processes

MDPI

Article

Boosted Arc Flow Formulation Using Graph Compression for the Two-Dimensional Strip Cutting Problem

Tamer G. Ali [1], Mehdi Mrad [2,*], Ali Balma [3], Anis Gharbi [1,*], Ali Samhan [1] and Mohammed A. Louly [4]

[1] Industrial Engineering Department, College of Engineering, King Saud University, P.O. Box 800, Riyadh 11421, Saudi Arabia
[2] Ecole Supérieure des Sciences Economiques et Commerciales de Tunis, Université de Tunis, Tunis 1089, Tunisia
[3] Ecole Nationale Supérieure d'Ingénieurs de Tunis, Université de Tunis, Tunis 1008, Tunisia
[4] Pôle Scientifique et Technique, Ecole Supérieure Polytechnique Mauritanie, Nouakchott P.O. Box 4303, Mauritania
* Correspondence: mehdi.mrad@essect.u-tunis.tn (M.M.); a.gharbi@ksu.edu.sa (A.G.)

Abstract: Since the requirement for a material cutting process occurs in a wide variety of applied contemporary manufacturing, the cutting stock problem plays a critical role in optimizing the amount of raw material utilized in everyday production operations. In this paper, we address the two-dimension strip-cutting problem and implement the graph compression technique to improve the performance of the arc-flow formulation. The number of variables of the obtained mathematical model are substantially reduced. A comparative study on a large set of benchmark instances shows that our compressed model yields very good results for the non-unitary item demand case in contrast to the state-of-the-art mathematical models. Moreover, improved bounds are provided for 24 unsolved benchmark instances, among which 8 have been solved to optimality.

Keywords: integer programming; cutting stock problem; arc flow formulation

Citation: Ali, T.G.; Mrad, M.; Balma, A.; Gharbi, A.; Samhan, A.; Louly, M.A. Boosted Arc Flow Formulation Using Graph Compression for the Two-Dimensional Strip Cutting Problem. *Processes* **2023**, *11*, 790. https://doi.org/10.3390/pr11030790

Academic Editors: Emad Abouel Nasr, Abdulrahman Al-Ahmari and Adham Ragab

Received: 30 January 2023
Revised: 22 February 2023
Accepted: 1 March 2023
Published: 7 March 2023

1. Introduction

Achieving a good level of profit requires a well-prepared production plan, settings, and optimized usage of available resources in production operations. In many services and industries (such as logistics, manufacturing [1], and health services [2]), one of the challenging problems with wide applications in the industrial and production field is the cutting problem. In particular, finding the best way to trim raw materials in order to manufacture finished goods while maintaining the maximum profit and minimizing the environmental effect is difficult in the manufacturing industry. The cutting problem has been tackled to solve various real-life manufacturing cases. These include, among others, applications in coronary stent manufacturing [2], the glass manufacturing industry [3], silicon steel coil manufacturing [4], paper production processes [5], green manufacturing [6], plastic rolls manufacturing [7], multiple manufacturing modes applied to construction industry [8], and TFT-LCD manufacturing [9].

In the cutting stock problem, a set of items with specified dimensions must be cut from an available raw material to meet the required demand. The problem and its variants are in strong connection to real field applications in cutting processes. One variant of the problem consists of the two-level strip-cutting problem (hereinafter 2D-SCP). The 2D-SCP is theoretically equivalent to the two-dimensional strip packing problem (2D-SPP). The difference between the two versions is explained by the data structure of the instances related to the two problems and from the different developed methods to solve them. In fact, cutting instances are known by the high demand for items (high multiplicity factor) while the packing instances have generally unitary or small demands (low multiplicity

factor). Thus, in some cases, the developed algorithms to solve cutting instances may not solve the packing case efficiently and vice versa [1].

Formally, the 2D-SCP can be described as follows: a set of rectangular items $i = \{1, 2, .., m\}$ with width w_i, height h_i and demand d_i are to be cut from a strip with fixed width W. In the context of two-dimensional cutting problems, Lodi et al. [10] introduced the following classification of the 2D-SCP based on the items' orientation and the cutting types (guillotine/non-guillotine).

- RF: Rotation of items by 90° is allowed (R) and no guillotine restriction is imposed (F)
- RG: Rotation of items by 90° is allowed (R) and guillotine restriction is imposed (G)
- OF: All items have a fixed orientation (O) and no guillotine restriction is imposed (F)
- OG: All items have a fixed orientation (O) and guillotine restriction is imposed (G)

Our addressed problem belongs to the OG class. The guillotine cut constraint obliges the tool to cut the plate from edge to edge. Moreover, the number of cutting stages is only two. In the case of a guillotine cutting process, on one hand, increasing the number of cutting stages can reduce the waste and increase the efficiency of the cutting process. On the other hand, it increases the hardness of the problem and may cause disorder in the workshop.

In this paper, we modified the so-called graph compression technique and proposed it to enhance the arc-flow formulation of the two-dimensional strip cutting problem. The proposed compression steps reduced the size of the mathematical model that is based on arc-flow formulation. The impact of such an improvement is assessed on a large set of benchmark instances. Our experimental study shows the good performance of the compressed model where new improved lower/upper bounds are provided for open instances.

The sequel of this paper is organized as follows. Section 2 is devoted to the literature review of the problem and highlights the most familiar solution methods. In Section 3, three recent mathematical models presented in [11] are described. The graph compression technique is explained and applied on the 2D-SCP through a detailed example in Section 4. Section 5 includes the details of the experimental studies and the analysis of the obtained results. Finally, the conclusion of the paper is considered in Section 6.

2. Literature Review

In a pioneering work, Ref. [12] provided a set covering formulation for different versions of the two-dimensional cutting stock problem. This has been improved by Refs. [13,14] who introduced one-dimensional bounded knapsacks and branch-and-bound procedures, respectively. New models and bounds have been proposed by Ref. [15] to solve the two-dimensional bin packing problem (2BP) and the two-dimensional strip packing problem (2SP). The tightness of the proposed bounds were tested on 10 classes of instances, class 1–4 in Ref. [10] and class 5–10 in Ref. [16]. An effective branch-and-price algorithm associated with the generation of Chvatal–Gomory cutting planes is introduced by Ref. [17] for the two-dimensional cutting stock problem. Better bounds obtained by decomposition techniques and constraint programming were used by Ref. [18]. Dichotomous-based lower bounds introduced by Ref. [19] showed good computational results in terms of optimality. An extension of the one-cut model of Ref. [20] for the one-dimensional cutting stock problem has been performed by Ref. [21]. Macedo et al., in Ref. [22], extended the arc-flow model of one-dimensional case in Ref. [23] to solve the two-dimensional cutting stock problem and obtained better results than those in Ref. [15]. An SOS-based branch-and-price algorithm has been derived from the arc-flow formulation by Ref. [24].

Rinaldi and Franz [25] developed two heuristics to solve the two-dimensional strip-cutting problem with sequencing constraints. An efficient Branch-and-Price algorithm for the two-dimensional cutting stock and the two-dimensional strip-packing problems has been developed by Ref. [26]. Another column generation-based algorithm is presented by Ref. [27] with a set covering formulation of the two dimensional level strip packing problem. Mrad [28] developed a strong arc-flow model for the two-stage guillotine strip cutting problem. Bezerra et al. Ref. [11] proposed three mathematical formulations to

solve the 2D-SCP that turned out to be very sensitive to the structure of the considered instances. A hybrid heuristic based on new improved rules and reinforcement learning, proposed by Ref. [29], outperformed many heuristics from the literature for the 2D-SCP. For a comprehensive review of the two-dimensional cutting and packing problems, the reader may refer to the recent paper of Ref. [30].

