Korean J. Chem. Eng., **30**(2), 278-285 (2013)
DOI: 10.1007/s11814-012-0125-x

Developing a heuristics for glass cutting process optimization: A case of two-dimensional two-stage guillotine cutting with multiple stock sizes

Kyung Tae Park*, Jun-Hyung Ryu*†, Ho-Kyung Lee**, and In-Beum Lee***

*Department of Chemical Engineering, POSTECH, Pohang 790-784, Korea
**LG Chemistry Ltd., Research Park, Deajeon 305-380, Korea
***Department of Nuclear and Energy System, Dongguk University, Gyeongju 780-714, Korea
(*Received 29 March 2012 • accepted 2 August 2012*)

Abstract−This paper presents a heuristic algorithm for a two-dimensional two-stage guillotine cutting problem with multiple stock sizes by allowing the rotation of items by 90°. The proposed algorithm generates *levels or strips* where the *first item* or *base item* is selected according to the length of the strip and packs the next items beside the base item in the strip. For each type of item, strips are generated for packing each type of item in a base item. The best *n* orders in a yield of strips or the best *n* strips are selected for each type of item. The selected best *n* strips are packed in one type of bin. For the other types of bins, another best *n* strips are selected and packed in each type of bin. The best yield in all types of bins is then selected. This iteration is executed until the number of item demands in the overall demands is less than the number of item demands in the bin. Four numerical examples generated from actual industries are illustrated to highlight the applicability of the proposed algorithm with some comments.

Key words: Glass Production, Two-dimensional Two-stage, Guillotine Cut, Level Packing Problem

INTRODUCTION

Process systems engineering (PSE) principles have been actively implemented in a wide range of industry practices. Nevertheless, frequent communications between academia and industry are needed to take full advantage of the established knowledge in academia and potentially important experiences in industry. This paper reports the implementation experience of an academic based research in actual industry for the case of glass production.

Industrial glass production consists of hollow glass production, such as bottles and glasses, and flat glass production, such as windows and mirrors. This paper deals with the cutting stock problem in flat glass production where large glass *stocks* or *bins* are cut by cutting machines according to a cutting pattern, and small pieces *items* are then produced from those bins. A notable feature of the glass industry is the 'guillotine cut' where glass is cut from one side edge to another. For the guillotine cut, we use a level packing method that generates *levels or strips* where the *first item* or *base item* is the length of the strip and packs next items beside the base item in the strip.

In the literature Wäscher et al. [1] and Dyckhoff [2] presented the topology of the cutting and packing problem and Lodi et al. [3] reviewed the two-dimensional packing problem. Several exact methods based upon mathematical programming [4,5] were reported for the 2-dimensional cutting stock problems. They were not desirable because the computational times involved in using them are long in handling a large number of discrete variables. Many papers were thus concerned with computational time reducing heuristic algorithms for generating good cutting patterns. One of the famous heuristic algorithms is the level packing method [6-8], which creates levels (or strips or shelves) in a sense that the length of the first item is equal to the length of the level and items are packed in the leftmost of the strips: The next item is packed beside the first item in the strip until the width of the strip exceeds the width of the bin, and the created strips are then packed in the bin. Algorithms considering multiple stock sizes that deal with several types of bins have also been presented [9].

Lodi et al. [5] presented the knapsack algorithm using a level packing method and heuristic based Tabu search algorithm. These algorithms showed excellent performance for one type of bin but could not be applied to several distinctive types of bins because the algorithm was used for the 0-1 knapsack problem. Glass cutting problems have demanded several types of items from several types of bins for the glass. Therefore, it is important to develop an algorithm for several types of bins. In many papers, the rotation of items was considered. Previous papers considered the width and length of all oriented and rotated items in the rotation case, but the computational time was long, considering all cases in several types of bins. Not all cases were taken into account, but several cases were considered for the rotation of items with a reduced computational time and preserved yield of the strip. A special feature in this glass cutting is the minimum distance, which is the space to cut glass by glass cutting workers. The minimum distance was also considered. In the glass cutting problem, a 2-stage guillotine cut was only allowed because of the working environment, such as a state of glass and cutting machine. This paper focuses on the 2-dimensional 2-stage guillotine cutting problem using a level packing method in the glass cutting problem. Multiple stock sizes and the rotation of items were also considered.

