An algorithm for determination of the guillotine restrictions for a rectangular cutting-stock pattern

DANIELA MARINESCU
Transilvania University of Braşov
Department of Computer Science
Iuliu Maniu 50, 500091, Braşov
ROMANIA
mdaniela@unitbv.ro

ALEXANDRA BĂICOIANU
Transilvania University of Braşov
Department of Computer Science
Iuliu Maniu 50, 500091, Braşov
ROMANIA
a.baicoianu@unitbv.ro

Abstract: Starting from a two-dimensional rectangular Cutting-Stock pattern with gaps, this paper is focuses on the problem of determination if the pattern is with guillotine restrictions or not and proposes an algorithm for solving it. First we present two new graph representations of the cutting pattern, weighted graph of downward adjacency and weighted graph of rightward adjacency. Using these representations we propose a method to verify guillotine restrictions of the pattern which can be apply for cutting-stock pattern with gaps but also for the cutting or covering pattern without gaps and overlapping.

Key–Words: two-dimensional cutting-stock problems, cutting pattern representation, guillotine restrictions.

1 Introduction

One of the NP-hard problems presented in the literature is the problem of cutting of concrete or abstract objects. This problem, together with the dual problem of covering (packing), appears under various specifications [4]: cutting-stock problems, knapsack problems, container and vehicle loading problems, pallet loading, bin-packing, assembly line balancing, etc. The problem of cutting and covering arises in various production processes with applications varying from the home-textile to the glass, steel, wood, and paper industries, where rectangular figures are cut from large rectangular sheets of materials. Furthermore, an arguably more complicated problem called Cutting and Covering [5] can be derived by cutting a piece of material into small pieces which are then used to cover a surface without overlapping or leaving any gaps.

Dyckhoff provided in [4] a classification of the various types of cutting problems such as: one dimensional, two dimensional and three dimensional with different types of constrains. For these problems it is also possible to find the packing order by using a topological sorting algorithm [13, 14]. A frequent constrain, imposed by industrial applications of the two or three dimensional problem, is the so-called guillotine restriction which states that the resulting patterns need to be guillotine cuttable. Many applications of two-dimensional cutting and covering in the glass, wood, and paper industries, for example, are restricted to guillotine cutting.

Because of the strong link between cutting problem and covering problem based on the duality of material and space, it seems to be obvious to examine both within a general framework. There are still some differences between cutting and covering problems.

In the literature have been proposed several techniques that solve the cutting and covering problems such as formulating the problem as a mixed integer problem, heuristics [17], genetic algorithms [7] as well as approximation algorithms [6]. All these methods result in a pattern or a set of patterns but they are not adequate for constructing cutting patterns when the approach is used to solve the guillotine cutting-stock problem.

In [1] a polynomial algorithm is presented in case of guillotine cutting a rectangle into small rectangles of two kinds: rectangles of the first kind with the same width but their heights can be various and rectangles of the second kind with the same height, and their widths can be various. However, the guillotine restrictions are difficult to respect in the general pattern-generation process. So instead of generation of a cutting-stock pattern with guillotine restrictions it is possible to use an analytic method to verify if the pattern, obtained by some methods used in case of non-guillotine cutting, is with or without guillotine restrictions [11]. Nevertheless, this method is rather unpractical since the cutting pattern is represented as an array pattern [12], which implies a large matrix representation. Another analytical method, presented in [15], used the graph representation [8, 9] of a covering pattern without gaps or overlapping. This method

developed an algorithm for guillotine restrictions verification, based on connections between guillotine cut and the connex components of the graphs. Unfortunately this algorithm is useless in case of a cutting-stock pattern with gaps. For this kind of pattern we propose in [16] another method based on two new kinds of graph representations, weighed graph representations, which it is more general comparing with the method presented in [15].

This paper is an extended paper of [16] and presents an algorithm for determination of the guillotine restrictions for a rectangular cutting-stock pattern with gaps.

2 Problem formulation

Let $\mathcal{P}$, a rectangular plate, characterized by length L and width W. From plate $\mathcal{P}$ we cut k rectangular items, C_i , i =1, 2, ..., k. An item C_i is characterized by length l_i and width w_i.

Definition 1 *A rectangular cutting-stock pattern is an arrangement of the k rectangular items C_i on the supporting plate $\mathcal{P}$, so that the borders of the items C_i to be parallel with the borders of the plate $\mathcal{P}$.*

For this kind of patterns we have presented in [8, 9] two graph representations. Starting from these representations we complete the graphs by adding a value for each arc from the two graphs.

Definition 2 *A rectangular cutting pattern has guillotine restrictions if at every moment of the cutting process the remaining supporting rectangle is separated in two new rectangles by a cut from an edge to the opposite edge of the rectangle and the cutting line is parallel with the two remaining edges.*

In the set of the rectangles $\{C_1, C_2, \ldots, C_k\}$ from the covering pattern we define a downwards adjacency relation and a rightwards adjacency relation.

