The Egyptian International Journal of Engineering Sciences and Technology

https://eijest.journals.ekb.eg/

Vol. 47 (2024) 92–100

DOI: 10.21608/EIJEST.2023.233755.1245

ISSN: 1687-1006 p
ISSN: 2682-3640 e

Improved Best Fit Heuristics for Offline Not Oriented Two-dimensional Rectangular Strip Packing Problem

Asmaa Yehia [a*], Mostafa Ashour [a], Ahmed Abed [a,b], Raafat Elshaer [c]

[a] Zagazig University, Faculty of Engineering, Industrial Engineering Department, Zagazig, Egypt
[b] Industrial Engineering Department, College of Engineering, Prince Sattam bin Abdulaziz University, P.O.Box 11942, 16273, Saudi Arabia.
[c] Industrial Engineering Department, Faculty of Engineering, King Khalid University, Abha, P.O.Box 960, 61421, Suadi Arabia.

A R T I C L E I N F O

Article history:

Received 03 September 2023
Received in revised form 11 November 2023
Accepted 21 November 2023
Available online 21 November 2023

Keywords:

Packing Problems, 2D strip packing,
, Heuristics.

A B S T R A C T

Cutting and Packing problems have been recognized as a sub-discipline of operation research for more than half a century. Such problems are involved in a range of circumstances such as pallet loading, wood or glass cutting, strip packing, and positioning problems. The focus of this study is on 2D rectangular strip packing problems in which rectangle items are not oriented and guillotine constraint is not considered aiming to pack all these items without overlapping into an open-ended bin called a strip of fixed width while the objective is obtaining the minimum total height of cutting. The main aim of this paper is to introduce two proposed heuristics for solving the problem under study. The two heuristics are improved versions of the known Best Fit heuristics, Tower Checker Best Fit (TCBF) and Waste Priority Best Fit (WPBF). The performance of the proposed heuristics is tested using well-known benchmark problems. The computational analysis of the results shows that the proposed heuristics can generate near-optimal solutions to large-scale problems. In addition, the performance of the heuristics outperforms other well-known heuristics from literature by 3%.

1. Introduction:

The initial classification of 2D cutting and packing problems splits packing problems into two types: spatial and non-spatial. Typically, spatial bin packing problems are divided into two types: 2D bin and 2D strip packing problems. The category of 2D bin packing problems includes single and multiple bin packing problems and their variations (offline, online, almost online). On the other hand, in non-spatial concerns such as capital budgeting, projects represent little objects while share capital is a vast object that must be distributed or allocated to these projects. Projects include things like brand-new machinery or its replacement, new factories or goods, and other research and development initiatives.

In 1990, Dyckhoff [1] revealed the first cutting and packing typology, which is based on four traits, dimensionality, and assignment type, both huge and small items. In 2007, the previous typology was refined by Wascher [2] and several criteria were adjusted based on Dyckhoff's original concept for defining all forms of cutting and packing problems corporately (see Figure 1). To the best of our knowledge, the most common typology utilized by researchers is the Wascher typology. In the same year, Ntene [3] developed a more straightforward sub-typology made up of six fields: dimensionality, forms of the objects packed, an area where the items will be packed, level of information, the aim of packing, and packing constraints.

According to [2] Wäscher's typology, the specific topic under consideration is an offline rectangular two-dimensional orthogonal, open-dimension problem. This is referred to the "RF" (rotated, free cutting) subtype since all rectangular shapes may be rotated by 90 degrees without needing to be cut with a guillotine.

* Corresponding author. Tel.: +2-01117395090
E-mail address: asmaayahia2593@gmail.com

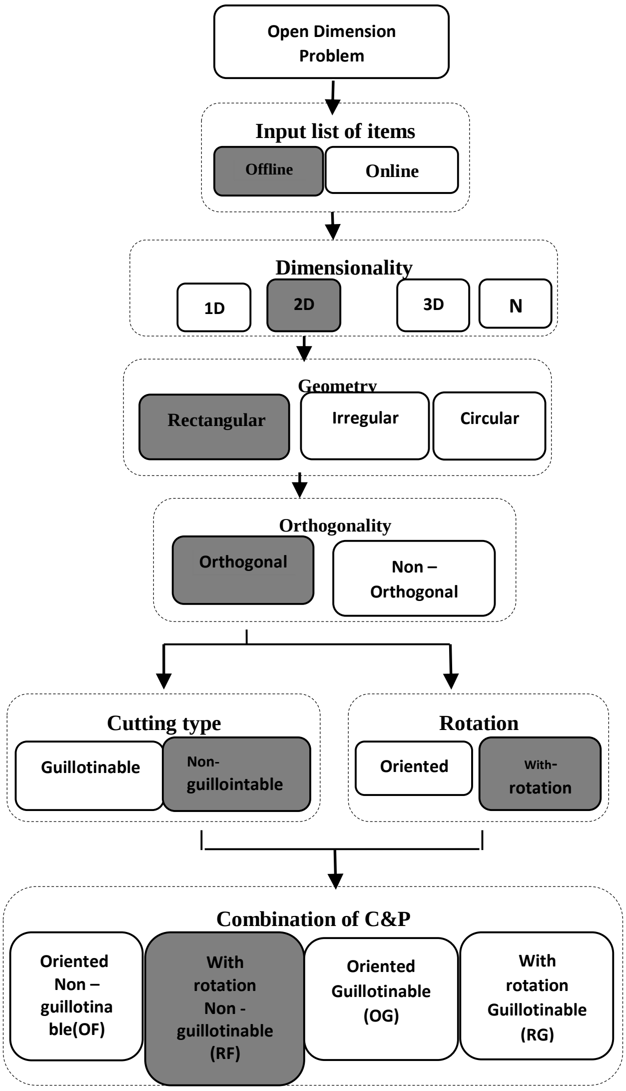


Figure 1. Wascher 's classification [2]

