INDERSCIENCE PUBLISHERS
Linking academia, business and industry through research

European J. of Industrial Engineering

ISSN online: 1751-5262 - ISSN print: 1751-5254
https://www.inderscience.com/ejie

An integrated two dimensional cutting stock and lot sizing problem with two criteria

Ayse Burcu Sanan, Meral Azizoğlu

DOI: 10.1504/EJIE.2023.10054219

Article History:

Received:	29 March 2022
Last revised:	14 June 2022
Accepted:	07 August 2022
Published online:	01 September 2023

An integrated two dimensional cutting stock and lot sizing problem with two criteria

Ayse Burcu Sanan and Meral Azizoğlu*

Department of IE,
METU,
Ankara 06800, Turkey
Email: burcu.ergen@metu.edu.tr
Email: ma@metu.edu.tr
*Corresponding author

Abstract: In this study, we consider an integrated two dimensional cutting stock and lot sizing problem arising in an aircraft manufacturing plant. The items are to be cut from steel panels of identical size to satisfy all periodic demands over a specified planning horizon. Two objectives, minimising the number of panels cut and the total inventory carrying cost of the items, are defined and all non-dominated objective vectors concerning the defined objectives are generated. To generate each non-dominated objective vector, we propose a mixed integer linear programming model whose efficiency is improved by optimality properties and bounding mechanisms. The results of our experiments have revealed that the instances with few items can be solved for up to 14 periods and the instances with more items can be solved for up to seven periods, in two hours. [Submitted: 29 March 2022; Accepted: 7 August 2022]

Keywords: two dimensional cutting stock problems; 2DCSPs; lot sizing problems; integrated problems; multi-objective programming.

Reference to this paper should be made as follows: Sanan, A.B. and Azizoğlu, M. (2023) ‘An integrated two dimensional cutting stock and lot sizing problem with two criteria’, *European J. Industrial Engineering*, Vol. 17, No. 5, pp.766–794.

Biographical notes: Ayse Burcu Sanan received her BS and MS from the Industrial Engineering Department of the Middle East Technical University, Ankara, Turkey. She is currently working at Turkish Aerospace Industries as an industrial engineering.

Meral Azizoğlu is a Full Professor of Industrial Engineering at the Middle East Technical University, Ankara, Turkey. She currently works on network optimisation, design and analysis of manufacturing systems, production planning, and scheduling.

1 Introduction

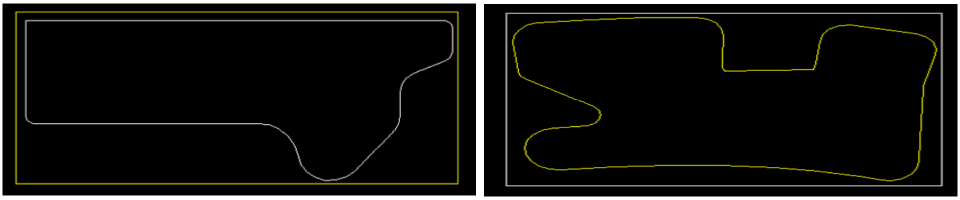
The cutting stock problem and the lot sizing problem have been studied by many researchers for several decades owing to their practical importance. The majority of the past research has considered those problems separately due to their individual computational complexity. Recent researchers have recognised that their integration leads to improved product volumes and reduced production costs. Moreover, the newly developed advanced computing technologies could handle higher complexities, hence offering a challenge to studying the integrated problems.

In this study, we consider an integrated two dimensional cutting stock and lot sizing problem. We assume a given planning horizon where the periodic demand for the small rectangular items should be obtained from larger rectangular blocks of identical size. We assume two-stage guillotine cuts, where the first stage is horizontal. The problem is to find the number of rectangular items to be cut in each period. Two objectives of our concern are to minimise the number of panels (cutting stock problem related) and to minimise the total inventory holding cost of the small items (lot size problem related).

The two dimensional cutting stock and lot sizing problem has many application areas including but not limited to copper, furniture, glass and fibreglass industries. Our particular interest in the problem is from an application to an aircraft company located in Ankara, Turkey, to manage its steel cutting operations. The steel cutting operations in the plant are performed using a guillotine cutting machine. The items having specified daily demand have to be cut from identical big steel plates.

The number of plates used in each period defines the total raw material cost and raw material inventory holding amounts at the stock area that should be minimised. The small steel items have irregular shapes as shown in Figure 1.

Figure 1 The items to be cut (see online version for colours)



The cutting operations at the plant involve two stages. First, the smallest rectangular shape covering the entire irregular item is cut by the guillotine cutting machine and then a precise shape is obtained using a programmable rotating cutter. The items have rigid orientations such that their widths and lengths should fit those of the panels.

The irregular-shaped items are used as components in the final products of the company, hence their daily requirements are projected from the production levels of the final products. As the items are used in final products, any delay in their cutting times would delay the promised delivery times of the final products; hence the items should be cut no later than their required times. Moreover, their inventory carryings over long periods are undesirable, as some of them are fragile and some occupy space till their required times.

The production managers of the aircraft company want to see a set of solutions that demonstrates meaningful trade-offs between raw material costs of the panels and inventory carrying costs of the items. To help the production managers, we generate all non-dominated objective vectors considering these two objectives. From this set, they can make a final choice from the presented cutting plans using their preferences. To generate the set of non-dominated objective vectors, we develop several mathematical models along with several properties of the best solutions.

To the best of our knowledge, our study is the first reported research on the multi-criteria integrated two dimensional cutting stock and lot sizing problem.

The rest of the paper is organised as follows. In Section 2, we review the related research. Section 3 defines the problem and the mathematical models. In Section 4, we discuss the details of our solution approach to generate all non-dominated objective vectors. We discuss the results of our experiments in Section 5. In Section 6, we give the main conclusions and discuss the future research directions.

2 Literature review

In this section, we first review single criterion two dimensional cutting stock problems (2DCSPs) and then multi-criteria cutting stock problems. Finally, the literature on the integrated cutting stock and lot sizing problems is summarised.

2.1 Single criterion 2DCSPs

A comprehensive review of single criterion one-dimensional cutting stock was given in Delorme et al. (2016). Gilmore and Gomory (1965) introduced the 2DCSP along with *k*-stage and multiple stock size extensions. They proposed a mathematical model and a column generation technique that became the basis for the evolving research.

Extensive research on the 2DCSP exists. Furini et al. (2016) and Martin et al. (2020) considered 2DCSP with an unlimited number of stages and propose integer programming models for their solutions. Some noteworthy studies that consider two-stage 2DCSP are due to Cintra et al. (2008), Furini et al. (2012), Silva et al. (2010), Furini and Malaguti (2013) and Ayasandır and Azizoğlu (2022). Cintra et al. (2008) developed heuristic algorithms for the two-stage 2DCSP using the linear programming relaxations of Gilmore and Gomory's (1965) model. Later on, Furini et al. (2012) proposed a heuristic algorithm for the two-stage 2DCSP using the column generation idea and showed that the algorithm is superior to that of Cintra et al. (2008). Pseudo-polynomial mathematical models for the two and three stages 2DCSP were developed by Silva et al. (2010). Furini and Malaguti (2013) extended Silva et al.'s (2010) model and Gilmore and Gomory's (1965) column generation idea to the two-stage 2DCSP with multiple stock sizes and proposed three mathematical programming models. Their objective was to minimise the total area of the panels. Recently, Ayasandır and Azizoğlu (2022) extended the most efficient model of Furini and Malaguti (2013) to their total net profit objective. They also proposed a new model along with some optimality properties and showed that the new model outperforms Furini and Malaguti's (2013) model.

2.2 Multi-criteria cutting stock problems

We are aware of too few studies on multi-criteria cutting stock problems. These are due to Kolen and Spieksma (2000), Cui and Yang (2010) and Aliano Filho et al. (2018, 2019) for one, De Armas et al. (2011) and Mellouli et al. (2019) for two and Gonzalez et al. (2016) for three dimensional problems. Kolen and Spieksma (2000) studied total cutting loss and the number of different patterns used in objectives. They presented a branch and bound algorithm to find the Pareto set. Cui and Yang (2010) considered three objectives: the total panel cost, the profit from the leftovers (the unused length of a panel is leftover once it is longer than a threshold) and the profit from the leftovers coming from past cutting operations. They proposed a heuristic algorithm that first cuts the major part of the demand and then the rest. Aliano Filho et al. (2018) studied two objectives: the number of times a cutting pattern is used and the number of different cutting patterns. They generated all non-dominated objective vectors and presented four methods: the weighted sum, the Chebyshev's metric, the ε-constraint and an improved version of Chebyshev's metric. Aliano Filho et al. (2019) considered two objectives: the cost of using cutting patterns and the cost of different cutting patterns. They generated all non-dominated objective vectors using an exact scalarisation approach. De Armas et al. (2011) considered two objectives: the total profit and the number of panels. To generate the set of non-dominated objective vectors, they proposed a non-dominated sorting genetic algorithm, strength Pareto evolutionary algorithm, and indicator-based evolutionary algorithm. Mellouli et al. (2019) considered the material waste and the setup costs as two objectives. They used a genetic algorithm to find the set of all non-dominated objective vectors. Gonzalez et al. (2016) studied two objectives: the total volume and weight of the items placed. They applied evolutionary algorithms and developed a multiple-level filling heuristic to obtain all non-dominated objective vectors.

2.3 Integrated cutting stock and lot sizing problems

We first review the most-noteworthy single criterion integrated cutting stock and lot sizing studies and then discuss two multi-criteria studies that exist in the literature. We refer the reader to Melega et al. (2018) for the classification and extensive review of the integrated cutting stock and lot sizing problems.