Definition 3 *The rectangle C_i is downward adjacent (rightward adjacent) with rectangle C_j if in the cutting pattern, C_j is to be found downward (respectively rightward) C_i and their borders have at least two common points.*

Let $C = \{ C_1 , C_2, ..., C_k \}$ and R_d, $R_r \notin C$. For any covering pattern, we defined in [8, 9], a graph of *downwards adjacency* and another one of *rightwards adjacency*. We define now two new graphs a weighted graph of *downwards adjacency*, G_d, and another one of *rightwards adjacency*, G_r. We will use in these definitions the notation $V(X, Y)$ for the value of the arc (X, Y).

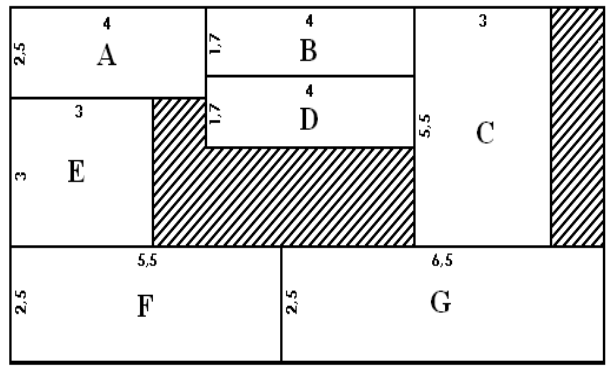

Figure 1: The rectangular cutting-stock pattern

Definition 4 *The weighted graph of downward adjacency G_d = ($C \cup \{R_d\}$, Γ_d) has as vertices the rectangles C_1, C_2, ..., C_k and a new vertex R_d symbolizing the northern borderline of the supporting plate P. The Γ_d is defined as follows:*

$$\begin{cases} \Gamma_d(C_i) \ni C_j \text{ if } C_i \text{ is downward adjacent with } C_j \\ \Gamma_d(R_d) \ni C_i \text{ if } C_i \text{ touches the North border of the support plate } P \\ V(X, C_j) = w_j, \forall X \in C \cup R_d \text{ and } C_j \in C \end{cases}.$$

Definition 5 *The weighted graph of rightward adjacency G_r = ($C \cup \{R_r\}$, Γ_r), where R_r symbolizes the western border. The Γ_r is defined as follows:*

$$\begin{cases} \Gamma_r(C_i) \ni C_j \text{ if } C_i \text{ is rightward adjacent with } C_j \\ \Gamma_r(R_r) \ni C_i \text{ if } C_i \text{ touches the West border of the support plate } P \\ V(X, C_j) = l_j, \forall X \in C \cup R_r \text{ and } C_j \in C \end{cases}.$$

Let the cutting-stock pattern from Figure 1. The weighted graphs G_d and G_r, are represented in Figure 2 and Figure 3.

We remark that in the graphs G_d and G_r the vertex R_d (respectively R_r) is connected by an arc to the vertex C_i if and only if C_i touches the northern (respectively the western) border of the support $\mathcal{P}$.

Remark 6 *In the following we consider only the rectangular cutting-stock pattern where the rectangles are not situated under or to the right of an empty spaces. When it is not true (see Figure 4), we can define another pattern (equivalent) by moving the rectangles (in Figure 4 this rectangle is C) down or to the left till they touch the border of another rectangle. In the sense of the cutting-stock problem with a minimum rest, for every cutting-stock pattern there is always an equivalent pattern of this form.*

