An algorithm for the guillotine restrictions verification in a rectangular covering model

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: We consider the Cutting and Covering problem with guillotine restrictions. In one of our previous works, we have shown the connection between the connex components of a graph representation for the covering model and the guillotine cuts. Based on this, in this paper we propose an algorithm which can be used to verify guillotine restrictions in a two-dimensional covering model.

Key–Words: bidimensional covering problem, guillotine restrictions, conex components of a graph

1 Introduction

Cutting and Covering problems are common yet difficult problems, known to be NP-hard or NP-complete. Also referred to as Cutting and Packing problems, these problems have many practical applications [22, 21]. A comprehensive classification of such problems is given in [4]. Furthermore, an arguably more complicated problem called Cutting and Covering [6, 7] 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.

Many variants of the Cutting and Covering problem were previously considered: 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 [18, 19]. 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, i.e. the items can be obtained through a sequence of edge-to-edge cuts parallel to the edges of the support.

Several techniques that solve the Cutting and Covering problem have been proposed, such as formulating the problem as a mixed integer problem [10], heuristics [23], genetic algorithms [11] as well as approximation algorithms [8, 9]. All these methos result in a pattern or a set of patterns. However, the guillotine restrictions are difficult to respect in this pattern-generation process. It is possible to use an analytic method to verify if the obtained pattern is with or without guillotine restrictions [15, 16]. Nevertheless, this method is rather unpractical since the cutting pattern is represented as an array model [17], which implies a large matrix representation.

In [20] we used the graph representation of the cutting and covering pattern [12, 13] to prove the connection between guillotine and the connex components of the graph. In this paper, we start from this connection and present an algorithm which can be used to verify the guillotine restrictions in a two-dimensional covering model.

2 Problem formulation

Let $\mathcal{P}$, a rectangular plate, characterized by length l and width w. The plate $\mathcal{P}$ is covered with k rectangular items, C_i , i =*1, 2, ...,* k, from $\mathcal{C}$ without gaps or overlapping. An item is characterized by length l_i and width w_i.

Definition 1 *A rectangular covering model is an arrangement of the k rectangular items C_i on the supporting plate $\mathcal{P}$, so that $\mathcal{P}$ is completely covered by the components C_i, without gaps or overlapping.*

Example 1. Let the covering model from Figure 1 where $\mathcal{P}$(340x172), C_1(96x47), C_2(44x123), C_3(51x59), C_4(51x62), C_5(110x39), C_6(110x59), C_7(135x94), C_8(72x77), C_9(83x77), C_{10}(89x77).

Definition 2 *A rectangular covering model 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*

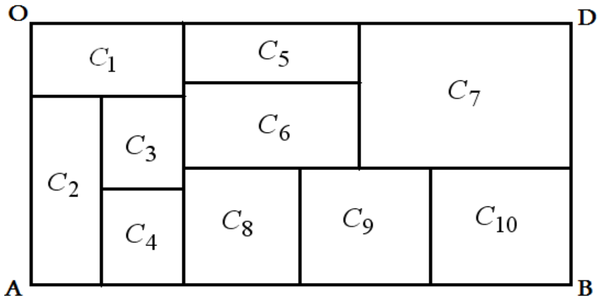

Figure 1: The rectangular covering model

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 model we define a downwards adjacency relation and a rightwards adjacency relation.

Definition 3 *The rectangle C_i is downward adjacent with rectangle C_j if in the covering model, C_j is to be found downward C_i and their borders have at least two common points.*

Definition 4 *The rectangle C_i is rightward adjacent with rectangle C_j if in the covering model, C_j is to be found rightward C_i and their borders have at least two common points.*

Let *C* = $\{ C_1, C_2, \ldots, C_k \}$ and $R_d, R_r \notin C$. For any covering model, we can define a graph of *downwards adjacency*, G_d, and another one of *rightwards adjacency*, G_r.

