Bidirectional Best-fit Heuristic Considering Compound Placement for Two Dimensional Orthogonal Rectangular Strip Packing

Ender Özcan, Zhang Kai, John H. Drake*

School of Computer Science, University of Nottingham Jubilee Campus, Wollaton Road, Nottingham, NG8 1BB, UK

*corresponding author

The two dimensional orthogonal rectangular strip packing problem is a common NP-hard optimisation problem whereby a set of rectangular shapes must be placed on a fixed width stock sheet with infinite length in such a way that wastage is minimised and material utilisation is maximised. The bidirectional best-fit heuristic is a deterministic approach which has previously been shown to outperform existing heuristic methods as well as many metaheuristics from the literature. Here, we propose a modification to the original bidirectional best-fit heuristic whereby combinations of pairs of rectangles are considered generating improved results over standard benchmark sets.

1. Introduction

Cutting and packing problems arise in many areas, particularly in manufacturing industries where a given stock material must be cut into a smaller set of shapes. The problem addressed in this paper is the NP-hard (Garey & Johnson, 1979) two dimensional orthogonal rectangular strip packing problem (SPP). In this problem a set of rectangular shapes must be arranged on a given sheet of fixed width and infinite height with the objective of minimising the highest point of any rectangle in the solution. Using the typology of Wäscher et al. (2007) the exact problem considered is a *two dimensional, open dimension* problem. As all rectangular shapes may be rotated by 90° and no guillotine cutting is required, this is referred to as the 'RF' (rotated, free cutting) subtype using the categorisation of Lodi et al. (1999). More information on cutting and packing problems can be found in a number of survey papers (Dyckhoff, 1990; Dowsland & Dowsland, 1992; Lodi et al., 2002; Wäscher et al., 2007)

Aşik & Özcan (2009) introduced the bidirectional best-fit heuristic for solving such problems. The bidirectional best-fit heuristic (BBF) is an elegant approach which considers both orientations of each rectangle yet to be packed when deciding which rectangle to place. Here we consider an extension to the original bidirectional best-fit heuristic which considers not only placing single rectangles but also placing different combinations of pairs of rectangles. We show that considering rectangles in pairs yields improved results over the original bidirectional best-fit heuristic and a number of metaheuristics from the literature.

Section 2 provides an overview of the two dimensional strip packing literature outlining previously used heuristic, metaheuristic and exact methods. Section 3 describes the original bidirectional best-fit heuristic in detail. Section 4 introduces a modified version of the bidirectional best-fit heuristic which considers compound shapes in the form of pairs of combined rectangles when searching for a shape to place. A number of new policies are described which are required in the modified heuristic. Section 5 defines the benchmarks which will be used and provides results and discussion of the application of the new method to these instances. Finally, Section 6 draws some conclusions based on our results.

2. The Two Dimensional Orthogonal Strip Packing Problem

2.1. Exact Methods

Although not the subject of our approach, there have been a large amount exact methods used to solve strip packing problems. Gilmore & Gomory (1961) developed a linear programming approach to solve very small strip packing problems to optimality. Christofides & Whitlock (1977) and Beasley (1985) used methods based on tree-search to solve the guillotine and non-guillotine variants of the strip packing problems respectively. The approach of Christofides & Whitlock (1977) was improved by Hifi & Zissimopoulos (1997) and further by Cung et al. (2000) however solving large instances was still impractical in a reasonable amount of time. Martello et al. (2003) and Lesh et al. (2004) both proposed variations of branch and bound techniques to solve small strip packing instances. More recently Kenmochi et al. (2009), Macedo et al. (2010), Alvarez-Valdes et al. (2009) and Boschetti & Montaletti (2010) all proposed exact methods for two-dimensional strip packing. Due to the difficulty in solving large problems, many heuristics and metaheuristics have been used in the literature to provide good solutions in an acceptable amount of time.

2.2. Heuristic Methods

Baker et al. (1980) introduced the bottom-up left-justified heuristic (BL). BL considers items sequentially, placing each rectangle at the top right corner of the stock sheet before moving it down to the lowest possible location and then as far left as it can without breaking the constraints of the size of the stock sheet or overlapping rectangles already placed. Another well recognised issue with this method is the tendency to create empty areas during the placement process leading to wasted space. Chazelle (1983) improved on the original BL heuristic with the bottom-left fill heuristic (BLF). BLF attempts to fill empty areas between rectangles already placed lower down the stock sheet before placing a rectangle at the top of the stock sheet. Although these methods are fast and effective at obtaining an approximation of a good packing, the quality of solution obtained is highly dependent on the order in which rectangles are considered. Hopper & Turton (2001) observed that a performance improvement of between 5% and 10% can be obtained when using BL and BLF by ordering rectangles by decreasing width or height. Lesh et al. (2005) exploited this idea by randomly perturbing the order in which rectangles are considered. Such approaches are still somewhat limited due to the existence of adversary instances for which no possible ordering will result in the optimal solution when using BL and BLF. Zhang et al. (2006) described a recursive heuristic based on the idea of divide and conquer with a worst case running time complexity of $O(n^3)$. Burke et al. (2004) proposed the best-fit heuristic (BF) to solve RF-SPPs. Unlike traditional traditional heuristics such as BL (Baker et al., 1980) and BLF (Chazelle, 1983) which greedily place a rectangle in the order in which they are processed, the best-fit heuristic selects the most appropriate rectangle to place at each step. The best-fit heuristic consists of a pre-processing phase, a packing phase and a post-processing phase. In the pre-processing phase each of the rectangles to be packed are arranged so that its width is greater than its height and then sorted in order of descending width. In the case that two rectangles have a shared width they are ordered by decreasing height. The packing phase maintains a '*skyline*' of the lowest available space at which a rectangle can be placed consisting of a series of line segments. At each stage of the packing phase, the lowest available line segment of the skyline is considered and widest rectangle which fits in the current lowest segment is placed. The skyline is then updated to include the most recently placed rectangle. Following the packing phase, a post-processing phase is carried out to eliminate any towers which are placed at the top of the packing. Imahori & Yagiura (2010) improved the efficiency of the best-fit heuristic from $O(n^2)$ to $O(n \log n)$, where n is the number of rectangles to be packed by using efficient data structures to maintain the current skyline, store remaining rectangles to be packed and efficiently search for the best-fit rectangle at each step. Aşik & Özcan (2009) introduced bidirectional best-fit heuristic (BBF). BBF builds on the ideas on BF by considering both horizontal and vertical gaps for the best placement location. This heuristic is discussed in more detail in Section 3.