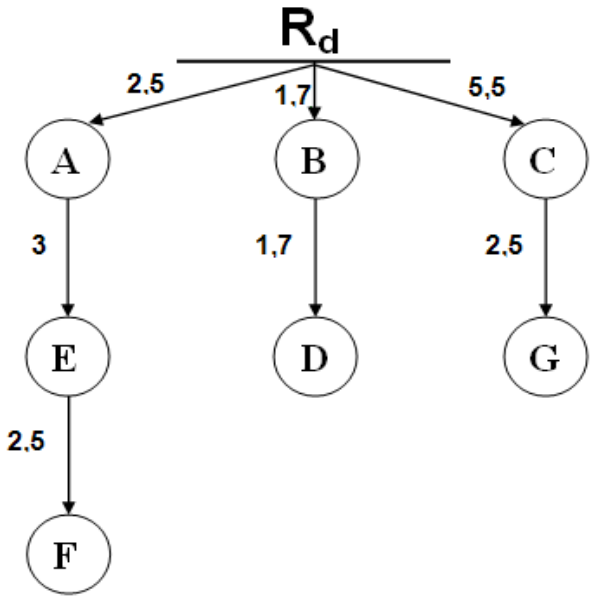

Figure 2: The graph G_d

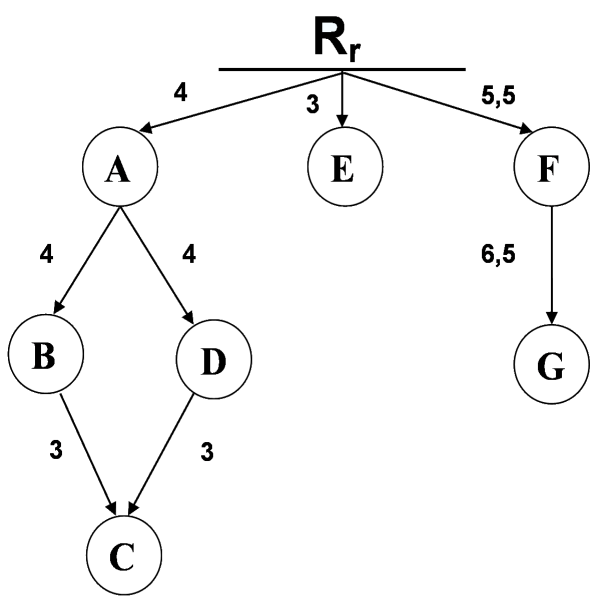

Figure 3: The graph G_r

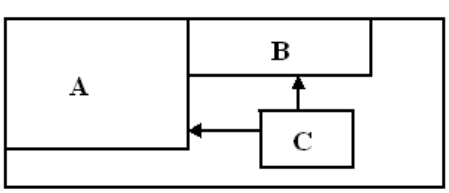

Figure 4: Moving directions

From the Remark 6 it results that the weighted graphs G_d and G_r attached to a cutting-stock pattern have the following properties [9]: the graphs are quasi strongly connected and have no circuit.

Let us have a cutting-stock pattern with guillotine restrictions. From [10] it follows that it is possible to represent a rectangular cutting-stock pattern with guillotine restrictions using an expression with two operations:

1. $\ominus$ - the s-line concatenation, an operation for horizontal cuts;
2. $\oslash$ - the s-column concatenation, an operation for vertical cuts.

For example, the cutting pattern from Figure 1 will be described by the following expression:

$$\ominus \oslash \ominus AE \oslash \ominus BDC \oslash FG.$$

3 Cuts determination

In [15] we presented an algorithm for cuts determination in case of a cutting pattern without gaps. But it is not possible to apply this algorithm in the case of a cutting-stock pattern with gaps.

Starting from a rectangular cutting-stock pattern with gaps we intend to find a connection between guillotine restrictions and the two weighted graphs of adjacency, G_d and G_r.

For this purpose we will use the notation $Lpd(R_d, C_i)$, respectively $Lpr(R_r, C_i)$ for the length of the path from R_d to C_i in the graphs G_d, respectively G_r. We remark that $Lpd(R_d, C_i)$ represents the distance from the northern border of the plate *P* to the southern border of piece C_i and similarly $Lpr(R_r, C_i)$ represents the distance from the western border of the plate *P* to the eastern border of piece C_i.

Remark 7 *If a cutting-stock pattern has an horizontal guillotine cut situated to a distance M from the North border of the supporting plate P then the set of the items, C, can be separated in two subsets S_1, the set of the items situated above this cut, and S_2 the set of the items situated below this cut. Of course in the weighted graph G_d we have:*

1. $Lpd(R_d, C_i) \leq M$ *for every* $C_i \in S_1$;
2. $Lpd(R_d, C_i) > M$ *for every* $C_i \in S_2$.

We obtain a similar result if the cutting-stock pattern has a vertical cut.

The two conditions from the above remarks are necessary but not sufficient because it is possible the

cut to intersect some items from the set S_2. We present in the following necessary and sufficient conditions for a guillotine cut.

Theorem 8 *Let a rectangular cutting-stock pattern with possible gaps and the weighted graph G_d attached to the pattern. The cutting-stock pattern has an horizontal guillotine cut on the distance M from the northern border of the supporting plate if and only if it is possible to separate the sets of the items, C, in two subsets, S_1 and S_2 so that:*

1. *$C = S_1 \cup S_2, S_1 \cup S_2 = \emptyset$;*
2. *For every $C_j \in C$ so that $(R_d, C_j) \in \Gamma_d$ it follows that $C_j \in S_1$;*
3. *$Lpd(R_d, C_i) \leq M$ for every $C_i \in S_1$;*
4. *If there is $C_j \in S_1$ so that $Lpd(R_d, C_j) < M$ then all direct descender of C_j will be in S_1.*

Proof:

i. Suppose that the cutting-stock pattern has a vertical guillotine cut. That means the sets of items $\mathcal{C}$ can be separated in two subsets, S_1, the set of the vertices situated above the cut, and S_2, the set of the vertices situated below the cut. From the Remark 7 it follows that the conditions 1, 2 and 3 are fulfilled.

Suppose that the condition 4 is not fulfilled. That means there are two items $C_j \in S_1$ and $C_i \in S_2$ so that $Lpd(R_d, C_j) < M$ and the item C_i is a successor of C_j. Because $C_i \in S_2$ it follows that $Lpd(R_d, C_i) > M$ and an horizontal cut situated on the distance M from the northern border of the supporting plate will intersect the item C_i. It means that without the condition 4 it is impossible to separate the set of the items by an horizontal cut. So our supposition that the condition 4 is not fulfilled is false.

ii. Suppose all the conditions 1-4 are fulfilled but it is not possible to make an horizontal cut on the distance M in the cutting-stock pattern. It follows that there is at least item $C_i \in S_2$ which is intersected by such a cut. It means that the distance from the northern border of the supporting plate to the northern border of the item C_i is less than M and the distance from the northern border of the supporting plate to the southern border of the item C_i is greater than M.