2.3.1 Single criterion integrated cutting stock and lot sizing problems

Farley (1988) was the first author who addressed the integrated cutting stock and lot sizing problem. He considered the clothing industry and proposed integer and quadratic programming models to minimise the total cutting, sewing, and storing costs. Hendry et al. (1996) proposed a two-stage solution approach for the integrated cutting stock and lot sizing problem in the copper industry. In the first stage, the cutting stock problem was solved heuristically and in the second stage, the lot sizing problem was solved using an integer programming model. Nonås and Thorstenson (2000, 2008) studied the integrated two dimensional cutting stock and lot sizing problem arising in a Norwegian truck company. They assumed irregular shapes and stochastic demand and aimed to minimise the total raw material cost and setup cost. Nonås and Thorstenson (2000) presented a column generation procedure to solve small-sized instances and Nonås and Thorstenson (2008) improved the procedure using a sequential heuristic by Haessler (1971) to solve large-sized instances. Poltroniere et al. (2008) studied an integrated one dimensional cutting stock and lot sizing problem arising in the paper industry. They aimed to minimise the sum of inventory costs, setup costs, material waste costs and final item inventory costs. They developed an integer programming model and two heuristic procedures. Gramani and França (2006) studied an integrated two dimensional cutting stock and lot sizing problem in the wooden industry. They aimed to minimise the total setup costs, number of plates, and inventory carrying costs of items, and proposed a network flow-based solution approach. Gramani et al. (2009) extended Gramani and França's (2006) model to include the production and inventory holding costs of the final products. They proposed a Lagrangian relaxation-based heuristic approach where the subproblems require an exponential effort. Silva et al. (2014) studied an integrated two dimensional cutting stock and lot sizing problem in the furniture industry. They aimed to minimise the total material, waste and storage costs. They proposed two integer programming models by extending the previously reported models in the literature. Leao et al. (2017) studied the integrated one dimensional cutting stock and lot sizing problem in the paper industry. They developed three mathematical models: pattern-oriented, period decomposition, and machine decomposition. To solve the machine decomposition model, they proposed a heuristic that uses a column generation method along with an adaptive neighbourhood search. Vanzela et al. (2017) studied the integrated two dimensional cutting stock and lot sizing problem arising in the Brazilian furniture industry. They proposed a mathematical model and a column generation-based heuristic to minimise the raw material waste, production, and inventory costs and to include the safety stock level and saw capacity constraints. Signorini et al. (2022) studied a one dimensional cutting stock and lot sizing problem arising in the concrete industry. They proposed two mathematical models and two heuristic procedures to minimise total production and inventory costs. Cristofoletti et al. (2021) studied a three dimensional cutting stock and lot sizing problem arising in the mattress industry. They presented a mathematical model to minimise total costs and waste. Andrade et al. (2021) considered a one dimensional cutting stock and lot sizing problem for a manufacturer of automotive springs to minimise inventory costs and losses in the steel bar cutting process. They proposed a branch and price algorithm using a column generation method.

2.3.2 Multi-criteria integrated cutting stock and lot sizing problems

Campello et al. (2020) and Oliveira et al. (2021) studied integrated one dimensional cutting stock and lot sizing problems. Campello et al. (2020) presented multi-criteria approach arising in the paper industry. Their objectives were minimising total production costs, inventory costs of paper rolls, and setup costs of machines and minimising total material waste and inventory costs of items. They proposed a weighting approach and an ε-constraint method to generate all non-dominated objective vectors. Oliveira et al. (2021) proposed a general goal programming methodology and tested their approach on one-dimensional problems.

To the best of our knowledge, there is no reported study on the multi-criteria integrated two dimensional cutting stock and lot sizing problem.

3 Problem definition and mathematical models

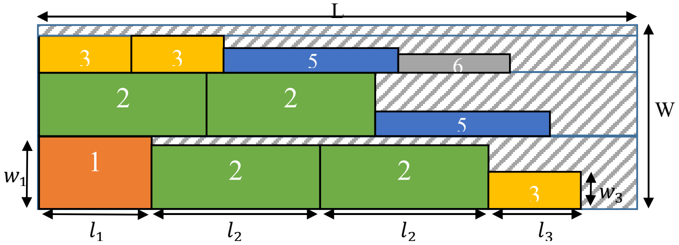
We consider a single panel type and multiple periods. There are n item types to be cut from several copies of the single panel type that is defined by its length L and width W. Item j, $j \in \{1, \dots, n\}$ is characterised by its length l_j and width w_j. We assume that the items are sorted in their non-increasing order of widths, i.e., $w_1 \geq w_2 \geq \dots \geq w_n$.

The cuts are guillotine type and orientations of the items on the panels are important, i.e., the width and length of the item should be consistent with those of the panel. Figure 2 depicts the placement of the items on a particular panel. The figure shows a two-stage guillotine cut where the first stage is horizontal. Each horizontal cut defines a level and the width of the level is defined by the first item, which is, by construction, the largest-width item assigned to that level.

Note that we define horizontal levels because the first stage guillotine cut is horizontal. Vertical levels would be defined if the first stage of the guillotine cuts was vertical. The shaded area of the panel shows the waste resulting from the placements of items on the panel. We say an item initialises a level if it is the first assigned item to that level. In Figure 2, items 1, 2 and 3 initialise levels 1, 2 and 3, respectively. The other items assigned to a level are referred to as additional items. In Figure 2, items 2 and 3 are the additional items assigned to level 1; items 2 and 5 are the additional items assigned to level 2; items 3, 5 and 6 are the additional items assigned to level 3.

There are *T* time periods whose cutting assignments should be planned together. Item *j* has a demand of d_{jt} units in period *t*. The demand of period *t* should be satisfied from the cuts of periods 1 through *t*. The cost of holding a single unit inventory of item *j* is h_j money units. No backlogging or lost sales are allowed.

Figure 2 The placement of items on a panel (see online version for colours)



We aim to define a cutting plan for each panel used at each period *t* with a perspective of minimising the following two measures over the planning horizon.

1 Total inventory holding cost – *TIC*.

2 Total number of panels used, i.e., the total cost of panels – *TP*.

We propose two mixed integer linear programming (MILP) models that are modified from the models by Ayasandır and Azizoğlu (2022) for the single period, single objective, multiple panel types 2DCSP. Ayasandır and Azizoğlu (2022) extended the first model from Furini and Magnanti (2013) who had proved its outperforming performance over the previously reported models. Ayasandır and Azizoğlu (2022) showed that their second model outperforms the first model, hence it is even better. Our contribution to those models is to consider additional decisions to capture the inventory carrying amounts between two consecutive periods.

We next explain these MILP models.

3.1 MILP model I

We define $D = \sum_{j=1}^{n} \sum_{t=1}^{T} d_{jt}$ as the total item demand over all periods. At least one item is cut at each level of a panel, hence there are at most *D* levels over all panels.

Recall that the items are in their non-increasing order of widths. Hence in period t, any item j may be cut from levels in range $[1, \alpha_{jt}]$ where $\alpha_{jt} = \sum_{s=1}^{j} \sum_{r=t}^{T} d_{sr}$ and $\alpha_{0t} = 0$. Any level i in period t can be used to cut the items in the range $[\beta_{it}, n]$ where $\beta_{it} = \min \{r\text{: } 1 \leq r \leq n, \alpha_{rt} \geq i\}$; $i = 1, \ldots, D$. Hence, β_{it} returns the item that initialises level i in period t.

The binary decision variables are defined as:

$$q_{kt} = \begin{cases} 1, & \text{if panel } k \text{ is used in period } t \\ & k \in \{1, \mathrm{A}\ , D\}, t \in \{1, \mathrm{A}\ , T\} \\ 0, & \text{otherwise} \end{cases}$$

$$y_{it} = \begin{cases} 1, & \text{if item } i \text{ initialises a level in period } t \\ & i \in \{1, \mathrm{A}\ , D\}, t \in \{1, \mathrm{A}\ , T\} \\ 0, & \text{otherwise} \end{cases}$$

$$z_{kit} = \begin{cases} 1, & \text{if level } i \text{ is used by panel } k \text{ in period } t \\ & k \in \{1, \mathrm{A}\ , D-1\}, i \in \{k, \mathrm{A}\ , D\}, t \in \{1, \mathrm{A}\ , T\} \\ 0, & \text{otherwise} \end{cases}$$

The quantity-based decision variables are as stated below:

x_{ijt} the number of additional items (addition to the initialising item) of type j put in level i in period t. $i \in \{1, \ldots, D-1\}, j \in \{\beta_{it}, \ldots, n\}, t \in \{1, \ldots, T\}$

I_{jt} amount of item j carried (inventory) from period t to $t + 1$. $j \in \{1, \ldots, n\}, t \in \{1, \ldots, T\}$.

The constraint set is defined below.

- Inventory – cut amount balance: The amount that initialises a level together with the amount further cut at that level plus inventory carried from the previous period is equal to the total demand in the current period plus the inventory carried to the next period.

$$\sum_{i=1}^{\alpha_{jt}} x_{ijt} + \sum_{i=\alpha_{j-1,t}+1}^{\alpha_{jt}} y_{it} + I_{j,t-1} = d_{jt} + I_{jt} \qquad j \in \{1, \mathrm{A}\ , n\}, t \in \{1, \mathrm{A}\ , T\} \qquad (1)$$

- Length capacity: For each period, the sum of the lengths of the items cut from any level should not exceed the length of the panel.

$$\sum_{j=\beta_{it}}^{n} x_{ijt} * l_j \leq \left(L - l_{\beta_{it}}\right) * y_{it} \qquad i \in \{1, \mathrm{A}\ , D-1\}, t \in \{1, \mathrm{A}\ , T\} \qquad (2)$$

- Width capacity: The sum of the widths of the items that initialise the levels of the panel should not exceed the width of the panel.

$$\sum_{i=k+1}^{D} z_{kit} * w_{\beta_{it}} \leq \left(W - w_{\beta_{kt}}\right) * q_{kt} \qquad k \in \{1, \mathrm{A}\ , D-1\}, t \in \{1, \mathrm{A}\ , T\} \qquad (3)$$

- Level-panel relation: A level should either initialise or be assigned to a level of a panel.

$$\sum_{k=1}^{i-1} z_{kit} + q_{it} = y_{it} \qquad i \in \{1,\ldots,D\}, t \in \{1,\ldots,T\} \tag{4}$$

- Binary variables, non-negativity and integrality:

$$z_{kit} \in \{1, 0\} \qquad k \in \{1,\ldots,D\}, i \in \{1,\ldots,D\}, t \in \{1,\ldots,T\} \tag{5}$$

$$y_{it} \in \{1, 0\} \qquad i \in \{1,\ldots,D\}, t \in \{1,\ldots T\} \tag{6}$$

$$q_{kt} \in \{1, 0\} \qquad k \in \{1,\ldots,D\}, t \in \{1,\ldots,T\} \tag{7}$$

$$x_{ijt} \geq 0 \text{ and integer} \qquad i \in \{1,\ldots,D\}, j \in \{1,\ldots,n\}, t \in \{1,\ldots,T\} \tag{8}$$

$$I_{jt} \geq 0 \qquad j \in \{1,\ldots,n\}, t \in \{1,\ldots,T\} \tag{9}$$

The performance criteria are defined as:

1 minimise total inventory cost, $TIC = Min \sum_{j=1}^{n} \sum_{t=1}^{T} h_j * I_{jt}$

2 minimise total number of panels, $TP = Min \sum_{k=1}^{D} \sum_{t=1}^{T} q_{kt}$.