2.3. Metaheuristic Methods

Jakobs (1996) and Liu & Teng (1999) used a genetic algorithm (GA) to evolve the ordering of rectangles to be packed using the BL heuristic. Dagli & Poshyanonda (1997) used two hybrid approached based on neural networks however the best results obtained suffer due to the excessive computational time required. Hopper & Turton (2001) investigated a number of metaheuristics to produce a placement ordering including Simulated Annealing and GAs and combined them with a number of piece placement strategies such as BL and BLF. Beltrán et al. (2004) combined a greedy randomized adaptive search procedure (GRASP) with variable neighbourhood search (VNS). '*Hyper-heuristics*' are high-level methodologies which operate on a search space of heuristics for solving complex problems (Burke et al., 2003, 2010a,b). Terashíma et al. used a GA (Terashima-Marín et al., 2005a) and a classifier system (Terashima-Marín et al., 2005b) as hyper-heuristics to evolve sequences of low-level strip packing heuristics. Bortfeldt (2006) implemented a GA (SPGAL) which operates directly on a search space of complete packing, rather than on an encoding of orderings such as those of Jakobs (1996) and Liu & Teng (1999). Burke et al. (2009) enhanced the the BF heuristic (Burke et al., 2004) with the hybrid simulated annealing and BLF heuristic of Hopper & Turton (2001). Alvarez-Valdes et al. (2009) proposed a reactive greedy randomized adaptive search procedure (reactive GRASP). This approach contains two phases, a constructive phase and improvement phase. The algorithm builds a solution based on a dynamically changing greedy function. Local search is then applied to improve the solution generated in the constructive phase. Belov et al. (2008) describe the SVC(SubKP) framework which iteratively applies a single constructive heuristic (SubKP) updating a number of parameters at each step. Burke et al. (2011) used a simple squeaky wheel optimisation methodology (SWP) for the oriented version of the strip packing problem where no rotations are allowed. Leung et al. (2011) introduced a two-stage stochastic 'intelligent search algorithm' (ISA) which again relies on a constructive phase and improvement phase based on simulated annealing resulting in some improvement on average over reactive GRASP (Alvarez-Valdes et al., 2009) and SVC(SubKP) (Belov et al., 2008). A simplified parameterless adaptation of this algorithm (SRA) is described by Yang et al. (2013). Wei et al. (2011) proposed iterative doubling binary search (IDBS) which when combined with tabu search outperformed many of the approaches from the literature.

3. The Bidirectional Best-fit Heuristic (BBF)

Aşik & Özcan (2009) proposed the bidirectional best-fit heuristic (BBF) as an improvement to the best-fit (BF) heuristic of Burke et al. (2004). The core idea of BF is to first find the lowest available portion of the skyline a rectangle can be placed, the lowest *horizontal gap*, before searching for a rectangle that best fits that space. This idea is extended in BBF by also considering placement of rectangles into the *vertical niche* formed between the left edge of the skyline and the expected best height of the current instance of the strip packing problem. The *expected best height* is a simple lower bound on the quality of solution that can be obtained for a given problem instance. The expected best height is calculated as:

$$expected\_Best\_Height = \sum_{\forall i, r_i \in R} \frac{area\_Of(r_i)}{W} \tag{1}$$

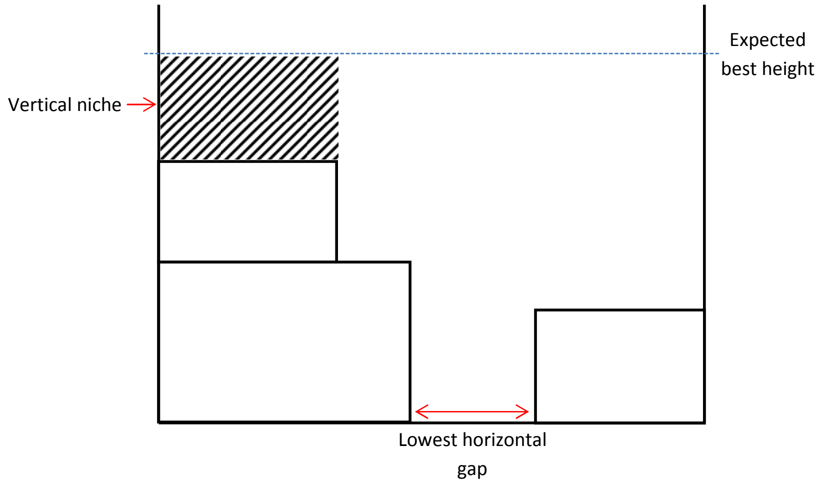
where R is the set of rectangles to be packed, r_i is the ith rectangle and W is the width of the stock sheet. Once a heuristic has obtained a solution which equals the expected best height it will return this solution. These concepts are shown in Figure 1. There are some adversary instances for which this bound is not appropriate whereby the one side of a single rectangle is greater than the width of the stock sheet and the expected best height calculated using Equation 1. The bound can be improved in this case by setting the expected best height to the length of the longest side of any rectangle in R.

The bidirectional best-fit heuristic consists of a pre-processing phase and a packing phase. The pre-processing phase is identical to the BF heuristic where all rectangles are oriented so that their width is greater than or equal to their height and are sorted in order of decreasing width and in the case of a common width by decreasing height. At each step a rectangular area (or *niche*) is considered and an appropriate rectangle is selected for packing in this niche. A number of placement policies are used during this process and are summarised in Table 1. Deciding which rectangle to place at each step of the packing phase consists of three stages; Exact Fit, Best-Fit and No Fit.

Table 1: Summary of the placement policies used in the BBF heuristic

Id.	Policy	Value	Brief explanation
1	Rectangle selection policy for exact fit into the vertical niche	Enabled	Placement into the vertical niche is considered during the exact fit phase
		Disabled	Only placement into horizontal niche is considered for an exact fit and Policy 3 is ignored
2	Rectangle selection policy for exact fit into the horizontal niche	TRE	The first rectangle that fits into the niche is chosen
		NRE	An exact fit into two different rectangular regions are considered consecutively, if not possible, the tallest rectangle that fits into the niche is chosen
3	Exact fit ordering policy	eHV	Possibility of an exact fit into the horizontal niche is sought first, if it is not possible, then the possibility of an exact fit into the vertical niche is considered
		eVH	Opposite of eHV
4	Rectangle selection policy for best-fit into the vertical niche	Enabled	Placement into the vertical niche is considered during the best-fit phase
		Disabled	Only placement into horizontal niche is considered for an best-fit, Policy 6 and Policy 7 are ignored
5	Horizontal (gap) best-fit policy	BP	Choose the rectangle that minimizes the remaining gap
		FP	Choose the first fitting tallest rectangle
6	Vertical (gap) best-fit policy	FH	Fix the height, choose the rectangle with this height and the largest width
		WR	Chooses the widest rectangle that fits into the vertical niche
7	Best-fit ordering policy	bHV	Possibility of a best-fit into the horizontal niche is thought first, if it is not possible, then the possibility of a best-fit into the vertical niche is considered
		bVH	Opposite of bHV
8	Placement policy (horizontal gap)	LM	Next to the left most neighbour
		TN	Next to the tallest neighbour
		SN	Next to the shortest neighbour

Figure 1: Illustration of *horizontal gap* and *vertical niche*



The policy definitions here vary slightly from the original definitions of Aşik & Özcan (2009). Policy 4 has been included explicitly to clarify that the consideration of vertical niche can be enabled or disabled independently in either the exact or best-fit stages.