3. State-of-the-Art Mathematical Models

In this section, the most recent mathematical models from literature for the 2D-SCP are presented. The literature stated the high flexibility of using a formulation of packing problem to solve the cutting version and vice versa. Three different formulations were rewritten by Ref. [11]. The first one is based on the model of Lodi et al. [15], while the second formulation is based on the work of Furini et al., Ref. [31], about the two dimensional two-staged guillotine cutting stock problem with multiple stock size. The third formulation is developed based on the model of Silva et al. [21]. An interesting model which is based on arc-flow formulation has been introduced by Ref. [28]. The proposed model made it feasible to efficiently solve some classes of the benchmark instances of the 2D-SCP.

3.1. The Model Based on Lodi et al. [15]

To each individual item, j is associated a shelf with height, h_j, that may include any item with height less than or equal to h_j (including item j itself). Note that not all the defined shelves need to be used. The decision variables are therefore defined as follows:

$$x_{jk} = \begin{cases} 1, \ if \ item \ j \ is \ cut \ from \ level \ k; (j, k = 1, .., n) \\ 0, \ otherwise \end{cases}$$

where $n = \sum_{i=1}^{m} d_i$ denotes the total number of items. Let $\alpha_i = \sum_{s=1}^{i} d_s, i = 1, \ldots, m$ with $\alpha_0 = 0$. The proposed model can be described by the following:

$$Min \ H \tag{1}$$

$$\sum_{k=1}^{j} x_{jk} = 1, \ j = 1, \ldots, n \tag{2}$$

$$\sum_{j=k+1}^{n} w_j x_{jk} \leq (W - w_k) x_{kk}, \ k = 1, \ldots, n-1 \tag{3}$$

$$\sum_{k=1}^{n} h_k x_{kk} \leq H \tag{4}$$

$$x_{tt} \geq x_{t+1,t+1}, t \in [\alpha_{i-1} + 1, \alpha_i - 1], \ i = 1, \ldots, m \tag{5}$$

$$\sum_{s=t+1}^{\alpha_i} x_{st} \geq \sum_{s=t+2}^{\alpha_i} x_{s,t+1}, t \in [\alpha_{i-1} + 1, \alpha_i - 1], i = 1, \ldots, m \tag{6}$$

$$x_{jk} \in \{0, 1\}, k = 1, .., n, j = k, \ldots, n \tag{7}$$

The objective function (1) aims to minimize the total height used from the strip. Constraints (2) ensure that each item is placed once on its own level or on a higher level. Constraints (3) oblige the width of any level to be enough to include all the assigned items. Constraint (4) is related to the objective function and forces its value to be larger or equal to the total used height. Constraints (5) and (6) are two valid inequalities of the two-dimensional knapsack problem that were introduced by Ref. [32]. The purpose of these two inequalities is to break symmetry that might occur in the solution. The model (1)–(7) is denoted by $M1ineq$.

3.2. The Model Based on Furini et al. [31]

The same method of cutting items into shelves, as explained in $M1ineq$, is adopted for this model. The latter is introduced by Ref. [31] for the two-dimensional two-staged guillotine cutting stock problem with multiple stock size. In addition to the decision

variables x_{jk}, a continuous decision variable y_k is included that represents the height of shelf k ($k = 1, \ldots, n$). The model is formulated as follows:

$$Min\ H \tag{8}$$

$$\sum_{k=1}^{n} y_k \leq H \tag{9}$$

$$\sum_{j=1}^{n} w_j x_{jk} \leq W \quad k = 1, \ldots, n \tag{10}$$

$$\sum_{k=1}^{n} x_{jk} = 1 \quad j = 1, \ldots, n \tag{11}$$

$$h_j x_{jk} \leq y_k \quad j, k = 1, \ldots, n \tag{12}$$

$$\sum_{j=1}^{n} (w_j h_j) x_{jk} \leq y_k W \quad k = 1, \ldots, n \tag{13}$$

$$y_k \leq y_{k+1} \quad k = 1, \ldots, n-1 \tag{14}$$

$$y_k \geq 0 \quad k = 1, \ldots, n \tag{15}$$

$$x_{jk} \in \{0, 1\} \quad j, k = 1, .., n \tag{16}$$

The objective function (8) minimizes the total height H used to cut all items. Constraint (9) assures that the total height of the shelves is less than or equal to the used strip height H. Constraints (10) state that the total width of shelves should be less than or equal to the strip width W. Constraints (11) amount to the initialization of each shelf k, which means that a shelf k is initialized by cutting item j. Constraints (12) ensure that the height of each item which is cut from shelf k should be smaller than or equal to the height y_k. Constraints (13) and (14) are valid inequalities that were introduced by Ref. [31] to enhance their formulation. Inequality (13) ensures that the overall area of items placed in shelf k is smaller than the area of the shelf k. Inequality (14) prevents the occurrence of symmetry, where the shelves are sorted in increasing order. The model (8)–(16) is denoted by *M*2*ineq*.

3.3. The Model Based on Silva et al. [21]

In fact, cutting one item produces two residual boards: the right residual board and the top residual board. For each item, the model checks iteratively the possibility of cutting some other items from the corresponding residual boards (see Example 1). Thus, all cuts and residual boards are known in advance.

Let B be the set of all residual boards. Note that for each $k \in B$, the residual boards with height and width (H_k, W_k) are such that $H_k \leq H$ and $W_k \leq W$. In addition, N designates the subset of B which dimensions are equal to one of the items. The model variables are:

$$x_{jk} = \begin{cases} 1, \ if\ item\ j\ is\ cut\ from\ level\ k; (j = 1, .., n; k = 0, \ldots, m) \\ 0, \ otherwise \end{cases}$$

$$a_{jki} = \begin{cases} 1, \ if\ board\ type\ i\ results\ from\ cutting\ item\ j\ from\ board\ k; (j = 1, .., n; k = 0, \ldots, m;\ i = 1, \ldots, m) \\ 0, \ otherwise \end{cases}$$

The model can then be formulated as follows:

$$Min\ H \tag{17}$$

$$\sum_{j=1}^{n} h_j x_{j0} \leq H \tag{18}$$

$$\sum_{k=0}^{m} x_{jk} = 1\ j = 1, \ldots, n \tag{19}$$

$$\sum_{k=0}^{m} \sum_{j=1}^{n} a_{jki} x_{jk} \geq \sum_{j=1}^{n} x_{ji}\ i \in N \tag{20}$$

$$\sum_{k=0}^{m} \sum_{j=1}^{n} a_{jki} x_{jk} \geq \sum_{j=1}^{n} x_{ji}\ i \in B \backslash N \tag{21}$$

$$x_{jk} \in \{0,1\}\ j = 1,\ldots,n;\ k = 0,..m \tag{22}$$

The objective function (17) minimizes the total consumed height H defined by (18). Constraints (19) ensure the demand satisfaction. In constraints (20) and (21), the number of residual boards obtained from cutting board i is greater than or equal to the number of cut items from board i. The difference between (20) and (21) is that in (20), the residual boards from item i still offer possible cutting smaller items. Finally, the domain of decision variables is given in (22). The model (17)–(22) is denoted by $M3$.

