



HAL
open science

An anytime tree search algorithm for two-dimensional two- and three-staged guillotine packing problems

Florian Fontan, Luc Libralesso

► To cite this version:

Florian Fontan, Luc Libralesso. An anytime tree search algorithm for two-dimensional two- and three-staged guillotine packing problems. 2020. ⟨hal-02531031v2⟩

HAL Id: hal-02531031

<https://hal.science/hal-02531031v2>

Preprint submitted on 18 Apr 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



HAL Authorization

An anytime tree search algorithm for two-dimensional two- and three-staged guillotine packing problems

Florian Fontan*, Luc Libralesso^a

^a*Univ. Grenoble Alpes, CNRS, Grenoble INP[☆], G-SCOP, 38000 Grenoble, France*

Abstract

Libralesso and Fontan (2020) proposed an anytime tree search algorithm for the 2018 ROADEF/EURO challenge glass cutting problem¹. The resulting program was ranked first among 64 participants. In this article, we generalize it and show that it is not only effective for the specific problem it was originally designed for, but is also very competitive and even returns state-of-the-art solutions on a large variety of Cutting and Packing problems from the literature. We adapted the algorithm for two-dimensional Bin Packing, Multiple Knapsack, and Strip Packing Problems, with two- or three-staged exact or non-exact guillotine cuts, the orientation of the first cut being imposed or not, and with or without item rotation. The combination of efficiency, ability to provide good solutions fast, simplicity and versatility makes it particularly suited for industrial applications, which require quickly developing algorithms implementing several business-specific constraints. The algorithm is implemented in a new software package called PackingSolver.

Keywords: two-dimensional guillotine packing, bin packing, knapsack, strip packing, anytime algorithm, tree search algorithm

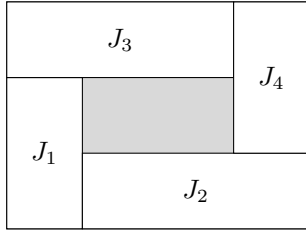
1 The 2018 ROADEF/EURO challenge featured an industrial glass cutting problem arising at the French
2 company Saint Gobain. Libralesso and Fontan (2020) proposed an anytime tree search algorithm that was
3 ranked first in the final phase of the challenge. They showed that the algorithm performs very well on
4 this specific variant with the specific instances considered. Indeed, some of the industrial constraints of the
5 problem seem to favor this kind of constructive approach. In particular, the problem includes precedence
6 constraints, which highly penalize other approaches such as local search, dynamic programming, mixed-
7 integer linear programming or column generation. Therefore, it was not obvious *a priori* whether the
8 algorithm would be competitive on other variants. In this article, we show that even on pure Packing
9 Problems from the literature, it is competitive compared to the other dedicated algorithms, and is even able
10 to return state-of-the-art solutions on several variants.

¹<https://www.roadef.org/challenge/2018/en/index.php>

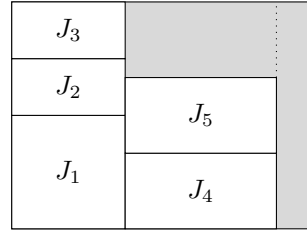
[☆]Institute of Engineering Univ. Grenoble Alpes

*Corresponding author

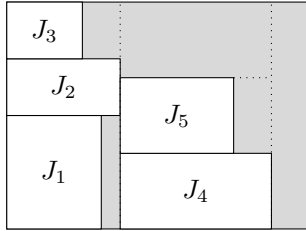
Email addresses: `dev@florian-fontan.fr` (Florian Fontan), `luc.libralesso@grenoble-inp.fr` (Luc Libralesso)



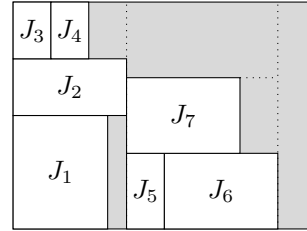
(a) Non-guillotine pattern



(b) Two-staged exact guillotine pattern, first stage vertical



(c) Two-staged non-exact guillotine pattern, first stage vertical



(d) Three-staged exact guillotine pattern, first stage vertical

Figure 1: Pattern type examples

11 Even though most of the new constraints taken into account integrate naturally within the algorithm,
 12 several improvements need to be made to make it efficient on the large variety of problems and instances from
 13 the literature: two new guide functions are proposed to deal with instances with different item distributions;
 14 an additional guide is designed for the Knapsack objective; and some flexibility has been introduced in the
 15 symmetry breaking strategy.

16 Libralesso and Fontan (2020) proposed an efficient algorithm for a specific problem with specific con-
 17 straints and instances. Here, we propose an efficient approach which should be useful for almost any
 18 (guillotine for now) Packing Problem. Also, as discussed in Section 5, experimenting on all these variants
 19 greatly improved our understanding of the effectiveness of MBA* and other tree search algorithms.

20 1. Introduction

21 We consider two-dimensional guillotine Packing Problems: one has to pack rectangles of various sizes
 22 into larger bins while only edge-to-edge cuts are allowed. In a solution, guillotine cuts can be partitioned
 23 into stages, *i.e.* series of parallel cuts, and it is common to limit the number of allowed stages. Here, we
 24 restrict to two- or three-staged guillotine patterns. In both cases, we consider both exact and non-exact
 25 variants. In the non-exact variant, an additional cut is allowed to separate items from waste. Figure 1
 26 illustrates the different pattern types.

27 We consider the three main objectives studied in the literature: Bin Packing, Knapsack and Strip Packing.
 28 In Bin Packing and Strip Packing Problems, all items need to be produced. In Bin Packing Problems,
 29 the number of used bins is minimized, while in Strip Packing Problems, there is only one container with
 30 one infinite dimension and the objective is to minimize the length used in this dimension. In Knapsack
 31 Problems, the number of containers is limited, every item has a profit and the total profit of the packed
 32 items is maximized.

33 Finally, for each variant, we consider the oriented case where item rotation is not allowed and non-oriented
 34 case where it is.

35 Throughout the article, the different variants are named following our notations illustrated with the
 36 following examples:

- 37 • BPP-O: (non-guillotine) Bin Packing Problem, Oriented
- 38 • G-BPP-R: Guillotine cuts, Bin Packing Problem, Rotation
- 39 • 2G-KP-O: 2-staged exact guillotine cuts, first cut horizontal or vertical, Knapsack Problem, Oriented
- 40 • 3NEGH-SPP-O: 3-staged non-exact guillotine cuts, first cut horizontal, Strip Packing Problem, Ori-
 41 ented

42 We also use the following vocabulary: a k -cut is a cut performed in the k -th stage. Cuts separate bins
 43 into k -th level sub-plates. For example, 1-cuts separate the bin in several first level sub-plates. S denotes a
 44 solution or a node in the search tree (a partial solution).

45 The following definitions are given for the case where the first cut in the last bin is vertical, but naturally,
 46 adapt to the case where it is horizontal. We call the last first level sub-plate, the rightmost one containing
 47 an item; the last second level sub-plate, the topmost one containing an item in the last first level sub-plate;
 48 and the last third level sub-plate the rightmost one containing an item in the last second level sub-plate.
 49 $x_1^{\text{prev}}(S)$ and $x_1^{\text{curr}}(S)$ are the left and right coordinates of the last first level sub-plate; $y_2^{\text{prev}}(S)$ and $y_2^{\text{curr}}(S)$
 50 are the bottom and top coordinates of the last second level sub-plate; and $x_3^{\text{prev}}(S)$ and $x_3^{\text{curr}}(S)$ are the left
 51 and right coordinates of the last third level sub-plate. Figure 2 presents a usage example of these definitions.
 52 We define the area and the waste of a solution S as follows:

$$\text{area}(S) = \begin{cases} A + x_1^{\text{curr}}(S)h & \text{if } S \text{ contains all items} \\ A + x_1^{\text{prev}}(S)h \\ \quad + (x_1^{\text{curr}}(S) - x_1^{\text{prev}}(S))y_2^{\text{prev}}(S) \\ \quad + (x_3^{\text{curr}}(S) - x_1^{\text{prev}}(S))(y_2^{\text{curr}}(S) - y_2^{\text{prev}}(S)) & \text{otherwise} \end{cases}$$

$$\text{waste}(S) = \text{area}(S) - \text{item_area}(S)$$

53 with A the sum of the areas of all but the last bin, h the height of the last bin and $\text{item_area}(S)$ the sum
 54 of the area of the items of S . Area and waste are illustrated in Figure 2.