Definition 5 *[13] The graph of downward adjacency $G_d = (C \cup \{R_d\}, \Gamma_d)$ 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:

$$\begin{cases} \Gamma_d(C_i) \ni C_j \text{ if } C_i \text{ is downward adjacent with } C_j \\ \Gamma_d(R_d) \ni C_i \text{ if } C_i \text{ touches the North border } OD \text{ of the support plate } P \end{cases}$$

Definition 6 *[13] The graph of rightward adjacency $G_r = (C \cup \{R_r\}, \Gamma_r)$, where R_r symbolizes the western border.*

The Γ_r is defined as follows:

$$\begin{cases} \Gamma_r(C_i) \ni C_j \text{ if } C_i \text{ is rightward adjacent with } C_j \\ \Gamma_r(R_r) \ni C_i \text{ if } C_i \text{ touches the West border } OA \text{ of the support plate } P \end{cases}$$

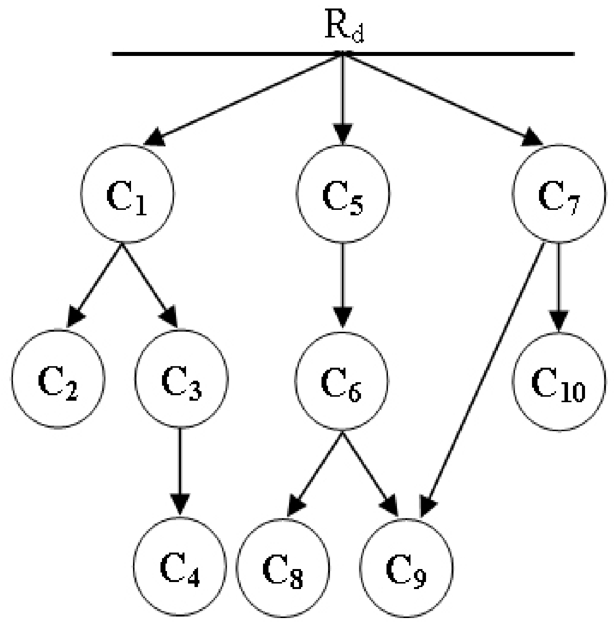

Figure 2: The graph G_d

Example 2. Let the covering model from Figure 1. The 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 arch to the vertex C_i if and only if C_i touches the northern (respectively the western) border of the support $\mathcal{P}$.

Definition 7 *[13] Let a covering model $\mathcal{M}$ and the graph of downward adjacency G_d. We say that the rectangle C_i is situated above the rectangle C_j in the covering model $\mathcal{M}$ if in the graph of downward adjacency G_d there is a path from C_i to C_j. Similarly, we say that the rectangle C_i is situated on the left of the rectangle C_j in the covering model $\mathcal{M}$ if in the graph of rightward adjacency G_r there is a path from C_i to C_j.*

Remark 8 *From [14] it follows that it is possible to represent a rectangular covering model with guillotine restrictions using an expression with two operations:*

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

For the rectangular covering model from Example 1 we obtain the following representation:

$$(C_1 \ominus (C_2 \oslash (C_3 \ominus C_4)))$$

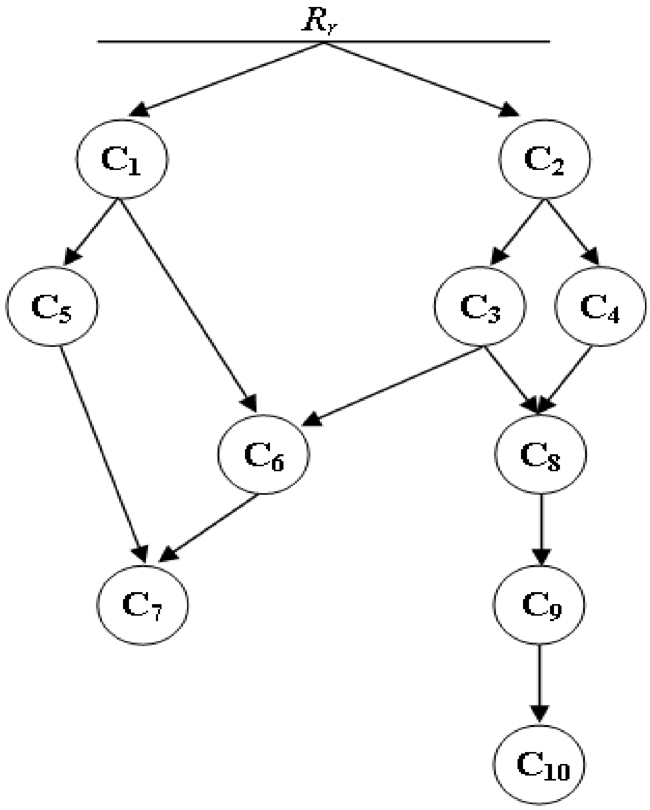

Figure 3: The graph G_r

$$\oslash$$

$$(((C_5 \ominus C_6) \oslash C_7) \ominus ((C_8 \oslash C_9) \oslash C_{10}))$$

where $C_1, \ldots, C_{10}$ are the items from our model.