Example 1. *Consider a strip with an infinite height and fixed width W = 20. Three items are to be cut from the strip. The height and width (h_j,w_j) of each item j are presented in Table 1.*

Table 1. Data for Example 1.

Item j	Height h_j	Width w_j
1	40	20
2	30	10
3	20	40

The first phase is to establish all initial boards by applying a horizontal cut for each item j. In this cutting phase, three boards are initialized. As shown in Figure 1, boards (a), (b) and (c) correspond to items 1, 2 and 3, respectively.

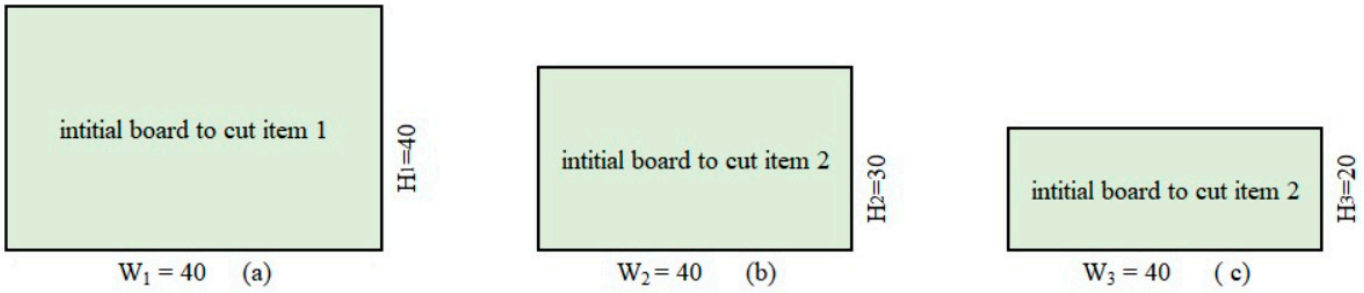


Figure 1. Initial boards in first phase of cutting items of Example 1.

Then, we proceed by cutting board (a) to produce item 1 with $(h_1, w_1) = (40, 20)$ and a residual board with $(H_4, W_4) = (40, 20)$. This generates the variable $x_{1,0} = 1$ and means that item 1 is cut from initial board 0. In addition to the $x_{1,0}$ variable, a parameter $a_{1,0,4}$ is generated and means that a new board of type 4 is created (see Figure 2). The same step is applied to obtain item 2 with $(h_2, w_2) = (30, 10)$. In Figure 1, a vertical cut of board (b) will generate $x_{2,0} = 1$ and a parameter $a_{2,0,5}$. Then, a residual board type 5 is generated with $(H_5, W_5) = (30, 30)$. The same process generates for item 3 a variable $x_{3,0}$ and a parameter $a_{3,0,6}$.

For all the generated residual boards, a step for fitting items into residual boards is applied. For example, consider board type 4 $(H_4, W_4) = (40, 20)$, which is generated in cutting phase 1. It is applicable to cut item 2 and to generate $x_{2,4}$ and parameter $a_{2,4,7}$ (see Figure 3). All of these generations related to cuts and residual boards could be implemented by using the algorithm in [21]. This algorithm is used by [11] to only generate right residual boards, while in [21] they generate both top and right residual boards.

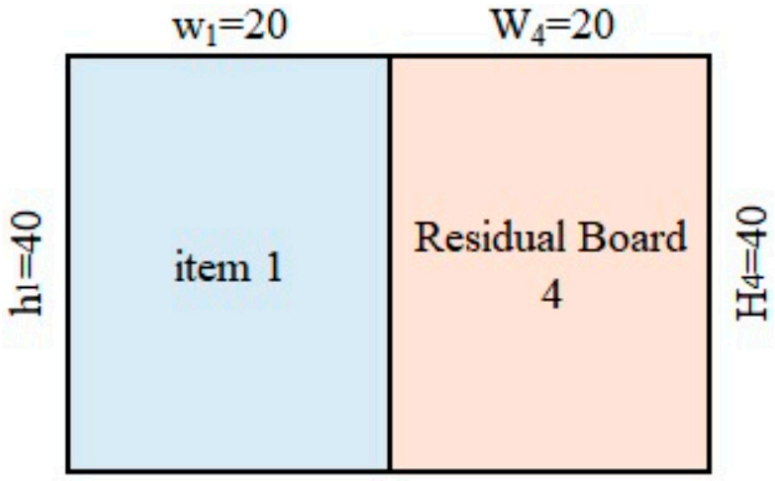


Figure 2. Deduction of item 1 and the residual board 4 of Example 1.

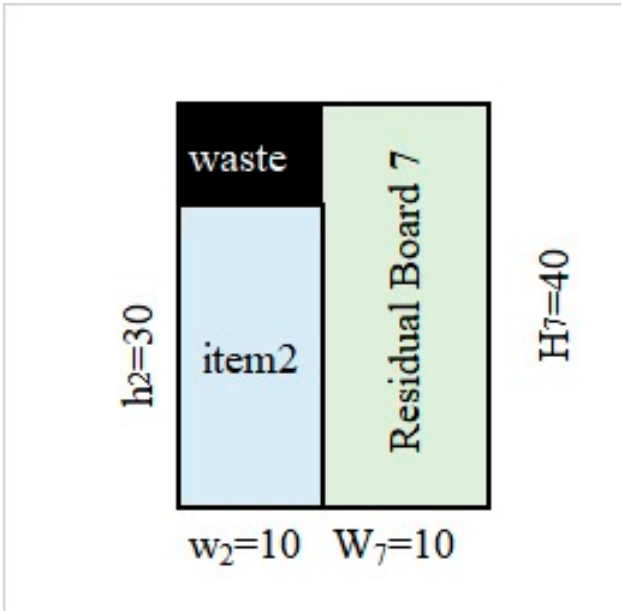


Figure 3. Using residual board 4 to cut item 2 of Example 1.

3.4. The Model of Mrad [28]

Mrad [28] extended the model introduced in Ref. [22] and used the graph technique in Ref. [23] to solve the 2D-SCP, where the items are cut through two stages. In the first stage, the strip is cut into shelves, denoted by π_k for each item k. In the second stage, each shelf is cut to produce the required items. Clearly, for each shelf π_k, a subset of items $S_k = \{i|h_i \leq h_k\}$ could be cut. Thus, the cutting of items in set S_k could be presented as a path in a specific graph as introduced in Ref. [23]. For each shelf π_k, $k \in \{1, \ldots, n\}$, a graph $G_k = (V_k, A_k)$ is constructed where V_k is the set of vertices and A_k is the set of arcs. Actually, any path in G_k, between nodes 0 and W, represents a sequence of no-overlapping items, which total width is less than or equal to W. This path amounts to a cutting pattern. Algorithm 1 is used to build the graphs G_k $(k = 1, \ldots, n)$. Interestingly, the following simple breaking-symmetry rules significantly reduce the size of the graphs:

1. The items are ordered according to a decreasing order of their widths.
2. In each shelf, any path must include an item with the same height as the shelf.
3. The number of occurrences of an item in a path cannot exceed its own demand.

Algorithm 1. Graph Construction Algorithm

Let M be a matrix with $W + 1$ rows and n_k columns. Where n_k is the number of items in the set S_k. $M[i][j]$ takes 0 if it is allowed to build an arc representing item j and starting from node i and 1 otherwise.