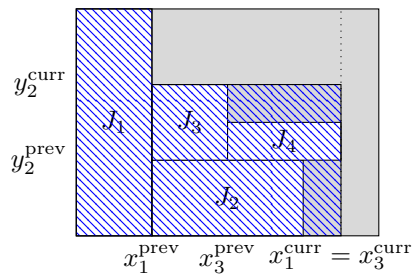


Figure 2: Last bin of a solution which does not contain all items. The area is the whole hatched part and the waste in the grey hatched part.

2. Literature review

Two-dimensional guillotine Packing Problems have been introduced by Gilmore and Gomory (1965) and have received a lot of attention since. Researchers usually focus on one specific variant or only on a few ones.

Algorithms are sometimes adapted for both the oriented and the non-oriented cases. Velasco and Uchoa (2019) developed a heuristic for G-KP-O and G-KP-R, Wei et al. (2014) for G-SPP-O and G-SPP-R, Charalambous and Fleszar (2011), Fleszar (2013) and Cui et al. (2018) for G-BPP-O and G-BPP-R, Lodi and Monaci (2003) an exact algorithm for 2NEGH-KP-O and 2NEGH-KP-R.

Some methods have been designed to work on more variants. do Nascimento et al. (2019) developed an exact algorithm for G-KP-O, 3NEGH-KP-O, 2NEGH-KP-O and the three-dimensional variants, Bortfeldt and Winter (2009) developed a genetic algorithm for G-KP-O, G-KP-R, and the non-guillotine variants. Alvelos et al. (2009) and Silva et al. (2010) respectively developed a heuristic and an exact algorithm for 3NEGH-BPP-O, 3GH-BPP-O, 2NEGH-BPP-O and 2GH-BPP-O, and the non-oriented cases. Furini et al. (2016) introduced a model for G-KP-O and G-SPP-O. Lodi et al. (2004) proposed a unified tabu search for two- and three-dimensional Packing Problems. They provide computational experiments for BPP-O and the three-dimensional variant. They also describe how to adapt the algorithm for several variants such as Strip Packing or Multiple Knapsack. However, adapting the algorithm requires to provide a heuristic procedure, on which the efficiency of the algorithm highly relies. We did not find any use of their tabu search in the subsequent literature. Also, a framework has been proposed by Nepomuceno et al. (2008); unfortunately, it has only been implemented for BPP-O and we did not find any use of their framework in the subsequent literature either.

Regarding our methodology, even though tree search algorithms have been widely used to solve Packing Problems, the search algorithm that we implemented does not seem to have been proposed before. We may notice that many packing algorithms rely on Beam Search which is relatively close, as discussed in Section 5. Akeb et al. (2009), Hifi and M'Hallah (2009), Akeb et al. (2010) and Akeb et al. (2011) implemented

80 it for Circular Packing Problems; Bennell and Song (2010) and Bennell et al. (2018) for Irregular Packing
81 Problems; Wang et al. (2013), Araya and Riff (2014) and Araya et al. (2020) for three-dimensional Packing
82 Problems; and Hifi et al. (2012) for 2NEGH-KP-O. However, these Beam Search implementations signifi-
83 cantly differ from our tree search implementation. Most of them do not use a restart strategy, are block-based
84 approaches and use probing (filling partial solutions with a greedy heuristic) to evaluate the quality of nodes.
85 Furthermore, they are globally more complex than our tree search implementation, suggesting that we better
86 captured the key ideas that make tree search algorithms efficient for Packing Problems.

87 3. Algorithm description

88 We propose an anytime tree search algorithm.

89 Anytime is a terminology usually found in automated planning and scheduling (AI planning) communi-
90 ties. It means that the algorithm can be stopped at any time and still provides good solutions. In other
91 words, it produces feasible solutions quickly and improves them over time (as classical meta-heuristics do).

92 Tree search algorithms represent the solution space as an implicit tree called “branching scheme” and
93 explore it completely in the case of exact methods or partially in the case of heuristic methods. The
94 branching scheme is described in Section 3.1 and the tree search algorithm in Section 3.2.

95 3.1. Branching scheme

96 We describe the branching scheme for the 3-staged cases with vertical cuts in the first stage. For the
97 2-staged cases, we merely impose the position of the first cut to be at the end of the bin and adjust the
98 computation of parameters accordingly; and when the cuts in the first stage are horizontal, we simply adapt
99 the computation of coordinates.

100 The branching scheme is rather straightforward. The root node is the empty solution without any items,
101 and at each stage, a new item is added. All items that do not belong to the current node are considered.
102 However, items in a solution are inserted according to the following order: rightmost first level sub-plates
103 first; within a first level sub-plate, bottommost second level sub-plates first; and within a second level sub-
104 plate, rightmost items first. Thus, a new item can be inserted in a new bin; in a new first level sub-plate
105 to the right of the current one; in a new second-level sub-plate above the current one; in a new third-level
106 sub-plate, to the right of the last added item. If the cuts of the first stage can be vertical or horizontal, then
107 two different insertions in a new bin are considered: an insertion in a new bin with vertical cuts in the first
108 stage, and an insertion in a new bin with horizontal cuts in the first stage.

109 To handle exact guillotine cuts, we simply fix the position of the 2-cut above an item inserted in a new
110 bin, first or second level sub-plate, *i.e.* the next items inserted in the same second level sub-plate will only
111 be those of the same height.

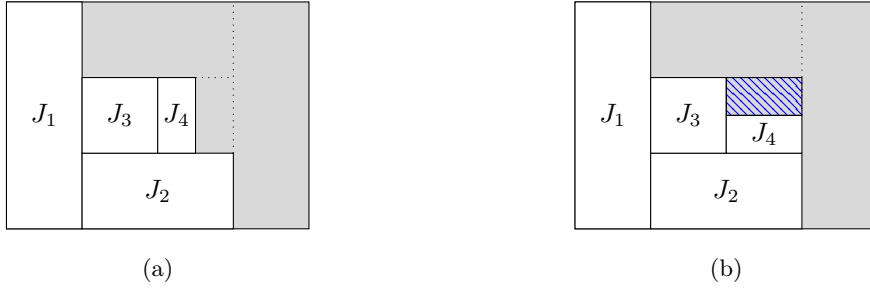


Figure 3: Solution (a) dominates solution (b) because the hatched area will not be used

112 Item rotation or not is naturally handled in the branching scheme.

113 To reduce the size of the tree, we apply some simple dominance rules.

114 First, if an item can be inserted in the current bin, we do not consider insertions in a new bin; and if an
 115 item can be inserted in the current first (resp. second) level sub-plate without increasing the position of its
 116 left 1-cut (resp. top 2-cut), we do not consider insertions in a new first (resp. second) level sub-plate.