But from the Remark 7 it follows that the northern border of the item C_i is identical with the southern border of some item C_j, situated above C_i. That means $(C_j, C_i) \in \Gamma_d$ and $Lpd(R_d, C_j) < M$ and so $C_j \in S_1$. From condition 4, because C_i is a successor of C_j, it follows that C_i must be in S_1 in contradiction with our hypothesis. That means that if the conditions 1-4 are fulfilled then there is an horizontal guillotine cut in the cutting-stock pattern. □

We obtain a similar result if we consider the weighted graph of rightwards adjacency.

4 The algorithm for verification of the guillotine restrictions

The results from the previous theorem suggest an algorithm for verification of the guillotine restrictions, in case of a cutting-stock pattern with gaps.

Input data: The weighted graphs G_d or G_r attached to a rectangular cutting pattern.

Output data: The s-pictural representation of the cutting pattern like a formula in a Polish prefixed form.

Method: The algorithm constructs the syntax tree for the s-pictural representation of the cutting pattern, starting from the root to the leaves (procedure PRORD). For every vertex of the tree it verifies if it is possible to make a vertical (procedure VCUT) or horizontal cut (HCUT), using an algorithm for decomposition of a graph in two components, S_1 and S_2.

We will use the following notations:

- G'_r, G'_d are the subgraphs of $G_r|_X$, respectively $G_d|_X$, where we can add, if it is necessary, the root $R_r(R_d)$ and the arcs starting from R_r (R_d), like in Definition 4.

- $L_1(L_2)$ is the weight of the first (second) cutting support which contains all the items from $S_1(S_2)$.

- $succ(C_i|_G)$ is the set of successors of the item C_i in the graph G.

The method ADD() is used for addition of the next member in the Polish prefixed form.

We remark that we can apply this algoritm also in case of a cutting-stock pattern without gaps and, of course, in the case of covering pattern with or without gaps.

4.1 Example

Let us have the cutting-stock pattern from Figure 1 with the weighted graphs, G_d and G_r.

For this example $L = 8$, $W = 5.5$, and first we are trying an horizontal cut. We obtain two sets, one composed from nodes $\{A, E, B, D, C\}$ and $\{F, G\}$, see Figure 5, Figure 6 .

In the syntactic tree from Figure 7 we have 2 components connected using the operation column concatenation $\ominus$ for the horizontal cut. Like we said before, the 2 components are obtained taking in consideration $L = 8$ and $W = 5.5$, in the first component we have the nodes from the left side of the cut and in the second the remaining nodes.

PROCEDURE PRORD$(G, C, L, W, ADD())$
begin
  VCUT$(G_r, C, L, W, err, S_l, S_2, L_1, L_2)$;
  if $err = 0$ **then**
    if $|C| = 1$ **then ADD**(C)
    else ADD$(\oslash)$;
    PRORD$(G_d, S_1, L_1, W, ADD())$;
    PRORD$(G_d, S_2, L_2, W, ADD())$;
  end
  else
    HCUT$(G_d, C, L, W, err, S_1, S_2, W_1, W_2)$;
    if $err = 0$ **then**
      if $|C| = 1$ **then ADD**(C)
      else ADD$(\ominus)$;
      PRORD$(G_r, S_1, L, W_1, ADD())$;
      PRORD$(G_r, S_2, L, W_2, ADD())$;
    end
    else No guillotine restrictions
  end
end
PROCEDURE
VCUT$(G_r, X, L, W, err, S_1, S_2, L_1, L_2)$
begin
  $err = 0$;
  CONSTRUCT-SUBGRAPH(G_r, G'_r, X, R_r);
  V $:= \bigcup\{C_i | C_i \in X, (R_r, C_i) \in \Gamma_r\}$, where all the elements are unmarked
  maxM $:= max\{l_i | C_i \in V\}$
  $P_i := \{l_i | C_i \in V\}$
  while $\exists C_i \in V$ *unmarked element* **do**
    mark C_i;
    if $P_i < maxM$ **then**
      for $C_j \in succ(C_i|_{G'_r})$ **do**
        $V := V \bigcup\{C_j|$ where C_j is an unmarked element$\}$;
        $P_j := P_i + l_j$;
        if $P_j > maxM$ **then**
          $MaxM := P_j$;
        end
      end
    end
  end
  maxM $:= max\{Lpd(R_r, C_i) | C_i \in V\}$
  if $maxM = L$ **then**
    $err = 1$;
  end
  $L_1 := maxM$;
  $L_2 := L - maxM$;
  $S_1 := V$;
  $S_2 := X - V$;
end