$M[i][j] \leftarrow 0 \; \forall \; i = 0, \ldots, W \; \forall \; j \in S_k$
$V_k \leftarrow \{0, W\}$
$for \; j \in S_k / \{k\} \; do$
$for \; i = 0, \ldots, w_k - 1$
$M[i][j] \leftarrow 1$
$end \; for$
$end \; for$
$for \; i \in S_k \; do$
$SetOfNewNodes \leftarrow \varnothing$
$for \; j \in V_k / \{W\} \; do$
$\alpha \leftarrow j$
$r \leftarrow 0$
$while \; (\alpha + w_i \leqslant W) \; and \; (r \leqslant d_i) \quad do$
$if \; M[\alpha][i] = 0 \; do$
$A_k \leftarrow A_k \cup (\alpha, \alpha + w_i)$
$M[\alpha][i]=1$
$SetOfNewNodes \leftarrow SetOfNewNodes \cup \{\alpha + w_i\}$
$end \; if$
$r \leftarrow r + 1$
$\alpha \leftarrow \alpha + w_i$
$end \; while$
$end \; for$
$V_k \leftarrow V_k \cup SetOfNewNodes$
$end \; for$
$for \; j \in V_k / \{W\} \; do$
$if \; \nexists (a, b) \in A_k \; such \; that \; a = j$
$A_k \leftarrow A_k \cup (j, W)$
$end \; if$
$end \; for$

Example 2. *Consider a strip with width W = 8 and infinite height H, and a set of four items I_1 (7, 5, 2), I_2 (6, 4, 1), I_3 (5, 3, 2) and I_4 (4, 2, 2) (where I (h, w, d) denotes an item I of height h, width w and demand d). Four shelves π_1, π_2, π_3 and π_4 are then required to be used in the second stage. Figure 4 depicts the corresponding four graphs G_1, G_2, G_3 and G_4, generated by Algorithm 1. In each graph, each non-dashed arc represents an associated item. Dashed arcs represent dummy items (waste material) and are added in order to respect the path between the source node and the target node.*

For a given shelf π_k, denote by H_k the set of different item heights in S_k. Let x^k_{abh} be an integer variable associated to each arc $(a,b) \in A_k$ that takes the number of items of width $(b-a)$ and height $h \in H_k$ placed at position a from the beginning of shelf π_k. This variable represents the flow on the arc (a, b) associated to the item of height h in the graph G_k. For example, for shelf π_3, we have $S_3 = \{3, 4\}$ and $H_3 = \{5, 4\}$. Consider the arc (5,7) in graph G_3, then the variable x^3_{685} (resp. x^3_{684}) denotes the number of items of width $8 - 6 = 2$ and height 5 (resp. 4) placed at position 6 from the beginning of shelf π_3.

Consider the decision variable z_k that represents the total flow on the graph corresponding to the π_k. The arc flow-based mathematical formulation of the 2D-SCP is the following:

$$Min \sum_{k=1}^{n} h_k z_k \tag{23}$$

$$\sum_{(a,b)\in A_k,\ h\in H_k} x^k_{abh} - \sum_{(b,c)\in A_k,\ h\in H_k} x^k_{bch} = \begin{cases} z_k & if\ b = 0 \\ 0 & if\ 1 \le b \le W-1; k = 1,..,n \\ -z_k & if\ b = W \end{cases} \quad (24)$$

$$\sum_{k,h_k\ge h_j} \sum_{(a,a+w_i)\in A^k} x^k_{a,a+w_j,h_j} \ge b_j \qquad j = 1,\ldots,n \quad (25)$$

$$x^k_{abh} \ge 0 \text{ and integer for all } (a,b) \in A_k, h \in H_k, \text{ and } k = 1,\ldots,n \quad (26)$$

$$z_k \ge 0 \qquad k = 1,\ldots,n \quad (27)$$

The objective function (23) consists of minimizing the total height used from the strip. Constraints (24) correspond to the flow conservation equality. That is, the value of the flow leaving each node $b \in \{1,2,\ldots,W-1\}$ in any graph G_k must be equal to the value of the flow entering this node. In addition, the flow leaving node 0 must be equal to the flow entering node W. Constraints (25) imply that the demand of each item should be satisfied, i.e., the total number of cut items of $j \in \{1,\ldots,n\}$ from any π_k ($k = 1,\ldots,n$) should be larger than or equal to the demand of this item. This mathematical formulation is strongly related to the structure of the proposed graphs and presents the strip-cutting problem as a minimum cost multi-flow problem with uncapacitated arcs and additional demand constraints. The model (23)–(27) is denoted by *Arc_Flow*.

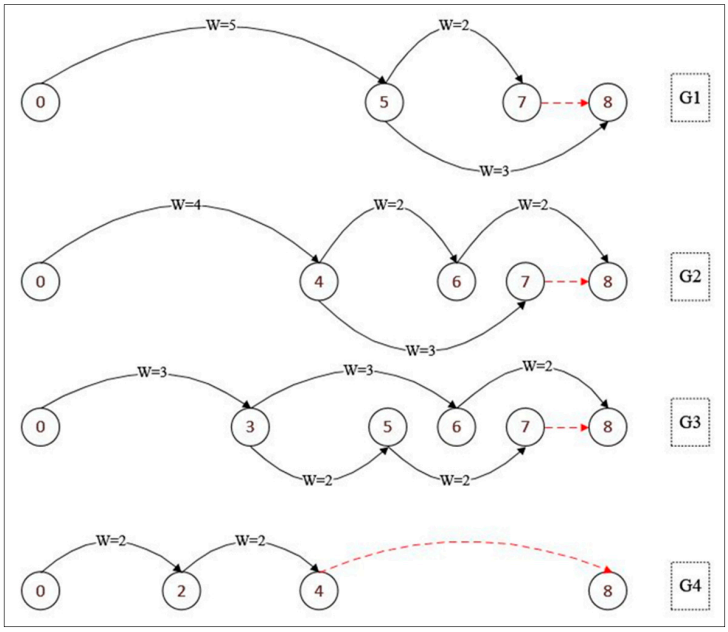


Figure 4. Graphs corresponding to Example 2.

4. Application of the Graph Compression Technique to the 2D-SCP

In this section, we introduce the graph compression technique that we applied to the arc flow-based model for the 2D-SCP. Our experimental results showed a substantial improvement of the performance of this model through applying this technique.

The graph compression method is introduced by Ref. [33] to decrease the resulting large size graphs related to the arc-flow formulation of the one-dimensional packing and cutting problems. It proved a substantial saving in the computational effort required to solve hard packing problems. Based on Ref. [33], the implementation of graph compression on HARD4 instances in Ref. [34] leads to 97% and 95% reduction in the number of nodes and arcs, respectively. Thus, the reduction of graph size can significantly hit the time needed by MIP solvers.

The graph compression requires at least two stages: Breaking Symmetry and Compression. In the first stage, the symmetry of the existing graph is broken. In fact, graphs

delivered by Algorithm 1 may include different paths that represent the same cutting pattern. Example 3 explains the occurrence of similar cutting patterns.