3.1. Exact Fit Stage

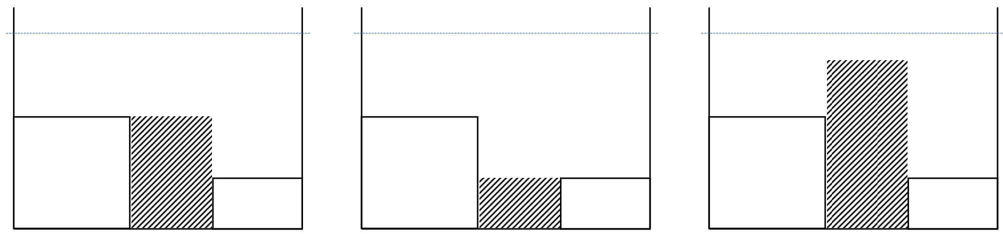
During this first stage the heuristic attempts to fill a gap entirely. As there are two potential areas for placement (lowest horizontal gap and the vertical niche) an *exact fit ordering policy* (Policy 3) is used to control the order in which gaps are considered for placement. If this policy is set to *eHV* a rectangle is sought to fit exactly into the horizontal gap is sought first before the vertical niche is considered, *eVH* provides the opposite considering the vertical niche first. Two *rectangle selection policies* (Policy 2) are used to control which rectangle is selected for placement in a given horizontal gap. *TRE* is the traditional rectangle selection method taken from the BF heuristic of Burke et al. (2004). *TRE* selects and places the first rectangle from the set of rectangles yet to be packed which fits exactly into the current gap. In BBF an additional rectangle selection policy *NRE* is used. *NRE* attempts to increase the size of the horizontal gaps on the skyline in order to find a placement for larger rectangles which are often difficult to pack towards the end of a run. *NRE* searches among the rectangles whose width is exactly that of the current horizontal gap looking for a rectangle with the same height as the tallest neighbour. If such a rectangle does not exist in the set of rectangles yet to be packed a rectangle is sought with the same height as the shortest neighbour, if no appropriate rectangle is found the traditional *TRE* rectangle selection policy is applied. These three potential placement areas are shown in Figure 2. It may be the case for a particular instance that it is not beneficial to consider a vertical niche. Policy 1 is included to dictate whether or not a vertical niche is considered at all. In the case that Policy 1 is disabled, Policy 3 can be ignored as only placement in horizontal gaps is possible.

Figure 2: Potential placements into the lowest horizontal gap during the exact fit phase

(a) Exact fit next to tallest neighbour

(b) Exact fit next to shortest neighbour

(c) Exact fit at an arbitrary level



3.2. Best Fit Stage

If no rectangle can be found to fit exactly into the lowest horizontal gap or the vertical niche this stage seeks to find the best fitting rectangle from the set of rectangles left to pack. Again a number of policies are required to manage the selection and placement of the next rectangle. Policy 4 is similar to Policy 1 and controls whether or not a vertical niche is considered in the best-fit stage. If Policy 4 is disabled, Policy 6 and Policy 7 are ignored. As with the exact fit stage an ordering policy (Policy 7) is used to determine the order in which gaps are considered for placement. bHV will attempt to find an appropriate rectangle to fit into the lowest horizontal gap before the vertical niche (and vice versa for bVH). Two rectangle selection policies *BP* and *FP* (Policy 5) are used to select a rectangle for placement in a given horizontal gap. *BP* is another policy taken from the original BF heuristic which finds the rectangle which minimises the gap remaining once it has been placed. *FP* places the first fitting rectangle from the sorted list of rectangles yet to be placed in the lowest horizontal gap. For the case of a vertical niche under consideration two rectangle selection policies are also used (Policy 6). *FH* fixes the height of potential rectangles to be placed to the height of the vertical niche, selecting the widest rectangle of this fixed height. *WR* selects the widest rectangle which will fit in the vertical niche. An additional placement policy (Policy 8) is required if a rectangle has been selected to fit into the lowest horizontal gap. Once the rectangle which best fits the lowest horizontal gap has been decided, a decision must be made as to which part of this gap it is placed in. *LM* (Figure 3(a)), *TN* (Figure 3(a)) and *SN* (Figure 3(b)) place a rectangle next to the leftmost, tallest and shortest neighbour of the current gap respectively.

3.3. No Fit Stage

If a suitable rectangle can not be found for placement in the first two stages the lowest horizontal gap is raised to be equal to the level of its shortest neighbour.

4. Modified Bidirectional Best-fit Heuristic

So far the bidirectional best-fit heuristic has been used to place a simple set of rectangles onto a fixed with strip of a given stock material. Here we propose a modification of the BBF heuristic which also considers compound polygons consisting of combinations

Figure 3: Potential placements into the lowest horizontal gap during the best-fit phase

(a) Next to tallest (leftmost) neighbour (b) Next to shortest neighbour

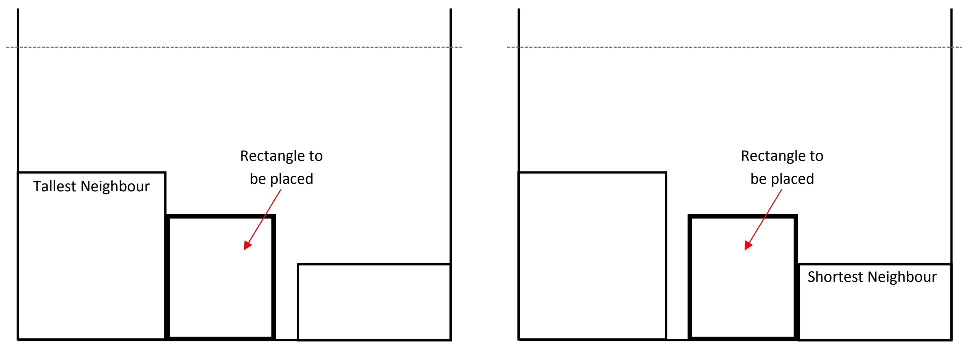
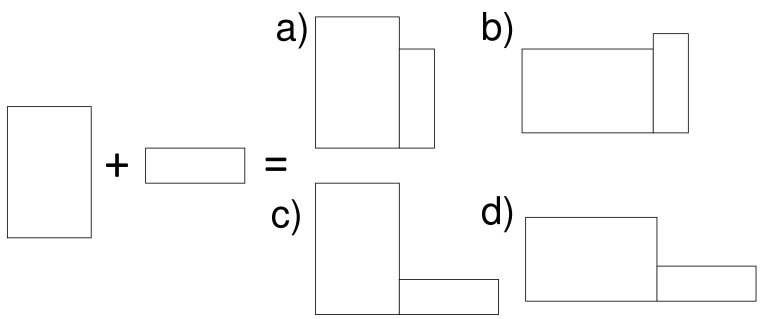