As well, according to the sub-typology of Ntene [3] , our research problem can be denoted in this way:

2D | R | SP | Off | MiS | 1,0,0,0

Where:

2D: means two dimensions,
R: regular shapes (rectangles) are to be packed.
SP: strip packing, off: level of information is offline, Mis: objective is minimizing strip total height,
1: rotation is allowed,
first 0: expresses no pre-imposed constraints before packing rectangles.
Second 0: indicates modifications in resources (width, height, time any resources needed to complete the packing mission) is not permitted.
last 0: shows guillotine packing does not exist. Guillotine packing means rectangles' sides cutting in the same order or (edge-to-edge cuts).

Most researchers considered heuristics algorithms used to solve this problem to be in three categories (Level, Shelf, Plane) algorithms.

- In level algorithms, the strip is divided into levels or sections. The level's height is determined by the height of the tallest rectangle packed in the level.
- On the other hand, Shelf algorithms compute the height of any new generating shelf in certain ways.
- On the contrary, in Plane algorithms, the strip is not divided. Rectangles can be cut or packed in any position inside the strip plane. The nature of a predefined problem can contribute to selecting the best technique used to solve it.

First of all, Bakert [4] are the first ones who specifically suggest the 2D strip packing problem in 1980. This problem is the packing of small shapes (often assumed rectangles) without overlapping inside an open-ended rectangle bin often referred to as a strip of fixed width and infinite height. The objective is to minimize the total height of the packing. In not oriented orthogonal packing, we can rotate rectangles during packing with 0 or 90 degrees, each side of all rectangles must be parallel to the two sides of the strip. To sum up the literature on the strip packing problem, the next part of this section concentrates on reviewing the literature on the 2D rectangular strip packing problem, especially offline plane strip packing problem heuristics.

Sleator [5] presented a technique for tackling this type of problem in 1980, where rotation was not permitted. His algorithm, which was 2.5 times optimum, was essentially simple. For solving oriented strip packing problems while taking into account the guillotine cut, Kenyon and Rémila [6] presented an asymptotic completely polynomial-time approximation method (AFPTAS). The algorithm performed poorly in many cases and was more theoretical than practical.

In 2004, Burke et al. [7] introduced the Fest-Fit Approach (BFA) for solving strip packing problems. This algorithm will be illustrated in detail in the next section. His proposed heuristic has outperformed the Bottom-Left and Bottom-Left-Fill algorithms.

The bidirectional best-fit heuristic (BBF) has been established by Aşik and Scan [8] in 2009. The performance of BBF was comparable to or better than most of the previously reported meta-heuristics for solving the non-guillotine-able rectangular strip packing problem. They specifically enhanced the results of the BFA heuristic, but unfavorably they achieved at a high computational cost.

By 2010 Imahori and Yagiura [9] introduced the worst-case approximation ratio of the BFA algorithm [7] and suggested an implementation of this algorithm. He employed data structures to quickly identify the best-fit rectangle for each stage in order to preserve the existing skyline while storing the remaining rectangles to be packed. In 2011, Wei et al. [10] Proposed a skyline heuristic for solving 2D rectangular strip packing. To improve their heuristic, they used the tabu search technique as a subroutine. In the same year, Leung et al [11] introduced a two-stage intelligent search approach (ISA) consisting of local search (LS) and simulated annealing (SA). Yang et al. [12] enhanced Leung et al.'s technique [11] by replacing the simulated annealing algorithm with a simple randomized approach (SRA) that does not require any parameters.

In 2013, Cui et al. [13] presented a heuristic solution for the 2D guillotine-able, non-oriented rectangular strip packing problem. Thirteen benchmark instance groups have been addressed using this heuristic. It has the power to raise the quality of all groups' solutions.

Da Silveira, Miyazawa and Xavier [14] in 2013, made research on Strip Packing problem with Unloading constraints (SPU). It was the first time to consider this type of problem. They proposed a GRASP heuristic and two approximation methods. Overall, their strategies worked effectively and produced excellent results.

For 2DSP and 3DSP, Wauters et al. [15] reported a shacking process. Based on the Bottom-Left-Fill (BLF) heuristic he developed an approach for 2DSP and used the Deepest-Bottom-Left-Fill (DBLF) approach for 3DSP. Verstichel et al. [16] improved the BFA heuristic [7] for non-oriented non-guillotine instances. The authors introduced three new item positioning policies and item orderings. In 2013 Ender Ozcan [17] modified the original BBF by considering combinations of pairs of rectangles. The performance of modifications, according to the authors, was comparable to other existing metaheuristics in that time but there was an increase in running time.

Wei et al. [18] developed a block-based layer-building technique for 2D guillotine strip packing. In addition, they declared that "Simple heuristics such as best-fit remains the most effective tool for handling large scale instances" but leave waste space at the end of the packing. They conclude that the block-based layer algorithm reduces search space and could get all the benefits of the two techniques combined.