Example 3. *Consider a strip with width W = 9 and infinite height H, and a set of three items I_1 (5, 4, 1), I_2 (5, 3, 3) and I_3 (3, 2, 1) (using the same notations of Example 2). Figure 5 shows the graph corresponding to h = 5 (without dummy arcs). Clearly, the two paths 0-4-6-9 and 0-4-7-9 represent the same cutting pattern.*

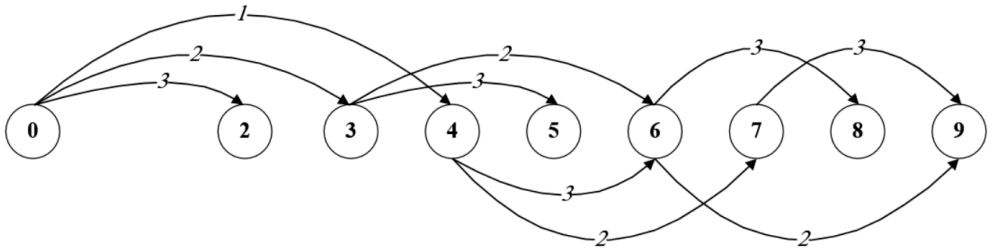


Figure 5. Graph corresponding to Example 3.

An easy way to break symmetry is to devote a level to each item. Then, the so-called level graph is built in a manner that only arcs related to a given item are added in its level. Each path in the level graph is devoted to a different pattern (see Example 4). Once the level graph is ready, the second step (compression step) can be applied. For that purpose, we introduce for each node u the so-called labeling function $\rho(u)$ which amounts to the length of the longest path between nodes 0 and u. Let s_{uv} denote the size of the item represented by the arc (u,v), T denote the node representing the width of the strip W in the last level of the level graph, and S denote the source node representing 0 in the first level. Then, the formula of $\rho(u)$ is given by the following equation:

$$\rho(u) = \begin{cases} 0 & if\ u = S, \\ W & if\ u = T, \\ min_{(u,v)}\{\rho(v) - s_{uv}\} & otherwise \end{cases} \tag{28}$$

Now, each node u in the level graph is translated to the node $\rho(u)$ in the compressed graph. Moreover, each arc (u,v) in the level graph will be replaced by ($\rho(u)$, $\rho(v)$) in the compressed graph. Interestingly, the nodes having the same label can be combined into one single node leading to a substantial reduction of the graph size. Indeed, if $\rho(u) = \rho(v)$ then the two nodes u and v will be represented by only one node in the compressed graph and the arc (u,v) will not be considered in the compressed graph. In addition, many different arcs in the level graph may be represented by only one arc in the compressed graph. Thus, the number of arcs and nodes will be potentially reduced.

It is worth noting that the recursive function (28) is applied on the nodes of the level graph after being sorted in the decreasing topological order. Once the compressed graph is obtained, a second round of compression can be applied. In this case, the reverse topological order of the compressed graph is considered and the following recursive function $\phi(u)$ is used:

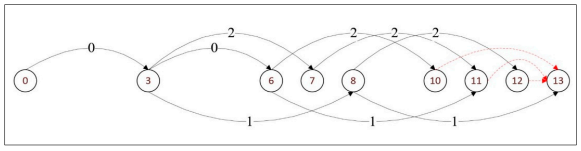
$$\phi(u) = \begin{cases} 0 & if\ u = 0 \\ max_{(v,u)}\{\phi(v) + s_{vu}\} & otherwise \end{cases} \tag{29}$$

Since the arc-flow formulation of the 2D-SCP requires n different graphs (one graph for each shelf), then the compression is applied on each graph independently. The resulting graphs with a reduced size will potentially require less variables and constraints in the formulation. Consequently, the solving time can be enhanced. Example 4 is illustrating the breaking symmetry and the main compression step on the graph of each shelf in the case of the 2D-SCP. For the sake of simplicity, we only considered the first compression round.

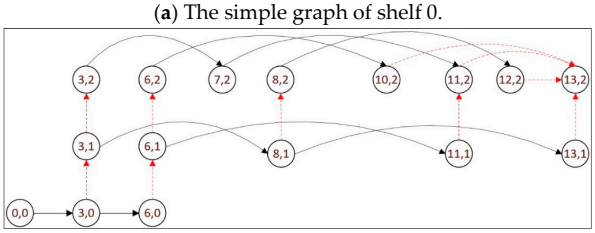
Example 4. *Consider an instance with a roll of width* $W = 13$*and three items to be cut as shown in Table 2. Algorithm 1 is applied to construct the graph related to each resulting strip. Figures 6–8 display the simple graph, the level graph and compressed graph for shelves 0, 1 and 2, respectively.*

Table 2. Data of Example 3.

Item i	h_i	w_i	d_i
0	3	7	2
1	5	6	2
2	4	5	2

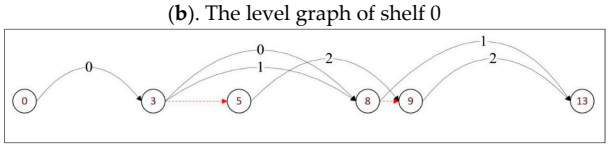


(**a**) The simple graph of shelf 0.



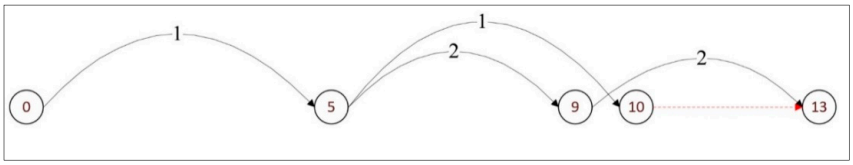
$\rho((13,2)) = 13$	$\rho((12,2)) = 13$	$\rho((11,2)) = 13$	$\rho((10,2)) = 13$	$\rho((13,1)) = 13$	$\rho((11,1)) = 13$	$\rho((8,2)) = 9$	$\rho((7,2)) = 9$
$\rho((6,2)) = 9$	$\rho((3,2)) = 5$	$\rho((8,1)) = 8$	$\rho((6,1)) = 8$	$\rho((3,1)) = 3$	$\rho((6,0)) = 8$	$\rho((3,0)) = 3$	$\rho((0,0)) = 0$

(**b**). The level graph of shelf 0

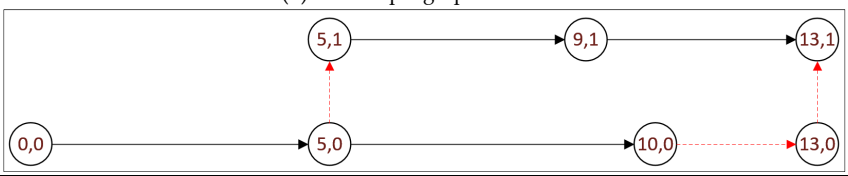


(**c**). The compressed graph of shelf 0

Figure 6. The three graphs of shelf 0.