We hereafter refer to equations (1) through (9) sets as $x \in X_{M1}$ and refer to model I as:

Min TIC
Min TP
s.t. $x \in X_{M1}$

3.2 MILP model II

Ayasandır and Azizoğlu (2022) noted that once the total demand over all items (D) is high, the model with decision variables defined on the total demand value – like model I – may become hard to solve. Recognising this fact, they proposed an alternative model that defines the decision variables on the number of available panels. We extended their alternative model to include the inventory-related decisions over multiple periods.

We let P be an upper bound on the number of panels cut and let UBL_{jt} be an upper bound on the number of levels for item j in period t. Later, we will discuss the way P is defined in our implementation.

Note that $\sum_{r=1}^{T} d_{jr}$ is the maximum amount of item j cut in period t, as no backlogs are allowed and all demand should be met. From each panel, up to $\left\lfloor \frac{W}{w_j} \right\rfloor$ levels for item j can be cut and over P panels $P * \left\lfloor \frac{W}{w_j} \right\rfloor$ levels can be cut. We define UBL_{jt} as follows:

$$UBL_{jt} = Min\left\{\sum_{r=t}^{T} d_{jr}, P*\left|\frac{W}{w_j}\right|\right\} \qquad j \in \{1,\mathrm{A}\ ,n\}, t \in \{1,\mathrm{A}\ ,T\}$$

We define the decision variables as:

$$q_{kt} = \begin{cases} 1, & \text{if panel } k \text{ is used in period } t \;\; k \in \{1,\mathrm{A}\ ,P\}, t \in \{1,\mathrm{A}\ \ T\} \\ 0, & \text{otherwise} \end{cases}$$

$$z_{imkt} = \begin{cases} 1, & \text{if item } i \text{ initialises level } m \text{ of panel } k \text{ in period } t \\ 0, & \text{otherwise} \end{cases}$$

The quantity-based variables are defined as:

x_{jimkt} additional amount of type j cut at level m initialised by item i in panel k in period t. $i, j \in \{1, \ldots, n\}$ and $i \leq j$ and $\left\lfloor \frac{L-l_i}{l_j} \right\rfloor \geq 1$, $m \in \{1, \ldots, UBL_{it}\}$, $k \in \{1, \ldots, P\}$, $t \in \{1, \ldots, T\}$ $\left\lfloor \frac{L-l_i}{l_j} \right\rfloor \geq 1$ is used to force an integer solution.

I_{jt} amount of item j carried from period t to $t + 1$. $t \in \{1, \ldots, T\}, j \in \{1, \ldots, n\}$.

The constraint set is given below.

- Inventory – cut amount balance: The amount that initialises a level together with the amount further cut at that level plus inventory carried from the previous period is equal to the total demand in the current period plus the inventory carried to the next period.

$$\sum_{i=1}^{n}\sum_{m=1}^{UBL_{it}}\sum_{k=1}^{P} x_{jimkt} + \sum_{m=1}^{UBL_{jt}}\sum_{k=1}^{P} z_{jmkt} + I_{j,t-1} = d_{jt} + I_{jt} \qquad j \in \{1,\mathrm{A}\ ,n\}, t \in \{1,\mathrm{A}\ ,T\} \tag{10}$$

- Length capacity: The sum of the lengths of the items assigned to any level cannot exceed the length of the panel for each period.

$$\sum_{j=1}^{n} l_j * x_{jimkt} + l_i * z_{imkt} \leq L * z_{imkt} \qquad i \in \{1,\mathrm{A}\ ,n\}, m \in \{1,\mathrm{A}\ ,UBL_{it}\}, \; k \in \{1,\mathrm{A}\ ,P\}, t \in \{1,\mathrm{A}\ ,T\} \tag{11}$$

- Width capacity: The sum of the widths of the items that initialise the levels of the panel cannot exceed the width of the panel.

$$\sum_{i=1}^{n}\sum_{m=1}^{UBL_{it}} w_i * z_{imkt} \leq W * q_{kt} \qquad k \in \{1,\mathrm{A}\ ,P\}, t \in \{1,\mathrm{A}\ ,T\} \tag{12}$$

- Binary variables, non-negativity and integrality:

$$z_{imkt} \in \{1, 0\} \qquad i \in \{1,\mathrm{A}\ ,n\}, m \in \left\{1,\mathrm{A}\ ,UBL_{it}\right\}, \; k \in \{1,\mathrm{A}\ ,P\}, t \in \{1,\mathrm{A}\ ,T\}, \tag{13}$$

$$q_{kt} \in \{1, 0\}, \qquad k \in \{1, \ldots, P\}, t \in \{1, \ldots, T\} \tag{14}$$

$$x_{jimkt} \geq 0 \text{ and integer} \qquad \begin{array}{l} j \in \{1, \ldots, n\}, i \in \{1, \ldots, n\}, m \in \{1, \ldots, UBL_{it}\}, \\ k \in \{1, \ldots, P\}, t \in \{1, \ldots, T\} \end{array} \tag{15}$$

$$I_{jt} \geq 0 \qquad j \in \{1, \ldots, n\}, t \in \{1, \ldots, T\} \tag{16}$$

Note that the value of P is important in terms of the number of decision variables. The smaller the value of P is, the fewer the number of decision variables.

The performance criteria are defined as:

1 minimise total inventory cost, $TIC = Min \sum_{j=1}^{n} \sum_{t=1}^{T} h_j * I_{jt}$

2 minimise total number of panels cut, $TP = Min \sum_{k=1}^{p} \sum_{t=1}^{T} q_{kt}$.

We refer to constraints (10) through (16) sets as $x \in X_{M2}$ and refer to model II as:

$Min\ TIC$

$Min\ TP$

s.t. $x \in X_{M2}$

4 Solution approach

We aim to generate all non-dominated objective vectors and an efficient solution corresponding to each non-dominated objective vector. In doing so, the constraint set of model II, i.e., $x \in X_{M2}$, is used.

The section is organised as follows: in Subsection 4.1, we define non-dominated objective vectors and extreme non-dominated objective vectors. Subsection 4.2 discusses the generation of all non-dominated objective vectors. In Subsection 4.3, we develop some mechanisms to reduce the computational burden of the models.

4.1 Non-dominated objective vectors

A solution (cutting and lot sizing plan) S in $x \in X_{M2}$ is said to be efficient if there does not exist a solution S' in $x \in X_{M2}$ such that $TIC(S') \leq TIC(S)$ and $TP(S') \leq TP(S)$ with strict inequality holding at least once. The objective vector $(TIC(S), TP(S))$ corresponding to the efficient solution S is said to be a non-dominated objective vector.

An efficient solution S is said to be extreme efficient if it has the best possible value for one objective. The associated non-dominated objective vector is said to be an extreme non-dominated objective vector.

We now discuss the generation of the two extreme non-dominated objective vectors.

4.1.1 Extreme non-dominated objective vector having smallest TIC value

Consider the following MILP:

$$\begin{aligned}&\textit{Min TIC}\\ &\text{s.t. } x \in X_{M2}\end{aligned}$$

The optimal solution produces all $\sum_{j=1}^{n} d_{jt}$ units in period *t* and has a TIC value of zero. Hence, the problem decomposes into *t* single period models with no inventory decisions.

We let X'_{M2} be the set of constraint (11) through (15) and define the following single period problem *P*(*t*) for each *t*.

$$\begin{aligned}&\textit{Min } P(t)\\ &\text{s.t.} \sum_{i=1}^{n}\sum_{m=1}^{UBL_{it}}\sum_{k=1}^{P} x_{jimkt} + \sum_{m=1}^{UBL_{jt}}\sum_{k=1}^{P} z_{jmkt} = d_{jt} \quad j \in \{1,\mathrm{A}\ , n\}\\ &x \in X'_{M2}\end{aligned}$$

Let $P^{*}(t)$ be the optimal solution.

The minimum number of panels with zero inventory is $P_{\max} = \sum_{t=1}^{T} P^{*}(t).$ Note that $P_{\max}$ is an upper bound on the number of the panels of all non-dominated objective vectors and (*TIC*, *TP*) = (0, $P_{\max}$) is an extreme non-dominated objective vector.

4.1.2 Extreme non-dominated objective vector having smallest TP value

Consider the following problem:

$$\begin{aligned}&\textit{Min TP}\\ &\text{s.t. } x \in X_{M2}\end{aligned}$$

The optimal solution to the above problem gives the smallest *TP* value of all efficient solutions. The resulting solution may not be efficient as there may exist another solution with a smaller *TIC* value. Such a solution can be found through the following two-step procedure.

Procedure 1: Generating an extreme non-dominated objective vector with the smallest TP value

Step 1 Solve the problem below for *T* = 1 and $d_{j1} = \sum_{t=1}^{T} d_{jt}\ \forall j \in \{1,\mathrm{A}\ , n\}$:

$$\begin{aligned}&\textit{Min TP}\\ &\text{s.t. } x \in X_{M1}\end{aligned}$$

Let $P_{\min}$ be the optimal solution.

Step 2 Solve the below problem:

$$\begin{aligned}&\textit{Min TIC}\\ &\text{s.t.} \sum_{k=1}^{TP^{*}}\sum_{t=1}^{T} q_{kt} = P_{\min}\\ &x \in X_{M2}\end{aligned}$$

Let TIC^* be the optimal solution.