A priority heuristic for the guillotine-not-oriented rectangular packing problem was reported by Zhang et al. [19]. It was the first time a priority technique had been used as a heuristic to choose an available item for a predefined place. By 2017, Wei et al. [20] improved Burke's heuristic (BFA) [7]. In their heuristic, instead of selecting the rectangle with the largest width, they used the fitness number to determine which rectangle would fit the gap the best. They used a random local search and evaluated several sequences to enhance the findings. In 2019, Wei et al. [21] proposed the First-Fit heuristic (FFH) for solving 2DSP with unloaded, non-guillotine, oriented cutting constraints.

Zhu et al. [22] introduced a hybrid heuristic approach that makes use of enhanced rules and reinforcement learning to solve the not oriented 2DSP without using guillotine cuts. The scoring methods based on the skyline algorithm are extended in this hybrid heuristic to minimize space waste. A Reinforcement Learning (RL) approach is utilized to improve local search ability and reduce the number of iterations, and the Deep Q-Network (DQN) is developed to retrieve the initial rectangles sequence.

After this review, we can conclude that there are variant characteristics of 2D strip packing problems and searching for this type of problem is still active and worthwhile to be searched. Although the best-fit Algorithm (BFA) heuristic, in general, is simple and generates good-quality packing in large-scale instances, it leaves so much wasted spaces. A few frequent constraints were considered by researchers like unloading, load balancing and multidrop constraints.

The main objective of this study is to propose two heuristics for solving the offline not oriented 2D strip packing problem. The two heuristics, TCBF and WPBF, are based on improving the steps of BFA heuristic. In the TCBF heuristic, post packing stage of the BFA is omitted where other steps are proposed for preventing towers in the solution. In the WPBF heuristic, the process of choosing the best-fit rectangle in the BFA heuristic is improved. For more detail, see section 3.

The outline of this paper is organized as follows: the problem statement is presented in section 2. In section 3 the proposed two heuristics are introduced in detail. The results are discussed and analyzed in section 4. The last section provides the conclusions of this work.

2. Problem Statement

Before declaring the mathematical model, let S be a rectangular strip with fixed width W and infinite height, and consider S embedded into a two-dimensional cartesian reference frame such that the left-bottom corner coincides with the origin. Let us assume there will be n rectangular pieces placed into the strip, with each piece i having a width w_i and height h_i. The goal is to fit all n pieces onto the strip without overlapping them in order to reduce the overall height of the pieces. Let (x_{i1}, y_{i1}) and (x_{i2}, y_{i2}) represent the coordinates of the left-bottom and right-top corners for each piece i that is inserted into the strip, respectively. Based on He's paper [23], the problem can be formulated mathematically as follows:

$$min \max_{i\in\{1,2,\dots,n\}} y_{i2}$$

Subject To:

$$(x_{i2} - x_{i1}, y_{i2} - y_{i1}) \epsilon \{(h_i, w_i), (w_i, h_i)\} \quad (1)$$

$$max(x_{i1} - x_{j2}, x_{j1} - x_{i2}, y_{i1} - y_{j2}, y_{j1} - y_{i2}) \geq 0 \quad (2)$$

$$0 \leq x_{ik} \leq W, y_{ik} \geq 0, k \in \{1,2\} \quad (3)$$

In constraints 1-3, i and j corresponds to $1, 2, \ldots, n$ and $i \neq j$. Constraint (1) requires that each piece be arranged orthogonally in the strip. Constraint (2) prevents any two rectangular pieces from overlapping. Additionally, all parts must fit entirely inside the strip according to constraint 3. The placements must adhere to the three constraints, and the goal is to arrange all the n pieces in the strip so that the overall height of the pieces is reduced.

3. Proposed heuristics

3.1 An Overview

The BFA heuristic proposed in [7], consists of three basic stages, the preprocessing stage, packing stage and post-packing stage.

1) In the preprocessing stage, before sorting the rectangles in order of descending width, each rectangle is first organized so that its width is bigger than its height. When two rectangles share a width, they are arranged in decreasing order by height.
2) The packing stage preserves a "skyline" of the smallest area that can accommodate a rectangle that is composed of connected line segments. The lowest line segment of the skyline that is currently available is taken into consideration at each stage of the packing phase, and the widest rectangle that will fit there is placed. Placing rectangle will be in one of three placement polices, Leftmost (LM), Tallest Neighbor (TN), and Smallest Neighbor (SN). Figure 2 visualizes the latest placement based on LM, TN, and SN policies. If there is more than one lowest segment sharing the same Y-axis height the priority will be to the one that has the smallest X-axis coordinate. Then, the latest rectangle placement is added to the skyline.
3) After the packing stage, the post-processing stage is conducted to remove any towers that have been positioned on the upper edge of the packing. Basically, Figure 3 illustrates it well. The result of packing using the best-fit technique is shown in Figure 3A. The tallest shape (Shape 4) is erased in post-processing to produce better results, and the skyline is accordingly reduced as seen in Figure 3B. Rotating the deleted item, an attempt is made to reposition it in the nest's lowest position. The lowest gap is raised to its lowest neighbor to create a larger gap because this form cannot fit, as seen in Figure 3C. The gap is raised once more because it still does not fit (see Figure 3D). The form can now fit in this gap; thus, it is positioned as indicated in Figure 3E. It is permitted if this new arrangement enhances the resolution (as in this case). It is approved if this alternative arrangement enhances the proposed solution (as in this case). The next-highest shape is used for the same operation (Shape 6). Shape 6 is positioned in its new location in Figure 3F. It is permitted if it improves the quality of the solution (as in this case). Since all prior attempts have resulted in higher-quality packing, the best shape is once more chosen. Shape 6 is once again the highest shape because its width is greater than its height, the process is stopped and the packing in Figure 3F is considered as the final solution. Burke, in [7], also stated that the main cause of adding post stage was that long and thin rectangles weren't inserted until the very end of a packing where towers are created. Towers indeed yield lower quality packaging. Thus, from this point, we introduced the following two proposed heuristics.