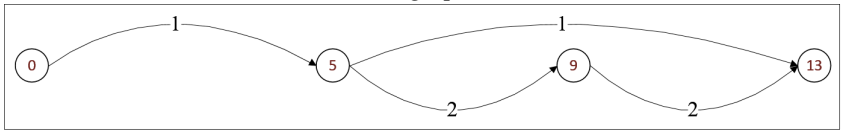


(**a**) The simple graph of shelf 1



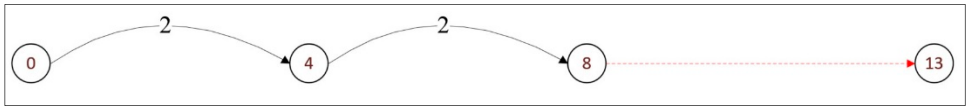
$\rho((13,1)) = 13$	$\rho((13,0)) = 13$	$\rho((9,1)) = 9$	$\rho((10,0)) = 13$	$\rho((5,1)) = 5$	$\rho((5,0)) = 5$	$\rho((0,0)) = 0$

(**b**) The level graph of shelf 1

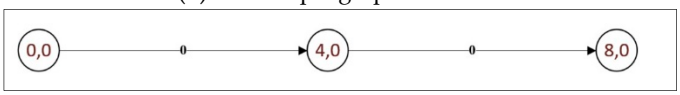


(**c**) The compressed graph of shelf 1

Figure 7. The three graphs of shelf 1.

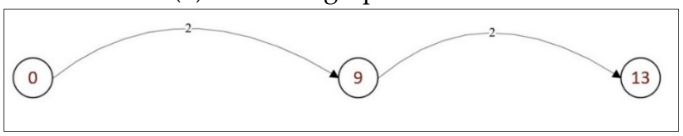


(**a**) The simple graph of shelf 2



$\rho((8,0)) = 13$	$\rho((4,0)) = 9$	$\rho((0,0)) = 0$

(**b**) The level graph of shelf 2



(**c**)The compressed graph of shelf 2

Figure 8. The three graphs of shelf 2.

5. Computational Experiments

5.1. The Benchmark Instances

We assessed the three models $M1ineq$, $M2ineq$ and $M3$ as well as the proposed arc-flow formulation with graph compression for the two-dimensional strip-cutting problem. We considered the five sets of benchmark instances, utilized by Ref. [11], to evaluate the performance of the mentioned models. We describe these sets as follows:

- Set 1 consists of 21 instances proposed by Ref. [35]. These are distributed in seven categories C1 . . . C7 and three types P1, P2, P3. All the item demands are equal to one. The number of items and the strip width for each problem are shown in Table 3.

Table 3. Data parameters of Set 1.

Categories	Number of Items			Strip Width
	P1	P2	P3	
C1	16	17	16	20
C2	25	25	25	40
C3	28	29	28	60
C4	49	49	49	60
C5	73	73	73	60
C6	97	97	97	80
C7	196	197	196	160

- Set 2 consists of 16 instances with unitary item demands, three of them (cgcut1, cgcut2, cgcut3) were used by Ref. [36] for the two-dimensional cutting stock problem. The remaining 13 instances (gcut1, . . . , gcut13) are proposed by Ref. [37] for the non-guillotine two-dimensional cutting problem.
- Set 3 consists of 500 instances that are divided into 10 classes. Each class includes 50 instances where for each value of $n \in \{20, 40, 60, 80, 100\}$ there are 10 generated instances. Berkey and Wang [16] introduced the first six classes, while the remaining four classes were presented in Ref. [38]. The structure of instances and values of w_i and h_i are uniformly distributed in the listed intervals as shown in Table 4. All the item demands are equal to one.

Table 4. Data parameters of Set 3.

Class	W	w_i	h_i
1	10	[1, 10]	[1, 10]
2	30	[1, 10]	[1, 10]
3	40	[1, 35]	[1, 35]
4	100	[1, 35]	[1, 35]
5	100	[1, 100]	[1, 100]
6	300	[1, 100]	[1, 100]
7	100	$[\frac{2W}{3}, W]$	$[1, \frac{H}{2}]$
8	100	$[1, \frac{W}{2}]$	$[\frac{2H}{3}, H]$
9	100	$[\frac{w}{2}, W]$	$[\frac{H}{2}, H]$
10	100	$[1, \frac{W}{2}]$	$[1, \frac{H}{2}]$

- Set 4 contains 20 instances (ATP30, . . . , ATP49) that are used by Ref. [28].
- Set 5 includes 43 instances from real-world settings presented in Refs. [21,22].

All the models described in this paper have been coded using Cplex Concert technology under C++ environment and solved using IBM ILog Cplex 12.9 solver. All computational experiments have been carried out on a desktop with Intel (R) Core (TM) i7 4930 K CPU 3.4 GHz processor with 32 GB of memory under Windows environment. A time limit of 3600 s has been set for all the models.

5.2. Impact of the Graph Compression

The impact of graph compression on the five sets of instances has been assessed according to the following criteria:

- ***Size reduction***: it reflects the impact of the graph compression on the number of variables of the generated mathematical model. It is computed as 100 $(n_1 - n_2)/n_1$, where n_1 and n_2 represent the number of variables corresponding to the non-compressed and compressed graphs, respectively.
- ***Time ratio***: it represents the ratio of the CPU time of the non-compressed model over the one of the compressed model. It indicates the impact of the graph compression on increasing/decreasing the running time.
- ***Gap improvement***: it represents the percentage gap improvement for unsolved instances. The gap is equal to 100 $(UB - LB)/UB$, where UB and LB denote the best found upper and lower bounds, respectively. The gap improvement is computed as 100 $(gap_1 - gap_2)/gap_1$, where gap_1 and gap_2 represent the gaps obtained by the non-compressed and compressed models, respectively.

Table 5 shows the results of the graph compression impact over the five sets of instances according to the three mentioned criteria. From this table, we can see that the most significant impact is observed for Set 5 in terms of model size reduction and time ratio. Indeed, the compressed model was, on average, 60.32% smaller and 2.11 faster than the non-compressed one. Sets 1 and 3 are the least impacted ones with respect to these criteria. On the other hand, the graph compression substantially reduced the gap for the unsolved instances where some of them have been solved to optimality in Sets 3 and 4.

Table 5. Impact of graph compression.

	Set 1	Set 2	Set 3	Set 4	Set 5
Average size reduction	5.22%	34.70%	12.27%	30.96%	60.32%
Maximum size reduction	23.44%	58.17%	54.22%	59.55%	81.14%
Average time ratio	0.95	1.79	1.32	1.85	2.11
Maximum time ratio	2.12	5.79	17.53	9.85	17.27
Average gap improvement	20.53%	7.86%	7.36%	31.12%	-
Maximum gap improvement	56.09%	7.86%	100.00%	100.00%	-

5.3. Comparison with the State-of-the-Art Mathematical Models

In this section, we present a comparison of the performance of the presented mathematical models in terms of percentage of solved instances (i.e., solved to optimality within the time limit of 3600 s), average computation time, average gap of unsolved instances, and performance of the linear relaxation. The latter criterion is assessed by computing the percentage of times the linear relaxation of each model equals the maximum value over all the linear relaxations of the four ones.

Tables 6–9 depict the results of the four models in terms of these criteria for each of the five sets of instances. From these tables, we observe that the compressed model provides the best results in all criteria for Sets 4 and 5 (non-unitary item demands). Moreover, it yields the second best results for Sets 1–3 (unitary item demands) in terms of percentage of solved instances and average computation time. In particular, the average computation time of our model is drastically better than those of the state-of-the-art models in Set 5. Indeed, it is 41.24 times faster than the second fastest model. Moreover, Table 9 shows that the linear relaxation of the compressed model outperforms the three other ones in all of the five sets.

Table 6. Percentage of solved instances.