The aim of the proposed algorithm is to minimize the waste parts in the glass cutting problem by considering item rotation and the

†To whom correspondence should be addressed.
E-mail: jhryu@dongguk.ac.kr

minimum distance. Four numerical examples from actual industrial cases are presented. This paper compares the proposed method with several methods of the strip packing problem in example 1 for the purpose of comparison,. Examples 2, 3 and 4 show the applicability of the proposed algorithm in handling actual industrial cases.

HEURISTIC ALGORITHM

The proposed algorithm consists of two sections, strip generation and overall algorithm. The strip generation produces strips for each type of item and selects the best n order in a yield of strips for each type of item known as *the best n strips*. The overall algorithm assigns the generated strips to all types of bins and selects one bin type generating the best yield. Rotating items and bins in the strip are also allowed in the overall algorithm. The rotation of an item is needed in most glass cutting problems, and the computation time is long in multiple stock sizes if all rotations of items are considered. This section presents a simple rotation method of an item in strip generation. Bin rotation is essential in this algorithm because the arrangement of items in a strip differs according to the width and length of the bin.

This algorithm considers a minimum distance, which is a special constraint of the glass cutting problem. In glass cutting, the cutting machine only cuts the bin into items according to a cutting pattern given by the user. The cutting bin is not automatically divided, so workers decompose the cutting bin into items. In this process, the minimum distance for the operation space that executes glass decomposition by the worker is needed.

1. Strip Generation

This section proposes the strip generation applying a level packing method, which was recently reported by many researchers in the cutting and packing area. The first step of strip generation produces strips for each type of item, and the next step selects the best n strips for each type of item. In the original strip, the first item is packed in the leftmost strip. The first item is called a *base item* as shown in Fig. 1. The length of the base item equals the length of the strip and the next items packed in the strip do not exceed the length of the base item. In the strip, the items are packed until the sum of the item widths exceeds the width of the strip or no item packed remains (Fig. 2). Various strips are generated according to the base item. It is assumed that all types of items are packed in the base item and then originate strips in this paper. This method reduces the computational time because previous methods of packing items

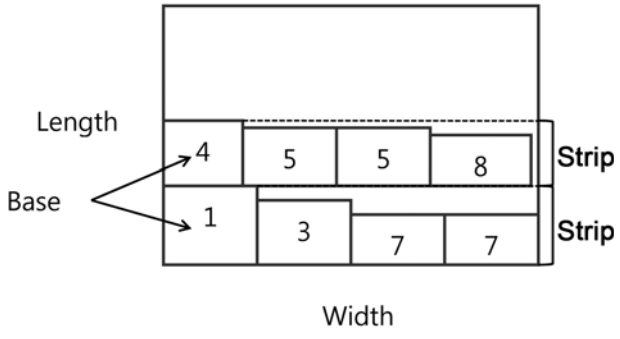

Fig. 1. Base items and strips.

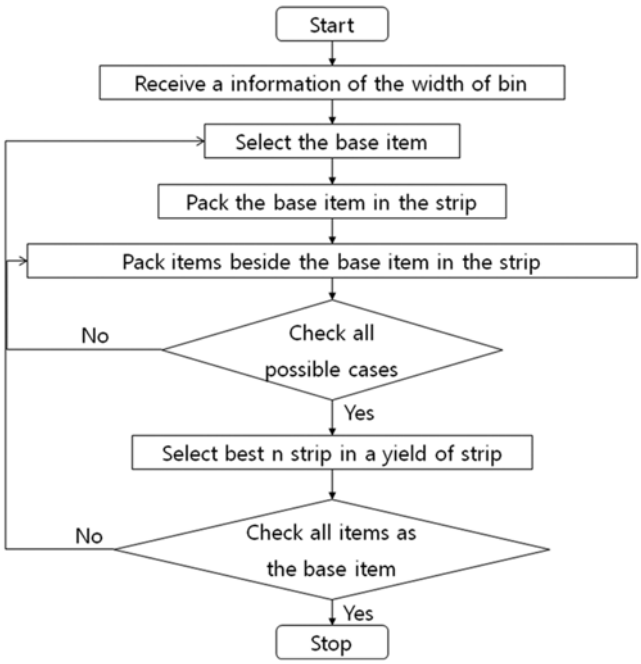

Fig. 2. Strip generation flowchart.

directly in the bin generated an overlapping computation when the strip of the same type in the bin was initiated.

