International Journal of Business and Management Study – IJBMS
Volume 2 : Issue 2 [ISSN : 2372-3955]
Publication Date: 19 October, 2015

A constructive heuristic for the two-dimensional bin packing problem with guillotine cuts and stacked boards

Johan Oppen, Eivind Bale and Torbjørn Hjelden

***Abstract*—We present a description and a solution algorithm for a new variant of the well-known two-dimensional bin packing problem with guillotine cuts. The problem is taken from the furniture industry, where boards are cut into parts to build cabinets, tables, shelves, etc. One of course wants to make cutting patterns which minimize the amount of waste, but it is also important to save operating time on the saw. This can be done by utilizing the maximum cutting height on the saw by cutting a stack of boards, all with the same cutting pattern, simultaneously. This means that in addition to minimizing waste, one seeks to minimize the number of stacks of boards to be cut by repeating the same cutting patterns a given number of times corresponding to the maximum height of a stack, which again depends on the maximum cutting height of the saw and the thickness of the boards. Computational testing shows that significant amounts of time can be saved without increasing the amount of waste.**

***Keywords*—Bin Packing, combinatorial optimization, heuristics**

I. Problem description and literature overview

Cutting and packing problems have received a lot of attention from reseachers during the last decades, and problem variants are found in many different industries. An early typology can be found in [2], this is improved and extended in [6]. In this paper, we consider a problem from the furniture industry, where parts for tables, shelves, cabinets, etc, are cut from boards of given sizes. According to the typology provided by [6], this problem should be classified as a Cutting Stock Problem if the small items are weakly heterogeneous, and as a Bin Packing Problem if the small items are strongly heterogeneous. The product mix at Grande Fabrikker in Innfjorden, Norway, where the problem presented here is taken from, is a mix of standard products produced in larger batches, and custom made products where the number of equally sized items typically is quite small. This means that the heterogeneity of small items varies quite much from one cutting job to the next, and we have chosen to deal with the problem as a variant of the two-dimensional Bin Packing Problem. The most recent survey papers we have found for two-dimensional packing problems are [3] and [5]. In both of these surveys, the focus is on problems where items cannot be rotated due to patterns, wood direction or other properties of the material which requires a fixed orientation. Although this property is present for some of the materials used by Grande Fabrikker, we have chosen to simplify the problem and allow for rotation. It would require only a very small adjustment of the algorithm to deal with problems where rotation is not allowed.

Most practical cutting problems have to be solved by applying only so-called *guillotine cuts*, which means that the cutting has to be done by performing a series of edge-to-edge cuts. This is typically done by first cutting the board into many strips, each of these are then cut into smaller items. This is the case also for the problem variant presented here.

In addition to physical materials, the sawing process also requires labor resources. The operator has to place boards onto the saw table, run the cutting program, reposition items to be cut again, and remove the finished items from the saw table before the next sawing operation is started. Depending on the number of cuts and repositioning operations needed, it takes between five and twenty minutes to go through this process. Multiple boards can be cut simultaneously as long as the height of the stack of boards does not exceed the distance from the table to the top of the saw blade. This maximum stack height is 125 mm for the saw currently used at Grande, which means that, for example, up to seven 16 mm boards, or six 19 mm boards, can be cut in the same operation. It takes the same amount of time to saw one board and a stack of full height, so it is obvious that time can be saved if cutting patterns are repeated in multiples of the maximum stacking height for the given board thickness. This aspect is what we focus on here, we want to find out if it is possible to generate cutting patterns to minimize the number of stacks or cutting operations without using more raw materials.

The commercial software currently used at Grande to generate cutting patterns can minimize waste or minimize the number of different cutting patterns used, but it lacks the possibility to generate cutting patterns so as to minimize the number of cutting operations based on a given stacking height.

To the best of our knowledge, the problem presented here has not been addressed in the literature earlier. This is a typical example of a practical problem where well-studied problems have to be extended or adjusted to fit to planning problems found in the real world..