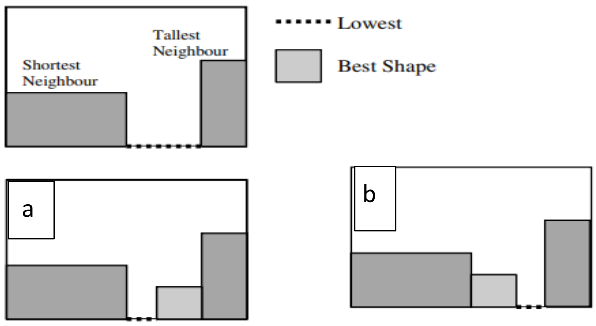


Figure 2. a) Placement next to TN. b) Placement next to SN and also represents LM [7].

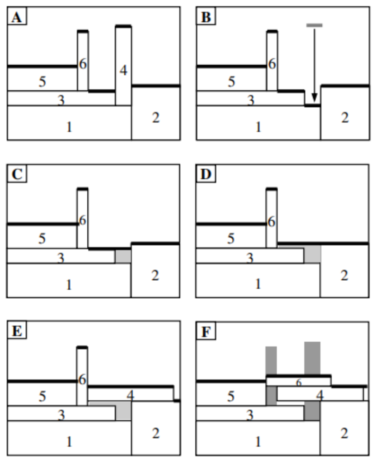


Figure 3. Processing of towers in the packing stage [7].

3.2 First proposed heuristic, TCBF

We can obviously note that in the BFA heuristic, all rectangles are packed in the first two stages, and then towers are repaired in stage 3(post-packing stage). Why don't we predict the chosen best-fit rectangle will cause the tower before placing it and prevent that from happening instead of correcting this situation later in the third stage? Thus, choosing the next best-fit rectangle that will not cause towers later will be a compromising alternative. As well as that we are still somehow not far from the concept of packing greater rectangles first. Adding to this, we shorten the whole packing process to two stages instead of three stages. This is accompanied by a decline in computation time. As well as that we are also hoping the quality of the solution is somehow enhanced. Aiming in that manner, we suggest the following steps to improve the quality of the solution based on one of Burke's predefined drawbacks. Basically, the detailed steps of the proposed TCBF algorithm are as shown below:

TCBF heuristic steps

1. Preprocessing stage:
   1.1. *Rotate all rectangles whose height is greater than their width.*
   1.2. *Sort rectangles descending based on width if there is a tie sort them descending based on height in list L, if there is more than one rectangle with the same width, we will sort these rectangles descending based on height.*
2. Packing stage:
   2.1. *Find the lowest gap which has the minimum Y-axis coordinates from the skyline. If the first rectangle of list L is to be packed, the lowest gap leftmost corner coordinates will be (0,0).*
   2.2. *Let* $C = \{r: r = i, w_i \leq W_l, i \in L\}$, $R = \{r: r = i, h_i \leq W_l, i \in L\}$ *if* $C \neq \emptyset \,\&R \neq \emptyset$. *Go to step4, then select the first rectangle from lists {C, R'} which has the highest area, if* $C = \emptyset \,\&\, R \neq \emptyset$, *go to step 4 then select the first rectangle from list R'. if* $C = \emptyset\& R = \emptyset$ *go to step3.*
   2.3. *Raise gap to the lowest neighbor and consider this gap* $W_l \times H_{min}$ *as a wasted area then go to step 2.2.*
   2.4. *Sort sets R descending based on height and put them in list R.' if there is more than one rectangle with the same height, we will sort these rectangles descending based on width.*
   2.5. *A very important hint before placing the chosen rectangle in the leftmost corner of the lowest gap, if the chosen rectangle from R' or C causes tower or (this rectangle will be the rectangle with the highest Y-axis coordinate skyline), as well as if we rotate it and placed in the worst case in the top of next higher skyline rectangle the maximum yield height will be less than before. Thus, we try to minimize the total height of cutting till now, by Not choosing this rectangle and choosing the next rectangle from the list not causing the tower or needing post packing stage.*
   2.6. *Then, put it in the leftmost corner of this gap (based on left most strategy) and update lists R' or C by deleting the last packed rectangle.*
   2.7. *Update skyline elements.*
   2.8. *Repeat the steps from 2.1 to 2.7 until no rectangle remains in list L.*

3.3 Second proposed heuristic, WPBF

As we mentioned before, Wei in [18] indicated that "Simple heuristics such as best-fit remains the most effective tool for handling large scale instances" but leaves waste space at the end of the packing. To get over this drawback, we aim to improve quality of the solution by reducing waste area by not placing always best-fit rectangle unless the remaining width of the chosen gap can fit at least one of the remaining unpacked rectangles. If there is no fitted rectangle in the remaining gap, we choose the rectangle with the smallest fitting factor in other words we inverse the rule of choosing the widest or best-fit rectangle, but post-packing stage in that case will be added. By the way, the fitting factor is computed by dividing the width of the rectangle by the width of the chosen gap. If there is a tie or more than one rectangle has the same fitting factor, the priority will be on the rectangle with the greatest height. When there are no left rectangles, in this case only the rectangle will be placed directly in the chosen gap. This heuristic consists of three stages (preprocessing, packing and post-packing). The following steps clearly illustrate the proposed heuristic.