After all types of items are packed in the base item and strips are made, the best n strips are selected for each type of item. One merit (merit 1) of selecting the best n strips gives other options when the demand of the item in the best strip exceeds the number of item demands in the overall demands. For example, the best strip consists of items 5, 6, 8 and 9, and the second best strip consists of items 5, 8, 10 and 12. If the demand of item 6 becomes 0, then the second best strip is used instead of the best strip. In the last part of the heuristic algorithm, a problem is generated when the same type of item only exists and the best strip is only selected. If the same type of item exists in the last part of the heuristic algorithm, the best strip is selected and the best strip is then packed in the bin. Although the present bin packing the best strip retains space for packing the remaining items, the remaining items is not packed in the present bin but packed in the next bin because the best strip for the same type of item only exists and other strips do not exist. For example, the demand for item 5 is 7, and the demands of other items are 0 in the last part. Suppose the best strip for item 5 includes 4 of item 5. The best strip including 4 of item 5 is only packed in the bin but the remaining items for item 5 (here 3 of item 5 remains) are not packed in this bin. This problem can be solved by selecting the best n strips, which is another merit (merit 2). The remaining items can be packed in the best n strips and available strips packed in this bin are increased. In this example, the best strip includes 4 of item 5 and the second best strip includes 3 of item 5. The best strip is packed in the bottom of the bin and the second best strip is then packed above the best strip, as shown (Fig. 3). To solve this problem, it is recommended that n, which is the number of the best strip, equals the number of items packed in the strip in the last part, because the case number is the number of items packed in the strip at the same type of item.

When assigning the items to the strip, the orientation and rota-

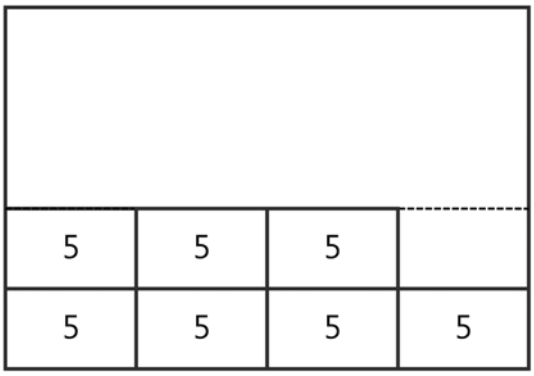

Fig. 3. The best strip and second best strip are packed in the bin.

tion of items can be considered. The orientation of the items packs items oriented in the strip, while the rotation case can pack items rotated by 90° in the strip. Since previous methods consider the width and length of all oriented and rotated items in the rotation case, they require considerable computational time. Here, a simple method is presented for the rotated items. After the base item is determined, this method compares the width with the length of the items when packing the next items approved to rotate, and then packs the long side of the item according to the height of the item in the strip. If the long side of an item exceeds the length of the base item, the short side of the item is then packed. One merit of this method is the reduced computational time because it does not consider all possible cases in packing the rotated items in the strip as previous methods. Nevertheless, the yield of the strip packing items by this method is high because packing the long side of an item as the height of an item in the strip is generally lower than the packing short side for the waste parts shown in Fig. 4.

Rotation of the base item was also considered. If the rotation of items is allowed, the rotation of the base item is also allowed. The types of items packed in the strip are different because the length of the strip is determined by the length of the base item. The best n

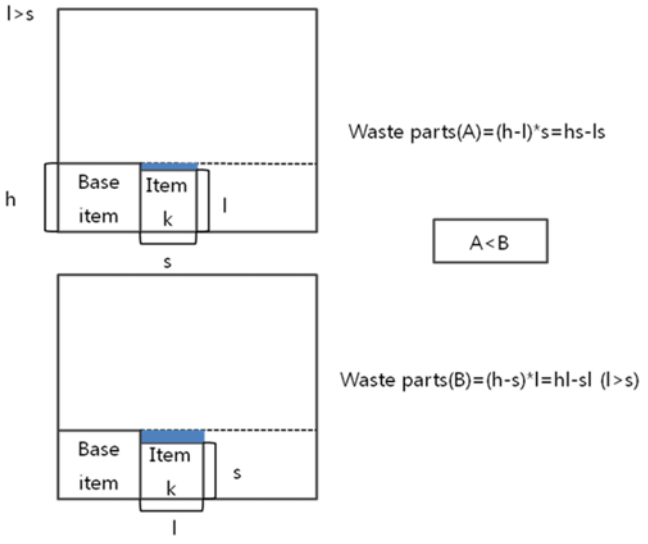

Fig. 4. Loss area in (a) item applying a simple method (b) oriented item.

strips for rotation of the base item as well as the orientation of the base item are generated. Because the possible strips packed in the bins are increased, the yield of this algorithm is increased but the computational item is also increased.