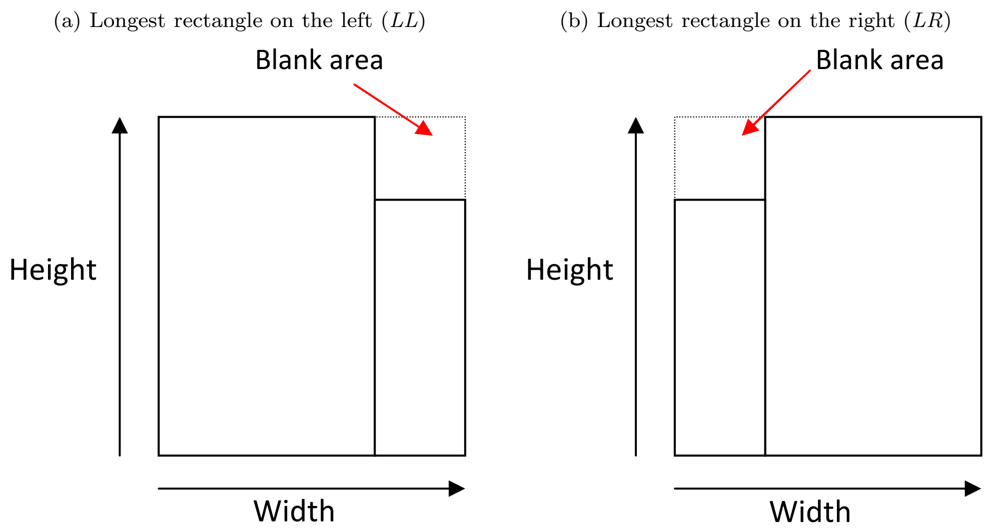
Figure 4: Possible combinations of two shapes X and Y



of two rectangles during the packing phase. A simple method to do this is to pair up two rectangles of equal length or height to form a larger rectangle. The drawback of this approach is that in the worst case where no any two rectangles have one side of equal length, the algorithm works exactly the same as the original bidirectional best-fit algorithm. During the pre-processing phase of the modified BBF, a second BST is created containing the set of combined shapes created from each pair of original rectangles in the set to be packed. For each pair of rectangles X and Y, four new shapes are produced as shown in Figure 4. In 4a X remains the same and Y is rotated by 90 degrees, in 4b both X and Y are rotated by 90 degrees, both rectangle X and rectangle Y remain the same in 4c and in 4d X is rotated by 90 degrees and Y remains the same. A notion of 'blank area' is also introduced to order combined shapes with equal width and height. Rectangles are again sorted in order of decreasing width and in the case of equal width in order of decreasing height. In the case that both the width and the height of the combined rectangles are equal they are ordered by increasing size of blank area.

In addition to the four possibilities outlined previously if rectangle Y is on the left hand side four more rectangles could be produced. This creates the need for an extra policy to be defined (Policy 9) to manage how rectangles are combined. *LL* places the rectangle with the longest side on the left hand side of the new shape whilst *LR* places

Figure 5: Possible rectangle order policies when combining shapes



the rectangle with the longest side on the right as shown in Figure 5.

A number of new policies must also be introduced into the packing phase to manage the selection and placement of the combined rectangles. In addition to the original options for Policy 2 *TREC*, *NREC*, *SBE* and *LBE* are defined. *TREC* is an extension the original *TRE* which selects and places the first rectangle which fits exactly into the current gap considering both single rectangles and combined rectangles. If both a single rectangle and a combined rectangle exist which fit exactly into the current gap the tallest of the two is placed. *NREC* first searches for a single rectangle which fits exactly into the current gap with the same height as the tallest neighbour before searching for combined rectangles meeting this criteria. If no shape is selected the same process is performed to find a rectangle which fits exactly into the current gap with the same height as the shortest neighbour. If a suitable shape is still not found *TREC* is applied. *SBE* first tries to find a single rectangle which fits exactly into the current gap before considering combined rectangles. If a placing a combined rectangle is necessary to fill the gap exactly the combined rectangle with the smallest blank area is chosen. *LBE* first seeks to place a combined rectangle with the largest blank area exactly into the gap, if this is not possible a single rectangle which fits exactly into the gap is sought.

Similarly for the best-fit stage of the packing phase a number of additional options are required for selecting a rectangle to place (Policy 5). *SBB* finds the rectangle which minimises the gap remaining once it has been placed considering both single rectangles and combined rectangles giving combined rectangles priority in the case of equal width. If more than one combined rectangle exists which minimise the current gap the combined rectangle with the smallest blank area is placed. *LBB* is similar to *LBE* and seeks to place the combined rectangle with the largest blank area which minimises the current horizontal gap before searching for a single rectangle which minimises the gap if this a combined rectangle does not exist. As all of the original policies are included in the

Table 2: Additional polices required when compound shapes are considered

Id.	Policy	Value	Brief explanation
2	Rectangle selection policy for exact fit into the horizontal niche	TREC	The first rectangle that fits into the niche is chosen considering both single rectangles and combined shapes
		NREC	An exact fit into two different rectangular regions are considered consecutively, single rectangles are considered before combined shapes. If this is not possible TREC is applied
		SBE	The first single rectangle that fits into the niche is chosen, if failed, the first combined rectangle that fits into the niche is chosen
		LBE	The combined rectangle with the largest blank area that fits into the niche is chosen, if failed, the first single rectangle that fits into the niche is chosen
4	Horizontal (gap) best-fit policy	SBB	Choose the shape which minimizes the remaining gap considering both single rectangles and combined shapes
		LBB	Choose the combined rectangle with largest blank area that minimizes the remaining gap, if none exist choose the single rectangle that minimizes the remaining gap
9	Compound shape rectangle order policy	LL	Longest rectangle on the left
		LR	Longest rectangle on the right

modified version of the BBF heuristic, it will always produce a solution at least as good as the original BBF heuristic. As this is a deterministic method the same solution will always be obtained from multiple runs. This is unlike many metaheuristic techniques from the literature which are measured in terms of average performance over a number of runs due to their stochastic nature. The new policies introduced for the modified BBF are summarised in Table 2.

5. Experimentation and Results

A number of datasets from the literature are used to test the modified BBF and compare it to both the original BBF heuristic and other methods from the literature and are summarised in Table 3. J1 and J2 are two small instances taken from Jakobs (1996) with known optimum heights. Ramesh Babu & Ramesh Babu (1999) introduced a single problem instance. The instances provided by Hopper & Turton (2001) are grouped into seven categories each with three instances. Each set of three instances share a common width and optimal height but sometimes vary slightly in the number of rectangles to pack. The Valenzuela & Wang (2001) dataset contains two different types of problem instance. The 'nice' instances consist of rectangles which are all of similar size whereas in the 'path' instances the size of one half of the rectangles differs vastly from the other half. Rectangle values in the 'nice' and 'path' problems are defined as floating point values so a scaling factor is used to convert these to integer values. The instances of Burke et al. (2004) (N1-N13) are a set of randomly generated problems of various sizes. All experiments were carried out on an Intel Core i7 3.20 GHz CPU with 16GB of RAM.