Johan Oppen
Møreforsking Molde
Norway

Eivind Bale, Torbjørn Hjelden
Grande Fabrikker AS
Norway

II. A constructive heuristic

Heuristic algorithms are widely used to solve variants of the two-dimensional bin packing problem, exact approaches are still not able to provide solutions for most instances found in the real world. This is quite typical for many optimization problems, as practical problems are often large and require extensions and adjustments of known models.

Our constructive heuristic is based on the *Finite First F*it algorithm, which is described in [4]. This algorithm is a so-called *one phase* algorithm which packs items directly into bins, without going via a strip packing phase. Our goal is not to find new best solutions for standard problem instances, but, in order to be able to trust the results, we need an algorithm which performs well.

As our algorithm allows items to be rotated, a choice about orientation always has to be made when a new *shelf* is opened. A shelf is a part or stripe of a bin, occupying a portion of the height and the full width of the bin.

We deal with the stack height by reducing the number of items to pack. For every item size, we divide the number of items to cut by the stack height, and round up if needed. This means that if, for example, 48 items of a certain size are needed and the stack height is six, we consider eight items of this size. If we need 50, we have to consider nine items to get enough, we then cut four more than we actually need. This is done for all item sizes in the cutting list/instance. When all cutting patterns are cut with full stack height, the number of items actually produced will cover at least the number of items needed. This procedure also reduces the instance size and in practice speeds up the algorithm.

We also need to consider the fact that the last bin may not be very well utilized, and that because all bins/cutting patterns are repeated several times, we may be able to reduce the number of boards used by packing the items from the last bin onto fewer boards than the full stack height. As an example, consider a problem with stack height six, meaning that all bins/cutting patterns are repeated six times, and where the last bin contains ten items of two different sizes, five items of each size. Cutting this pattern as a stack of six boards gives 60 items, 30 of each. If we can pack these items onto five boards instead of six, we would save one board without increasing the number of cutting operations. To see if this is possible, we try to pack 5x6/5=6 items of each size into one bin, which still corresponds to the number of items needed. If this can be done, we try to reduce the number of boards used to four, and so on, until we cannot pack the items into one bin.

We also need to take into account any surplus of items due to the fact that the original number of items may not be divisible by the stack height. If we, for example, need 500 of a given item size and the stack height is six, we need to pack 84 of this item. This corresponds to 504 items, so we have a surplus of four items. Assume we try to reduce the number of boards cut in the last stack according to the example given in the previous paragraph, and that we have a surplus of four of one of the item sizes, meaning that we actually need only 26 of the 30 items. We then still need to pack six items with a stack height of five, but we would need to pack only seven, not eight, items with a stack height of four, as this would give us 28 items.

Our algorithm proceeds in the following way:

- The stack height is computed, and the number of each item is divided by the stack height. If the resulting number of each item size is not an integer, the number is rounded up and the surplus is registered.
- All items are sorted by non-increasing length of the shortest edge.
- Pack every item into the first bin that can accommodate it, or on the bottom left of a new one if no such bin exists. If the item is packed into an already opened bin, it is packed onto the first shelf that can accommodate it, or to the leftmost side of a new one if no such shelf exists.
- Whenever a new shelf is opened, it has to be decided if the first item should be packed vertically or horizontally.
    - In the first run, we always pack the first item on a new shelf vertically.
    - In the second run, the first item on all shelves is packed horizontally.
    - The algorithm is then run 1000 times where vertical or horizontal packing of the first item on a shelf is chosen randomly.
- When an item is packed onto an already opened shelf, it is packed vertically if the longest edge does not exceed the height of the shelf, otherwise it is packed horizontally.
- When all items are packed, try to reduce the stack height of the last bin, taking any surplus into consideration, as explained earlier.
- Return the solution using the smallest number of boards.

III. Computational experiments

The algorithm is coded in c++, and the computational experiments are run on a 2.10 GHz HP EliteBook laptop with 8 GB of RAM, running Windows 7 Enterprise.