Note that TIC^* is an upper bound on the total inventory cost of all efficient solutions and $(TIC, TP) = (TIC^*, P_{\min})$ is an extreme non-dominated objective vector.

The other non-dominated objective vectors have TIC in $[1, TIC^* - 1]$ and TP in $[P_{\min} + 1, P_{\max} - 1]$ when the TIC and TP values are integers.

Note that $P_{\min}$ is the number of panels for cutting all items in a single period. Hence, it is an upper bound on the number of panels used per period. Recognising this fact, we use $P_{\min}$ in place of P (like using $k \in \{1, \ldots, P_{\min}\}$, $\sum_{k=1}^{P_{\min}} \sum_{t=1}^{T} q_{kt}$) in the model that returns the extreme solution with minimum total inventory cost.

The models that deliver the extreme solutions are two stage two-dimensional cutting stock models with guillotine cuts. 2DCSPs with guillotine cuts are strongly NP-hard (see Furini and Malaguti, 2013), so is our problem of generating extreme, thereby all, non-dominated objective vectors.

4.1.3 Generating all non-dominated objective vectors

We first present a property of the non-dominated objective vectors (Theorem 4.1) and use it to generate their set.

Theorem 4.1: There exists a non-dominated objective vector corresponding to each number of panels between $P_{\min}$ and $P_{\max}$.

Proof: Among all non-dominated objective vectors $P_{\max} = \sum_{t=1}^{T} P^*(t)$ is the minimum number of panels while carrying no inventory and $P_{\min}$ is the minimum number of panels while carrying maximum inventory.

Assume $P_m^*(t)$ is the number of panels used in period t in the $P_{\min}$ solution, i.e., $P_{\min} = \sum_{t=1}^{T} P_m^*(t)$. For any period t, for which $P_m^*(t) < P^*(t)$ putting an extra panel to $P_m^*(t)$ reduces the total inventory amount. Extra panels can be put till the number of panels reaches $P_{\max}$ and each addition reduces the total inventory amount. Hence, there exists a non-dominated objective vector for each value of P between $P_{\min}$ and $P_{\max}$.

#

Using the result of the above theorem, we define the following problem to generate an efficient solution with P panels.

$$\begin{aligned} & \textit{Min TIC} \\ & \text{s.t.} \sum_{k=1}^{P_{\min}} \sum_{t=1}^{T} q_{kt} = P \\ & x \in X_{M2} \end{aligned}$$

We hereafter refer to the above model as the *Min TIC|P* model.

Procedure 2 below generates all non-dominated objective vectors using the *Min TIC|P* model.

Procedure 2: Generating all non-dominated objective vectors

```
Step 1   Find $P_{\min}$ and $P_{\max}$ using the methods in Subsection 4.1.
         Let $r = 1$, $P = P_{\max}$.
Step 2   Let $r = r + 1$ and $P = P - 1$.
         Solve the Min TIC|P model.
         The optimal solution ($TIC^*$, $P$) is the $r^{th}$ non-dominated objective vector.
Step 3   If $P \geq P_{\min}$ then go to step 2.
         Else, all $r$ non-dominated objective vectors are generated.
```

4.2 Solving the min TIC|P model

In this section, we develop some mechanisms to reduce the computational burden of the *Min TIC|P* model. In Subsections 4.2.1 and 4.2.2, we present upper and lower bounds on the number of panels of any efficient solution. Subsection 4.2.3 defines lower bounds on the total inventory cost value. Subsection 4.2.4 discusses some reductions for the panel-related variables and Subsection 4.2.5 introduces some valid inequalities.

4.2.1 Upper bounds on the TP values

We derive several upper bounds on the number of panels in period t for any non-dominated objective vector. We use the best of those upper bounds to reduce the number of panel-related decision variables and constraints.

- *Upper bound 1:* Recall that $P_{\min}$ is the maximum number of panels that could be used for any period t. The minimum number of panels that should be used in periods 1 through $t - 1$ is $\left\lceil \frac{\sum_{r=1}^{t-1} \sum_{j=1}^{n} d_{jr} * l_j * w_j}{L * W} \right\rceil$. Hence, $P_{\min} - \left\lceil \frac{\sum_{r=1}^{t-1} \sum_{j=1}^{n} d_{jr} * l_j * w_j}{L * W} \right\rceil$ is a valid upper bound on the number of panels used in period t.
- *Upper bound 2:* $P_{\max_t}$ is the minimum number of panels for a single period t with $d_{j1} = d_{jt}$. $\sum_{r=1}^{T} P_{\max_r}$ is the number of panels with no inventory carrying for periods t through T, hence is a valid upper bound on the number of panels in period t.
- *Upper bound 3:* Let $TP^*(t)$ be the minimum number of panels for a single period t with $d_{j1} = \sum_{r=1}^{t-1} d_{jr}$. At least $TP^*(t)$ panels should be cut in the first $(t - 1)$ periods, leaving at most $P - TP^*(t)$ panels for period t.

An overall upper bound on the number of panels in period t, $P_{\max}(t)$, is found as:

$$P_{\max}(t) = Min \left\{ P_{\min} - \left\lceil \frac{\sum_{r=1}^{t-1} \sum_{j=1}^{n} d_{jr} * l_j * w_j}{L * W} \right\rceil, \sum_{r=t}^{T} P_{\max_r}, P - TP^*(t) \right\}$$

In our models, in place of P, we use $P_{\max}(t)$ at all appropriate places and update the parameter UBL_{jt} as $Min\left\{\sum_{r=1}^{T} d_{jr}, P_{\max}(t) * \left\lfloor \frac{W}{w_j} \right\rfloor\right\}$.

4.2.2 Lower bounds on the TP values

We derive two lower bounds on the number of the panels in periods 1 through t. We define a constraint using the best of the lower bounds.

- *Lower bound 1:* The minimum number of panels to be used in periods 1 through t is found by considering the total area required by the total demand of the periods and the total area available. The resulting expression is $\left\lceil \frac{\sum_{r=1}^{t} \sum_{j=1}^{n} d_{jr} * l_j * w_j}{L * W} \right\rceil$.
- *Lower bound 2:* Recall that $P_{\max_t}$ is the minimum number of panels required to satisfy the demand of period t. In the first t periods, at least $Max\{P_{\max_r}\}$ $r \in \{1, \ldots, t\}$ panels should be used. We add the following constraint to model II that uses the lower bounds.

$$\sum_{k=1}^{P_{\max}(t)} \sum_{r=1}^{t} q_{kr} \geq Max\left\{\left\lceil \frac{\sum_{r=1}^{t} \sum_{j=1}^{n} d_{jr} * l_j w_j}{L * W} \right\rceil, Max_r P_{\max_r}\right\} \; t \in \{1, \text{A} \; , T\} \quad (17)$$

4.2.3 Bounds on the TIC values

We included the following two constraints based on the total inventory cost.

$$TIC(P) \geq TIC(P-1) + 1 \quad (18)$$

$$TIC(P) \leq TIC\left(P_{\min}\right) - P + P_{\min} \quad (19)$$

where $TIC(P)$ is the optimal TIC value with P panels.

Constraint (18) uses the definition of the non-dominated objective vectors and the integrality of the TIC values. Constraint (19) uses the fact that there are $(P - P_{\min})$ non-dominated objective vectors with number of panels in $[P_{\min} + 1, P]$. Hence, the difference between TIC values of $P_{\min}$ and P solutions should be at least $P - P_{\min}$.

As $TIC(P_{\min})$ is not available, we find an upper bound using the below procedure:

Procedure 3: Finding an upper bound for the $TIC(P_{\min})$

Step 1 Find the item assignments to the panels using Procedure 1.

Step 2 Let t be the earliest period having positive demand.

Step 3 Let r be the item having the largest demand in period t.

Step 4 Let s be the panel that has maximum units of item r and assign panel s to period t.

Update the demands of the items considering the items assigned to panel s.

Step 5 Stop if all updated demands are zero. Else, go to step 2.

4.2.4 Bounds on the decision variables

We derive a theorem to restrict the panel cuts only to the periods having positive demand.

Theorem 4.2: If $\sum_{j=1}^{n} d_{jt} = 0$ for period t, then $\sum_{k=1}^{P_{\max}(t)} q_{kt} = 0$ for period t.

Proof: Assume there is a panel cut in period t with $\sum_{j=1}^{n} d_{jt} = 0.$ Then the panel can be shifted to period $t + 1$ without increasing the number of panels while reducing the total inventory cost. Hence, a solution in which there is a panel cut cannot be efficient.

#

Using the result of the above theorem, we only define decision variables q_{kt}, z_{imkt} and x_{jimkt} for period t, such that $\sum_{j=1}^{n} d_{jt} > 0,$ however, keep I_{jt} variables for all t.

4.2.5 Additional constraints for panel assignments

Margot (2010) states that "an integer linear program is symmetric if its variables can be permuted without changing the structure of the problem." Recognising this result, we introduce three symmetry elimination constraints for model II.

We first fix some q_{kt} values for the first period, $t = 1$, as follows;

$$q_{k1} = 1 \qquad k \in \{1, \ldots, P^*(1)\} \tag{20}$$

This is because at least $P^*(1)$ panels should be used in the first period as no backlogging is allowed. For the panels $P^*(1)$ through $P_{\max}(t)$ we use the following relation:

$$q_{k1} \geq q_{(k+1)1} \qquad k \in \{P^*(1), \ldots, P_{\max}(t)\} \tag{21}$$

$$q_{kt} \geq q_{(k+1)t} \qquad k \in \{1, \ldots, P_{\max}(t)\}, t \in \{2, \ldots, T\} \tag{22}$$

Constraints (21) and (22) sets are used to eliminate the solutions that skip panel k but use panel $k + 1$. Constraint (23) set uses the same idea for the levels and eliminates the solutions of the same panel that skips level m but uses level $m + 1$.

$$z_{imkt} \geq z_{i(m+1)kt} \qquad \begin{array}{l} i \in \{1, \ldots, n\}, m \in \{1, \ldots, UBL_{it}\}, \\ k \in \{1, \ldots, P_{\max}(t)\}, t \in \{1, \ldots, T\} \\ \text{where } \sum_{j=1}^{n} d_{jt} > 0 \end{array} \tag{23}$$

We let X''_{M2} be the set of constraints (10) through (23) sets and state improved model II as

$$Min\ TIC$$

$$\text{s.t.} \sum_{k=1}^{P_{\min}} \sum_{t=1}^{T} q_{kt} = P$$

$$z \in X''_{M2}$$

5 Computational experiments

In this chapter, we present the computational experiments designed to test the performances of our solution approaches. To generate the non-dominated objective vector set through mathematical model optimisation, we integrate CPLEX version 12.8.0 into the Java programming language. The heuristic algorithm for finding an extreme non-dominated objective vector is coded in the Java programming language using Netbeans IDE 8.2. The experiments are conducted on Intel® Xeon® E-2246G CPU @ 3.6 GHz, 16.0 GB RAM Windows 10.