5.1. Comparison of the Bidirectional Best-fit Heuristic and Modified Bidirectional Best-fit Heuristic with existing heuristics

Table 4 compares the results of the original BBF and the modified BBF with packing heuristics from the literature when applied to the instances of Jakobs (1996), Ramesh Babu & Ramesh Babu (1999) and Hopper & Turton (2001). Included in this table are results for the Bottom Left and Bottom Left Fill heuristics considering rectangles in order of decreasing height (BL-DH and BLF-DH) as presented by Aşik & Özcan (2009) and the Best-Fit heuristic (BF) of Burke et al. (2004). Ayy results which are unavailable are marked '-'. As mentioned in Section 4 BBFM will always obtain a solution at least as good as the original BBF as all of the policy combinations included in the BBF are also in BBFM. BBF was shown by Aşik & Özcan (2009) to outperform all of the traditional packing heuristics. The modified BBF heuristic (BBFM) achieves an improvement over the original BBF in 17 of the 24 instances in these sets. Table 5 shows the results of the original BBF with the modified BBF when applied to the instances of Valenzuela & Wang (2001) and Burke et al. (2004). The modified BBF heuristic (BBFM) achieves an improvement over the original BBF in 16 of the 25 instances in these sets. It can be seen from Table 4 and Table 5 that the runtime for BBFM is longer than for the original BBF. Much of this is attributed to the larger search space of shapes to be considered. Another factor is the larger number of policy combinations to be tested. The original BBF only has to test 288 policy combinations whereas due to the extra policies needed for the modified BBF (BBFM) 6912 policy combinations must be tested. Note that this is a constant factor and is not affected by the size of instance currently being solved.

5.2. Comparison to previous metaheuristic approaches

Table 6 compares the results of BBFM to a number of other approaches from the literature over the dataset provided by Hopper & Turton (2001). These results are presented in terms of %-gap distance from the optimal solution calculated as:

$$100 * \frac{SolutionFound - OptimalSolution}{OptimalSolution} \tag{2}$$

The number of optimal solutions found by each technique are also included. The techniques from the literature are: Best-fit (BF) (Burke et al., 2004), Best-fit with simulated annealing (BF-SA) (Burke et al., 2009), squeaky wheel optimisation (SWP) (Burke et al., 2011), SVC(SubKP) (Belov et al., 2008), GRASP (Alvarez-Valdes et al., 2009), an 'intelligent search algorithm' (ISA) (Leung et al., 2011) and iterative doubling binary search (IDBS) (Wei et al., 2011). In the case that an approach is stochastic, the average performance reported is used to calculate %-gap to compare to our deterministic method.

Table 3: Summary of the datasets used in these experiments where n is the number of rectangles, *Opt* is the known optimal height and W is the width of the stock sheet

Source	Problem Name	n	*Opt*	W
Jakobs (1996)	J1	25	15	40
	J2	50	15	40
Ramesh Babu & Ramesh Babu (1999)	RB	50	375	1000
Hopper & Turton (2001)	C1	16-17	20	20
(3 instances per set)	C2	25	15	40
	C3	28-29	30	60
	C4	49	65	60
	C5	72-73	90	60
	C6	97	120	80
	C7	196-197	240	160
Valenzuela & Wang (2001)	nice0025	25	1000	1000
	nice0050	50	1000	1000
	nice0100	100	1000	1000
	nice0200	200	1000	1000
	nice0500	500	1000	1000
	nice1000	1000	1000	1000
	path0025	25	1000	1000
	path0050	50	1000	1000
	path0100	100	1000	1000
	path0200	200	1000	1000
	path0500	500	1000	1000
	path1000	1000	1000	1000
Burke et al. (2004)	N1	10	40	40
	N2	20	50	30
	N3	30	50	30
	N4	40	80	80
	N5	50	100	100
	N6	60	100	50
	N7	70	100	80
	N8	80	80	100
	N9	100	150	50
	N10	200	150	70
	N11	300	150	70
	N12	500	300	100
	N13	3152	960	640

Table 4: Comparison between BL-DH, BLF-DH, the Bidirectional Best-fit Heuristic (BBF) and Modified Bidirectional Best-fit Heuristic (BBFM) when applied to datasets of Jakobs (1996), Ramesh Babu & Ramesh Babu (1999) and Hopper & Turton (2001)

Instance	*Opt*	BL-DH	BLF-DH	BF	BBF	Time(s)	BBFM	Time(s)
J1	15	-	-	-	16	0.006	15	0.134
J2	15	-	-	-	16	0.017	15	0.541
RB	375	-	-	400	400	0.064	375	0.092
C1.1	20	23	22	21	21	0.003	20	0.001
C1.2	20	22	22	22	21	0.003	21	0.292
C1.3	20	21	21	24	21	0.003	21	0.242
C2.1	15	17	17	16	16	0.006	16	0.543
C2.2	15	26	26	16	17	0.006	15	0.014
C2.3	15	17	17	16	16	0.006	16	0.585
C3.1	30	33	33	32	32	0.007	30	0.148
C3.2	30	33	32	34	33	0.007	31	0.733
C3.3	30	34	34	33	33	0.008	32	0.687
C4.1	60	67	66	63	62	0.017	62	2.333
C4.2	60	68	63	62	63	0.017	61	2.207
C4.3	60	64	63	62	62	0.016	61	2.205
C5.1	90	94	94	93	91	0.028	91	4.853
C5.2	90	99	95	92	92	0.029	91	4.839
C5.3	90	97	94	93	92	0.029	91	4.883
C6.1	120	130	126	123	123	0.042	121	8.631
C6.2	120	130	123	122	123	0.045	122	8.728
C6.3	120	131	128	124	123	0.047	121	8.763
C7.1	240	252	249	247	243	0.129	242	38.468
C7.2	240	264	247	244	242	0.124	242	38.885
C7.3	240	257	249	245	243	0.126	241	38.787

Table 5: Comparison between the Bidirectional Best-fit Heuristic (BBF) and Modified Bidirectional Best-fit Heuristic (BBFM) when applied to datasets of Valenzuela & Wang (2001) and Burke et al. (2004)

Instance	*Opt*	BF	BBF	Time(s)	BBFM	Time(s)
nice0025	1000	1074	1083	0.086	1069	2.249
nice0050	1000	1085	1079	0.14	1068	5.09
nice0100	1000	1070	1067	0.241	1063	14.089
nice0200	1000	1053	1053	0.439	1038	47.298
nice0500	1000	1035	1033	1.026	1024	307.285
nice1000	1000	1037	1037	2.317	1012	1497.162
path0025	1000	1101	1091	0.088	1091	2.499
path0050	1000	1138	1074	0.157	1074	5.215
path0100	1000	1073	1073	0.262	1073	15.040
path0200	1000	1041	1053	0.486	1053	49.968
path0500	1000	1037	1032	1.105	1031	305.560
path1000	1000	1028	1028	2.191	1026	1519.160
N1	40	45	40	0.003	40	0.000
N2	50	53	52	0.006	50	0.008
N3	50	52	52	0.009	52	0.760
N4	80	83	82	0.016	82	1.468
N5	100	105	103	0.022	102	2.360
N6	100	103	102	0.025	101	3.195
N7	100	107	106	0.037	105	4.596
N8	80	84	82	0.035	81	5.639
N9	150	152	152	0.057	151	9.184
N10	150	152	151	0.204	151	39.322
N11	150	152	151	0.337	150	15.297
N12	300	306	302	0.821	302	282.628
N13	960	964	964	8.557	960	422.601