2. Overall Algorithm

To compute the best solution of the 2-dimensional 2-stage guillotine problem allowing the rotation of items in multiple types of bins, the proposed algorithm can be summarized as follows:

1. Arrange the length of required items in descending order. If the rotation of the item is possible, this part is deleted. In the orientation of the item, the computational time is reduced because this algorithm does not consider that the length of the item exceeds the length of the base item. On the other hand, the rotation of the item this algorithm considers this case. Although the length of item exceeds the length of the base item during rotation, this algorithm introduces this item if the width of the item is lower than the length of the base item.

2. Select one type of bin from various bins: strip generation has begun.

3. Generate strips for the type of each item. In the strip, the type of each item is packed in the base item and strips are generated for the type of each item.

4. Select the best n orders in the yield of strips for the type of item (best n strips): strip generation is complete. Here, many strips are generated for each type of item.

5. Pack strips in the bin. The best strip for each type of item is selected and they are packed in the bin. For packing strips in the bins, all the possible cases are considered. If the best n strip packed in the bin for any type of item is impossible because the demand of any item in the best strip exceeds the number of item demands in the overall demands, the second best strip for any type of item can be then packed in the bin.

6. Select the best yield pattern in the bin packing strips.

7. Check the rotation of the bin. If the bin rotation cannot be checked, then the algorithm goes to step 3. Otherwise the algorithm goes to step 8.

8. Check all bin types? If all types of bins cannot be checked, then the algorithm again goes to step 2. Otherwise, the algorithm goes to next step 9.

9. Select one type of bin of the best yield pattern in all types of bins.

10. Create the bin with the best yield pattern until the number of item demands in the overall demands is less than the number of item demands in this bin. To prevent the overlapping computation, the bin pattern of the best yield is produced continuously. After the last production of the best yield bin, the number of item demands in the overall demands is less than the number of item demands in this bin.

11. Stop if all demands for each item are equal to 0. Otherwise, goes to step 2.

Fig. 5 shows the overall flowchart of this heuristic algorithm. This method is simple and gives a good yield.

This paper considers the minimum distance that widely appears in the glass cutting problem. The minimum distance depends on the thickness of the glass; the minimum distance increases with increasing glass thickness. In real glass plants, the minimum distance is generated in a trim loss area. To apply the minimum distance to the