In Subsection 5.1, we present the features of our instances and data generation scheme. In Subsection 5.2, we discuss the results of our experiments in detail.

5.1 The instance features and data generation

In our experiments, we use both real data from the aircraft manufacturing plant and data from the literature. We take a total of 56 real instances, 40 of which have seven periods and 16 of which have 14 periods. We define the instance features for seven and 14 periods in Tables A1 and A2, respectively. The tables report the number of item types, n, the total demand, D, $w_{\max}(w_{\min})$ is the maximum (minimum) width over all items of the problem instance, $l_{\max}(l_{\min})$ is the maximum (minimum) length over the items in the problem instance. W and L are the width and length of the panel, respectively. The measurement units of all dimensions are millimetres.

In Table A1, instances 1 through 8 have originally seven periods whereas instances 9 through 40 are formed by splitting one 14-period instance into two parts. We obtain 32 instances with seven periods by splitting 16 periods with 14 periods. We refer to the real data as *SetR*.

We further include 18 problem instances whose $w_{\max}$, $w_{\min}$, $l_{\max}$, $l_{\min}$, W and L values are taken from Hifi and Roucairol (2001). We refer to this dataset as *SetL*.

For each of the 18 instances in *SetL* we select two T values: 7 and 14, and two n values: 3 and 5, hence we use a total of 72 (18 * 2 * 2) problem instances. Table A3 gives the features of these instances together with their names known in the literature.

We generate the l_j values from a discrete uniform distribution (DU) between $l_{\min}$ and $l_{\max}$, w_j values from DU between $w_{\min}$ and $w_{\max}$. The demand of item j for period t, i.e., d_{jt} is generated from DU between 0 and 5. h_j values are taken from DU between 1 and 3 to be compatible with the values of our practical application.

5.2 Results of the experiments

In this subsection, we report the results of our experiment for *SetR* instances (Subsection 5.2.1) and *SetL* instances (Subsection 5.2.2).

5.2.1 Results of the real life instances, SetR

We first compare the performance of models I and II (without any improvement mechanism) on *SetR* instances. Tables 1 and 2 report the total CPU time of generating all non-dominated objective vectors and their average and maximum CPU times for $T = 7$ and 14, respectively.

Table 1 Models I and II performances, $T = 7$, *SetR*

Instance	*D*	*Number of non-dom. vectors*	*Model I*			*Model II*		
			Total CPU time	*Avg. CPU time*	*Max. CPU time*	*Total CPU time*	*Avg. CPU time*	*Max. CPU time*
1	45	2	7.77	3.88	7.42	1.17	0.59	1.13
2	50	2	15.23	7.62	14.81	4.77	2.38	4.23
3	32	2	1.72	0.86	1.31	3.09	1.55	3.05
4	54	3	91.03	30.34	85.25	1.42	0.47	1.41
5	300	3	14,572.25	4,857.42	7,200 (2)[1]	7,583.06	2,527.69	7,200 (1)
6	61	3	887.95	295.98	502.31	13.02	4.34	7.09
7	264	5	28,825.86	5,765.17	7,200 (4)	14,617.72	2,923.54	7,200 (2)
8	24	2	4.06	2.03	3.83	2.92	1.46	2.63
9	24	2	3.55	1.77	3.38	0.08	0.04	0.06
10	36	3	12.81	4.27	8.86	0.78	0.26	0.59
11	10	1	0	0	0	0.02	0.02	0.02
12	12	2	0.03	0.02	0.02	0.19	0.09	0.14
13	29	1	0.44	0.44	0.44	0.06	0.06	0.06
14	15	1	0.03	0.03	0.03	0.02	0.02	0.02
15	15	1	0	0	0	0	0	0
16	30	1	0.2	0.2	0.2	0	0	0
17	94	1	0.2	0.2	0.2	0	0	0
18	144	1	3.03	3.03	3.03	0.11	0.11	0.11
19	72	1	2.09	2.09	2.09	0.03	0.03	0.03
20	216	2	1,838.53	919.27	1,834.83	0.16	0.08	0.14
21	24	2	0.94	0.47	0.84	0.03	0.02	0.03

Notes: [1]numbers in parentheses is the number of non-dominated solutions that remain unsolved in two hours.

Table 1 Models I and II performances, $T = 7$, *SetR* (continued)

Instance	*D*	*Number of non-dom. vectors*	*Model I*			*Model II*		
			Total CPU time	*Avg. CPU time*	*Max. CPU time*	*Total CPU time*	*Avg. CPU time*	*Max. CPU time*
22	18	2	2	1	1.83	0.16	0.08	0.09
23	16	2	3.95	1.98	3.44	4.56	2.28	4.17
24	40	2	18.45	9.23	17.17	23.56	11.78	21.25
25	9	1	0	0	0	0	0	0
26	48	3	67.91	22.64	48.67	1.94	0.65	1.77
27	56	1	2.52	2.52	2.52	3.48	3.48	3.48
28	72	3	355.23	118.41	303.8	326.94	108.98	192.14
29	120	2	28.72	14.36	27.63	0.06	0.03	0.03
30	24	1	0.42	0.42	0.42	0	0	0
31	14	2	0.67	0.34	0.58	0.05	0.02	0.03
32	36	4	13,978.61	3,494.65	7,200 (1)	3.19	0.8	1.8
33	42	1	0.83	0.83	0.83	0.63	0.63	0.63
34	24	2	5.55	2.77	5.2	0.17	0.09	0.09
35	61	2	116.11	58.05	114.53	6.16	3.08	5.67
36	23	2	5.05	2.52	4.77	1.23	0.62	1.11
37	51	2	10.89	5.45	9.45	0.05	0.02	0.03
38	66	3	58.67	19.56	57.02	1.58	0.53	1.44
39	186	4	21,740.28	5,435.07	7,200 (2)	15,105.2	3,776.3	7,200 (2)
40	252	2	7,212.41	3,606.2	7,200 (1)	176.02	88.01	162.69
Average		*2*	*2,247*	*617*	*977*	*947*	*237*	*550*

Notes: [1]numbers in parentheses is the number of non-dominated solutions that remain unsolved in two hours.

The tables also include the number of non-dominated objective vectors that could not be found in 7,200 seconds. In finding the total, average, and maximum CPU times, the CPU times of the unsolved instances are taken as 7,200 seconds.

As can be observed from the above tables, the number of non-dominated vectors increases as the number of periods increases. Note that on average there are 2 and 4 non-dominated objective vectors when there are 7 and 14 periods, respectively. Moreover, the average CPU time, i.e., the average time to reach a non-dominated objective vector, increases from 617 seconds to 2,322 seconds for model I and from 237 seconds to 678 seconds for model II, as T increases from 7 to 14.

From Tables 1 and 2, we observe the significantly better performance of model II over model I. Note that for $T = 7$ the average total and average CPU times for model I are 2,247 and 617 seconds, respectively. The respective average total and average CPU times reduce to 947 and 237 seconds when model II is used. Moreover, when $T = 7$ and 14, model I cannot find 10 out of 82 and 19 out of 56 non-dominated objective vectors, respectively. On the other hand, model II cannot find 5 and 6 non-dominated objective vectors, for T = 7 and 14, respectively. Those results are due to the fact the decision variables of model I are related to the total demand, whereas model II generates the decision variables based on the total number of panels used.

In many practical applications and in our real case, the number of panels is significantly smaller than the total demand, hence giving a nice challenge for model II.

Table 2 Models I and II performances, T = 14, *SetR*

Instance	*D*	*Number of non-dom. vectors*	*Model I*			*Model II*		
			Total CPU time	*Avg. CPU time*	*Max. CPU time*	*Total CPU time*	*Avg. CPU time*	*Max. CPU time*
1	60	5	22,312.14	4,462.43	7,200 (3)[1]	18.28	3.66	7.03
2	22	3	1.13	0.38	0.56	4.94	1.65	2.75
3	44	2	178.22	89.11	176.03	0.28	0.14	0.16
4	45	1	1.89	1.89	1.89	0.03	0.03	0.03
5	238	2	7,231.25	3,615.63	7,200 (1)	25.64	12.82	23.77
6	288	3	8,022.14	2,674.05	7,200 (1)	2.38	0.79	2.17
7	42	4	386.75	96.69	232.34	4.25	1.06	2.28
8	56	3	251.42	83.81	138.41	450.3	150.1	282.14
9	57	4	2,924.16	731.04	1,406.63	6.16	1.54	3.17
10	128	3	8,721.84	2,907.28	7,200 (1)	5,506.23	1,835.41	4,111.95
11	144	3	960.56	320.19	662.36	2.03	0.68	1.88
12	50	6	28,875.61	4,812.6	7,200 (4)	40.16	6.69	13.48
13	66	3	10,854.89	3,618.3	7,200 (1)	37.06	12.35	19.3
14	84	3	10,833.23	3,611.08	7,200 (1)	7,253.53	2,417.84	7,200 (1)
15	117	5	15,903.08	3,180.62	7,200 (2)	43.56	8.71	20.88
16	438	6	41,701.5	6,950.25	7,200 (5)	38,354.44	6,392.41	7,200 (5)
Average		*4*	*9,947*	*2,322*	*4,215*	*3,234*	*678*	*1,181*

Notes: [1]numbers in parentheses is the number of non-dominated solutions that remain unsolved in two hours.