Table 6: Performance in terms of %-gap and number of optimal results found of techniques from the literature over Hopper & Turton (2001) instances

Instance	BF	BBF	BF-SA	BBFM	SWP	SVC	GRASP	ISA	IDBS
C1.1	5.00	5.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C1.2	10.00	5.00	0.00	5.00	5.00	5.00	0.00	0.00	0.00
C1.3	20.00	5.00	0.00	5.00	0.00	0.00	0.00	0.00	0.00
C2.1	6.67	6.67	6.67	6.67	6.67	0.00	0.00	0.00	0.00
C2.2	6.67	13.33	6.67	0.00	0.00	0.00	0.00	0.00	0.00
C2.3	6.67	6.67	6.67	6.67	0.00	0.00	0.00	0.00	0.00
C3.1	6.67	6.67	3.33	0.00	0.00	0.00	0.00	0.00	0.00
C3.2	13.33	10.00	3.33	3.33	3.33	3.33	3.33	3.33	0.00
C3.3	10.00	10.00	3.33	6.67	0.00	0.00	0.00	0.00	0.00
C4.1	5.00	3.33	1.67	3.33	1.67	1.67	1.67	1.67	0.00
C4.2	3.33	5.00	1.67	1.67	1.67	1.67	1.67	1.67	0.00
C4.3	3.33	3.33	1.67	1.67	1.67	1.67	1.67	1.67	0.00
C5.1	3.33	1.11	1.11	1.11	1.11	1.11	1.11	1.11	0.00
C5.2	2.22	2.22	1.11	1.11	1.11	1.11	1.11	1.11	0.00
C5.3	3.33	2.22	2.22	1.11	1.11	1.11	1.11	1.11	0.00
C6.1	2.50	2.50	1.67	0.83	1.67	0.83	1.67	0.83	0.00
C6.2	1.67	2.50	0.83	1.67	0.83	0.83	1.67	0.83	0.00
C6.3	3.33	2.50	1.67	0.83	1.67	0.83	1.67	0.83	0.00
C7.1	2.92	1.25	1.67	0.83	1.25	0.83	1.67	0.83	0.00
C7.2	1.67	0.83	1.67	0.83	0.83	0.83	1.25	0.42	0.00
C7.3	2.08	1.25	2.08	0.83	1.25	0.83	1.25	0.83	0.00
Average	5.70	4.59	2.34	2.32	1.47	1.03	0.99	0.77	0.00
# of *Opt*	0	0	3	3	6	7	8	8	21

BBFM has similar performance many of the techniques in the literature in terms of %-gap including SVC(SubKP) (Belov et al., 2008) and GRASP (Alvarez-Valdes et al., 2009) which up until recently were considered state of the art. Although the performance in terms of %-gap initially looks poor, using such a metric for comparison can be misleading when working with instances with varied optimal solution values. For example, obtaining a solution with height 1 greater than the optimal for an instance in C1 gives %-gap of 5% whereas in C7 this would be 0.42%. This is unfortunate for methods such as ours which excel in the larger instances of the set. When compared directly, BBFM has equal performance SVC(SubKP) (Belov et al., 2008) in 10 instances, outperforms SVC in 5 instances and is outperformed by SVC in 6 instances showing very similar performance. When compared to the second best metaheuristic, BBFM matches the performance of ISA (Leung et al., 2011) in 13 of the 21 instances. The only method which finds more optimal solution in this set of instances is the IDBS approach of Wei et al. (2011) which finds optimal results for all 21 instances. Note that BBF and BBFM are deterministic unlike many of the metaheuristic techniques whose performance is often reported as an average of multiple runs, here the performance of our algorithm is guaranteed.

Table 7 shows the performance of the same nine techniques over the benchmark set provided by Burke et al. (2004). Again the performance of BBFM is close to that of

Table 7: Performance in terms of %-gap and number of optimal results found of techniques from the literature over Burke et al. (2004) instances

Instance	BF	BBF	BF-SA	BBFM	GRASP	SWP	SVC	ISA	IDBS
N1	12.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
N2	6.00	4.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
N3	4.00	4.00	2.00	4.00	2.00	0.00	0.00	0.00	0.00
N4	3.75	2.50	2.50	2.50	1.25	1.25	1.25	0.00	0.00
N5	5.00	3.00	3.00	2.00	2.00	1.00	1.00	1.00	0.00
N6	3.00	2.00	2.00	1.00	1.00	1.00	1.00	1.00	0.00
N7	7.00	6.00	4.00	5.00	1.00	1.00	1.00	0.00	0.00
N8	5.00	2.50	2.50	1.25	1.25	1.25	1.25	1.25	0.00
N9	1.33	1.33	1.33	0.67	0.67	0.67	0.67	0.67	0.00
N10	1.33	0.67	1.33	0.67	0.67	0.67	0.67	0.67	0.00
N11	1.33	0.67	2.00	0.00	0.67	0.67	0.67	0.67	0.00
N12	2.00	0.67	2.00	0.67	1.33	1.33	0.33	0.33	0.00
N13	0.42	0.42	0.42	0.00	0.52	0.63	0.31	0.00	0.00
Average	4.05	2.13	1.78	1.37	0.95	0.73	0.63	0.43	0.00
# *Opt*	0	1	2	4	2	3	3	6	13

some of the best metaheuristics in the literature. BBFM finds more optimal solutions in this set than the previously state-of-the-art SVC(SubKP) (Belov et al., 2008) and GRASP (Alvarez-Valdes et al., 2009) approaches. In terms of optimal solutions, only ISA (Leung et al., 2011) and IDBS Wei et al. (2011) are able to find more optimal solutions than BBFM. BBFM is the only technique other than the state-of-the-art IDBS able to find optimal packings for the instances N11 and N13 from this set. Figure 6 and Figure 7 show the optimal packings for these two instances.

Finally, the performance over the benchmark of Valenzuela & Wang (2001) is presented in Table 8. Here we see BBFM perform poorly when compared to metaheuristic techniques. This may be due to these instances creating a landscape which is difficult for a constructive heuristic to traverse.