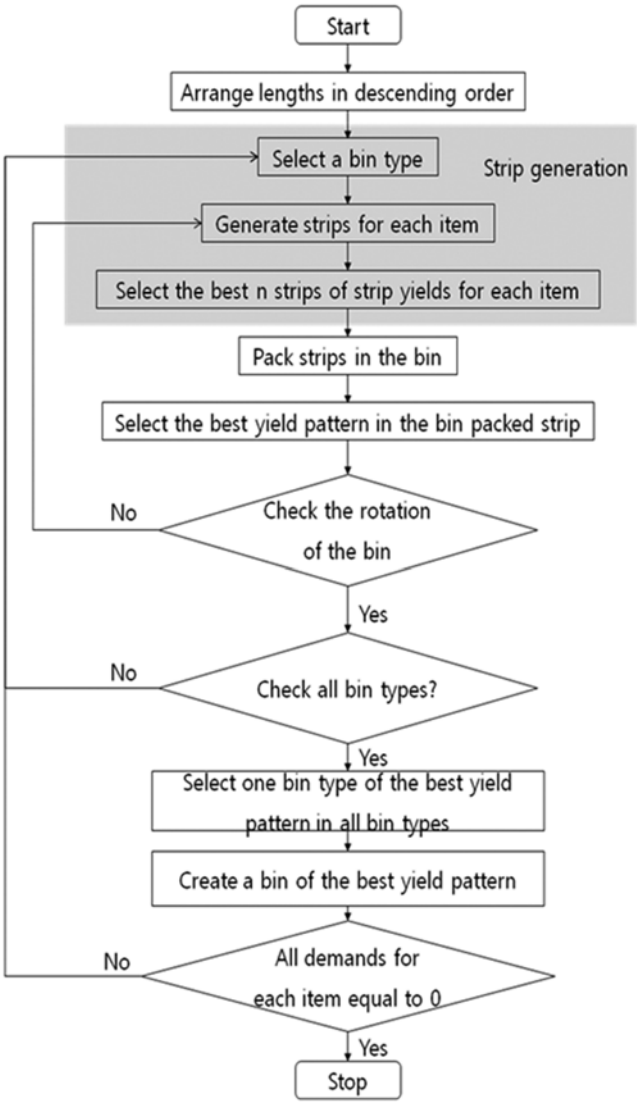

Fig. 5. Heuristic algorithm flowchart in the two dimensional 2 stage guillotine cutting problem.

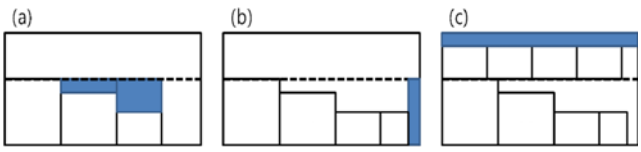

Fig. 6. The minimum distance in (a) the item, (b) the strip, and (c) the bin.

heuristic algorithm, three cases that appear in the item, strip and bin are considered. The minimum distance in the item is used to pack the item in the strip, as shown in Fig. 6(a). The minimum distance in the strip is used to pack items in the strip as shown in Fig. 6(b). The minimum distance in the bin appears in packing strips in the bin, as shown in Fig. 6(c). For the minimum distance, this algorithm adds some constraints that the remaining length of each case exceeds the minimum distance or equals to 0, where workers do not need to eliminate the trim loss area.

The rotation of the bin in this heuristic algorithm can be considered. According to the width in the bin, the components (items) of the strip for each type of item are changed. In real glass plants, the rotation of the real bin is generally not permitted. Although the rotation of the real bin is not allowed in plants, if the vertical and horizontal cuts are permitted, then this algorithm applying the rotation of the bin can be also used in this case. This is because the rotation of the bin means that the vertical cut is executed in the fixed bin, as shown in Fig. 7.

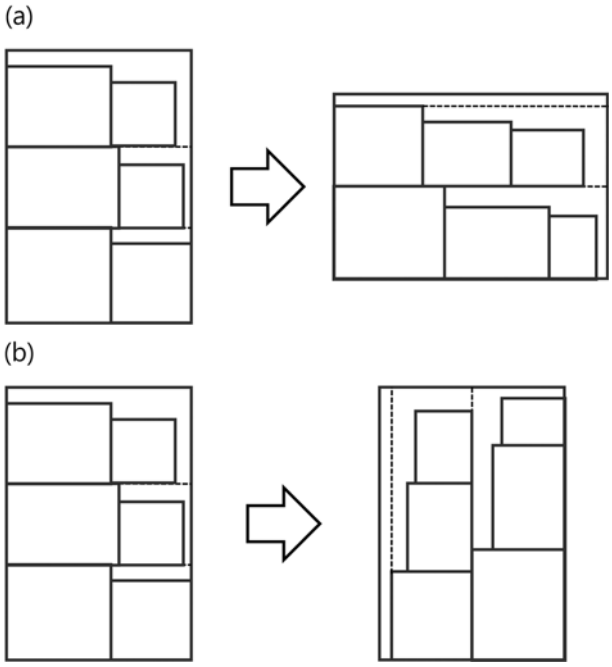

Fig. 7. The best case for (a) rotation of the real bin and (b) allowance of the vertical cut.

COMPUTATIONAL EXPERIMENTS

To illustrate how the proposed algorithm is utilized, numerical examples are prepared. Particularly, numerical examples from actual industrial practice are prepared. Since they represent the typical industrial case, the actual applicability of the proposed algorithm is going to be clearly highlighted. Four numerical examples are prepared. Few studies have presented problems of the same type in a 2-dimensional 2-stage guillotine cut with the minimum distance. Therefore, this paper compares this method with several methods of the strip packing problem [10] in Example 1. Examples 2 and 3 deal with a medium-sized problem on the 2-dimensional 2-stage guillotine cut with the minimum distance. A relatively large-sized problem is addressed in example 4. The solution of all problems was calculated using Excel 2010 on an Intel Core i5 2.67 GHZ with 8 GB of RAM.