117 Then, if item rotation is allowed, some insertions can be discarded as illustrated in Figure 3.

118 We also impose an order on identical items.

119 Finally, we add the following symmetry breaking strategy: a k -level sub-plate is forbidden to contain
 120 an item with a smaller index than the previous k level sub-plate of the same $(k - 1)$ -level sub-plate. The
 121 symmetry breaking strategy is controlled with a parameter s , $1 \leq s \leq 4$. If $s = k$, then the symmetry
 122 breaking strategy is only used with k' level sub-plates, $k' \geq k$. For example, if $s = 4$, no symmetry breaking
 123 strategy is used. The choice of the value of s is discussed in Section 5.

124 3.2. Tree search algorithm

125 The tree described in the previous section is too large to be entirely explored. Therefore, we use a
 126 tree search algorithm that we called Memory Bounded A* (MBA*) to explore the most interesting parts
 127 in priority. The pseudo-code is given in Algorithm 1. MBA* starts with a queue containing only the root
 128 node. At each iteration, the “best” node is extracted from the queue and its children are added to the queue.
 129 If the size of the queue goes over a pre-defined threshold value, the “worst” nodes are discarded. We start
 130 with a threshold of 2, and each time the queue becomes empty, we start over with a threshold multiplied by
 131 the growth factor f . We choose $f = 1.5$ as discussed in Section 5.

132 The function used to define “better” and “worse” is called a guide. The lower the value of the guide
 133 function is, the better the solution. For Bin Packing and Strip Packing Problems, we designed the following
 134 guide functions:

$$c_0(S) = \text{waste_percentage}(S)$$

Algorithm 1 Memory Bounded A* (MBA*)

```
1: queue  $\leftarrow$  {root}
2: while |queue|  $\neq$   $\emptyset$  and time < timelimit do
3:    $n \leftarrow$  extractBest(queue)
4:   queue  $\leftarrow$  queue  $\setminus$  { $n$ }
5:   for all  $v \in$  children( $n$ ) do
6:     queue  $\leftarrow$  queue  $\cup$  { $v$ }
7:   while |queue| >  $D$  do
8:      $n \leftarrow$  extractWorst(queue)
9:     queue  $\leftarrow$  queue  $\setminus$  { $n$ }
```

135

$$c_1(S) = \frac{\text{waste_percentage}(S)}{\text{mean_item_area}(S)}$$

136

$$c_2(S) = \frac{0.1 + \text{waste_percentage}(S)}{\text{mean_item_area}(S)}$$

137

$$c_3(S) = \frac{0.1 + \text{waste_percentage}(S)}{\text{mean_squared_item_area}(S)}$$

138 with

- 139 • $\text{waste_percentage}(S) = \text{waste}(S)/\text{area}(S)$;
- 140 • $\text{mean_item_area}(S)$ the mean area of the items of S ;
- 141 • $\text{mean_squared_item_area}(S)$ the mean squared area of the items of S .

142 For Knapsack Problems, we use the following guide function:

$$c_4(S) = \frac{\text{area}(S)}{\text{profit}(S)}$$

143 with $\text{profit}(S)$ the sum of profit of the items of S .

144 The importance and design of these guide functions are discussed in Section 5.

145 **4. Computational experiments**

146 The algorithm has been implemented in C++ in a new software package called PackingSolver. The
147 code is available online². The repository also contains all the scripts used to conduct the experiments so
148 that results can be reproduced. The results presented above have been obtained with PackingSolver 0.2³
149 running on a personal computer with an Intel Core i5-8500 CPU @ 3.00GHz \times 6. We allow running up to

²<https://github.com/fontanf/packingsolver>³<https://github.com/fontanf/packingsolver/releases/tag/0.2>

150 3 threads with different settings in parallel. The settings have been chosen following the observations given
151 in Section 5. Better settings may exist, we try to reproduce the results one would obtain in a practical
152 situation where the global characteristics of the instances are known.

153 We compare the performances of our algorithm with the best algorithms from the literature for each
154 variant. Due to a large number of problems, we only provide a synthesis of the results here. However,
155 detailed results are available online⁴ and the interested reader is encouraged to have a look at them.

156 Results are summarized in Tables 1, 2 and 3. The first column of the tables indicates the article from
157 which the results have been extracted or the parameters we used for our algorithm. c_a^b indicates a thread
158 with guide function c_a and symmetry breaking parameter b . TL stands for “time limit”. The time limit has
159 been chosen to yield a good compromise between computation time and the best solution value. We only
160 indicate the frequencies of the processors used to evaluate the other algorithms when they significantly differ
161 from ours, *i.e.* below 2GHz.

162 For Bin Packing Problems, the second column contains the total number of bins used in Table 1a and
163 the average of the average percentage of waste of each sub-dataset in Table 1b. For Knapsack and Strip
164 Packing Problems, it contains the average gap to the best-known solutions. The third one indicates the
165 average time to best when available, or the average computation time.

166 Dataset “hifi” is a dataset composed of instances from Christofides and Whitlock (1977), Wang (1983),
167 Oliveira and Ferreira (1990), Tschöke and Holthöfer (1995), Fekete and Schepers (1997), Fayard et al. (1998),
168 Hifi (1997) and Cung et al. (2000). Researchers usually test their algorithms on a subset of these instances,
169 but often not the same. Dataset “bwmv” refers to datasets from Berkey and Wang (1987) and Martello and
170 Vigo (1998) which are usually used together.

171 Other datasets are

- 172 • “beasley1985” from Beasley (1985)
- 173 • “fayard1998” from Fayard et al. (1998)
- 174 • “kroger1995” from Kröger (1995)
- 175 • “hopper2000” from Hopper (2000)
- 176 • “hopper2001” from Hopper and Turton (2001)
- 177 • “alvarez2002” from Alvarez-Valdés et al. (2002)
- 178 • “morabito2010” from Morabito and Pureza (2010)
- 179 • “hifi2012” from Hifi et al. (2012)
- 180 • “velasco2019” from Velasco and Uchoa (2019)

⁴https://github.com/fontanf/packingsolver/blob/0.2/results_rectangleguillotine.ods

Article / Parameters	Total	Time (s)
3NEGH-BPP-O, "bwmv"		
PS, $c_0^2c_2^2c_3^3$, TL 60s	7278	0.790
3GH-BPP-O, "bwmv"		
Puchinger and Raidl (2007)	7325	160.68
PS, $c_0^2c_2^2c_3^3$, TL 60s	7344	0.808
2NEGH-BPP-O, "bwmv"		
Alvelos et al. (2014)	7372	29.42
Alvelos et al. (2014)	7364	84.04
PS, $c_0^2c_2^2c_3^3$, TL 60s	7391	0.814
2NEGH-BPP-O, "hifi"		
Cui and Zhao (2013)	260	0.19
PS, $c_2^3c_3^3c_3^4$, TL 10s	255	0.106
2NEGH-BPP-O, "alvarez2002"		
Cui and Zhao (2013)	219	9.5
PS, $c_2^3c_3^3c_3^4$, TL 10s	218	0.346
2NEGH-BPP-R, "bwmv"		
Cui et al. (2016)	7034	20.72
PS, $c_0^2c_2^2c_3^3$, TL 60s	7029	0.590

(a)

Article / Parameters	Waste	Time (s)
3NEGH-BPP-O, "bwmv"		
Alvelos et al. (2009)	26.52	
PS, $c_0^2c_2^2c_3^3$, TL 60s	20.93	0.790
3GH-BPP-O, "bwmv"		
Alvelos et al. (2009)	26.29	
PS, $c_0^2c_2^2c_3^3$, TL 60s	22.34	0.808
2NEGH-BPP-O, "bwmv"		
Alvelos et al. (2009)	26.12	
PS, $c_0^2c_2^2c_3^3$, TL 60s	23.21	0.807
2GH-BPP-O, "bwmv"		
Alvelos et al. (2009)	49.06	
PS, $c_0^3c_2^3c_3^4$, TL 60s	49.45	0.181

(b)