6. Conclusions

The bidirectional best-fit heuristic is a deterministic heuristic which has previously been shown to outperform existing packing heuristics in the literature when applied to the two dimensional orthogonal rectangular strip packing problem. We have presented a modified version of the bidirectional best-fit heuristic which also considers placement of rectangles in pairs. We have shown that modifying the heuristics behaviour in such a way leads to improved performance in over two-thirds of the instances tested. As this is an extension to the original heuristic performance is equal in all other cases. The cost of this improvement is an increase in runtime due to a increase in the number of policy combinations to be tested. Future work will include investigating whether it is possible to 'learn' which policies perform well on which instances and decide in advance a subset of policy combinations to apply. In addition to outperforming existing heuristics we have shown that the performance of this method is comparable to many of the state-of-the-art

Figure 6: Optimal packing found by BBFM for instance N11 from Burke et al. (2004)

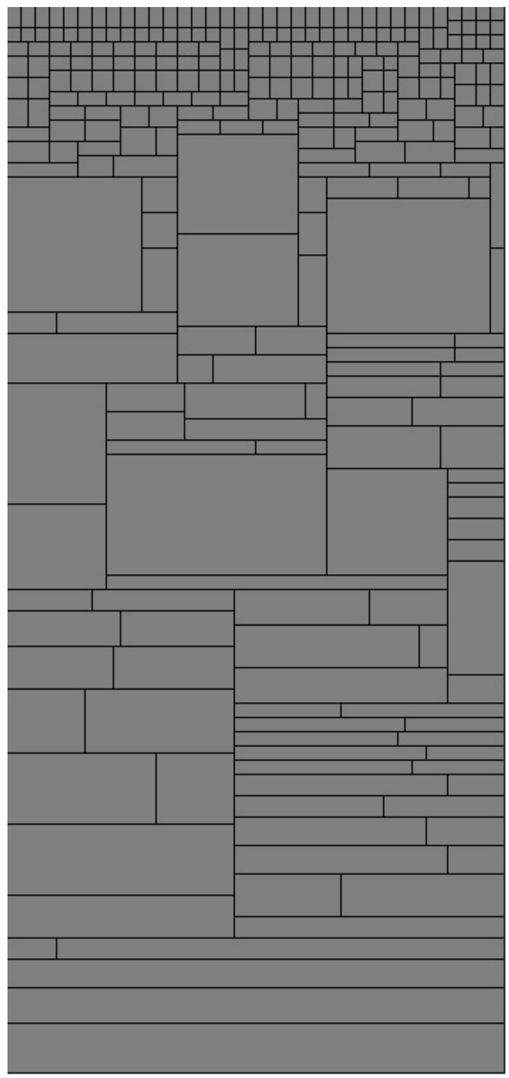


Figure 7: Optimal packing found by BBFM for instance N13 from Burke et al. (2004)

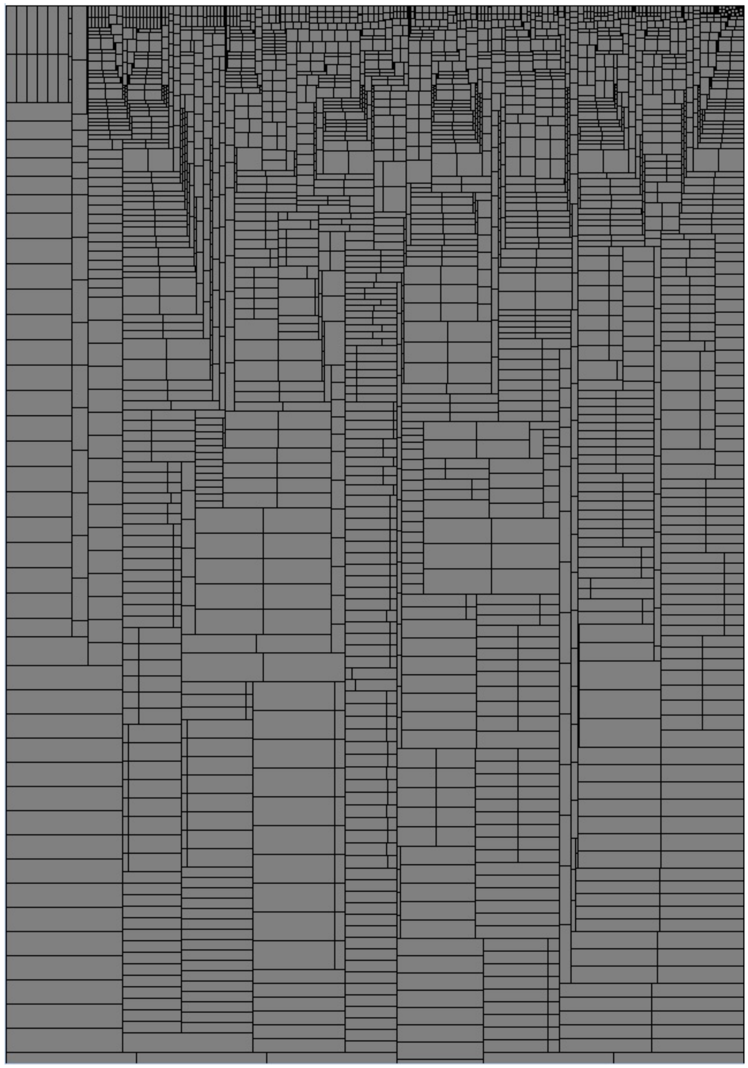


Table 8: Performance in terms of %-gap of techniques from the literature over Valenzuela & Wang (2001) instances

Instance	BF	BBF	BBFM	BF-SA	SWP	GRASP	ISA	SVC	IDBS
nice0025	7.40	8.30	6.90	4.00	3.70	3.40	4.10	3.70	0.30
nice0050	8.50	7.90	6.80	4.40	4.90	4.70	4.70	3.80	2.30
nice0100	7.00	6.70	6.30	5.00	4.60	4.10	3.70	3.50	2.00
nice0200	5.30	5.30	3.80	4.70	3.80	3.70	3.10	2.60	1.40
nice0500	3.50	3.30	2.40	3.50	3.30	2.40	1.50	1.70	0.50
nice1000	3.70	3.70	1.20	3.80	2.90	2.00	1.10	1.40	0.20
path0025	10.10	9.10	9.10	3.10	6.90	4.20	4.20	4.20	0.60
path0050	13.80	7.40	7.40	3.40	1.70	1.90	1.50	1.40	1.00
path0100	7.30	7.30	7.30	3.00	2.90	2.70	2.30	2.20	2.30
path0200	4.10	5.30	5.30	3.40	2.00	2.30	1.80	1.80	2.10
path0500	3.70	3.20	3.10	3.50	3.20	3.40	2.00	2.20	1.50
path1000	2.80	2.80	2.60	2.90	2.80	2.60	1.10	1.80	0.70
Average	6.43	5.86	5.18	3.73	3.56	3.12	2.59	2.53	1.24

metaheurstics taken from the literature. Unlike many of these techniques, the modified bidirectional best-fit heuristic is a deterministic method which results in the same best packing for each run.

References

Aşik, O. B., & Özcan, E. (2009). Bidirectional best-fit approach for orthogonal rectangular strip packing. *Annals of Operations Research*, *172*, 405–427.

Alvarez-Valdes, R., Parreño, F., & Tamarit, J. (2009). A branch and bound algorithm for the strip packing problem. *OR Spectrum*, *31*, 431–459.

Baker, B., Coffman-Jr, E. G., & Rivest, R. L. (1980). Orthogonal packing in two dimensions. *SIAM J. Comput.*, *9*, 846–855.

Beasley, J. E. (1985). An exact two-dimensional non-guillotine cutting tree search procedure. *Operations Research*, *33*, 49–64.