1. (Example 1) Compare this Method with the Methods of the Strip Packing Problem

Ntene et al. [10] compared level algorithms for the 2-dimensional oriented strip packing problem. The strip packing problem packs items in a rectangle with a limited width and unlimited length. Here, the method was revised to be applied to the strip packing problem with no rotation and no minimum distance. This example compares this method with the methods of strip packing problems used

Table 1. Dimensions of items of example instance in Ntene et al. [10]

Number	Height	Width
1	6	8
2	16	32
3	20	3
4	24	24
5	4	13
6	4	2
7	6	7
8	16	11
9	4	8
10	6	12
11	3	13
12	3	28

Fig. 8. Comparing this strip packing method with previous methods for the example reported by Ntene et al. [10]. (a) Our strip generation algorithm (b) The knapsack algorithm [5] (c) The next-fit decreasing height algorithm [8] (d) The first-fit decreasing height algorithm [8].

in the example reported by Ntene et al. Table 1 lists dimensions of the items of the example, packed in a width of 40 centimeter and an unlimited length. The objective of the strip packing problem is to minimize the length of the rectangle.

Fig. 8 shows the results of this example to compare this method with previous methods in the strip packing problem using the results reported by Ntene et al. The results from the proposed method were

Table 2. Comparison of the knapsack algorithm [11] with the proposed algorithm

	Knapsack algorithm [11]	Our algorithm
Strip generation	0-1 Knapsack algorithm	Find the best n strips for each item
Bin generation	1 Bin packing (1BP)	Enumeration for packing strips in the bin

Table 3. (a) Types of bins and (b) types and demands of items for example 2

(a)

Type	Length	Width	Area
1	2,398	3,245	7,781,510
2	2,048	2,987	6,117,376

(b)

Type	Length	Width	Area	Demand
1	1,120	1,212	1,357,440	10
2	1,078	1,140	1,228,920	10
3	937	1,183	1,108,471	20
4	937	1,030	965,110	20
5	653	937	611,861	20
6	543	1,023	555,489	20
7	543	937	508,791	40
8	486	943	458,298	20

good and similar to the results from the knapsack problem. Table 2 compares the knapsack algorithm with the proposed method. The proposed algorithm is similar to the knapsack algorithm in a case of a single bin. On the other hand, in the case of multiple bins, it is not possible to employ the knapsack algorithm because the knapsack algorithm is concerned with how to fill identical rooms with multiple items.

2. (Example 2) Small-sized Problem with Two Bins

This example presents the Small-sized problem with two bins for a 2-dimensional 2-stage guillotine cut with the minimum distance and rotation. Table 3 lists dimensions of the bins and items along with the demands of the items. The yield for this problem is 90.83%. The solution time was 2 sec. Fig. 9 shows the cutting pattern of a solution of a problem.

3. (Example 3) Small-sized Problem with Several Bins

In industrial glass production, the number of bins actually used in glass companies is fewer than ten because of the limited machine performance and the limited space for each bin. This example allows use of the maximum ten types of bin and of same types of item described in example 1. Table 4 lists dimensions of the bins and items along with the demands of the items. The yield for this problem is 93.69% and the computation time was 8 sec. Example 3 is five-times more than the type of bin described in example 2 but the computational time is four-times longer and the yield is better.

4. (Example 4) Large Size Problem

This example presents a large-sized problem for 2-dimensional 2-stage guillotine cut with the minimum distance and rotation. Table