Remark 9 *From the Property 4 of [13] it follows that for two items C_i and C_j from a covering model there is only one of the following situations:*

1. *C_i is situated above C_j (there is a path from C_i to C_j in the graph G_d);*
2. *C_j is situated above C_i (there is a path from C_j to C_i in the graph G_d);*
3. *C_i is situated on the left of C_j (there is a path from C_i to C_j in the graph G_r);*
4. *C_j is situated on the left of C_i (there is a path from C_j to C_i in the graph G_r);*
5. *There is no path between C_i and C_j neither in the graph G_d nor in G_r.*

3 Cuts determination

Starting from a rectangular covering model without gaps and overlapping we intend to find a connection between guillotine restrictions and the two graphs of adjacency, G_d and G_r, attached to the covering model.

For this purpose we construct two new subgraphs $G'_d$and G'_r where:

1. $G'_d = (C, \Gamma'_d)$ is obtained from G_d by elimination of the vertex R_d together with the arches starting from R_d;
2. $G'_r = (C, \Gamma'_r)$ is obtained from G_r by elimination of the vertex R_r together with the arches starting from R_r.

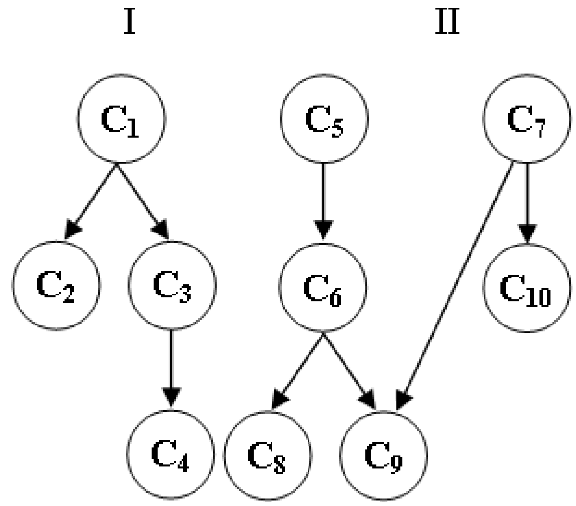

Figure 4: The subgraphs G'_d extracted from G_d

In Figure 4 is presented the subgraph G'_d for the covering model from Figure 1.

In [12] we proved that the graph G_d, respectively G_r, is a conex graph. By the elimination of a vertex together with the arches starting from this vertex it is not sure that the subgraph G'_d, respectively G'_r, remain a conex graph.

We will prove that for a model with guillotine restrictions at least one of these subgraphs is not a conex graph.

Theorem 10 *[20] Let $\mathcal{M}$ be a rectangular covering model without gaps and overlapping and the graph G'_d attached to $\mathcal{M}$. In the covering model $\mathcal{M}$ there is a vertical guillotine cut if and only if in the graph G'_d there are at least two connex components.*

Proof:

i. Suppose that the covering model $\mathcal{M}$ has a vertical guillotine cut. That means the sets of items $\mathcal{C}$ can be separated in two subsets, $\mathcal{C}_l$, the set of the vertices

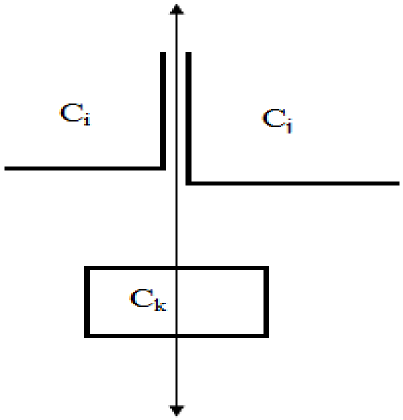

Figure 5: Particular case

situated on the left of the cut, and $\mathcal{C}_r$, the set of the vertices situated on the right of the cut.

Let $C_i \in \mathcal{C}_l$ and $C_j \in \mathcal{C}_r$ two items situated on the left, respectively on the right of the cut. Suppose that the graph G'_d is connex. It follows that there is a chain between C_i and C_j in the graph G'_d. On this chain there are at least two vertices, C_n, C_m so that $C_n \in \mathcal{C}_l$, $C_m \in \mathcal{C}_r$ and C_m is downward adjacent with C_n or C_n is downward adjacent with C_m, from Definition 7. That means C_m is situated above, respectively downward, C_n and the items C_m and C_n have at least two common points.