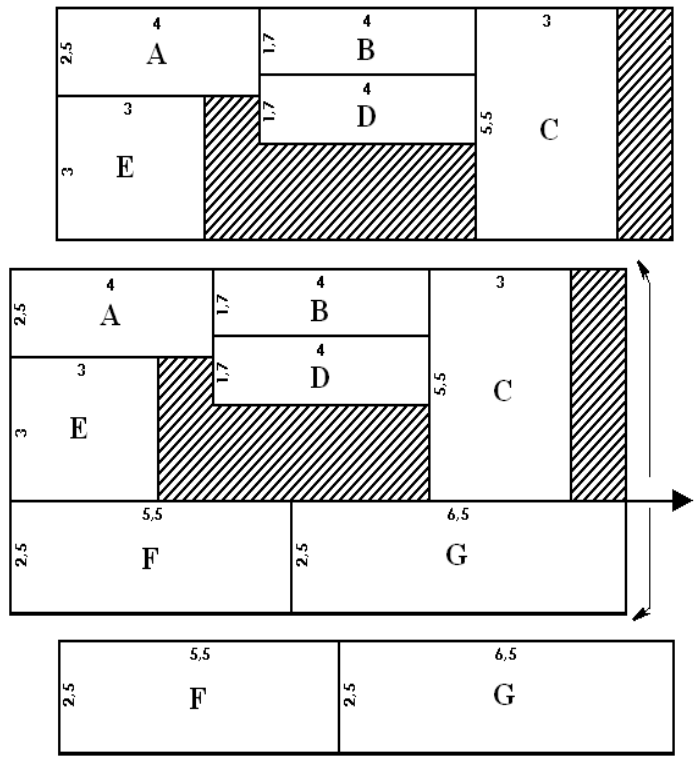

Figure 5: The first horizontal cut

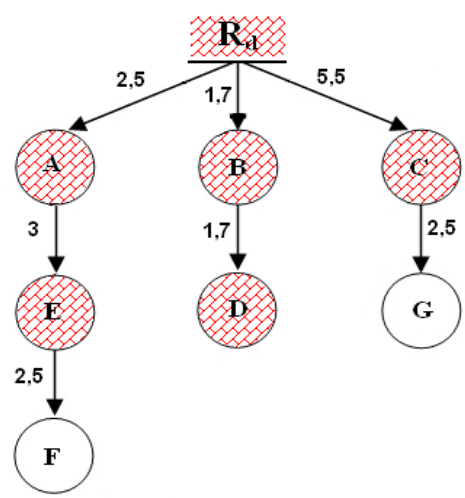

Figure 6: The two sets from the horizontal cut

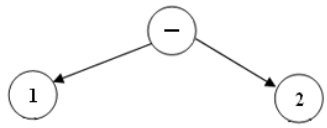

Figure 7: The first syntactic tree

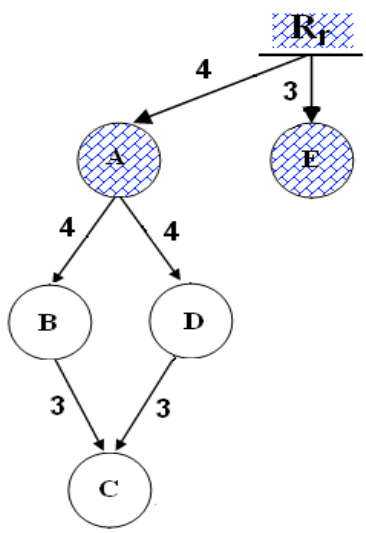

Figure 8: The two sets from the vertical cut

The prefix Polish notation for this syntactic tree from Figure 7 is:

$$\ominus.$$

Using the first component we are trying a vertical cut, we have $maxM = 2.5$. In Figures 8 and 9, we have the decomposition in two other sets, one of them contains the items $\{A, E\}$, and the other one $\{B, D, C\}$.

In Figure 10 we are adding to the syntactic tree the components 3 and 4, containing the sets of items above mentioned, which are connected using the operation of collumn concatenation $\oslash$ for the vertical cut.

The prefix notation for this tree from Figure 10 is:

$$\ominus \oslash .$$

We continue to make horizontal or vertical cut for the left and right components from the syntactic tree until every component contains only one item from the covering pattern.

