvements appears in situations with a high diversity of items and high demand. For all instances, the heuristics returned solutions in which the number of cycles is smaller than the number of cycles of the Factory V solutions. Using the Factory V solution as reference, on average, the total number of cycles is reduced in 12 % with an increase in the total number of objects of 8 %. For the single lot instances, on average there is a reduction of 9 % in the number of cycles and an increase of 11 % in the number of objects.

The non-dominated solutions given by the two heuristics increase the options of the decision makers. For most instances, in situations with a high diversity of items and high demand, the heuristics found solutions that improved on those used in practice at the factory. It is worth observing that the production manager of Factory V might take up to a week to take a decision related to what cutting patterns to use and how many objects must be cut to meet the demand for furniture.

4.4 Summary of the results

A numerical study was conducted to analyse the algorithms PCM-F and PCM-C using data collected at a small scale furniture plant. Four different values of the β parameter were used to diversify the set of solutions given by the two procedures. This parameter is used to impose the minimum use of the saw capacity in each saw cycle. For both algorithms, most of the non-dominated solutions were obtained with $\beta = 80\,\%$ and $\beta = 100\,\%$. Imposing a higher utilization of the saw capacity allows a better compromise between the conflicting objectives.

The criterion used to define the cutting patterns frequency in the PCH-C algorithm helped to improve the solution in cases where the cuts used in the PCH-F algorithm had no effect. When comparing the results of the two proposed algorithms, it is possible to see that the PCH-C algorithm provides a higher number of non-dominated solutions and, in most cases, better solutions than those given by PCH-F. This behaviour was expected considering that the criterion to generate cuts in the PCH-C takes into account the actual number of objects in each incomplete saw cycle. The quality of the solutions improves in situations with a high diversity of items and high demands. For all instances, the heuristics returned solutions in which the number of cycles is smaller than the number of cycles of the solutions adopted by the Factory V decision makers, with an increase of at most 12 % in the total number of objects.

The proposed solution approaches should be adopted in Factory V in situations with high demand since they produce a variety of good solutions taking into account the trade-off between minimizing the total number objects and cycles. Also, they are an automated way of planning, in a reasonable amount of (computational) time, which cutting patterns to use and how many objects must be cut according to them.

5 Conclusions

In this paper, the two-dimensional cutting stock problem is addressed considering two potentially conflicting objectives: the minimization of the total number of objects and the minimization of the total number of saw cycles, the CSP-C. Our work adds to the literature on the CSP-C proposing a novel heuristic method to solve the problem considering the conflict between these two objectives, and treating it in the context of the furniture factories. The procedure and the mathematical models can be useful for the one-dimensional case as well as in other industrial contexts.

The basic idea of the heuristic method proposed is to control the number of saw cycles by dynamically imposing constraints on the cutting pattern frequencies. Two strategies to add cuts are considered in a column generation framework. In the first strategy (PCH-F) an attempt to minimize the number of cycles is made by defining that, if a cutting pattern is used, its frequency must be greater than a pre-defined value. In the second strategy (PCH-C) the control is made by imposing that, in each saw cycle, at least a pre-defined number of objects are cut. Each algorithm returns a set of solutions that are analysed to select the non-dominated ones. The algorithms were tested using real data from a typical furniture plant.

Comparing the results of the proposed algorithms with the practice of Factory V, the heuristics PCH-F and the PCH-C were able to find solutions close to and, in most cases, better than those of Factory V, with respect to the number of cycles and objects. Furthermore, both algorithms can be helpful in finding a variety of non-dominated solutions allowing the decision maker to choose the one he or she considers the best.

Acknowledgements This work was partially funded by the Brazilian agencies CNPq, FAPESP and CAPES. The authors are grateful to Factory V for providing the data used in the numerical study. Thanks are also due to the anonymous reviewers for their careful reading, useful comments and suggestions.

References

Alves, C., Macedo, R., & Carvalho, J. M. V. (2009). New lower bounds based on column generation and constraint programming for the pattern minimization problem. *Computers and Operations Research*, *36*, 2944–2954.

Barnhart, C., et al. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, *46*(3), 316–329.

Calderon, C. M. A. (2013). *O segmento moveleiro na região do Alto Juruá-AC: perfil e uso de tecnologias alternativas para a caracterização das principais espécies madeireiras.* Doctorate Thesis, Universidade de Brasília.

Cherri, A., Arenales, M. N., Yanasse, H. H., Poldi, K. C., & Vianna, A. C. G. (2014). The one-dimensional cutting stock problem with usable leftovers—A survey. *European Journal of Operational Research*, *236*(2), 395–402.

Chu, C., & Antonio, J. (1999). Approximation algorithms to solve real-life multicriteria cutting stock problems. *Operations Research*, *47*, 495–508.

Deb, K. (2004). *Multi-objective optimization using evolutionary algorithms*. England: Wiley.

Degraeve, Z., Gochet, W., & Jans, J. (2002). Alternative formulations for a layout problem in the fashion industry. *European Journal of Operational Research*, *143*, 80–93.

Degraeve, Z., & Vandebroeke, M. (1998). A mixed integer programming model for solving a layout problem in the fashion industry. *Management Science*, *44*, 301–310.

Diegel, A., Miller, G., Montocchio, E., Van Schalkwyk, S., & Diegel, O. (2006). Enforcing minimun run length in the cutting stock problem. *European Journal of Operational Research*, *171*, 708–721.

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

Faccio, A. P., & Rangel, S. (2009). Geração de padrões de corte *n*-grupos para a indústria moveleira. *Boletim da Sociedade Paranaense de Matemática*, *27*, 41–57.