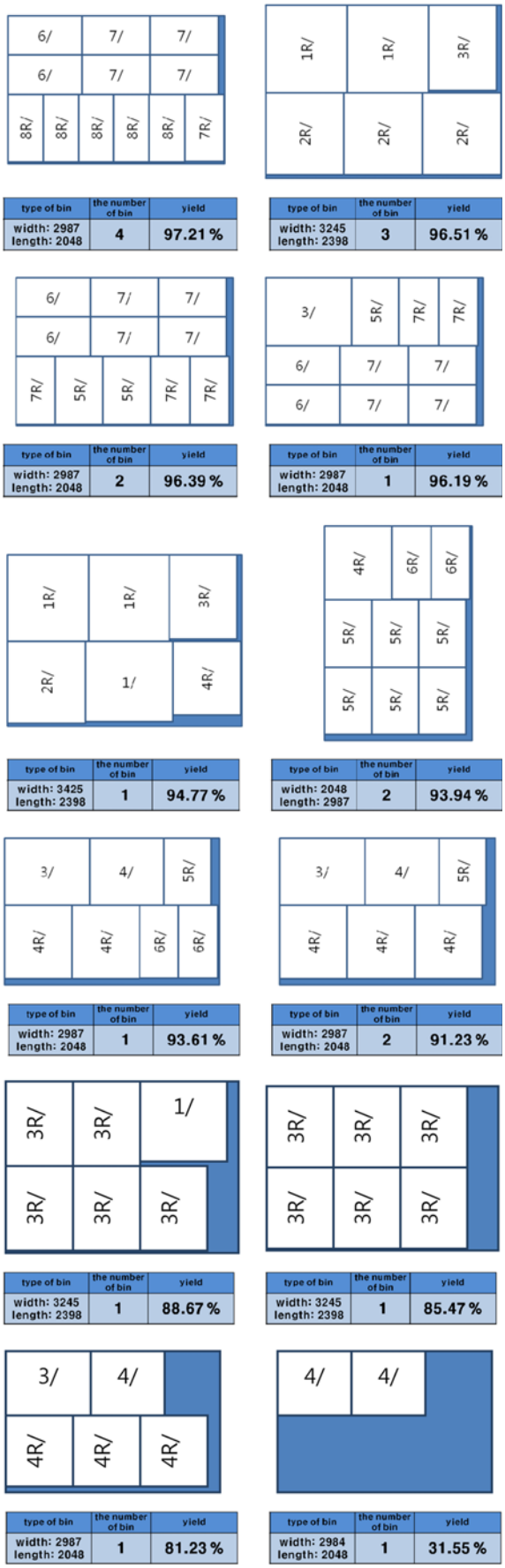

Fig. 9. Cutting patterns for the solution of example 2.

Table 4. (a) Types of bins and (b) types and demands of items for example 3

(a)

Type	Width	Length	Area
1	3,363	2,441	8,209,083
2	3,038	2,241	6,808,158
3	3,349	2,199	7,364,451
4	3,349	2,140	7,166,860
5	3,349	2,048	6,858,752
6	3,038	2,205	6,698,790
7	3,298	1,998	6,589,404
8	3,038	2,140	6,501,320
9	3,363	1,831	6,157,653
10	3,038	1,831	5,562,578

(b)

Type	Width	Length	Area	Demand
1	1,212	1,120	1,357,440	30
2	1,140	1,078	1,228,920	20
3	1,183	937	1,108,471	40
4	1,030	937	965,110	50
5	937	653	611,861	120
6	1,023	543	555,489	85
7	937	543	508,791	75
8	943	486	458,298	100

4 lists the dimensions of the bins and items along with demands of items. The yield for this problem was 96.59%. The solution time was 85 sec. Fig. 10 shows the cutting pattern of a solution for the problem. Example 4 is the same type of bin and same amount of item described in example 3, and two-times more than the type of item, but the computational time is ten-times longer. According to the increasing of the type of item, computational time for generating strips increased rapidly.

In strip generation, several methods are presented, including generating strips for each item, selecting the best n strip in all strips, and considering the rotation of items. The algorithm showed good results when it was applied to the strip packing problem as in example 1. In the overall algorithm, the rotation of the bins and the minimum distance for three cases, such as the item, strip, and bin, was considered.

For small and large problems, such as examples 2, 3 and 4, the result according to the proposed algorithm was good and obtained quickly. This is suitable for case 1 with a few types of items and large demands in terms of computational time, because the recursive computational times are reduced in the packing strips in the bin. On the other hand, for a large number of items with small demands like case 2, it takes relatively a longer computation time than the previous cases because recursive computation was not performed. This would be a good future research topic to expand the current algorithm.

One defect of the proposed algorithm occurred when selecting the bin pattern at several bins of each stage. The algorithm selects the best bin pattern at several bins of each stage, but the result of this algorithm is not always optimal. This defect occurred in almost

Fig. 10. Cutting patterns for the solution of example 4.