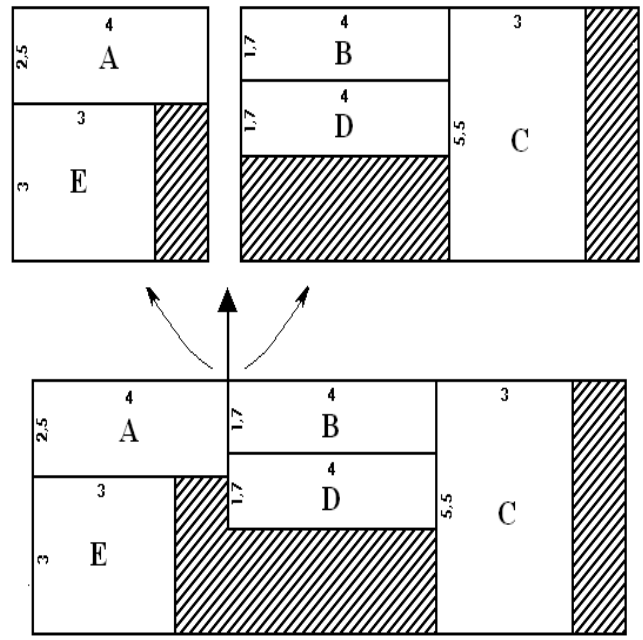

Figure 9: A vertical cut

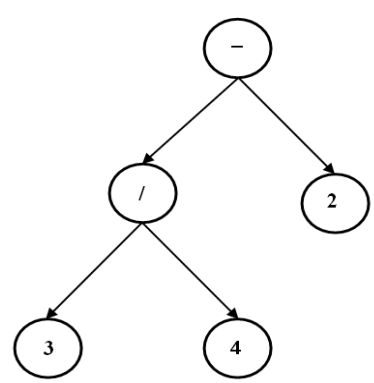

Figure 10: The syntactic tree

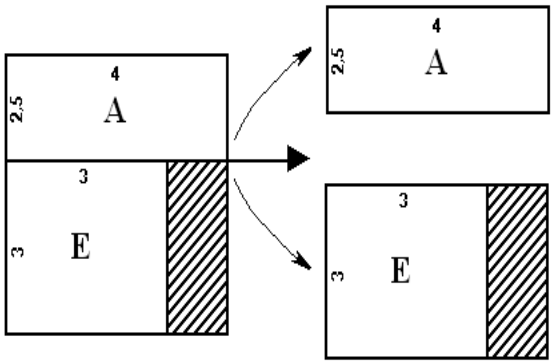

Figure 11: One horizontal cut

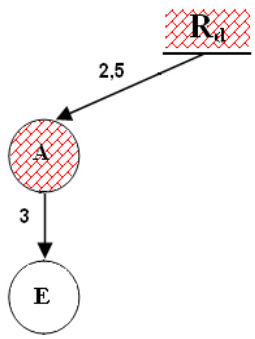

Figure 12: The sets obtained from the previous cut

On the third component, consists of items $\{A, E\}$ we are trying an horizontal cut, a decomposition of the component 3 in A and E. Both of them are leaves of the syntactic tree. We did the horizontal cut in Figure 11, and we can see the sets obtained in Figure 12.

The syntactic tree associated is presented in Figure 13.

The prefixed notation associated to the syntax tree from Figure 13 is:

$$\ominus \oslash \ominus AE.$$

On the fourth component, we are looking for an horizontal cut, but this is nor possible so we are doing a vertical one, see Figure 14 which divides it in a simple node, C and a new component, component number 5. In Figure 15 we have effective the cuts and in Figure 16 we have the syntactic tree, wuth the leaf of the tree, C.

On the fifth component we are trying an horizontal cut, Figure 17, Figure 18 and we have in Figure 19 the two new leaves added to the syntactic tree.

We determinated the all left side of the syntactic tree corresponding to our covering pattern using the two kinds of cuts.