WPBF heuristic steps

1. Preprocessing stage:
   1.1. *Rotate all rectangles whose height greater than width.*
   1.2. *Sort rectangles descending based on width if there is a tie sort them descending based on height in list L, if there is more than one rectangle with the same width, we will sort these rectangles descending based on height.*
2. Packing stage:
   2.1. *Find the lowest gap which has the minimum Y-axis coordinates from skyline. If the first rectangle of list L is to be packed, the lowest gap leftmost corner coordinates will be (0,0).*
   2.2. *Let* $C = \{r: r = i, w_i \leq W_l, i \in L\}$, $R = \{r: r = i, h_i \leq W_l, i \in L\}$ *if* $C \neq \emptyset \,\&R \neq \emptyset$. *Go to step4, then select the first rectangle from lists {C, R'} which has the highest area, if* $C = \emptyset \,\&\, R \neq \emptyset$, *go to step 4 then select the first rectangle from list R'. if* $C = \emptyset\& R = \emptyset$ *go to step3.*
   2.3. *Raise gap to the lowest neighbor and consider this gap* $W_l \times H_{min}$ *as a wasted area then go to step 2.2.*
   2.4. *Sort sets R descending based on height and put them in list R.' if there is more than one rectangle with the same height, we will sort these rectangles descending based on width.*
   2.5. *Very important hint before placing the chosen rectangle in leftmost corner of the lowest gap, if the chosen rectangle from R' or C causes tower or (this rectangle will be the rectangle with the highest Y-axis coordinate skyline), as well as if we rotate it and*

placed in the worst case in the top of next higher skyline rectangle the maximum yield height will be less than before.as well as after putting it in the selected gap will be remained space with no fit rectangle. Thus, we will inverse the rule by choosing the rectangle with the lowest fitness number in order to minimize wasted area as much as possible.

2.6. *Then, put it in the leftmost corner of this gap (based on left most strategy) and update lists R' or C by deleting the last packed rectangle.*

2.7. *Update skyline elements.*

2.8. *Repeat the steps from 2.1 to 2.7untill no rectangles remain in list L.*

3. *Post Packing stage:*

3.1. *If the rectangle with the highest Y-axis coordinate skyline, the height of it is larger than its width, it can be removed from this gap. if not, don't complete this stage, stop and return to the last packing solution.*

3.2. *Following this, rotating the removed rectangle, and placing it in the minimum possible gap. (Minimum possible gap means raising the current minimum gap to the shortest neighbor then update and search for minimum gap again then test this gap if its width fits the removed rectangle if yes put the removed rectangle and go to step 1, if not raise gap to the shortest neighbor, then repeat the test until the width of minimum current gap fits the chosen rectangle). (In the worst case the removed rectangle will be placed on the top of the next higher skyline rectangle). To our knowledge, the maximum yield height after this stage will be less than before.*

3.3. *Update skyline elements and repeat steps from step 3.1 again.*

4. Results and discussion

In this section, we seek to compare the proposed heuristic approach with more traditional approaches. Two datasets are used to assess the relative performance of the proposed two heuristics. The first dataset is provided in [24]. In this data set, there are seven different-sized categories with 21 problem sets of rectangle data each containing three problems with a comparable size and object dimension. The second datasets provided by Valenzuela and Wang [25] include rectangles with dimensions that are both very comparable (the "nice" data) and quite different (the "path" data). Data for each category ranges from 25 to 1,000 rectangles. For the sake of comparison, we use the same tested datasets that Burke used in his paper. Let us know that the best replacement policy of his three placement policies based on [21] data sets was a leftmost policy which we used here in our research. The PC used for all experiments had 1.35 GHz CPU and 4.00 GB RAM.

Table 1 shows the results of our two proposed heuristics, TCBF and WPBF, and BFA heuristic using the leftmost strategy (LM). The best solutions are stated in bold type. In Table 1, TCBF heuristic outperforms the BFA in five cases, C1P2, C2P3, C6P2, C7P1, and C7P2. On the other hand, WPBF heuristic outperforms it in only one case, C7P1. Adding to this, we tied in almost all the remaining data sets.

In Table 1, which depicts the percentage of deviation over the optimal solution for the same data sets, we achieved the lowest percentage deviation in four data sets. On the other hand, Burke` leftmost policy got the lowest percentage deviation over optimal in only one data set and in another data set we tied. To sum up, using the first proposal can improve the results by 3% than using the best-fit with the leftmost policy of Burke. We prefer before making the decision of using any one of two proposed heuristics, to use two proposed algorithms simultaneously and choose the best solutions of both of them which can improve results by 18% than using Burke (LM) best-fit separately.

Table 1. Comparison between our results and Burke with "Left most "policy results for Hopper & Turton benchmark problems.

Instance no.	Solution Height				% deviation		
	Opt.	BFA heuristic (LM)	TCBF heuristic	WPBF heuristic	BFA heuristic (LM)	TCBF heuristic	WPBF heuristic
C1P1	20	21	22	21	5.0	10.0	5.0
C1P2	20	22	21	22	10.0	**5.0**	10.0
C1P3	20	24	24	24	20.0	20.0	20.0
C2P1	15	17	18	17	13.3	20.0	13.3
C2P2	15	16	16	16	6.7	6.7	6.7
C2P3	15	18	17	18	20.0	**13.3**	20.0
C3P1	30	32	32	32	6.7	6.7	6.7
C3P2	30	34	34	34	13.3	13.3	13.3
C3P3	30	33	35	33	10.0	16.7	10.0
C4P1	60	63	63	63	5.0	5.0	5.0
C4P2	60	64	67	64	6.7	11.7	6.7
C4P3	60	62	62	62	3.3	3.3	3.3
C5P1	90	94	95	95	4.4	5.6	5.6
C5P2	90	93	96	93	3.3	6.7	3.3
C5P3	90	94	94	99	4.4	4.4	10.0
C6P1	120	124	124	124	3.3	3.3	3.3
C6P2	120	124	123	124	3.3	**2.5**	3.3
C6P3	120	124	124	124	3.3	3.3	3.3
C7P1	240	246	245	245	2.5	**2.1**	**2.1**
C7P2	240	246	245	246	2.5	**2.1**	2.5
C7P3	240	245	245	247	2.1	2.1	2.9
Average percentage deviation					7.1	7.8	7.4