FICO Xpress optimization suite. Getting Started with Xpress. www.fico.com. Last visited 27 January 2015.
Foerster, H., & Waescher, G. (2000). Pattern reduction in one-dimensional cutting stock problem. *International Journal of Production Research*, *38*, 1657–1676.
Gilmore, P. C., & Gomory, R. E. (1963). A linear programming approach to the cutting-stock problem. II. *Operations Research*, *13*, 94–120.
Gilmore, P. C., & Gomory, R. E. (1965). Multistage cutting stock problems of two and more dimensional. *Operations Research*, *14*, 1045–1074.
Glass, C. A., & Oostrum, J. M. (2010). Bun splitting: a practical cutting stock problem. *Annals of Operations Research*, *179*, 15–33.
Haessler, R. W. (1975). Controlling cutting pattern changes in one-dimensional trim problems. *Operations Research*, *23*, 483–493.
Kallrath, J., Rebennack, S., Kallrath, J., & Kusche, R. (2014). Solving real-world cutting stock-problems in the paper industry: Mathematical approaches, experience and challenges. *European Journal of Operational Research*, *238*(1), 374–389.
Li, Y., Chu, C. & Wang, K. L. (2008). A heuristic procedure based on column generation to solve a cutting stock problem. *Proceedings of the 2008 IEEE IEEM*, 158-162.
Malaguti, E., Duràn, R. M., & Toth, P. (2014). Approaches to real world two-dimensional cutting problems. *Omega*, *47*, 99–115.
McDiarmid, C. (1999). Pattern minimisation in cutting stock problems. *Discrete Applied Mathematics*, *98*, 121–130.
Mobasher, A., & Ekici, A. (2013). Solution approaches for the cutting stock problem with setup cost. *Computers and Operations Research*, *40*(1), 225–235.
Morabito, R., & Arenales, M. N. (2000). Optimizing the cutting of stock plates in a furniture company. *International Journal of Production Research*, *38*, 2725–2742.
Morabito, R., Arenales, M. N., & Yanasse, H. H. (2009). Special issue on cutting, packing and related problems. *International Transactions in Operational Research*, *16*, 659. doi:10.1111/j.1475-3995.2009.00739.x.
Moretti, A. C., & Salles Neto, L. L. (2008). Nonlinear cutting stock problem model to minimize the number of different patterns and objects. *Computational and Applied Mathematics*, *27*, 61–78.
Mosquera, G. P., & Rangel, S. (2007). Redução de ciclos da serra no problema de corte de estoque bidimensional na indústria de móveis. *Anais do XXX CNMAC*. SBMAC - Brasilian Society of Applied and Computational Mathematics.
Munari, P., & Gondzio, J. (2013). Using the primal-dual interior point algorithm within the branch-price-and-cut method. *Computers and Operations Research*, *40*(8), 2026–2036.
Poldi, K. C., & Arenales, M. N. (2009). Heuristics for the one-dimensional cutting stock problem with limited multiple stock lengths. *Computers and Operations Research*, *36*(6), 2074–2081.
Queiroz, T. A., & Miyazawa, F. K. (2014). Order and static stability into the strip packing problem. *Annals of Operations Research*, *223*, 137–154.
Ranck R. Jr., (2008). *Desenvolvimento de alguns métodos de solução para o problema de redução de ciclos da serra*. M.Sc. dissertation. INPE.
Rangel, S., & Figueiredo, A. (2008). O problema de corte de estoque em indústria de móveis de pequeno e médio portes. *Pesquisa Operacional*, *28*(3), 451–472.
Rebennack, S., Kallrath, J., & Pardalos, P. M. (2009). Column enumeration based decomposition techniques for a class of non-convex MINLP problems. *Journal of Global Optimization*, *43*(2–3), 277–297.
Silva, E., Oliveira, J. F., & Waescher, G. (2014). 2DCPackGen: a problem generator for two-dimensional rectangular cutting and packing problems. *European Journal of Operational Research*, *237*(3), 846–856.
Toscano, A., Rangel, S., & Yanasse, H. H. (2010). Um algoritmo para a redução de ciclos da serra e de objetos no problema de corte de estoque bidimensional de uma indústria moveleira. *Anais do XLII SBPO*. SOBRAPO - Brazilian Operational Research Society.
Vanderbeck, F. (2000). Exact algorithm for minimizing the number of setups in the one-dimensional cutting stock problem. *Operations Research*, *48*, 915–926.
Vasko, F. J., Newhart, D. D., Stott, J. R., & K. L. & Wolf, F. E. (2000). Fiddler on the roof: Balancing trim loss and setups. *OR Insight*, *13*, 9–14.
Waescher, G., & Gau, T. (1996). Heuristics for the integer one-dimensional cutting stock problem: A computational study. *OR Spektrum*, *18*, 131–144.
Waescher, G., Haußner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, *183*(3), 1109–1130.
Wang, P., & Waescher, G. (Eds.). (2002). Special issue on cutting and packing. *European Journal of Operational Research*, *141*(2), 239–240.
Williams, H. P. (1990). *Model building in mathematical programming*. England: Wiley.

Yanasse, H. H. (2008). A note on the minimization of the number of cutting cycles problem. In R. J. Rio de Janeiro (Eds.), *SPOLM 2008, XI Simpósio de Pesquisa Operacional e Logística da Marinha*. Published in CD, ISSN 1806-3632, file 012.

Yanasse, H. H., Harris, R. G., & Zinober, A. S. I. (1993). Two-dimensional cutting stock with multiple stock sizes. *Journal Operations Research Society*, *42*, 673–683.

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.

Yanasse, H. H., & Morabito, R. (2008). A note on linear models for two-group and three-group two-dimensional guillotine cutting problems. *International Journal of Production Research*, *46*, 6189–6206.