We next study the effect of the improvement mechanisms on the efficiency of model II and report the results for model II and improved model II using *SetR* instances for 7 and 14 periods, in Tables 3 and 4, respectively.

As can be observed from Tables 3 and 4, the performance of model II improves significantly by the improvement mechanisms. This implies the effort spent to generate the mechanisms is much less than the CPU time reductions obtained through their use. For almost all instances – with few exceptions having too small differences – the CPU time required by improved model II is less than that required by model II. The average CPU times for T = 7 and 14 reduce from 237 to 12 seconds and from 678 to 173 seconds, respectively. The number of unsolved instances is 5 for model II while improved model II finds all non-dominated objective vectors when T = 7. When T = 14, the respective unsolved instances are 6 and 2 for model II and improved model II.

Table 3 Model II and improved model II performances, $T = 7$, *SetR*

Instance	*D*	*Number of non-dom. vectors*	*Model II*			*Improved model II*		
			Total CPU time	*Avg. CPU time*	*Max. CPU time*	*Total CPU time*	*Avg. CPU time*	*Max. CPU time*
1	45	2	1.17	0.59	1.13	0.06	0.03	0.05
2	50	2	4.77	2.38	4.23	0.13	0.06	0.08
3	32	2	3.09	1.55	3.05	0.14	0.07	0.09
4	54	3	1.42	0.47	1.41	0.13	0.04	0.11
5	300	3	7,583.06	2,527.69	7,200 (1)[1]	180.83	60.28	173.58
6	61	3	13.02	4.34	7.09	2.70	0.90	1.20
7	264	5	14,617.72	2,923.54	7,200 (2)	1,856.50	371.30	1,808.53
8	24	2	2.92	1.46	2.63	0.20	0.10	0.19
9	24	2	0.08	0.04	0.06	0.03	0.02	0.03
10	36	3	0.78	0.26	0.59	0.09	0.03	0.05
11	10	1	0.02	0.02	0.02	0	0	0
12	12	2	0.19	0.09	0.14	0.06	0.03	0.03
13	29	1	0.06	0.06	0.06	0	0	0
14	15	1	0.02	0.02	0.02	0.05	0.05	0.05
15	15	1	0	0	0	0	0	0
16	30	1	0	0	0	0.03	0.03	0.03
17	94	1	0	0	0	0	0	0
18	144	1	0.11	0.11	0.11	0	0	0
19	72	1	0.03	0.03	0.03	0	0	0
20	216	2	0.16	0.08	0.14	0.02	0.01	0.02
21	24	2	0.03	0.02	0.03	0.02	0.01	0.02
22	18	2	0.16	0.08	0.09	0.03	0.02	0.03
23	16	2	4.56	2.28	4.17	0.13	0.06	0.11
24	40	2	23.56	11.78	21.25	0.14	0.07	0.08
25	9	1	0	0	0	0	0	0
26	48	3	1.94	0.65	1.77	0.08	0.03	0.05
27	56	1	3.48	3.48	3.48	0.11	0.11	0.11
28	72	3	326.94	108.98	192.14	9.89	3.30	4.72
29	120	2	0.06	0.03	0.03	0	0	0
30	24	1	0	0	0	0	0	0
31	14	2	0.05	0.02	0.03	0.03	0.02	0.03
32	36	4	3.19	0.80	1.80	0.14	0.04	0.05
33	42	1	0.63	0.63	0.63	0.03	0.03	0.03
34	24	2	0.17	0.09	0.09	0.03	0.02	0.02
35	61	2	6.16	3.08	5.67	3.53	1.77	2.97

Notes: [1] numbers in parentheses is the number of non-dominated solutions that remain unsolved in two hours.

Table 3 Model II and improved model II performances, $T = 7$, *SetR* (continued)

Instance	*D*	*Number of non-dom. vectors*	*Model II*			*Improved model II*		
			Total CPU time	*Avg. CPU time*	*Max. CPU time*	*Total CPU time*	*Avg. CPU time*	*Max. CPU time*
36	23	2	1.23	0.62	1.11	0.02	0.01	0.02
37	51	2	0.05	0.02	0.03	0.02	0.01	0.02
38	66	3	1.58	0.53	1.44	0.11	0.04	0.09
39	186	4	15,105.20	3,776.30	7,200 (2)	118	29.5	96.59
40	252	2	176.02	88.01	162.69	2.25	1.125	1.47
Average		*2*	*947*	*237*	*550*	*54*	*12*	*52*

Notes: [1]numbers in parentheses is the number of non-dominated solutions that remain unsolved in two hours.

Table 4 Model II and improved model II performances, $T = 14$, *SetR*

Instance	*D*	*Number of non-dom. vectors*	*Model II*			*Improved model II*		
			Total CPU time	*Avg. CPU time*	*Max. CPU time*	*Total CPU time*	*Avg. CPU time*	*Max. CPU time*
1	60	5	18.28	3.66	7.03	0.77	0.15	0.34
2	22	3	4.94	1.65	2.75	0.13	0.04	0.08
3	44	2	0.28	0.14	0.16	0.02	0.01	0.02
4	45	1	0.03	0.03	0.03	0.09	0.09	0.09
5	238	2	25.64	12.82	23.77	0.98	0.49	0.92
6	288	3	2.38	0.79	2.17	0.08	0.03	0.06
7	42	4	4.25	1.06	2.28	0.48	0.12	0.44
8	56	3	450.30	150.10	282.14	1.86	0.62	0.78
9	57	4	6.16	1.54	3.17	0.14	0.04	0.06
10	128	3	5,506.23	1,835.41	4,111.95	34.09	11.36	14.92
11	144	3	2.03	0.68	1.88	0.06	0.02	0.06
12	50	6	40.16	6.69	13.48	1.50	0.25	0.66
13	66	3	37.06	12.35	19.30	0.48	0.16	0.30
14	84	3	7,253.53	2,417.84	7,200 (1)[1]	26.13	8.71	24.66
15	117	5	43.56	8.71	20.88	1.53	0.31	1.02
16	438	6	38,354.44	6,392.41	7,200 (5)	16,436.77	2,739.46	7,200 (2)
Average		*4*	*3,234*	*678*	*1,181*	*1,032*	*173*	*453*

Notes: [1]numbers in parentheses is the number of non-dominated solutions that could not be found in two hours.

5.2.2 Results for the instances from the literature, SetL

We continue our experiments using the data from the literature. We solve the instances with improved model II, attributing to its superiority over models I and II. Tables 5 and 6 report the results of the experiments with $T = 7$ and 14, respectively. For each T, we use two different values of *n* (3 and 5 item types). We observe that the *D* values increase as *n* increases.

As *n* (thereby *D*) or *T* increases, the minimum and the maximum number of panel values in the non-dominated objective vectors set increase; however, the range of the minimum and the maximum number of panel values, thereby the number of non-dominated objective vectors, may not be affected. So we do not expect any remarkable increase in the number of non-dominated objective vectors with increases in *n* or *T*. The results in Tables 5 and 6 are in line with our expectations. The number of non-dominated objective vectors increases slightly as *T* increases. Note that the respective average number of non-dominated objective vectors are 4 and 5 for $n = 3$ and 5 when $T = 7$ and the respective average number of non-dominated objective vectors are 9 and 8 for $n = 3$ and 5 when $T = 14$. We also observe that there is no relation between *D* and the number of the non-dominated objective vectors. For example for instances 2s, 3s and there are seven periods (Table 5), the number of non-dominated objective vectors increases from 2 to 4 as *D* increases from 45 to 55 when $n = 3$ and decreases from 4 to 10 as *D* increases from 89 to 112 when $n = 5$.

Table 5 Improved model II performance, $T = 7$, *SetL*

Instance	*n = 3*					*n = 5*				
	D	*Number of non-dom. vectors*	*Total CPU time*	*Avg. CPU time*	*Max. CPU time*	*D*	*Number of non-dom. vectors*	*Total CPU time*	*Avg. CPU time*	*Max. CPU time*
A1	59	5	13.64	2.73	6.13	113	5	230.67	46.13	99.34
A2	62	4	35.77	8.94	25.64	116	5	12,736.02	2,547.2	7,200 (1)[1]
A3	42	3	0.81	0.27	0.44	104	6	251.94	41.99	78.03
A4	65	5	28.73	5.75	9	99	5	74.28	14.86	40.95
A5	60	4	10.44	2.61	5.88	109	5	158.33	31.67	74.31
CHL2	52	5	8.86	1.77	3.55	106	5	167.72	33.54	119.34
CHL5	55	5	3.16	0.63	1.02	90	5	20.47	4.09	6.22
Hchl3s	52	3	13.56	4.52	12.94	120	4	79.42	19.86	62.23
Hchl4s	50	4	13.7	3.43	7.34	107	5	183.64	36.73	94.31
Hchl6s	53	4	3.41	0.85	1.58	110	4	34.28	8.57	21.11
Hchl8s	65	7	4.64	0.66	1.56	107	4	2.64	0.66	1.13
HH	55	3	3.42	1.14	1.81	103	5	229.95	45.99	170.81
Of1	49	3	7.92	2.64	3.77	116	7	108	15.43	36.36
Of2	41	5	3.17	0.63	1.39	101	6	179.61	29.93	100.09
Sts4	44	5	11	2.2	6.34	120	4	59.92	14.98	38.78
W	65	5	36.66	7.33	13.31	121	6	258.28	43.05	78.17
2s	45	2	6.38	3.19	5.91	112	4	183.13	45.78	117.75
3s	55	4	16.36	4.09	10.34	89	10	242.58	24.26	97.28
Average		*4*	*12*	*3*	*7*		*5*	*844*	*167*	*469*

Notes: [1]numbers in parentheses is the number of non-dominated solutions that could not be found in two hours.

We first observe the effect of the total item demand on the complexity of the solutions. For fixed *n* and *T*, an increase in the total demand increases the CPU times. Note from Table 6 that when *n* = 3 and *T* = 14, A2 and Of2 instances have a total demand value of 103 and 76, respectively and their respective average CPU times are 455.62 and 6.57 seconds. When *n* = 5 those instances have total demand values of 191 and 148 leaving six instances and a single instance unsolved in two hours.