Table 1: Results on Bin Packing Problems

181 4.1. Bin Packing Problems

182 Results for Bin Packing Problems are summarized in Table 1. On 2NEGH-BPP-O and 2NEGH-BPP-R,
183 PackingSolver respectively needs fewer bins than the algorithms from Cui and Zhao (2013) and Cui et al.
184 (2016) for the considered datasets. Furthermore, the average time to best is of the order of a second, which
185 is significantly smaller than the average time reported for the other algorithms. On 3NEGH-BPP-O, 3GH-
186 BPP-O, and 2NEGH-BPP-O, the average of the average percentage of waste of PackingSolver is smaller
187 than the one of the algorithms from Alvelos et al. (2009). However, on 2GH-BPP-O, it is greater. Finally,
188 compared to the algorithms from Puchinger and Raidl (2007) and Alvelos et al. (2014), it needs more bins,
189 but the average time to best is two orders of magnitude smaller than the average time reported for those
190 algorithms. We also note that PackingSolver respectively needs significantly fewer bins on 3NEGH-BPP-O
191 and 3GH-BPP-O compared to the algorithms from Puchinger and Raidl (2007) and Alvelos et al. (2014) for
192 3GH-BPP-O and 2NEGH-BPP-O,

193 4.2. Knapsack Problems

194 Results for Knapsack Problems are summarized in Table 2. We include comparisons with algorithms
195 designed for the non-staged variants. In these cases, PackingSolver usually fails to find the best solutions.
196 It seems likely that they often cannot be reached with only 3 stages. However, its average gap to best is
197 generally less than 1% and on datasets “velasco2019” it is even better than the recent algorithm from Velasco
198 and Uchoa (2019). The same happens on dataset “fayard1998” for G-KP-R, but the algorithm developed by
199 Bortfeldt and Winter (2009) seems to perform significantly worse than more recent algorithms and none of
200 them has been tested on this dataset.

201 On 3NEGV-KP-O, the average gap to best of PackingSolver is better than Cui et al. (2015), but at the
202 expense of longer computation times. For 2NEGH-KP-O, as Alvarez-Valdes et al. (2007), it finds all the
203 best solutions, but faster. Compared to the algorithm from Hifi et al. (2008), it performs slightly worse on
204 dataset “alvarez2002” (even if the average gap is 0.0, it fails to find the best solution on two instances) but
205 better on dataset “hifi2012”.

206 On variants 2NEG-KP-R, 2G-KP-O, 2GH-KP-O, and 2GV-KP-O for which Lodi and Monaci (2003) and
207 Hifi and Roucairol (2001) developed exact algorithms, PackingSolver finds all optimal solutions in reasonable
208 computation times.

209 Note that, to the best of our knowledge, only Cui et al. (2008) proposed an algorithm for a variant of
210 a Multiple Knapsack Problem. However, they consider homogenous T-shaped patterns which we do not
211 consider in this article.

212 4.3. Strip Packing Problems

213 Not many variants of guillotine Strip Packing Problems have been studied in the literature; only G-
214 SPP-O, G-SPP-R, and 2NEGH-SPP-O. This makes comparisons with PackingSolver difficult since it is
215 limited to three-staged patterns, and 2NEGH-SPP-O has several specific structural properties that dedicated
216 algorithms can exploit, but not a more generic one. We still provide computational experiments for these
217 variants in Table 3. As expected, PackingSolver does not perform as well. Still, on dataset “bwmv”, it
218 returns strictly better average solutions on 16 out of 50 groups of instances for G-SPP-O and on 14 out of
219 50 groups of instances for G-SPP-R than the algorithm from Wei et al. (2014). To highlight a bit more the
220 contribution of our algorithm for Strip Packing Problems, we provide a comparison of the solutions from
221 Lodi et al. (2004) and from Cui et al. (2017) for 2NEGH-SPP-O with the solutions returned by PackingSolver
222 for 2NEGH-SPP-R, *i.e.* when item rotation is allowed. The average solutions returned by PackingSolver are
223 strictly better on each of the 50 groups of instances of dataset “bwmv”.

224 5. Discussion

225 In this section, we discuss some items related to the algorithm.

Article / Parameters	Gap	Time (s)
G-KP-O, "fayard1998"		
Velasco and Uchoa (2019)	0.00	0.06
PS, 3NEG-KP-O, $c_4^2c_4^3$, TL 10s	0.16	0.182
G-KP-O, "alvarez2002"		
Wei and Lim (2015)	0.02	21.987
Velasco and Uchoa (2019)	0.00	93.681
PS, 3NEG-KP-O, $c_4^2c_4^3$, TL 60s	0.48	13.264
G-KP-O, "hopper2001"		
Wei and Lim (2015)	0.31	22.214
PS, 3NEG-KP-O, $c_4^2c_4^3$, TL 10s	4.69	1.283
G-KP-O, "morabito2010"		
Velasco and Uchoa (2019)	0.01	19.57
PS, 3NEG-KP-O, $c_4^2c_4^3$, TL 10s	0.17	0.332
G-KP-O, "beasley1985"		
Dolatabadi et al. (2012)	0.00	1397.738
Wei and Lim (2015)	0.44	20.923
PS, 3NEG-KP-O, $c_4^2c_4^3$, TL 10s	0.56	0.204
G-KP-O, "velasco2019"		
Velasco and Uchoa (2019)	1.42	165.618
PS, 3NEG-KP-O, $c_4^2c_4^3$, TL 120s	0.47	34.682
G-KP-R, "hopper2001"		
Wei and Lim (2015)	0.00	5.04
PS, 3NEG-KP-R, $c_4^2c_4^3$, TL 30s	1.71	8.049
G-KP-R, "fayard1998"		
Bortfeldt and Winter (2009)	1.57	
PS, 3NEG-KP-R, $c_4^2c_4^3$, TL 30s	0.00	2.578
G-KP-R, "velasco2019"		
Velasco and Uchoa (2019)	1.05	170.20
PS, 3NEG-KP-R, $c_4^2c_4^3$, TL 120s	0.51	38.590
3NEGV-KP-O, "alvarez2002"		
Cui et al. (2015)	0.09	2.06
PS, $c_4^1c_4^2c_4^3$, TL 60s	0.01	11.879

Article / Parameters	Gap	Time (s)
2NEGH-KP-O, "hifi"		
Alvarez-Valdes et al. (2007)	0.00	0.5
PS, $c_4^2c_4^3$, TL 3s	0.00	0.032
2NEGH-KP-O, "alvarez2002"		
Hifi et al. (2008)	0.00	0.2
PS, $c_4^2c_4^3$, TL 10s	0.00	0.410
2NEGV-KP-O, "alvarez2002"		
Hifi et al. (2008)	0.00	0.2
PS, $c_4^2c_4^3$, TL 10s	0.00	0.382
2NEGH-KP-O, "hifi2012"		
Hifi et al. (2008)	0.26	368.365
PS, $c_4^2c_4^3$, TL 300s	0.12	138.742
2NEGV-KP-O, "hifi2012"		
Hifi et al. (2008)	0.24	310.105
PS, $c_4^2c_4^3$, TL 300s	0.00	121.014
2NEGH-KP-R, "hifi"		
Lodi and Monaci (2003) (533 Mhz)	0.00	34.348
PS, $c_4^2c_4^3$, TL 3s	0.00	0.161
2G-KP-O, "hifi"		
Hifi and Roucairol (2001) (250 Mhz)	0.00	1.253
PS, $c_4^2c_4^3$, TL 1s	0.00	0.003
2GH-KP-O, "hifi"		
Hifi and Roucairol (2001) (250 Mhz)	0.00	1.145
PS, $c_4^2c_4^3$, TL 1s	0.00	0.002
2GV-KP-O, "hifi"		
Hifi and Roucairol (2001) (250 Mhz)	0.00	1.147
PS, $c_4^2c_4^3$, TL 1s	0.00	0.005