The polish notation obtained from the left side of

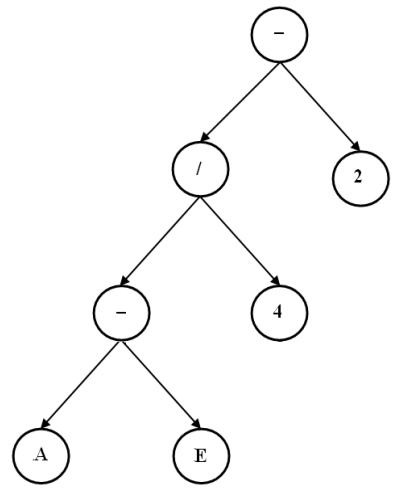

Figure 13: The syntactic tree

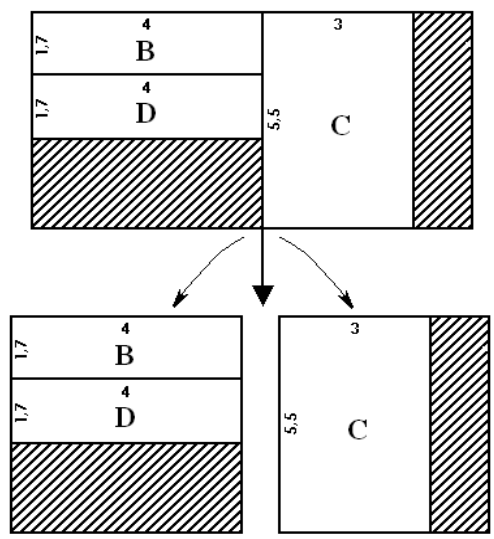

Figure 14: A vertical cut on the fourth component

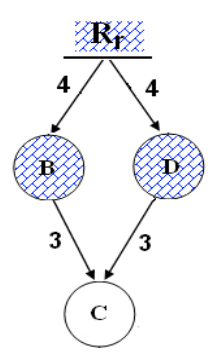

Figure 15: The cuts obtained

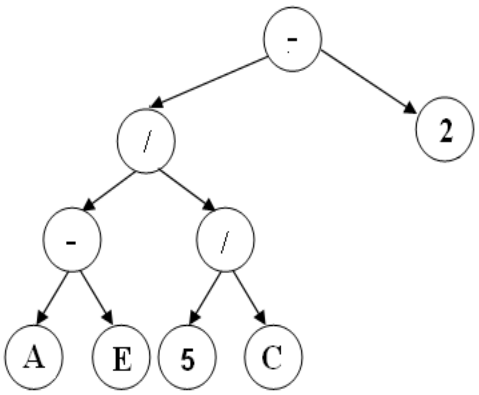

Figure 16: The syntactic tree

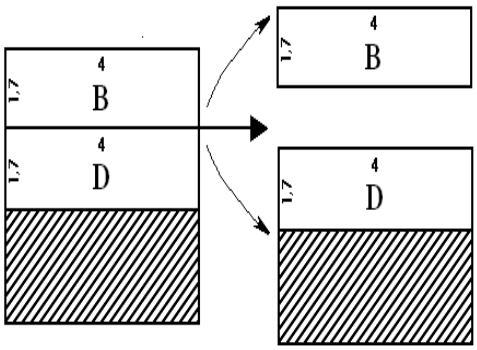

Figure 17: The horizontal cut on the fifth component

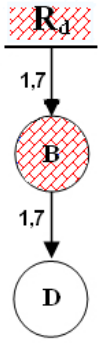

Figure 18: The cuts

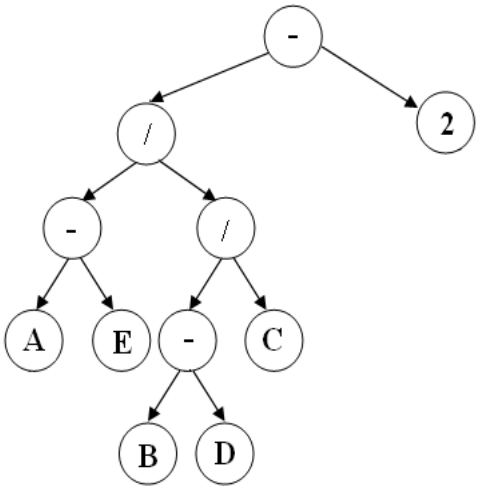

Figure 19: The syntactic tree

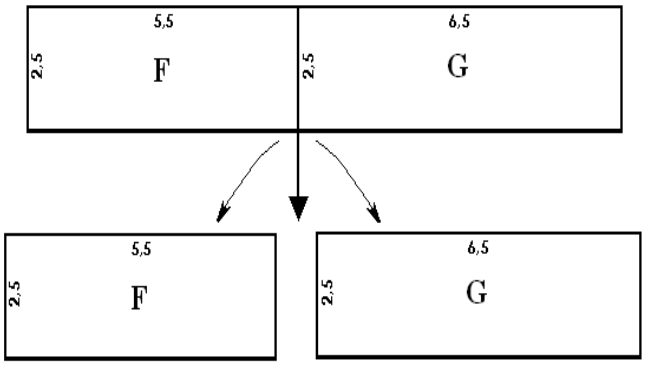

Figure 20: The horizontal cut on the second component

the syntax tree is:

$$\ominus \oslash \ominus AE \oslash \ominus BDC.$$

Let's consider now the right side of the syntactic tree. Using the second component obtained from the first cut that we did at the beginning, we are trying an horizontal cut, but it is not possible, so in Figure 20 we have the vertical cut, which means the decomposition of the component 2 in two nodes, F and G.

The derived syntactic tree is presented in Figure 21.

The polish notation for the tree from Figure 21 is:

$$\ominus \oslash \ominus AE \oslash \ominus BDC \oslash FG.$$

4.2 Correctness and Complexity

The correctness of the algorithm follows from the theorems 1, that make the connection between a guillotine cut and the decomposition of a graph in two subgraphs.

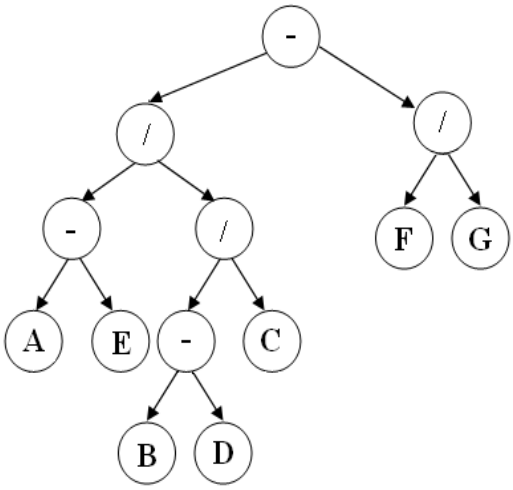

Figure 21: The syntactic tree

The procedure *PREORD()* represents a preorder traversal of a graph, so the complexity is $O(k)$ [3, 2], where k is the number of the cutting items. Also, in the procedure *VCUT*, respectively *HCUT* we traverse a subgraph of the initial graph. So, the complexity of the algorithm is $O(k^2)$.