Table 6 Improved model II performance, *T* = 14, *SetL*

	n = 3					*n = 5*				
Instance	*D*	*Number of non-dom. vectors*	*Total CPU time*	*Avg. CPU time*	*Max. CPU time*	*D*	*Number of non-dom. vectors*	*Total CPU time*	*Avg. CPU time*	*Max. CPU time*
A1	101	12	10,658.95	888.25	7,200 (1)	173	11	38,566.27	3,506.02	7,200 (5)[1]
A2	103	11	5,011.80	455.62	1,116.56	191	9	44,942.13	4,993.57	7,200 (6)
A3	82	8	219.25	27.41	81.66	190	5	1,510.39	302.08	1,040.80
A4	95	9	160.67	17.85	48.08	175	9	18,345.39	2,038.38	7,200 (2)
A5	98	6	741.17	123.53	611.86	172	7	6,299.38	899.91	5,320.84
CHL2	89	8	1,762.98	220.37	656.50	192	7	10,109.05	1,444.15	3,933.86
CHL5	85	9	48.53	5.39	12.09	151	11	26,229.19	2,384.47	7,200 (1)
Hchl3s	91	9	371.14	41.24	135.19	182	9	12,255.89	1,361.77	6,195.52
Hchl4s	91	8	109.28	13.66	68.09	182	12	40,627.78	3,385.65	7,200 (4)
Hchl6s	89	8	93.33	11.67	21.66	179	7	6,187.81	883.97	5,175.73
Hchl8s	98	8	33.81	4.23	10.47	174	7	6,69.61	95.66	142.89
HH	93	6	45.52	7.59	13.42	176	11	47,924.64	4,356.79	7,200 (5)
Of1	78	9	407.42	45.27	153.73	183	8	31,202.17	3,900.27	7,200 (4)
Of2	76	6	39.44	6.57	14.22	148	9	15,035.69	1,670.63	7,200 (1)
Sts4	83	10	107.59	10.76	34.31	192	9	13,493.09	1,499.23	7,200 (1)
W	88	9	2,727.08	303.01	1,197.28	193	14	32,849.84	2,346.42	7,200 (4)
2s	79	10	234.06	23.41	87.45	176	10	41,454.72	4,145.47	7,200 (5)
3s	96	11	418.70	38.06	72.44	155	11	32,449.22	2,949.93	7,200 (2)
Average		*9*	*1,288*	*125*	*641*		*8*	*23,342*	*2,342*	*6,012*

Notes: [1] numbers in parentheses is the number of non-dominated solutions that remain unsolved in two hours.

Recall that D values increase as n increases. Tables 5 and 6 show that as n, thereby D, increases, the CPU times increase significantly. When $T = 7$, the respective average CPU times are 3 and 167 seconds for $n = 3$ and 5 and when $T = 14$, the respective average CPU times are 125 and 2,342 seconds for $n = 3$ and 5. This is because as n and D increase the number of binary variables, thereby the complexity of the mixed integer programs, increases exponentially. Note from Table 5 that for the problem with $n = 3$ and $T = 7$, all 76 non-dominated objective vectors are solved very quickly with an average maximum CPU time of seven seconds. When n becomes 5, in only 1 out of 95 non-dominated objective vectors, the optimal solution cannot be found in two hours. Once the unsolved instance is excluded, the average CPU time decreases from 167 to 102 seconds.

5.2.3 Comparison of SetR and SetL instances

We next discuss the performances of *SetR* and *SetL* for $T = 7$ when $n = 3$ and 5 and $T = 14$ when $n = 3$. When $T = 7$, *SetR* resides three instances with $n = 3$, having an average of seven non-dominated objective vectors and four instances with $n = 5$ having an average of 11 non-dominated objective vectors. *SetL* has 4 and 5 non-dominated objective vectors when $n = 3$ and $n = 5$, respectively. The total demand of *SetR* (71 and 136 for $n = 3$ and 5, respectively) is higher than the total demand of *SetL* (54 and 90 for $n = 3$ and 5, respectively) hence *SetL* has fewer average non-dominated objective vectors.

The results for $T = 7$ show that *SetL* has higher CPU times than *SetR* which is due to the distribution of the total demand over the planning horizon. The total demand of *SetR* (71 and 136 for $n = 3$ and 5, respectively) is associated with a couple of periods while the total demand of *SetL* (54 and 90 for $n = 3$ and 5, respectively) is distributed evenly over all periods. When improved model II is used, the average CPU times are 0.02 and 3 seconds for *SetR* and *SetL*, respectively for $n = 3$, while the respective average CPU times are 12 and 167 seconds for *SetR* and *SetL*, for $n = 5$. A similar observation can be done for $n = 3$ and $T = 14$ combination. There are five instances with 11 non-dominated objective vectors with an average total demand of 127 in *SetR*. In *SetL*, the average total demand over 18 instances is 108. The average CPU times are 0.13 and 125 seconds for *SetR* and *SetL*, respectively.

We observe the number of non-dominated solutions for both sets is almost below 10. Both in the real life and literature instances the total demand is not low however the units come from few item types, thereby serving to the same purpose. This keeps similar inventory carrying amounts among different solutions, thereby producing no so high non-dominated objective vectors.

To summarise, we obtain exact non-dominated objective vectors for the instances with up to five items and up to 200 units of total demand when there are seven periods and three items having up to 100 units of total demand when there are 14 periods. For the problems of bigger sizes, approximate non-dominated objective vectors might be of great help. To get such vectors, decomposition-based heuristic approaches that benefit from our models can be developed. The decomposition might use from item decomposition or period decomposition ideas.

6 Conclusions

In this study, we consider an integrated cutting stock and lot sizing problem where two dimensional items are to be cut from two dimensional blocks of identical size to satisfy all demands over a specified planning horizon. We study two objectives: minimising the number of panels (cutting-stock related) and total inventory carrying cost (lot sizing related) and aim to generate all non-dominated objective vectors. We show that the objectives are conflicting and there is a non-dominated objective vector corresponding to each number of panels in a specified range. To generate each objective vector, we use a MILP model that minimises the total inventory cost subject to a specified number of panels. We enhance the efficiency of the model by incorporating optimality properties and bounding mechanisms.

Our interest in the problem is from an aircraft manufacturing plant in Turkey whose production planners want to reduce the total purchasing and keeping costs of the big steel panels while carrying small amounts of fragile steel items. Our approaches might also be used to help the managers of other industries like furniture manufacturing having a high demand for small wooden items that form final furniture products over long periods. To the best of our knowledge, we propose the first multi-criteria approach for the integrated two dimensional cutting stock and lot sizing problem. Future research may consider some extensions like allowing backorders (lot sizing problem) and non-guillotine cuts (cutting stock problem). Another promising extension might be to analyse the preferences of the company managers and find a representative set of non-dominated objective vectors that favour those preferences.

Acknowledgements

We thank Professor Murat Koksalan for his invaluable suggestions and two reviewers for their constructive comments.

References

Aliano Filho, A., Moretti, A.C., Pato, M.V. and Oliveira, W.A. (2019) ‘An exact scalarization method with multiple reference points for bi-objective integer linear optimization problems’, *Annals of Operations Research*, Vol. 296, No. 1, pp.1–35.

Aliano Filho, A.A., Moretti, A.C. and 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*, Vol. 69, No. 1, pp.91–107.

Andrade, P.R.d.L., de Araujo, S.A., Cherri, A.C. and Lemos, F.K. (2021) ‘The integrated lot sizing and cutting stock problem in an automotive spring factory’, *Applied Mathematical Modelling*, Vol. 91, No. 1, pp.1023–1036.

Ayasandır, U. and Azizoğlu, M. (2022) ‘Two dimensional cutting stock problem with multiple stock sizes’, *International Journal of Planning and Scheduling*, DOI: 10.1504/IJPS.2021.10043393.

Campello, B.S.C., Ghidini, C.T.L.S., Ayres, A.O.C. and Oliveira, W.A. (2020) ‘A multiobjective integrated model for lot sizing and cutting stock problems’, *Journal of the Operational Research Society*, Vol. 71, No. 9, pp.1466–1478.

Cintra, G.F., Miyazawa, F.K., Wakabayashi, Y. and Xavier, E.C. (2008) 'Algorithms for two-dimensional cutting stock and strip packing problems using dynamic programming and column generation', *European Journal of Operational Research*, Vol. 191, No. 1, pp.61–85.

Cristofoletti, M.M., de Araujo, S.A. and Cherri, A.N. (2021) 'Integrated lot-sizing and cutting stock problem applied to the mattress industry', *Journal of the Operational Research Society*, Vol. 72, No. 6, pp.1279–1293.

Cui, Y. and Yang, Y. (2010) 'A heuristic for the one-dimensional cutting stock problem with usable leftover', *European Journal of Operational Research*, Vol. 204, No. 2, pp.245–250.

De Armas, J., Miranda, G. and León, C. (2011) 'A multi-objective approach for the 2D guillotine cutting stock problem', *Lecture Notes in Computer Science, LNCS Part 2*, Vol. 6692, pp.292–299.

Delorme, M., Iori, M. and Martello, S. (2016) 'Bin packing and cutting stock problems: mathematical models and exact algorithms', *European Journal of Operational Research*, Vol. 255, No. 1, pp.1–20.

Farley, A.A. (1988) 'Mathematical programming models for cutting-stock problems in the clothing industry', *Journal of the Operational Research Society*, Vol. 39, No. 1, pp.41–53.

Furini, F. and Malaguti, E. (2013) 'Models for the two-dimensional two-stage cutting stock problem with multiple stock size', *Computers and Operations Research*, Vol. 40, No. 8, pp.1953–1962.

Furini, F., Malaguti, E. and Thomopulos, D. (2016) 'Modeling two-dimensional guillotine cutting problems via integer programming', in *INFORMS Journal on Computing*, Vol. 28, No. 4, pp.736–751.

Furini, F., Malaguti, E., Medina Durán, R., Persiani, A. and Toth, P. (2012) 'A column generation heuristic for the two-dimensional two-staged guillotine cutting stock problem with multiple stock size', *European Journal of Operational Research*, Vol. 218, No. 1, pp.251–260.