Table 2: Results on Knapsack Problems

Article / Parameters	Gap	Time (s)
G-SPP-O, "kroger1995"		
Wei et al. (2014)	0.27	22.67
PS, 3NEGH-SPP-O, $c_0^2 c_0^3 c_0^4$, TL 30s	3.65	10.416
G-SPP-O, "hopper2001"		
Wei et al. (2014)	0.00	6.267
PS, 3NEGH-SPP-O, $c_0^2 c_0^3 c_0^4$, TL 30s	6.75	4.364
G-SPP-O, "hopper2000"		
Wei et al. (2014)	0.00	20.647
PS, 3NEGH-SPP-O, $c_0^2 c_0^3 c_0^4$, TL 30s	8.72	5.899
G-SPP-O, "bwmv"		
Wei et al. (2014)	0.15	17.736
PS, 3NEGH-SPP-O, $c_0^2 c_5^2 c_6^3$, TL 60s	1.10	12.831
G-SPP-R, "kroger1995"		
Cui et al. (2013)	0.00	56
PS, 3NEGH-SPP-R, $c_0^2 c_0^3 c_0^4$, TL 30s	1.84	9.716
G-SPP-R, "hopper2001"		
Wei et al. (2014)	0.00	13.466
3NEGH-SPP-R, $c_0^2 c_0^3 c_0^4$, TL 30s	3.00	4.153
G-SPP-R, "hopper2000"		
Wei et al. (2014)	0.00	13.465
PS, 3NEGH-SPP-R, $c_0^2 c_0^3 c_0^4$, TL 30s	3.30	10.7
G-SPP-R, "bwmv"		
Wei et al. (2014)	0.13	18.253
PS, 3NEGH-SPP-R, $c_0^2 c_5^2 c_6^3$, TL 30s	0.58	12.592
2NEGH-SPP-O, "alvarez2002"		
Cui et al. (2013)	0.02	4.78
PS, $c_4^1 c_4^2 c_4^3$, TL 30s	1.13	3.726
2NEGH-SPP-O, "bwmv"		
Lodi et al. (2004)	0.02	66.71
Cui et al. (2017)	0.13	1.77
PS, $c_0^2 c_5^2 c_6^3$, TL 30s	0.68	0.992
2NEGH-SPP-R, "bwmv"		
Lodi et al. (2004)	7.96	66.71
Cui et al. (2017)	8.08	1.77
PS, $c_0^2 c_5^2 c_6^3$, TL 30s	0.00	1.773

Table 3: Results on Strip Packing Problems

226 *Growth factor of the queue size threshold:*. In Section 3.1, we indicated that we set the growth factor of the
227 queue size threshold to 1.5. The greater the threshold, the better the solutions will be, but the longer MBA*
228 will take to terminate. Furthermore, for Bin Packing and Strip Packing Problems, full solutions are usually
229 found shortly before it terminates. Therefore, by choosing a too large value for the growth factor, we take
230 the risk to reach the time limit having to spend a lot of time with a given threshold without obtaining any
231 solutions from it. On the other hand, if the growth factor is too small, then only small thresholds value will
232 be explored and no good solutions will be found. In our experiments, 1.5 proved to be a good compromise.

233 *Choice of guide functions:*. The effectiveness of MBA* highly relies on the definition of its guide function.
234 For MBA*, the guide function should be relevant to compare two nodes at different stages of the tree.
235 Therefore, the waste-percentage c_0 appears much more relevant than the waste alone for Bin Packing and
236 Strip Packing variants. Guide function c_1 is adapted from c_0 , but it favours solutions containing larger
237 items. This helps to avoid situations where all small items are packed in the first bins and the last bins
238 get all the large items, creating large waste areas. Guide function c_2 is adapted from c_1 : indeed, even if c_1
239 favors large items first, solutions with no waste at all will always be extracted first, even if they contain only
240 small items. The constant in c_2 aims at fixing this behavior and will lead to better solutions on instances
241 in which optimal solutions contain significant waste (more than 10%). c_3 is adapted from c_2 and favours
242 even more large items first. This guide function is useful for some instances containing several very large
243 items. Finally, c_4 is a natural adaption of c_0 for Knapsack variants. An experimental comparison of several
244 guide functions for the 2018 ROADEF/EURO challenge glass cutting problem is presented in Libralesso and
245 Fontan (2020).

246 *Depth of the symmetry breaking strategy:*. In exact tree search algorithms, it is usually worth breaking
247 symmetries. However, this is not the case when the tree is not meant to be explored completely. For
248 example, consider two symmetrical nodes, the first one normally appearing in the queue, but the second one
249 never being added to the queue because one of its ancestors has been removed to reduce the size of the queue.
250 If the first one is not explored because the symmetry has been detected, then this solution will not be found
251 during the search. How to determine the ideal depth of the symmetry breaking strategy for an instance is
252 not clear yet. The relative size of the items compared to the bin might be an influential factor. For the
253 experiments, we chose 2 or 3 as “standard” values. For some instances containing many items (more than
254 1000), only a value of 4 ensures finding a feasible solution quickly; in contrast, for some knapsack instances
255 with few first-level sub-plates, a value of 1 gives access to better solutions. An experimental evaluation
256 of the influence of the symmetry breaking strategy for the 2018 ROADEF/EURO challenge glass cutting
257 problem is presented in Libralesso and Fontan (2020).

258 *MBA* vs Beam Search.*: Beam Search is another popular tree search algorithm in the packing literature.
259 Beam Search also starts with a queue containing only the root node. However, at each iteration, all nodes
260 of the queue are expanded, and as in MBA*, if the size of the queue goes over a pre-defined threshold, the
261 worst nodes are discarded. Thus, at each iteration, the queue always contains nodes belonging to the same
262 level of the tree. Beam Search seems therefore effective when the guide function is relevant to compare
263 nodes belonging to the same level. This is for example generally not the case in Branch-and-Cut trees where
264 branching consists in fixing a variable to 0 or 1. With our branching scheme for Packing Problems, it is
265 easier to compare such solutions, but the guide functions we presented in Section 3 make it even possible
266 to compare nodes at different levels of the search tree. Thus, Beam Search expands many nodes which are
267 not that much interesting, whereas MBA* always expands only the best current node. An experimental
268 comparison of MBA* and Beam Search for the 2018 ROADEF/EURO challenge glass cutting problem is
269 presented in Libralesso and Fontan (2020). It shows that MBA* finds significantly better solutions than the
270 equivalent Beam Search implementation.

271 *Higher staged guillotine cuts.*: Our branching scheme generates up to three-staged patterns. One could
272 wonder whether it could be possible to adapt it for four-staged or non-staged guillotine patterns. However,
273 if a similar branching scheme seems possible, it may significantly increase symmetry issues. We believe that
274 this would be prohibitive. MBA* might be used to solve these variants, but new branching schemes need to
275 be designed.

276 *Item-based vs block-based.*: Many researchers highlighted the benefits of using block-based approaches, *i.e.*
277 inserting several items at each stage of the tree (Bortfeldt and Jungmann 2012, Wei et al. 2014, Lodi et al.
278 2017). It is interesting to note that it is not what we implemented, yet our algorithm is competitive.

279 **6. Conclusion and future work**

280 We showed that the algorithm proposed by Libralesso and Fontan (2020) for the 2018 ROADEF/EURO
281 challenge glass cutting problem is actually also very competitive compared to other dedicated algorithms
282 for guillotine Packing Problems from the literature, and is even able to return state-of-the-art solutions on
283 several variants. Its performances seem to rely on two key components: a branching scheme which limits
284 symmetry issues; and a tree search algorithm fully exploiting guide functions which make it possible to
285 compare nodes at different levels of the search tree.