5 Conclusions

Indifferently if it is a guillotine covering [15] or cutting-stock, with gaps or without gaps, the problem, the so-called two-dimensional guillotine problem, is a constraint on a complete partition of two-dimensional space. The partitioning of two-dimensional space is a ubiquitous problem in industry. It appears in many forms from pallet loading to floor tile tessellation.

A subset of the problem, the two-dimensional guillotine problem, is almost as pervasive. Various aspects of the problem are found in industries that produce two dimensional sheets of glass, textiles, paper or other material. A similar problem arises in the design of layouts for integrated circuitshow should the subcircuits be arranged to minimize the total chip area required or in the design of an optimal placement of a set of solar panels. Like the complete partition, the guillotine problem remains NP hard. For this reason it is better to use an algorithm for generating an unconstrained covering or cutting-stock pattern and, after that or in each step, to use our algorithms for verifying the guillotine restrictions of the generated pattern.

Acknowledgements: The research was supported by the Grant IDEI no. 134/2007.

References:

[1] M.Z. Arslanov, D.U. Ashigaliev and E.E. Ismail Polynomial algorithms for guillotine cutting of a rectangle into small rectangles of two kinds European, *Journal of Operational Research*, Vol. 185, Issue 1, 2008, Pp 105-121.

[2] Ciurea, E., Ciupala, L.: *Algoritmi - Introducere in algoritmica fluxurilor in retele*, Ed. Matrix ROM Bucureşti 2006.

[3] Cormen, T.H., Leiserson, C.E., Rivest, R.R.: *Introduction to Algorithms*, MIT Press, 1990.

[4] Dyckhoff, H.: A typology of cutting and packing problems, *European Journal of Operational Research*, 44 (1990), pp. 145-159.

[5] Iacob, P., Marinescu, D., Luca, C.: Covering with Rectangular Pieces, *Analele Ştiinţifice Univ. Ovidius Constanţa*, Vol. 11(2), 2003, pp 75-86.

[6] Iacob, P.,Marinescu, D. and Kiss-Jakab, K: A decomposition problem of a natural number for a rectangular cutting-covering pattern,*Proceeding of the 11-th WSEAS CSCC*, Crete, 2007, Vol. 4, pp. 76-81.

[7] Lewis, J.E.,Ragade, R.K.,Kumar, A. and Biles, W.E.: A distributed chromosome genetic algorithm for bin-packing, *14th Intern. Conf. on Flexible Automation and Intelligent Manufacturing*, Vol. 21, Issues 4-5, Aug-Oct 2005, pp. 486-495.

[8] Marinescu, D.: Graphs attached to a rectangular cutting-stock model (French), *Itinerant Seminar of Functional Equation, Approximation and Convexity*, Cluj-Napoca 1988 Preprint No. 6 pp. 209 -212.

[9] Marinescu, D.: A representation problem for a rectangular cutting - stock model, *Foundations of Computing and Decision Sciences*, Vol. 32, 2007, No. 3, pp. 239-250.

[10] Marinescu, D.: A s-picture language for a cutting-stock model with quillotine restrictions, *Buletin of the Transilvania University of Brasov* - seria C, Vol XXXIII 1991 pp 39-45.

[11] Marinescu, D.: A bidimensional Turing machine for the cutting stock problem with quillotine restrictions, *Proceeding of the scientific symposion with the contribution of teachers and researchers from the Republic of Moldova*, Brasov 1991, pp 73-83.

[12] Marinescu, D.: A monotonic array grammar for a two-dimensional cutting-stock problem, *Proceeding of the third colloquium on Logic, Language, Mathematics Linquistics*, Brasov 1991 pp 51-60.

[13] D. Marinescu, P. Iacob and A. Băicoianu, A Plan of Lauding the Boxes for a Three Dimensional Bin Packing Model, *WSEAS Transactions on Systems*, Issue 10, Volume 7, October 2008, pp. 830-839.

[14] D. Marinescu, P. Iacob and K. Kiss-Jakab, A topological order for a rectangular covering model, *Proc. of the 11-th WSEAS CSCC*, Crete, Vol. 4, 2007, pp. 82-85.

[15] D. Marinescu and A. Băicoianu: An Algorithm for the Guillotine Restrictions Verification in a Rectangular Covering Model, *WSEAS Transactions on Computers*, ISSN: 1109-2750, Issue 8, Vol. 8, Aug. 2009, pp. 1306-1316.

[16] D. Marinescu and A. Băicoianu: The Determination of the Guillotine Restrictions for a Rectangular Cutting-Stock Pattern, *Proceeding of the 14-th WSEAS CSCC*, Corfu, 2010, pp.121-126.

[17] Xiang Song, Chengbin Chu and Yiyong Nie, A Heuristic Dynamic Programming Algorithm for 2D Unconstrained Guillotine Cutting, *WSEAS Transactions on Mathematics*, Issue 1, Volume 3, January 2004, pp. 230-238.