Gilmore, P.C. and Gomory, R.E. (1965) 'Multistage cutting stock problems of two and more dimensions', *Operations Research*, Vol. 13, No. 1, pp.94–120.

Gonzalez, Y., Miranda, G. and Leon, C. (2016) 'Multi-objective multi-level filling evolutionary algorithm for the 3D cutting stock problem', *Procedia Computer Science*, Vol. 96, pp.355–364.

Gramani, M.C.N. and França, P.M. (2006) 'The combined cutting stock and lot-sizing problem in industrial processes', *European Journal of Operational Research*, Vol. 174, No. 1, pp.509–521.

Gramani, M.C.N., França, P.M. and Arenales, M.N. (2009) 'A Lagrangian relaxation approach to a coupled lot-sizing and cutting stock problem', *International Journal of Production Economics*, Vol. 119, No. 2, pp.219–227.

Haessler, R.W. (1971) 'A heuristic programming solution to a nonlinear cutting stock problem', *Management Science*, Vol. 17, No. 12, pp.B-793–B-802.

Hendry, L.C., Fok, K.K. and Shek, K.W. (1996) 'A cutting stock and scheduling problem in the copper industry', *Journal of the Operational Research Society*, Vol. 47, No. 1, pp.38–47.

Hifi, M. and Roucairol, C. (2001) 'Approximate and exact algorithms for constrained (un)weighted two-dimensional two-staged cutting stock problems', *Journal of Combinatorial Optimization*, Vol. 5, No. 4, pp.465–494.

Kolen, A.W.J. and Spieksma, F.C.R. (2000) 'Solving a bi-criterion cutting stock problem with open-ended demand: a case study', *Journal of the Operational Research Society*, Vol. 51, No. 11, pp.1238–1247.

Leao, A.A.S., Furlan, M.M. and Toledo, F.M.B. (2017) 'Decomposition methods for the lot-sizing and cutting-stock problems in paper industries', *Applied Mathematical Modelling*, Vol. 48, No. 1, pp.250–268.

Margot, F. (2010) 'Symmetry in integer linear programming', in *50 Years of Integer Programming 1958–2008*, pp.647–686, Springer, Berlin, Heidelberg.

Martin, M., Birgin, E.G., Lobato, R.D., Morabito, R. and Munari. P. (2020) 'Models for the two-dimensional rectangular single large placement problem with guillotine cuts and constrained pattern', in *International Transactions in Operational Research*, Vol. 27, No. 2, pp.767–793.

Melega, G.M., de Araujo, S.A. and Jans, R. (2018) 'Classification and literature review of integrated lot-sizing and cutting stock problems', *European Journal of Operational Research*, Vol. 271, No. 1, pp.1–19.

Mellouli, A., Mellouli, R. and Masmoudi, F. (2019) 'An innovative genetic algorithm for a multi-objective optimization of two-dimensional cutting-stock problem', *Applied Artificial Intelligence*, Vol. 33, No. 6, pp.531–547.

Nonås, S.L. and Thorstenson, A. (2000) 'Combined cutting-stock and lot-sizing problem', *European Journal of Operational Research*, Vol. 120, No. 2, pp.327–342.

Nonås, S.L. and Thorstenson, A. (2008) 'Solving a combined cutting-stock and lot-sizing problem with a column generating procedure', *Computers and Operations Research*, Vol. 35, No. 10, pp.3371–3392.

Oliveira, W.A., Fiorotto, D.J., Song, X. and Jones, D.F. (2021) 'An extended goal programming model for the multiobjective integrated lot-sizing and cutting stock problem', *European Journal of Operational Research*, Vol. 295, No. 3, pp.996–1007.

Poltroniere, S.C., Poldi, K.C., Toledo, F.M.B. and Arenales, M.N. (2008) 'A coupling cutting stock-lot sizing problem in the paper industry', *Annals of Operations Research*, Vol. 157, No. 1, pp.91–104.

Signorini, C., de Araujo, S.A. and Melega, G.M. (2022) 'One dimensional multi-period cutting stock problems in the concrete industry', *International Journal of Production Research*, Vol. 80, No. 8, pp.2386–2403.

Silva, E., Alvelos, F. and Valério de Carvalho, J.M. (2010) 'An integer programming model for two- and three-stage two-dimensional cutting stock problems', *European Journal of Operational Research*, Vol. 205, No. 3, pp.699–708.

Silva, E., Alvelos, F. and Valério De Carvalho, J.M. (2014) 'Integrating two-dimensional cutting stock and lot-sizing problems', *Journal of the Operational Research Society*, Vol. 65, No. 1, pp.108–123.

Vanzela, M., Melega, G.M., Rangel, S. and de Araujo, S.A. (2017) 'The integrated lot sizing and cutting stock problem with saw cycle constraints applied to furniture production', *Computers and Operations Research*, Vol. 79, No. 1, pp.148–160.

Appendix

Table A1 Features of the problem instances, $T = 7$, *SetR*

Instance	*n*	*D*	w_{max}	w_{min}	l_{max}	l_{min}	*W*	*L*
1	3	45	60	55	2,785	1,930	1,220	3,900
2	4	50	165	56	3,048	66	1,220	3,660
3	5	32	508	102	3,048	152	1,220	3,660
4	5	54	170	75	230	75	1,220	2,500
5	5	300	400	85	1,200	110	1,250	2,500
6	7	61	279	89	2,438	140	1,220	3,660
7	11	264	146	76	965	117	1,220	3,660
8	14	24	914	70	2,235	152	1,220	3,660

Note: *split instances.

Table A1 Features of the problem instances, $T = 7$, *SetR* (continued)

Instance	n	D	w_{max}	w_{min}	l_{max}	l_{min}	W	L
9*	2	24	125	64	215	175	1,220	3,660
10*	2	36	125	64	215	175	1,220	3,660
11*	1	10	102	102	3,048	3,048	1,220	3,660
12*	2	12	1,016	508	2,540	1,092	1,220	3,660
13*	2	29	76	51	432	114	1,220	3,660
14*	1	15	254	254	254	254	1,220	3,660
15*	1	15	55	55	2,300	2,300	1,220	3,900
16*	2	30	60	55	2,325	2,300	1,220	3,900
17*	1	94	64	64	175	175	1,250	2,500
18*	2	144	125	125	215	200	1,250	2,500
19*	1	72	110	110	120	120	1,220	3,660
20*	2	216	90	45	150	110	1,220	3,660
21*	2	24	191	114	241	114	1,220	3,660
22*	2	18	216	51	330	211	1,220	3,660
23*	2	16	590	590	810	810	1,250	2,500
24*	4	40	620	590	810	670	1,250	2,500
25*	1	9	64	64	89	89	1,220	3,660
26*	3	48	76	33	279	84	1,220	3,660
27*	4	56	620	590	810	670	1,250	2,500
28*	4	72	620	590	810	670	1,250	2,500
29*	3	120	89	70	311	127	1,220	3,660
30*	1	24	95	95	622	622	1,220	3,660
31*	2	14	89	53	1,046	104	1,220	3,660
32*	4	36	254	53	2,515	104	1,220	3,660
33*	5	42	610	89	711	191	1,220	3,660
34*	2	24	140	89	140	102	1,220	3,660
35*	7	61	1,016	61	3,048	2,540	1,220	3,660
36*	2	23	229	97	3,048	838	1,220	3,660
37*	2	51	152	89	330	216	1,220	3,660
38*	7	66	152	51	229	89	1,220	3,660
39*	6	186	585	45	2,485	60	1,250	2,500
40*	5	252	410	40	1,810	45	1,250	2,500

Note: *split instances.

Table A2 Features of the problem instances, $T = 14$, *SetR*

Instance	*n*	*D*	w_{max}	w_{min}	l_{max}	l_{min}	*W*	*L*
1	2	60	125	64	215	175	1,220	3,660
2	3	22	1,016	102	3,048	1,092	1,220	3,660
3	3	44	254	51	432	114	1,220	3,660
4	3	45	60	55	2,325	2,300	1,220	3,900
5	3	238	125	64	215	175	1,250	2,500
6	3	288	110	45	150	110	1,220	3,660
7	4	42	216	51	330	114	1,220	3,660
8	4	56	620	590	810	670	1,250	2,500
9	4	57	76	53	279	84	1,220	3,660
10	4	128	620	590	810	670	1,250	2,500
11	4	144	95	70	622	127	1,220	3,660
12	6	50	254	53	2,515	104	1,220	3,660
13	7	66	610	89	711	102	1,220	3,660
14	9	84	1,016	61	3,048	838	1,220	3,660
15	9	117	152	51	330	89	1,220	3,660
16	10	438	585	40	2,485	45	1,250	2,500

Table A3 Features of the problem instances taken from the literature, *SetL*

							T = 7		*T = 14*	
Instance	w_{max}	w_{min}	l_{max}	l_{min}	*W*	*L*	*n = 3*	*n = 5*	*n = 3*	*n = 5*
							D	*D*	*D*	*D*
A1	43	11	33	9	54	60	59	113	101	173
A2	42	14	33	12	54	72	62	116	103	191
A3	43	14	35	15	72	84	42	104	82	190
A4	43	11	33	9	63	108	65	99	95	175
A5	63	12	69	13	90	158	60	109	98	172
CHL2	31	9	31	11	49	74	52	106	89	192
CHL5	14	2	20	1	18	24	55	90	85	151
Hchl3s	65	13	54	15	88	152	52	120	91	182
Hchl4s	65	13	54	15	88	152	50	107	91	182
Hchl6s	101	38	109	35	219	303	53	110	89	179
Hchl8s	14	2	20	1	18	58	65	107	98	174
HH	65	13	54	18	88	152	55	103	93	176
Of1	36	4	55	9	36	84	49	116	78	183
Of2	27	4	47	13	36	84	41	101	76	148
Sts4	49	16	44	14	89	118	44	120	83	192
W	33	9	43	11	36	84	65	121	88	193
2s	35	7	31	9	63	48	45	112	79	176
3s	43	11	33	9	63	48	55	89	96	155