286 In addition to effectiveness, the choice of a tree search algorithm makes the algorithm attractive for
287 problems with additional side constraints. Indeed, new constraints are likely to reduce the size of the search
288 tree.

289 The algorithm is implemented in a new software package intended for researchers in Packing Problems
290 to develop new branching schemes for other variants, for researchers in Artificial Intelligence to experiment
291 new tree search algorithms, and for OR practitioners to quickly develop efficient algorithms implementing
292 several business-specific constraints.

293 Future research will focus on developing algorithms for Cutting Stock and Variable-sized Bin Packing
294 Problems, as well as branching schemes to generate another kind of patterns such as non-guillotine or
295 non-staged guillotine ones.

296 L. Libralesso, F. Fontan, An anytime tree search algorithm for the 2018 ROADEF/EURO challenge glass cutting
297 problem, arXiv:2004.00963 [cs] URL <http://arxiv.org/abs/2004.00963>, arXiv: 2004.00963.

298 P. C. Gilmore, R. E. Gomory, Multistage Cutting Stock Problems of Two and More Dimensions, *Operations Research*
299 13 (1) (1965) 94–120, ISSN 0030-364X, doi:10.1287/opre.13.1.94, URL [https://pubsonline.informs.org/
300 doi/abs/10.1287/opre.13.1.94](https://pubsonline.informs.org/doi/abs/10.1287/opre.13.1.94).

301 A. S. Velasco, E. Uchoa, Improved state space relaxation for constrained two-dimensional guillotine cutting problems,
302 *European Journal of Operational Research* 272 (1) (2019) 106–120, ISSN 0377-2217, doi:10.1016/j.ejor.2018.
303 06.016, URL <http://www.sciencedirect.com/science/article/pii/S0377221718305393>.

304 L. Wei, T. Tian, W. Zhu, A. Lim, A block-based layer building approach for the 2D guillotine strip packing problem,
305 *European Journal of Operational Research* 239 (1) (2014) 58–69, ISSN 0377-2217, doi:10.1016/j.ejor.2014.04.
306 020, URL <http://www.sciencedirect.com/science/article/pii/S0377221714003464>.

307 C. Charalambous, K. Fleszar, A constructive bin-oriented heuristic for the two-dimensional bin packing problem with
308 guillotine cuts, *Computers & Operations Research* 38 (10) (2011) 1443–1451, ISSN 0305-0548, doi:10.1016/j.
309 cor.2010.12.013, URL <http://www.sciencedirect.com/science/article/pii/S0305054810003096>.

310 K. Fleszar, Three insertion heuristics and a justification improvement heuristic for two-dimensional bin packing with
311 guillotine cuts, *Computers & Operations Research* 40 (1) (2013) 463–474, ISSN 0305-0548, doi:10.1016/j.cor.
312 2012.07.016, URL <http://www.sciencedirect.com/science/article/pii/S0305054812001621>.

313 Y.-P. Cui, Y. Yao, D. Zhang, Applying triple-block patterns in solving the two-dimensional bin packing problem, *Journal of the Operational Research Society* 69 (3) (2018) 402–415, ISSN 0160-5682, doi:10.1057/s41274-016-0148-5,
314 URL <https://orsociety.tandfonline.com/doi/abs/10.1057/s41274-016-0148-5>,
315

316 A. Lodi, M. Monaci, Integer linear programming models for 2-staged two-dimensional Knapsack problems, *Mathematical Programming* 94 (2) (2003) 257–278, ISSN 1436-4646, doi:10.1007/s10107-002-0319-9, URL <https://doi.org/10.1007/s10107-002-0319-9>.
317
318

319 O. X. do Nascimento, T. A. de Queiroz, L. Junqueira, A MIP-CP based approach for two- and three-dimensional
320 cutting problems with staged guillotine cuts, *Annals of Operations Research* ISSN 1572-9338, doi:10.1007/
321 s10479-019-03466-x, URL <https://doi.org/10.1007/s10479-019-03466-x>.

322 A. Bortfeldt, T. Winter, A genetic algorithm for the two-dimensional knapsack problem with rectangular pieces, *International Transactions in Operational Research* 16 (6) (2009) 685–713, ISSN 1475-3995, doi:10.1111/j.1475-3995.
323 2009.00701.x, URL <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1475-3995.2009.00701.x>.
324

- 325 F. Alvelos, T. M. Chan, P. Vilaça, T. Gomes, E. Silva, J. M. V. d. Carvalho, Sequence based heuristics for two-
326 dimensional bin packing problems, *Engineering Optimization* 41 (8) (2009) 773–791, ISSN 0305-215X, doi:
327 10.1080/03052150902835960, URL <https://doi.org/10.1080/03052150902835960>.
- 328 E. Silva, F. Alvelos, J. M. Valério de Carvalho, An integer programming model for two- and three-stage two-
329 dimensional cutting stock problems, *European Journal of Operational Research* 205 (3) (2010) 699–708,
330 ISSN 0377-2217, doi:10.1016/j.ejor.2010.01.039, URL [http://www.sciencedirect.com/science/article/
331 pii/S0377221710000731](http://www.sciencedirect.com/science/article/pii/S0377221710000731).
- 332 F. Furini, E. Malaguti, D. Thomopulos, Modeling Two-Dimensional Guillotine Cutting Problems via Integer Program-
333 ming, *INFORMS Journal on Computing* 28 (4) (2016) 736–751, ISSN 1091-9856, doi:10.1287/ijoc.2016.0710,
334 URL <https://pubsonline.informs.org/doi/10.1287/ijoc.2016.0710>.
- 335 A. Lodi, S. Martello, D. Vigo, Models and Bounds for Two-Dimensional Level Packing Problems, *Journal of Combi-
336 natorial Optimization* 8 (3) (2004) 363–379, ISSN 1573-2886, doi:10.1023/B:JOCO.0000038915.62826.79, URL
337 <https://doi.org/10.1023/B:JOCO.0000038915.62826.79>.
- 338 N. Nepomuceno, P. Pinheiro, A. L. V. Coelho, A Hybrid Optimization Framework for Cutting and Packing Prob-
339 lems, in: C. Cotta, J. van Hemert (Eds.), *Recent Advances in Evolutionary Computation for Combinatorial
340 Optimization*, Studies in Computational Intelligence, Springer, Berlin, Heidelberg, ISBN 978-3-540-70807-0,
341 87–99, doi:10.1007/978-3-540-70807-0_6, URL https://doi.org/10.1007/978-3-540-70807-0_6, 2008.
- 342 H. Akeb, M. Hifi, R. M’Hallah, A beam search algorithm for the circular packing problem, *Computers &
343 Operations Research* 36 (5) (2009) 1513–1528, ISSN 0305-0548, doi:10.1016/j.cor.2008.02.003, URL [http:
344 //www.sciencedirect.com/science/article/pii/S0305054808000269](http://www.sciencedirect.com/science/article/pii/S0305054808000269).
- 345 M. Hifi, R. M’Hallah, Beam search and non-linear programming tools for the circular packing problem, *International
346 Journal of Mathematics in Operational Research* 1 (4) (2009) 476–503, ISSN 1757-5850, doi:10.1504/IJMOR.
347 2009.026278, URL [https://www.inderscienceonline.com/doi/abs/10.1504/IJMOR.
2009.026278](https://www.inderscienceonline.com/doi/abs/10.1504/IJMOR.2009.026278).
- 348 H. Akeb, M. Hifi, R. M’Hallah, Adaptive beam search lookahead algorithms for the circular packing problem, *Internation-
349 al Transactions in Operational Research* 17 (5) (2010) 553–575, ISSN 1475-3995, doi:10.1111/j.1475-3995.
350 2009.00745.x, URL <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1475-3995.2009.00745.x>.
- 351 H. Akeb, M. Hifi, S. Negre, An augmented beam search-based algorithm for the circular open dimension problem,
352 *Computers & Industrial Engineering* 61 (2) (2011) 373–381, ISSN 0360-8352, doi:10.1016/j.cie.2011.02.009,
353 URL <http://www.sciencedirect.com/science/article/pii/S0360835211000647>.
- 354 J. A. Bennell, X. Song, A beam search implementation for the irregular shape packing problem, *Journal of Heuris-
355 tics* 16 (2) (2010) 167–188, ISSN 1572-9397, doi:10.1007/s10732-008-9095-x, URL [https://doi.org/10.1007/
356 s10732-008-9095-x](https://doi.org/10.1007/s10732-008-9095-x).
- 357 J. A. Bennell, M. Cabo, A. Martínez-Sykora, A beam search approach to solve the convex irregular bin
358 packing problem with guillotine cuts, *European Journal of Operational Research* 270 (1) (2018) 89–102,
359 ISSN 0377-2217, doi:10.1016/j.ejor.2018.03.029, URL [http://www.sciencedirect.com/science/article/
360 pii/S0377221718302571](http://www.sciencedirect.com/science/article/pii/S0377221718302571).