	Set 1	Set 2	Set 3	Set 4	Set 5
Compressed Model	57.14%	93.75%	74.20%	20.00%	100.00%
M1ineq	71.43%	93.75%	85.60%	0.00%	76.74%
M2ineq	42.86%	62.50%	21.20%	0.00%	27.91%
M3	42.86%	93.75%	63.40%	0.00%	72.09%

Table 7. Average computation time.

	Set 1	Set 2	Set 3	Set 4	Set 5
Compressed Model	1617.43	225.08	962.63	2981.18	24.72
M1ineq	1035.99	225.58	627.89	3600.00	1019.47
M2ineq	2059.22	1395.06	2842.01	3600.00	2597.23
M3	2085.10	225.78	1363.38	3600.00	1106.27

Table 8. Average gap for unsolved instances.

	Set 1	Set 2	Set 3	Set 4	Set 5
Compressed Model	11.15%	7.57%	5.79%	0.53%	0.00%
M1ineq	1.83%	2.00%	1.81%	1.70%	0.53%
M2ineq	9.83%	7.57%	9.59%	16.54%	35.01%
M3	20.48%	7.11%	7.33%	9.58%	0.21%

Table 9. Performance of the linear relaxation.

	Set 1	**Set 2**	**Set 3**	**Set 4**	**Set 5**
Compressed Model	100.00%	100.00%	100.00%	100.00%	86.05%
M1ineq	0.00%	6.25%	0.00%	0.00%	0.00%
M2ineq	0.00%	0.00%	0.00%	0.00%	0.00%
M3	0.00%	25.00%	3.40%	0.00%	46.51%

From Tables 6 and 7, it can be observed that Set 4 is by far the hardest set to solve. Indeed, none of its instances has been solved by the models of the literature. The only model that makes it feasible to solve 20% of the instances is the proposed compressed model. Interestingly, by using this model we were able to substantially improve the upper/lower bounds of these hard benchmark instances, as will be detailed in the next section.

5.4. New Results for Open Benchmark Instances

In this section, we present new found upper and lower bounds for 24 unsolved benchmark instances of the two-dimensional strip cutting problem. Table 10 shows the details of these results where UB_{lit} (and, respectively, LB_{lit}) denotes the best obtained upper (and, respectively, lower) bound in the literature [11], and UB_{new} (and, respectively, LB_{new}) denotes the upper (and, respectively, lower) bound obtained by the proposed compressed model. The gap improvement is computed for each instance by 100 $(Dev_{lit} - Dev_{new})/Dev_{lit}$, where $Dev_{lit} = UB_{lit} - LB_{lit}$ and $Dev_{new} = UB_{new} - LB_{new}$. The asterisk symbol indicates that the new obtained upper/lower bound reaches the optimum.

Table 10. Improved benchmark results.

Instance	UB_{lit}	LB_{lit}	UB_{new}	LB_{new}	**Gap Improvement**
ATP30	1262	1242	1255 *	1255 *	100.00%
ATP31	13,068	12,866	12,964	12,893	64.85%
ATP32	1504	1491	1500	1496	69.23%
ATP33	11,802	11,742	11,771	11,770	98.33%
ATP34	2432	2421	2432	2426	45.45%
ATP35	4743	4707	4742 *	4742 *	100.00%
ATP36	1794	1778	1788	1783	68.75%
ATP37	9764	9683	9743	9722	74.07%
ATP38	3635	3611	3630 *	3630 *	100.00%
ATP39	5566	5481	5509	5506	96.47%
ATP40	2084	2037	2056	2041	68.09%
ATP41	4170	4123	4143	4137	87.23%
ATP42	4296	4227	4260	4231	57.97%
ATP43	11,235	11,001	11,118	11,030	62.39%
ATP44	4210	4150	4204	4165	35.00%
ATP45	4329	4270	4326	4293	44.07%
ATP46	5650	5534	5581	5549	72.41%
ATP47	6757	6698	6749	6708	30.51%
ATP48	1987	1950	1988	1958	21.62%

Table 10. *Cont.*

Instance	UB_{lit}	LB_{lit}	UB_{new}	LB_{new}	Gap Improvement
ATP49	2214	2195	2211 *	2211 *	100.00%
A11	94,890	94,598	94,873 *	94,873 *	100.00%
A14	139,959	139,938	139,959	139 959 *	100.00%
A21	57,348	57,340	57,348	57,348 *	100.00%
A35	34,554	34,525	34,554	34,554 *	100.00%
			Average gap improvement		69.82%

Interestingly, all of the upper/lower bounds of the hardest set of instances (namely Set 4) have been improved (except one upper bound for instance ATP 48). Moreover, optimality was reached for four of them. Moreover, the four open benchmark instances of Set 5 have all been solved to optimality. Furthermore, we can appreciate the dramatic improvement of the gap for the displayed instances since it equals 69.82%, on average, reaching more than 96% for some unsolved ones.

6. Conclusions

We addressed the two-dimensional strip-cutting problem and proposed an improved arc-flow-based mathematical model. It consists in extending the so-called graph compression approach formerly designed for the one-dimension case. The experimental results showed a significant impact of the graph compression technique on the efficiency of the proposed arc-flow model. Moreover, our computational comparison with three recent state-of-the-art mathematical models provided strong evidence of the good performance of our approach, especially for the instances with non-unitary item demands. New results for 24 unsolved benchmark instances were provided, with an average gap improvement of around 70%.

The present paper is actually a part of a Ph.D. thesis [39]. The next step would be to investigate the impact of the graph compression technique to more complex cutting and packing problems such as the two-dimensional cutting stock problem.

Author Contributions: Conceptualization, M.M. and T.G.A.; methodology, M.M. and T.G.A.; software, M.M., T.G.A. and A.B.; resources, A.S., M.A.L. and A.G.; data curation, A.G., A.S. and M.A.L.; writing—original draft preparation, T.G.A.; writing—review and editing, M.M. and A.G.; funding acquisition, M.M. and A.S. All authors have read and agreed to the published version of the manuscript.

Funding: This project was funded by the National Plan for Science, Technology and Innovation (MAARIFAH), King Abdulaziz City for Science and Technology, Kingdom of Saudi Arabia, Award Number (13-MAT1544-02).

Data Availability Statement: Data are available for all instances in this paper from corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Valério De Carvalho, J.V. LP models for bin packing and cutting stock problems. *Eur. J. Oper. Res.* **2002**, *141*, 253–273. [CrossRef]
2. Aktin, T.; Özdemir, R.G. An integrated approach to the one-dimensional cutting stock problem in coronary stent manufacturing. *Eur. J. Oper. Res.* **2009**, *196*, 737–743. [CrossRef]
3. Parreño, F.; Alonso, M.; Alvarez-Valdes, R. Solving a large cutting problem in the glass manufacturing industry. *Eur. J. Oper. Res.* **2020**, *287*, 378–388. [CrossRef]
4. Li, F.; Chen, Y.; Hu, X. Manufacturing-oriented silicon steel coil lengthwise cutting stock problem with useable leftover. *Eng. Comput.* **2021**, *39*, 477–492. [CrossRef]
5. Pierini, L.M.; Poldi, K.C. Lot Sizing and cutting stock problems in a paper production process. *Pesqui. Oper.* **2021**, *41*. [CrossRef]
6. Wattanasiriseth, P.; Krairit, A. An Application of Cutting-Stock Problem in Green Manufacturing: A Case Study of Wooden Pallet Industry. *IOP Conf. Ser. Mater. Sci. Eng.* **2019**, *530*, 012005. [CrossRef]