Belov, G., Scheithauer, G., & Mukhacheva, E. A. (2008). One-dimensional heuristics adapted for two-dimensional rectangular strip packing. *Journal of the Operational Research Society*, *59*, 823–832.

Beltrán, J. D., Calderón, J. E., Cabrera, R. J., Pérez, J. A. M., & Moreno-vega, J. M. (2004). Grasp/vns hybrid for the strip packing problem. In *In 1st International Workshop on Hybrid Metaheuristics* (pp. 79–90).

Bortfeldt, A. (2006). A genetic algorithm for the two-dimensional strip packing problem with rectangular pieces. *European Journal of Operational Research*, *172*, 814–837.

Boschetti, M. A., & Montaletti, L. (2010). An exact algorithm for the two-dimensional strip-packing problem. *Operations Research*, *58*, 1774–1791.

Burke, E., Kendall, G., & Whitwell, G. (2004). A new placement heuristic for the orthogonal stock-cutting problem. *Operations Research*, *52*, 655–671.

Burke, E. K., Hart, E., Kendall, G., Newall, J., Ross, P., & Schulenburg, S. (2003). Hyper-heuristics: An emerging direction in modern search technology. Handbook of Metaheuristics.

Burke, E. K., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., & Qu, R. (2010a). *Hyper-heuristics: A Survey of the State of the Art*. Technical Report No. NOTTCS-TR-SUB-0906241418-2747 School of Computer Science and Information Technology, University of Nottingham.

Burke, E. K., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., & Woodward, J. (2010b). Handbook of metaheuristics. chapter A Classification of Hyper-heuristics Approaches. (pp. 449–468). Springer.

Burke, E. K., Hyde, M. R., & Kendall, G. (2011). A squeaky wheel optimisation methodology for two-dimensional strip packing. *Computers and Operations Research*, *38*, 1035–1044.

Burke, E. K., Kendall, G., & Whitwell, G. (2009). A simulated annealing enhancement of the best-fit heuristic for the orthogonal stock-cutting problem. *INFORMS Journal on Computing*, *21*, 505–516.

Chazelle, B. (1983). The bottom-left bin packing heuristic: An efficient implementation. *IEEE Transactions on Computers*, *32*, 697–707.

Christofides, N., & Whitlock, C. (1977). An algorithm for two-dimensional cutting problems. *Operations Research*, *25*, 30–44.

Cung, V.-D., Hifi, M., & Cun, B. L. (2000). Constrained two-dimensional cutting stock problems a best-first branch-and-bound algorithm. *International Transactions in Operational Research*, *7*, 185–210.

Dagli, C. H., & Poshyanonda, P. (1997). New approaches to nesting rectangular patterns. *Journal of Intelligent Manufacturing*, *8*, 177–190.

Dowsland, K. A., & Dowsland, W. B. (1992). Packing problems. *European Journal of Operational Research*, *56*, 2–14.

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

Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co.

Gilmore, P., & Gomory, R. (1961). A linear programming approach to the cutting stock problem. *Operations Research*, *9*, 849–859.

Hifi, M., & Zissimopoulos, V. (1997). Constrained two-dimensional cutting: An improvement of christofides and whitlock's exact algorithm. *The Journal of the Operational Research Society*, *48*, 324–331.

Hopper, E., & Turton, B. C. H. (2001). An empirical investigation of meta-heuristic and heuristic algorithms for a 2d packing problem. *European Journal of Operational Research*, *128*, 34–57.

Imahori, S., & Yagiura, M. (2010). The best-fit heuristic for the rectangular strip packing problem: An efficient implementation and the worst case approximation ratio. *Computers & Operations Research*, *37*, 325–333.

Jakobs, S. (1996). On genetic algorithms for the packing of polygons. *European Journal of Operational Research*, *88*, 165–181.

Kenmochi, M., Imamichi, T., Nonobe, K., Yagiura, M., & Nagamochi, H. (2009). Exact algorithms for the two-dimensional strip packing problem with and without rotations. *European Journal of Operational Research*, *198*, 73–83.

Lesh, N., Marks, J., McMahon, A., & Mitzenmacher, M. (2004). Exhaustive approaches to 2d rectangular perfect packings. *Information Processing Letters*, *90*, 7–14.

Lesh, N., Marks, J., McMahon, A., & Mitzenmacher, M. (2005). New heuristic and interactive approaches to 2d rectangular strip packing. *ACM Journal of Experimental Algorithmics*, *10*, 1–18.

Leung, S. C., Zhang, D., & Sim, K. M. (2011). A two-stage intelligent search algorithm for the two-dimensional strip packing problem. *European Journal of Operational Research*, *215*, 57–69.

Liu, D., & Teng, H. (1999). An improved bl-algorithm for genetic algorithms of the orthogonal packing of rectangles. *European Journal of Operational Research*, *112*, 413–420.

Lodi, A., Martello, S., & Monaci, M. (2002). Two-dimensionalpacking problems: A survey. *European Journal of Operational Research*, *141*, 241–252.

Lodi, A., Martello, S., & Vigo, D. (1999). Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems. *INFORMS Journal on Computing*, *11*, 345–357.

Macedo, R., Alves, C., & Valério de Carvalho, J. M. (2010). Arc-flow model for the two-dimensional guillotine cutting stock problem. *Computers & Operations Research*, *37*, 991–1001.

Martello, S., Monaci, M., & Vigo, D. (2003). An exact approach to the strip-packing problem. *INFORMS Journal on Computing*, *15*, 310–319.

Ramesh Babu, A., & Ramesh Babu, N. (1999). Effective nesting of rectangular parts in multiple rectangular sheets using genetic and heuristic algorithms. *International Journal of Production Research*, *37*, 1625–1643.

Terashima-Marín, H., Flores-Álvarez, E. J., & Ross, P. (2005a). Hyper-heuristics and classifier systems for solving 2d-regular cutting stock problems. In *Proceedings of GECCO 2005* (pp. 637–643).

Terashima-Marín, H., Moran-Saavedra, A., & Ross, P. (2005b). Forming hyper-heuristics with gas when solving 2d-regular cutting stock problems. In *Proceedings of the IEEE CEC* (pp. 1104–1110). volume 2.

Valenzuela, C. L., & Wang, P. Y. (2001). Heuristics for large strip packing problems with guillotine patterns: an empirical study. In *Proceedings of the Metaheuristic International Conference* MIC'2001.

Wäscher, G., Haußner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, *183*, 1109–1130.

Wei, L., Oon, W.-C., Zhu, W., & Lim, A. (2011). A skyline heuristic for the 2d rectangular packing and strip packing problems. *European Journal of Operational Research*, *215*, 337–346.

Yang, S., Han, S., & Ye, W. (2013). A simple randomized algorithm for two-dimensional strip packing. *Computers & Operations Research*, *40*, 1–8.

Zhang, D., Kang, Y., & Deng, A. (2006). A new heuristic recursive algorithm for the strip rectangular packing problem. *Computers & Operations Research*, *33*, 2209–2217.