- 361 N. Wang, A. Lim, W. Zhu, A multi-round partial beam search approach for the single container loading
362 problem with shipment priority, *International Journal of Production Economics* 145 (2) (2013) 531–540,
363 ISSN 0925-5273, doi:10.1016/j.ijpe.2013.04.028, URL [http://www.sciencedirect.com/science/article/
364 pii/S0925527313001928](http://www.sciencedirect.com/science/article/pii/S0925527313001928).
- 365 I. Araya, M. C. Riff, A beam search approach to the container loading problem, *Computers & Operations Re-*
366 *search* 43 (2014) 100–107, ISSN 0305-0548, doi:10.1016/j.cor.2013.09.003, URL [http://www.sciencedirect.
367 com/science/article/pii/S0305054813002530](http://www.sciencedirect.com/science/article/pii/S0305054813002530).
- 368 I. Araya, M. Moyano, C. Sanchez, A beam search algorithm for the biobjective container loading problem, *Eu-*
369 *ropean Journal of Operational Research* ISSN 0377-2217, doi:10.1016/j.ejor.2020.03.040, URL [http://www.
370 sciencedirect.com/science/article/pii/S037722172030254X](http://www.sciencedirect.com/science/article/pii/S037722172030254X).
- 371 M. Hifi, S. Negre, R. Ouafi, T. Saadi, A parallel algorithm for constrained two-staged two-dimensional cutting
372 problems, *Computers & Industrial Engineering* 62 (1) (2012) 177–189, ISSN 0360-8352, doi:10.1016/j.cie.2011.
373 09.005, URL <http://www.sciencedirect.com/science/article/pii/S0360835211002737>.
- 374 N. Christofides, C. Whitlock, An Algorithm for Two-Dimensional Cutting Problems, *Operations Research* 25 (1)
375 (1977) 30–44, ISSN 0030-364X, doi:10.1287/opre.25.1.30, URL [https://pubsonline.informs.org/doi/10.
376 1287/opre.25.1.30](https://pubsonline.informs.org/doi/10.1287/opre.25.1.30).
- 377 P. Y. Wang, Two Algorithms for Constrained Two-Dimensional Cutting Stock Problems, *Operations Research* 31 (3)
378 (1983) 573–586, ISSN 0030-364X, doi:10.1287/opre.31.3.573, URL [https://pubsonline.informs.org/doi/
379 abs/10.1287/opre.31.3.573](https://pubsonline.informs.org/doi/abs/10.1287/opre.31.3.573).
- 380 J. Oliveira, J. Ferreira, An improved version of Wang’s algorithm for two-dimensional cutting problems, *European*
381 *Journal of Operational Research* 44 (2) (1990) 256–266, ISSN 0377-2217, doi:10.1016/0377-2217(90)90361-E,
382 URL <http://www.sciencedirect.com/science/article/pii/037722179090361E>.
- 383 S. Tschöke, N. Holthöfer, A new parallel approach to the constrained two-dimensional cutting stock problem, in:
384 A. Ferreira, J. Rolim (Eds.), *Parallel Algorithms for Irregularly Structured Problems*, Lecture Notes in Com-
385 *puter Science*, Springer, Berlin, Heidelberg, ISBN 978-3-540-44915-7, 285–300, doi:10.1007/3-540-60321-2_24,
386 1995.
- 387 S. P. Fekete, J. Schepers, A new exact algorithm for general orthogonal d-dimensional knapsack problems, in:
388 R. Burkard, G. Woeginger (Eds.), *Algorithms — ESA ’97*, Lecture Notes in Computer Science, Springer,
389 Berlin, Heidelberg, ISBN 978-3-540-69536-3, 144–156, doi:10.1007/3-540-63397-9_12, 1997.
- 390 D. Fayard, M. Hifi, V. Zissimopoulos, An efficient approach for large-scale two-dimensional guillotine cutting stock
391 problems, *Journal of the Operational Research Society* 49 (12) (1998) 1270–1277, ISSN 0160-5682, doi:10.1057/
392 palgrave.jors.2600638, URL <https://doi.org/10.1057/palgrave.jors.2600638>.
- 393 M. Hifi, An improvement of viswanathan and bagchi’s exact algorithm for constrained two-dimensional cutting stock,
394 *Computers & Operations Research* 24 (8) (1997) 727–736, ISSN 0305-0548, doi:10.1016/S0305-0548(96)00095-0,
395 URL <http://www.sciencedirect.com/science/article/pii/S0305054896000950>.
- 396 V.-D. Cung, M. Hifi, B. L. Cun, Constrained two-dimensional cutting stock problems a best-first branch-and-bound

397 algorithm, *International Transactions in Operational Research* 7 (3) (2000) 185–210, ISSN 1475-3995, doi:10.
 398 1111/j.1475-3995.2000.tb00194.x, URL [https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1475-3995.](https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1475-3995.2000.tb00194.x)
 399 2000.tb00194.x.

400 J. O. Berkey, P. Y. Wang, Two-Dimensional Finite Bin-Packing Algorithms, *Journal of the Operational Research*
 401 *Society* 38 (5) (1987) 423–429, ISSN 1476-9360, doi:10.1057/jors.1987.70, URL [https://doi.org/10.1057/](https://doi.org/10.1057/jors.1987.70)
 402 [jors.1987.70](https://doi.org/10.1057/jors.1987.70).

403 S. Martello, D. Vigo, Exact Solution of the Two-Dimensional Finite Bin Packing Problem, *Management Science*
 404 44 (3) (1998) 388–399, ISSN 0025-1909, doi:10.1287/mnsc.44.3.388, URL [https://pubsonline.informs.org/](https://pubsonline.informs.org/doi/abs/10.1287/mnsc.44.3.388)
 405 [doi/abs/10.1287/mnsc.44.3.388](https://pubsonline.informs.org/doi/abs/10.1287/mnsc.44.3.388).

406 J. E. Beasley, Algorithms for Unconstrained Two-Dimensional Guillotine Cutting, *Journal of the Operational Re-*
 407 *search Society* 36 (4) (1985) 297–306, ISSN 1476-9360, doi:10.1057/jors.1985.51, URL [https://doi.org/10.](https://doi.org/10.1057/jors.1985.51)
 408 [1057/jors.1985.51](https://doi.org/10.1057/jors.1985.51).

409 B. Kröger, Guillotineable bin packing: A genetic approach, *European Journal of Operational Research* 84 (3)
 410 (1995) 645–661, ISSN 0377-2217, doi:10.1016/0377-2217(95)00029-P, URL [http://www.sciencedirect.com/](http://www.sciencedirect.com/science/article/pii/037722179500029P)
 411 [science/article/pii/037722179500029P](http://www.sciencedirect.com/science/article/pii/037722179500029P).