7. Varela, R.; Vela, M.D.C.R.; Puente, J.; Sierra-Sanchez, M.R.; González-Rodríguez, I. An effective solution for a real cutting stock problem in manufacturing plastic rolls. *Ann. Oper. Res.* **2008**, *166*, 125–146. [CrossRef]
8. Lemos, F.K.; Cherri, A.C.; de Araujo, S.A. The cutting stock problem with multiple manufacturing modes applied to a construction industry. *Int. J. Prod. Res.* **2020**, *59*, 1088–1106. [CrossRef]
9. Huang, Y.-H.; Lu, H.-C.; Wang, Y.-C.; Chang, Y.-F.; Gao, C.-K. A Global Method for a Two-Dimensional Cutting Stock Problem in the Manufacturing Industry. In *Application of Decision Science in Business and Management*; IntechOpen: London, UK, 2020. [CrossRef]
10. Lodi, A.; Martello, S.; Vigo, D. Heuristic and Metaheuristic Approaches for a Class of Two-Dimensional Bin Packing Problems. *INFORMS J. Comput.* **1999**, *11*, 345–357. [CrossRef]
11. Bezerra, V.M.R.; Leao, A.A.S.; Oliveira, J.F.; Santos, M.O. Models for the two-dimensional level strip packing problem—A review and a computational evaluation. *J. Oper. Res. Soc.* **2019**, *71*, 606–627. [CrossRef]
12. Gilmore, P.C.; Gomory, R.E. Multistage Cutting Stock Problems of Two and More Dimensions. *Oper. Res.* **1965**, *13*, 94–120. [CrossRef]
13. Hifi, M. An improvement of viswanathan and bagchi's exact algorithm for constrained two-dimensional cutting stock. *Comput. Oper. Res.* **1997**, *24*, 727–736. [CrossRef]
14. Hifi, M. Exact algorithms for the guillotine strip cutting/packing problem. *Comput. Oper. Res.* **1998**, *25*, 925–940. [CrossRef]
15. Lodi, A.; Martello, S.; Vigo, D. Models and Bounds for Two-Dimensional Level Packing Problems. *J. Comb. Optim.* **2004**, *8*, 363–379. [CrossRef]
16. Berkey, J.O.; Wang, P.Y. Two-Dimensional Finite Bin-Packing Algorithms. *J. Oper. Res. Soc.* **1987**, *38*, 423. [CrossRef]
17. Belov, G.; Scheithauer, G. A branch-and-cut-and-price algorithm for one-dimensional stock cutting and two-dimensional two-stage cutting. *Eur. J. Oper. Res.* **2006**, *171*, 85–106. [CrossRef]
18. Pisinger, D.; Sigurd, M. Using Decomposition Techniques and Constraint Programming for Solving the Two-Dimensional Bin-Packing Problem. *INFORMS J. Comput.* **2007**, *19*, 36–51. [CrossRef]
19. Bekrar, A.; Kacem, I. An Exact Method for the 2D Guillotine Strip Packing Problem. *Adv. Oper. Res.* **2009**, *2009*, 1–20. [CrossRef]
20. Dyckhoff, H. A New Linear Programming Approach to the Cutting Stock Problem. *Oper. Res.* **1981**, *29*, 1092–1104. [CrossRef]
21. Silva, E.; Alvelos, F.; Valério de Carvalho, J.M.V. An integer programming model for two- and three-stage two-dimensional cutting stock problems. *Eur. J. Oper. Res.* **2010**, *205*, 699–708. [CrossRef]
22. Macedo, R.; Alves, C.; de Carvalho, J.V. Arc-flow model for the two-dimensional guillotine cutting stock problem. *Comput. Oper. Res.* **2010**, *37*, 991–1001. [CrossRef]
23. Carvalho, J.V. Exact Solution of Cutting Stock Problems Using Column Generation and Branch-and-Bound. *Int. Trans. Oper. Res.* **1998**, *5*, 35–44. [CrossRef]
24. Mrad, M.; Meftahi, I.; Haouari, M. A branch-and-price algorithm for the two-stage guillotine cutting stock problem. *J. Oper. Res. Soc.* **2013**, *64*, 629–637. [CrossRef]
25. Rinaldi, F.; Franz, A. A two-dimensional strip cutting problem with sequencing constraint. *Eur. J. Oper. Res.* **2007**, *183*, 1371–1384. [CrossRef]
26. Cintra, G.; Miyazawa, F.; Wakabayashi, Y.; Xavier, E. Algorithms for two-dimensional cutting stock and strip packing problems using dynamic programming and column generation. *Eur. J. Oper. Res.* **2008**, *191*, 61–85. [CrossRef]
27. Bettinelli, A.; Ceselli, A.; Righini, G. A branch-and-price algorithm for the two-dimensional level strip packing problem. *4OR* **2007**, *6*, 361–374. [CrossRef]
28. Mrad, M. An arc flow-based optimization approach for the two-stage guillotine strip cutting problem. *J. Oper. Res. Soc.* **2015**, *66*, 1850–1859. [CrossRef]
29. Zhu, K.; Ji, N.; Li, X.D. Hybrid Heuristic Algorithm Based on Improved Rules & Reinforcement Learning for 2D Strip Packing Problem. *IEEE Access* **2020**, *8*, 226784–226796. [CrossRef]
30. Iori, M.; de Lima, V.L.; Martello, S.; Miyazawa, F.K.; Monaci, M. Exact solution techniques for two-dimensional cutting and packing. *Eur. J. Oper. Res.* **2020**, *289*, 399–415. [CrossRef]
31. Furini, F.; Malaguti, E.; Durán, R.M.; Persiani, A.; Toth, P. A column generation heuristic for the two-dimensional two-staged guillotine cutting stock problem with multiple stock size. *Eur. J. Oper. Res.* **2012**, *218*, 251–260. [CrossRef]
32. Lodi, A.; Monaci, M. Integer linear programming models for 2-staged two-dimensional Knapsack problems. *Math. Program.* **2003**, *94*, 257–278. [CrossRef]
33. Brandão, F.; Pedroso, J.P. Bin packing and related problems: General arc-flow formulation with graph compression. *Comput. Oper. Res.* **2016**, *69*, 56–67. [CrossRef]
34. Scholl, A.; Klein, R.; Jürgens, C. Bison: A fast hybrid procedure for exactly solving the one-dimensional bin packing problem. Comput. *Oper. Res.* **1997**, *24*, 627–645. [CrossRef]
35. Hopper, E.; Turton, B. An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. *Eur. J. Oper. Res.* **2001**, *128*, 34–57. [CrossRef]
36. Christofides, N.; Whitlock, C. An Algorithm for Two-Dimensional Cutting Problems. *Oper. Res.* **1977**, *25*, 30–44. [CrossRef]
37. Beasley, J.E. An Exact Two-Dimensional Non-Guillotine Cutting Tree Search Procedure. *Oper. Res.* **1985**, *33*, 49–64. [CrossRef]

38. Martello, S.; Vigo, D. Exact Solution of the Two-Dimensional Finite Bin Packing Problem. *Manag. Sci.* **1998**, *44*, 388–399. [CrossRef]
39. Ali, T. Impact of graph compression on the two stage cutting stock problems. Ph.D. Thesis, Department of Industrial Engineering, College of Engineering, King Saud University, Riyadh, Saudi Arabia, 2023.