Table 5. (a) Types of bins and (b) types and demands of items for example 4

(a)

Type	Width	Length	Area
1	2,441	3,363	8,209,083
2	2,241	3,038	6,808,158
3	2,199	3,349	7,364,451
4	2,140	3,349	7,166,860
5	2,048	3,349	6,858,752
6	2,205	3,038	6,698,790
7	1,998	3,298	6,589,404
8	2,140	3,038	6,501,320
9	1,831	3,363	6,157,653
10	1,831	3,038	5,562,578

(b)

Type	Length	Width	Area	Demand
1	1,112	1,267	1,408,904	10
2	1,101	1,251	1,377,351	20
3	921	1,253	1,154,013	10
4	921	1,251	1,152,171	20
5	670	921	617,070	50
6	651	1,101	716,751	20
7	651	921	599,571	20
8	489	930	454,770	10
9	489	926	452,814	40
10	421	921	387,741	40
11	403	1,112	448,136	20
12	403	921	371,163	20
13	401	1,101	441,501	20
14	401	921	369,321	20

heuristic algorithm. To solve this defect, precise methods for considering all possible patterns are good alternatives, but precise methods are not suitable in this problem, which has many demands of items, because the computational time of solving exact methods is long. Therefore, the development of new heuristic algorithms to reduce this defect or find new exact methods for reducing the computational time using decomposition methods will be needed.

Real glass cutting problems generally allow a 3-stage or *n*-stage guillotine cut in the cutting process. If a 3-stage or n-stage case is applied to this problem, the result would be higher than the 2-stage case. Therefore, it is important to develop an algorithm that can be applied to a 3-stage or n-stage guillotine cut with the rotation of items.

CONCLUSIONS

This paper presents the 2-dimensional 2-stage guillotine cutting problem with multiple stock sizes and rotation of items. Strip generation, such as a generating strip for each item, selecting the best n strip in the yield of the strip, and considering the rotation of an item, was examined, and an overall algorithm, such as the rotation of bins and minimum distance, was also evaluated. By applying the strip generation and overall algorithm to the cutting stock problem, the computational time was reduced and the yield of this result was good. As an extension of the current algorithm, it would be possible to consider the 2-dimensional 3-stage case of the cutting stock problem with the rotation of items and multiple stock sizes.

This paper proposes an algorithm for minimizing the production cost in the flat glass industry. The proposed algorithm can be applied in many other important industrial cases, for instance, display materials such as liquid crystal display (LCD) panels, polarized films. Considering the common objective of making the best of the limited resource by minimizing the waste, it is also possible to introduce products like papers, steels, or carpets. We cannot make a huge amount of individual product orders separately. A number of common types of unit production are necessary. The resulting issue is to minimize the redundant area except the parts that are used for orders. By making a little bit more efficient algorithm, we can save more production cost and thereby stay more competitive. It is quite reasonable to mention that the attention on developing new algorithms will not be reduced under the increasing economic competition.

The proposed algorithm saves the production cost by reducing the number of mother glasses with the help of increased yield, utilization performance. There are other ways to remain competitive for glass production companies. Increasing overall performance of glass product supply chains in the context of so-called supply chain management would be a good alternative [11,12]. While glass production is widely known to be saturated technologies, more new approaches will be developed to survive in the industry.

ACKNOWLEDGEMENTS

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0023678).

REFERENCES

1. G. Wäscher, H. Hauβner and H. Schumann, *Eur. J. Oper. Res.*, **183**, 1109 (2007).
2. H. Dyckhoff, *Eur. J. Oper. Res.*, **44**, 145 (1990).
3. A. Lodi, S. Martello and M. Monaci, *Eur. J. Oper. Res.*, **141**(2), 241 (2002).
4. D. Pisinger and M. Sigurd, *Discrete Optim.*, **2**(2), 154 (2005).
5. J. Puchinger and G. R. Raidl, *Eur. J. Oper. Res.*, **183**(3), 1304 (2007).
6. A. Lodi, S. Martello and D. Vigo, *INFORMS J. Comput.*, **11**(4), 345 (1999).
7. F. R. K. Chung, M. R. Garey and D. S. Johnson, *SIAM J. ALGEBRA DISCR.*, **3**(1), 66 (1982).
8. E. G. Coffman Jr., M. R. Garey, D. S. Johnson and R. E. Tarjan, *SIAM J. Comput.*, **9**(4), 808 (1980).
9. T. H. Wua, J.-F. Chen, C. Lowa and P.-T. Tanga, *Int. J. Prod. Res.*, **41**(16), 3883 (2003).
10. N. Ntene and J. H. van Vuuren, *Discrete Optim.*, **6**(2), 174 (2009).
11. J. H. Ryu, *Korean J. Chem. Eng.*, **27**(6), 1681 (2010).
12. J. H. Ryu, *Korean Chem. Eng. Res.*, **46**(4), 792 (2008).