The purpose of this paper is to show that a substantial amount of time can be saved by making cutting patterns to allow for full stacks to be cut simultaneously, and that this can be done without using more raw materials. We have, however, also solved the 500 benchmark instances referred to in [1] to check that the algorithm performs decently. In this paper, it is reported that seven of the best known heuristics and metaheuristics use in total between 7064 and 7367 bins to pack all items when rotation is allowed. Our heuristic used 7205 bins in total, so we claim that the algorithm performs sufficiently well to give reliable results.

Our main computational test is performed on a set of real-world instances provided by Grande Fabrikker AS. We use ten different cutting lists with three to thirteen item sizes and a total number of items ranging from 48 to 4148. The

Tabell 1 Results from computational testing

				No stacking		Stacking	
Inst	Height	Sizes	Items	Boards	Stacks	Boards	Stacks
1	6	3	768	29	8	28	5
2	6	8	1172	70	22	70	12
3	6	3	672	25	16	25	5
4	6	3	432	29	8	29	5
5	6	4	576	45	13	44	8
6	6	4	528	19	13	19	4
7	6	13	4148	208	46	208	35
8	7	4	860	34	8	34	5
9	6	10	1511	88	25	89	15
10	6	5	48	6	4	5	1

number of boards/bins needed varies from five to 208. Detailed results are given in Table 1.

The column named ``Height" gives the stack height for each instance. Nine of the instances have a board thickness of 19mm and thus a stack height of six, one instance is for 16mm items and thus the stack height is seven. The columns named ``Sizes" and ``Items" gives the number of different item sizes and the total number of items, respectively. Then follow the number of boards used and the number of stacks, both if the stack height is ignored and if it is used to minimize the number of stacks.

For three of the instances, the algorithm finds a solution using one less board if stacking is applied, while the opposite happens for one instance. This means that in total, stacking saves two boards. For all instances, we see that the number of stacks and thus the number of cutting operations is significantly lower if we utilize the full stack height when cutting patterns are made. In total, it is possible to save 68 cutting operations or 41.7% of the time spent on cutting if stacking is applied. Observations have shown that it takes between five and twenty minutes to cut a stack of boards (or a single board). Even if we use a conservative estimate of ten minutes, this means that the company would save in total 680 minutes, or more than ten hours, of operation time, when all ten cutting lists were processed.

It takes less than a minute to run all ten instances, so the computational time is satisfactory.

IV. Conclusions

We have presented a new variant of the two-dimensional bin packing problem with guillotine cuts, where cutting patterns are repeated in order to utilize the maximum cutting height of the saw. We have shown that cutting patterns for the full stack height can be generated without using more raw materials, and that the amount of time saved can be substantial.

References

[1] C. Charalambous and K. Flezar, "A constructive bin-oriented heuristic for the two-dimensional bin packing problem with guillotine cuts," Computers & Operations Research, vol. 38, pp. 1443–1451, 2011.

[2] H. Dyckhoff, "A typology of cutting and packing problems," European Journal of Operational Research, vol. 44, pp. 145-159, 1990.

[3] A. Lodi, S. Martello and M. Monaci, "Two-dimensional packing problems: A survey," European Journal of Operational Research, vol. 141, pp. 241-252, 2002..

[4] A. Lodi, S. Martello and D. Vigo, "Heuristic and metaheuristic approach for a class of two-dimensional packing problems," INFORMS Journal on Compouting, vol.11(4), pp. 345-357, 1999.

[5] A. Lodi, S. Martello and D. Vigo, "Recent advances on two-dimensional bin packing problems," Discrete applied mathematics, vol. 123, pp. 379-396, 2002.

[6] G. Wäscher, H. Haussner and H. Schumann, "An improved typology of cutting and packing problems," European Journal of Operational Research, vol. 183, pp. 1109–1130, 2007.

Johan Oppen

"This is a typical example of a practical problem where well-studied problems have to be extended or adjusted to fit to planning problems found in the real world."
Johan Oppen, research scientist.

Eivind Bale

"It takes the same amount of time to saw one board and a stack of full height, so it is obvious that time can be saved."

Eivind Bale, production manager.

Torbjørn Hjelden

"In addition to physical materials, the sawing process also requires labor resources. We want to find out if it is possible to save time without using more raw materials."
Torbjørn Hjelden, general manager.