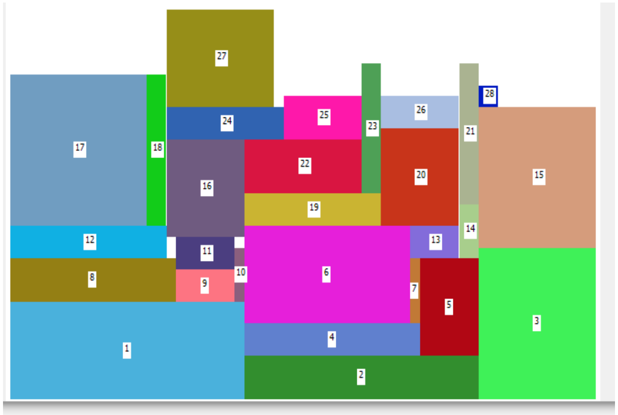


Figure 4. Representation of solution of problem C3P3 using first proposal algorithm.

As shown in Table 1 the worst cases for the proposed TCBF heuristic are the problems C2p1 and C3P3. To understand why the deviation is so high we will exhibit the solution by graph and then discuss the reasons. For problem C3p3 the rectangle number 27 is the rectangle of the highest skyline coordinates, (refer to Figure 4). In our point of view, the main cause of getting this biggest height is not involving this wide rectangle earlier in the packing stage. Due to the smallest lowest gaps (look like stair steps) which attract small rectangles. To avoid that happening again we can make the gap wider by closing these small gaps together earlier in the packing to acquire the widest rectangles or choose the widest gap not always the lowest gap. Thus, the results for sure will be better than our results and Burke's results. By the way, some researchers applied this idea before and indeed that enhanced the results. We also made some trials of closing the gaps in only cases of one unit difference of height of two neighbors of the lowest gap and found that almost all results are improved. This point is the most important one of our conclusions.

In the second proposed heuristic, WPBF, the worst case is only in problem C5p3. We can notice that the rectangles of convergent or similar dimensions are stuffed on top of each other which indeed means or inflects reducing wasted area, but the problem still was obtaining a medium or somehow wide rectangle at the final of packaging process. Figure 5 explains what we mean clearly.

Table 2 depicts the remaining results of benchmark problem data sets used by Burke in [7]. Figure 6 exhibits a better packing solution for benchmark problems using the suggested heuristics.

Table 2 Comparison between our results and Burke with “Left most “policy results for the rest of the benchmark problems that he used.

Instance no.	Solution Height				% deviation		
	Opt.	BFA heuristic (LM)	TCBF heuristic	WPBF heuristic	BFA heuristic (LM)	TCBF heuristic	WPBF heuristic
N1	40	48	**45**	**45**	20.00	12.50	12.50
N2	50	55	55	55	10.00	10.00	10.00
N3	50	**54**	55	**54**	8.00	10.00	8.00
N4	80	86	**83**	89	7.50	3.75	11.25
N5	100	105	**104**	106	5.00	4.00	6.00
N6	100	102	**102**	103	2.00	2.00	3.00
N7	100	110	113	**110**	10.00	13.00	10.00
N8	80	85	**84**	**84**	6.25	5.00	5.00
N9	150	163	163	163	8.67	8.67	8.67
N10	150	153	153	**152**	2.00	2.00	1.33
N11	150	153	154	**152**	2.00	2.67	1.33
N12	300	347	364	**305**	15.67	21.33	1.67
N13	960	986	**964**	**966**	2.71	0.42	0.63
path1	100	112.4	**107.5**	124.4	12.40	7.50	24.40
path2	100	130.3	**113.1**	**113.5**	30.30	13.10	13.50
path3	100	112.6	**111.2**	113	12.60	11.20	13.00
path4	100	106	**105.8**	108	6.00	5.80	8.00
path5	100	104.7	**103.2**	**103.8**	4.70	3.20	3.80
path6	100	103.3	**103.3**	103.6	3.30	3.30	3.60
nice1	100	**108**	111.5	117.5	8.00	11.50	17.50
nice2	100	111	**111**	117.6	11.00	11.00	17.60
nice3	100	109.5	**109.5**	109.6	9.50	9.50	9.60
nice4	100	**108.1**	108.3	**107.5**	8.10	8.30	7.50
nice5	100	**104.3**	105.3	104.4	4.30	5.30	4.40
nice6	100	104.3	104.6	**104.3**	4.30	4.60	4.30
babu1	375	400	**400**	**400**	6.67	6.67	6.67
Average percentage deviation					8.50	7.55	8.2