412 E. Hopper, Two-dimensional packing utilising evolutionary algorithms and other meta-heuristic methods, PhD Thesis,
 413 University of Wales. Cardiff, 2000.

414 E. Hopper, B. C. H. Turton, An empirical investigation of meta-heuristic and heuristic algorithms for a 2D pack-
 415 ing problem, *European Journal of Operational Research* 128 (1) (2001) 34–57, ISSN 0377-2217, doi:10.1016/
 416 S0377-2217(99)00357-4, URL <http://www.sciencedirect.com/science/article/pii/S0377221799003574>.

417 R. Alvarez-Valdés, A. Parajón, J. M. Tamarit, A tabu search algorithm for large-scale guillotine (un)constrained
 418 two-dimensional cutting problems, *Computers & Operations Research* 29 (7) (2002) 925–947, ISSN
 419 0305-0548, doi:10.1016/S0305-0548(00)00095-2, URL [http://www.sciencedirect.com/science/article/](http://www.sciencedirect.com/science/article/pii/S0305054800000952)
 420 [pii/S0305054800000952](http://www.sciencedirect.com/science/article/pii/S0305054800000952).

421 R. Morabito, V. Pureza, A heuristic approach based on dynamic programming and and/or-graph search for the
 422 constrained two-dimensional guillotine cutting problem, *Annals of Operations Research* 179 (1) (2010) 297–
 423 315, ISSN 1572-9338, doi:10.1007/s10479-008-0457-4, URL <https://doi.org/10.1007/s10479-008-0457-4>.

424 Y. Cui, Z. Zhao, Heuristic for the rectangular two-dimensional single stock size cutting stock problem with two-staged
 425 patterns, *European Journal of Operational Research* 231 (2) (2013) 288–298, ISSN 0377-2217, doi:10.1016/j.
 426 ejor.2013.05.042, URL <http://www.sciencedirect.com/science/article/pii/S0377221713004591>.

427 Y. Cui, Y. Yao, Y.-P. Cui, Hybrid approach for the two-dimensional bin packing problem with two-staged patterns,
 428 *International Transactions in Operational Research* 23 (3) (2016) 539–549, ISSN 1475-3995, doi:10.1111/itor.
 429 12188, URL <https://onlinelibrary.wiley.com/doi/abs/10.1111/itor.12188>.

430 J. Puchinger, G. R. Raidl, Models and algorithms for three-stage two-dimensional bin packing, *European Journal of*
 431 *Operational Research* 183 (3) (2007) 1304–1327, ISSN 0377-2217, doi:10.1016/j.ejor.2005.11.064, URL [http:](http://www.sciencedirect.com/science/article/pii/S0377221706003067)
 432 [//www.sciencedirect.com/science/article/pii/S0377221706003067](http://www.sciencedirect.com/science/article/pii/S0377221706003067).

- 433 F. Alvelos, E. Silva, J. M. V. de Carvalho, A Hybrid Heuristic Based on Column Generation for Two- and Three-
434 Stage Bin Packing Problems, in: B. Murgante, S. Misra, A. M. A. C. Rocha, C. Torre, J. G. Rocha, M. I.
435 Falcão, D. Taniar, B. O. Apduhan, O. Gervasi (Eds.), Computational Science and Its Applications – ICCSA
436 2014, Lecture Notes in Computer Science, Springer International Publishing, Cham, ISBN 978-3-319-09129-7,
437 211–226, doi:10.1007/978-3-319-09129-7_16, 2014.
- 438 Y.-P. Cui, Y. Cui, T. Tang, W. Hu, Heuristic for constrained two-dimensional three-staged patterns, Journal of the
439 Operational Research Society 66 (4) (2015) 647–656, ISSN 0160-5682, doi:10.1057/jors.2014.33, URL <https://orsociety.tandfonline.com/doi/abs/10.1057/jors.2014.33>.
- 441 R. Alvarez-Valdes, R. Martí, J. M. Tamarit, A. Parajón, GRASP and Path Relinking for the Two-Dimensional
442 Two-Stage Cutting-Stock Problem, INFORMS Journal on Computing 19 (2) (2007) 261–272, ISSN 1091-9856,
443 doi:10.1287/ijoc.1050.0169, URL <https://pubsonline.informs.org/doi/abs/10.1287/ijoc.1050.0169>.
- 444 M. Hifi, R. M'Hallah, T. Saadi, Algorithms for the Constrained Two-Staged Two-Dimensional Cutting Problem,
445 INFORMS Journal on Computing 20 (2) (2008) 212–221, ISSN 1091-9856, doi:10.1287/ijoc.1070.0233, URL
446 <https://pubsonline.informs.org/doi/10.1287/ijoc.1070.0233>.
- 447 M. Hifi, C. Roucairol, Approximate and Exact Algorithms for Constrained (Un) Weighted Two-dimensional Two-
448 staged Cutting Stock Problems, Journal of Combinatorial Optimization 5 (4) (2001) 465–494, ISSN 1573-2886,
449 doi:10.1023/A:1011628809603, URL <https://doi.org/10.1023/A:1011628809603>.
- 450 L. Wei, A. Lim, A bidirectional building approach for the 2D constrained guillotine knapsack packing problem,
451 European Journal of Operational Research 242 (1) (2015) 63–71, ISSN 0377-2217, doi:10.1016/j.ejor.2014.10.
452 004, URL <http://www.sciencedirect.com/science/article/pii/S037722171400808X>.
- 453 M. Dolatabadi, A. Lodi, M. Monaci, Exact algorithms for the two-dimensional guillotine knapsack, Computers &
454 Operations Research 39 (1) (2012) 48–53, ISSN 0305-0548, doi:10.1016/j.cor.2010.12.018, URL <http://www.sciencedirect.com/science/article/pii/S0305054811000190>.
- 456 Y. Cui, T. Gu, W. Hu, An algorithm for the constrained two-dimensional rectangular multiple identical large object
457 placement problem, Optimization Methods and Software 23 (3) (2008) 375–393, ISSN 1055-6788, doi:10.1080/
458 10556780701617163, URL <https://doi.org/10.1080/10556780701617163>.
- 459 Y.-P. Cui, Y. Zhou, Y. Cui, Triple-solution approach for the strip packing problem with two-staged patterns, Journal
460 of Combinatorial Optimization 34 (2) (2017) 588–604, ISSN 1573-2886, doi:10.1007/s10878-016-0088-7, URL
461 <https://doi.org/10.1007/s10878-016-0088-7>.
- 462 Y. Cui, L. Yang, Q. Chen, Heuristic for the rectangular strip packing problem with rotation of items, Computers
463 & Operations Research 40 (4) (2013) 1094–1099, ISSN 0305-0548, doi:10.1016/j.cor.2012.11.020, URL <http://www.sciencedirect.com/science/article/pii/S030505481200264X>.
- 465 A. Bortfeldt, S. Jungmann, A tree search algorithm for solving the multi-dimensional strip packing problem with
466 guillotine cutting constraint, Annals of Operations Research 196 (1) (2012) 53–71, ISSN 1572-9338, doi:10.
467 1007/s10479-012-1084-7, URL <https://doi.org/10.1007/s10479-012-1084-7>.
- 468 A. Lodi, M. Monaci, E. Pietrobuoni, Partial enumeration algorithms for Two-Dimensional Bin Packing Problem with

469 guillotine constraints, *Discrete Applied Mathematics* 217 (2017) 40–47, ISSN 0166-218X, doi:10.1016/j.dam.
470 2015.09.012, URL <http://www.sciencedirect.com/science/article/pii/S0166218X15004734>.