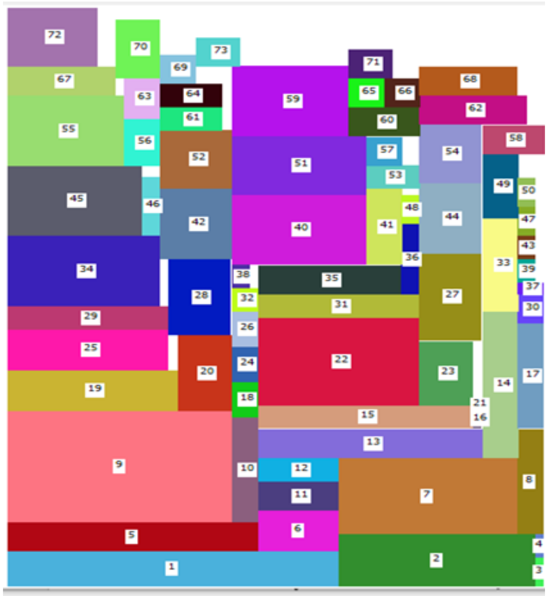
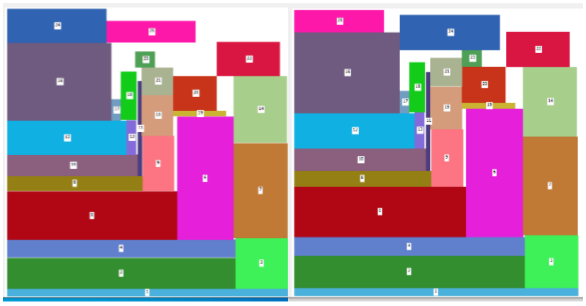
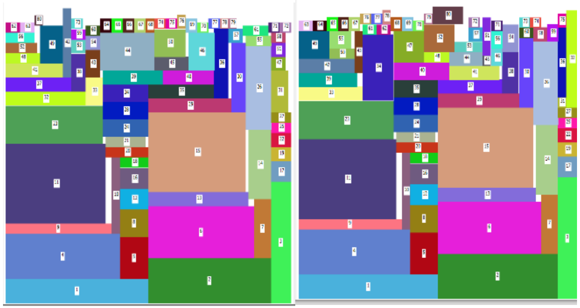


Figure 5. Representation of solution of problem C5P3 using second proposal algorithm.



Burke' solution for "path1" problem.

First proposal algorithm solution



Burke' solution for "N8" problem.

Second proposal algorithm solution for "N8" problem.

Figure 6. Packing' layout of our proposed heuristics and Burke's best-fit heuristic of problems Path1 and N8 from literature.

5. Conclusions:

Many other areas of operations research, such as memory allocation and multiprocessor scheduling, have a logical structure that is related to the problem presented here in this study. Consequently, the methods suggested in this paper could be used for these other domains where similar solution quality advancements could be made. In this research, we provided an efficient implementation of a new heuristic approach based on best-fit methodology. Our two new heuristics have provided better outcomes than the previous one using data from other researchers in the field of cutting and packing.

As we all know, Burke's best-fit heuristic outperformed the previous metaheuristic hybrids in terms of the quality of the packings created. His algorithm took less time to arrive at these nearly optimal solutions. However, based on our calculations, our suggested heuristic also took comparatively less time. Our two proposed techniques also work extremely well with data sets that have both different-sized and similar-sized rectangles (such as the problems represented by data from Valenzuela and Wang.

Eventually, our main contribution is to get rid of some predefined drawbacks of the best-fit heuristic that is very commonly used by almost of researchers. Additionally, we reached some points for enhancing the quality of solution such as considering the height of selected best-fit rectangle before placing it. As well as reducing waste by ensuring that there is a place for at least one rectangle instead of enclosing the next lowest yielded gap. Besides that, some drawbacks of our proposed heuristics algorithms were discussed in this paper. Adding to this, we proposed the way of enhancing our results more by enclosing gaps that are convergent in (Y-axis coordinate or height) to allow acquiring widest rectangles thus, these large rectangles will not remain at the end of packing like what happened in our solution of worst case of Hopper data set. The bottom line of our research is choosing or generating widest gaps rather than lowest gap is more effective and indeed improves our results more.

Acknowledgments:

The authors would like to thank Engineer **Sherif Salah Elden** a computer and systems engineer for assisting in coding C# program by his insightful suggestions to correct the mistakes. For more insight about the programs and results go to the following two links of three programs coded and the details of all results:
https://drive.google.com/drive/folders/1dCpEzLAlZ64rIE5PjoreQibMUJL82l04
https://drive.google.com/drive/folders/16lhvq4mOxtIYrylDQcHOx3oEXmfrBoPP

References:

[1] H. Dyckhoff, 'A typology of cutting and packing problems', Eur J Oper Res, vol. 44, no. 2, pp. 145–159, Jan. 1990, doi: 10.1016/0377-2217(90)90350-K.

[2] G. Wäscher, H. Haußner, and H. Schumann, 'An improved typology of cutting and packing problems', Eur J Oper Res, vol. 183, no. 3, pp. 1109–1130, Dec. 2007, doi: 10.1016/j.ejor.2005.12.047.

[3] N. Ntene, "An Algorithmic Approach to the 2D Oriented Strip Packing Problem", Ph.D Thesis, University of Stellenbosch, Dec. 2007.

[4] B. S. Bakert, E. G. Coffman, J. R. Arid, and R. L. Rivest, 'Orthogonal packings in two dimensions*' SIAM Journal on computing,vol. 9,no.4, pp. 846-855, 1980.

[5] D. Sleator, 'A 2.5 times optimal algorithm for packing in two dimensions'. Inf. Process. Lett. vol.10, no.1, pp. 37-40,1980.

[6] C. Kenyon and E. Rémila, 'Near-optimal solution to a two-dimensional cutting stock problem', Mathematics of Operations Research, vol. 25, no. 4, pp. 645–656, 2000, doi: 10.1287/moor.25.4.645.12118.

[7] E. K. Burke, G. Kendall, and G. Whitwell, 'A new placement heuristic for the orthogonal stock-cutting problem', Oper Res, vol. 52, no. 4, pp. 655–672, Jul. 2004, doi: 10.1287/opre.1040.0109.

[8] Ö. B. Aşik and E. Özcan, 'Bidirectional best-fit heuristic for orthogonal rectangular strip packing', Ann Oper Res, vol. 172, no. 1, pp. 405–427, Jan. 2009, doi: 10.1007/s10479-009-0642-0.

[9] S. Imahori and M. Yagiura, 'The best-fit heuristic for the rectangular strip packing problem: An efficient implementation and the worst-case approximation ratio', Comput Oper Res, vol. 37, no. 2, pp. 325–333, Feb. 2010, doi: 10.1016/j.cor.2009.05.008.

[10] L. Wei, W. C. Oon, W. Zhu, and A. Lim, 'A skyline heuristic for the 2D rectangular packing and strip packing problems', Eur J Oper Res, vol. 215, no. 2, pp. 337–346, Dec. 2011, doi: 10.1016/j.ejor.2011.06.022.

[11] S. C. H. Leung, D. Zhang, and K. M. Sim, 'A two-stage intelligent search algorithm for the two-dimensional strip packing problem', Eur J Oper Res, vol. 215, no. 1, pp. 57–69, Nov. 2011, doi: 10.1016/j.ejor.2011.06.002.

[12] Yang, S., Han, S., & Ye, W. 'A simple randomized algorithm for two-dimensional strip packing ' ,Computers and Operations Research, vol. 40,no.1, pp.1–8. ,2013. https://doi.org/10.1016/j.cor.2012.05.001

[13] Y. Cui, L. Yang, and Q. Chen, 'Heuristic for the rectangular strip packing problem with rotation of items', Comput Oper Res, vol. 40, no. 4, pp. 1094–1099, Apr. 2013, doi: 10.1016/j.cor.2012.11.020.

[14] J. L. M. Da Silveira, F. K. Miyazawa, and E. C. Xavier, 'Heuristics for the strip packing problem with unloading constraints', Comput Oper Res, vol. 40, no. 4, pp. 991–1003, Apr. 2013, doi: 10.1016/j.cor.2012.11.003.

[15] T. Wauters, J. Verstichel, and G. Vanden Berghe, 'An effective shaking procedure for 2D and 3D strip packing problems', Comput Oper Res, vol. 40, no. 11, pp. 2662–2669, 2013, doi: 10.1016/j.cor.2013.05.017.

[16] Verstichel, J., de Causmaecker, P., & Berghe, G. vanden. (2013). An improved best-fit heuristic for the orthogonal strip packing problem. International Transactions in Operational Research, 20(5), 711–730. https://doi.org/10.1111/itor.12030.

[17] E. Özcan, Z. Kai, and J. H. Drake, "Bidirectional best-fit heuristic considering compound placement for two dimensional orthogonal rectangular strip packing," Expert Syst. Appl., vol. 40, no. 10, pp. 4035–4043, 2013.

[18] L. Wei, T. Tian, W. Zhu, and A. Lim, 'A block-based layer building approach for the 2D guillotine strip packing problem', Eur J Oper Res, vol. 239, no. 1, pp. 58–69, Nov. 2014, doi: 10.1016/j.ejor.2014.04.020.

[19] D. Zhang, L. Shi, S. C. H. Leung, and T. Wu, 'A priority heuristic for the guillotine rectangular packing problem', Inf Process Lett, vol. 116, no. 1, pp. 15–21, Jan. 2016, doi: 10.1016/j.ipl.2015.08.008.

[20] L. Wei, Q. Hu, S. C. H. Leung, and N. Zhang, 'An improved skyline based heuristic for the 2D strip packing problem and its efficient implementation', Comput Oper Res, vol. 80, pp. 113–127, Apr. 2017, doi: 10.1016/j.cor.2016.11.024.

[21] L. Wei, Y. Wang, H. Cheng, and J. Huang, 'An open space based heuristic for the 2D strip packing problem with unloading constraints', Appl Math Model, vol. 70, pp. 67–81, Jun. 2019, doi: 10.1016/j.apm.2019.01.022.

[22] K. Zhu, N. H. Ji, and X. D. Li, 'Hybrid Heuristic Algorithm Based On Improved Rules & Reinforcement Learning For 2D Strip Packing Problem', IEEE Access, 2020, doi: 10.1109/ACCESS.2020.3045905.

[23] He, K., Jin, Y., & Huang, W. 'Heuristics for two-dimensional strip packing problem with 90 rotations'. Expert Systems with Applications,vol. 40,no.14,pp. 5542–5550.,2013. https://doi.org/10.1016/j.eswa.2013.04.005

[24] E. Hopper and B. C. H. Turton, 'An Empirical Investigation of Meta-heuristic and Heuristic Algorithms for a 2D Packing Problem', European Journal of Operational Research,vol. 128,no. 1,pp. 34-57, 2000.

[25] Valenzuela, C. L., P. Y. Wang, ' Heuristics for large strip packing problems with guillotine patterns: An empirical study', Proc. 4th Metaheuristics Internat. Conf., University of Porto, Porto, Portugal,pp.417–421, 2001.