But, in this case, it is impossible to separate C_m from C_n by a vertical cut. So our supposition that G'_d is a connex graph is false.

ii. Suppose the two connex components of the graph G'_d are G_{d1} and G_{d2}. Let $\mathcal{C}_1$ be the set of the vertices from G_{d1} and $\mathcal{C}_2$ be the set of the vertices from G_{d2}. Let $C_i \in \mathcal{C}_1$ and $C_j \in \mathcal{C}_2$ so that C_i is rightward adjacent with C_j. It follows that there is no chain between C_i and C_j in G'_d and surely no paths.

Suppose that a cut between C_i and C_j intersects an item C_k. This means we have a situation like in Figure 5. In this case, C_i and C_j are situated above the item C_k in the model $\mathcal{M}$ and that means there are two paths: one path from C_i to C_k and another path from C_j to C_k in G'_d. That means there is a chain between C_i and C_j. But this is impossible because C_i and C_j belong to two different connex components. It follows that our assumption that a cut between C_i and C_j intersects an item C_k is false.

So it is possible to separate the model $\mathcal{M}$ in two submodels by a vertical cut. $\square$

Theorem 11 *[20] Let $\mathcal{M}$ be a rectangular covering model without gaps and overlapping and the graph G'_r attached to $\mathcal{M}$. In the covering model $\mathcal{M}$ there is a horizontal guillotine cut if and only if in the graph G'_r there are at least two connected components.*

Proof: The proof is similar with the proof of previous theorem, for vertical cuts. $\square$

4 The algorithm for verification of the guillotine restrictions

The results from the previous theorems suggest an algorithm for verification of the guillotine restrictions, using the decomposition of graphs G'_d or G'_r in connex components.

Input data: The graphs G'_d or G'_r attached to a rectangular covering model.

Output data: The s-pictural representation of the covering model like a formula in a polish prefixed form.

Method: The algorithm constructs the syntax tree for the s-pictural representation of the covering model, defined in Remark 8, starting from the root to the leaves (procedure PREORDER). For every vertex of the tree it verifies if it is possible to make a vertical (procedure V-CUT) or horizontal cut (H-CUT), using an algorithm for decomposition of a graph in two components (procedure CONEXCOMP): one is a conex component and the other is the rest of the graph after extraction of the conex component.

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

PROCEDURE PREORDER $(G, ADD())$
V-CUT(G, err, G_l, G_r);
if $err = 0$ **then**
  if $|G| = 1$ **then ADD**(G)
  else ADD$(\oslash)$;
    PREORDER$(G_l, ADD())$;
    PREORDER$(G_r, ADD())$;
end
else
  H-CUT(G, err, G_u, G_d);
  if $err = 0$ **then**
    if $|G| = 1$ **then ADD**(G)
    else ADD$(\ominus)$;
      PREORDER$(G_u, ADD())$;
      PREORDER$(G_d, ADD())$;
  end
  else No guillotine restrictions
end

PROCEDURE V-CUT(G, err, G_l, G_r)
$err = 0$;
CONEXCOMP(G, p, N_l, N_r);
if $p = 1$ **then**
| $err = 1$;
end
else
| $G_l = N_l$;
| $G_r = N_r$;
end

PROCEDURE H-CUT(G, err, G_u, G_d)
$err = 0$;
CONEXCOMP(G, p, N_u, N_d);
if $p = 1$ **then**
| $err = 1$;
end
else
| $G_u = N_u$;
| $G_d = N_d$;
end

4.1 Example

Let us have the covering model from Figure 1 with the extracted subgraphs, G'_d and G'_r. In Figure 6 it is presented the first vertical cut of the covering model and the decomposition in two components: a conex component and the remainder component. In the syntax tree from Figure 7 the conex component is marked by 1, the remainder component is marked by 2 and they are connected using the operation of collumn concatenation $\oslash$ for the vertical cut. Let $A = \{C_1, C_2, C_3, C_4\}$ and $B = \{C_5, C_6, C_7, C_8, C_9, C_{10}\}$ be the sets of the vertices from these two components.

The partial prefix polish notation for this syntax tree from Figure 7 is:

$$\oslash.$$

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

Using the first component we are doing an horizontal cut in Figure 8, so we have the decomposition of the set A in two sets which are corresponding to the two new components, one of them contain only the item C_1 and another set $D = \{C_2, C_3, C_4\}$. In Figure 9 we are adding to the syntax tree the nodes C_1 and the component 3 which are connected using

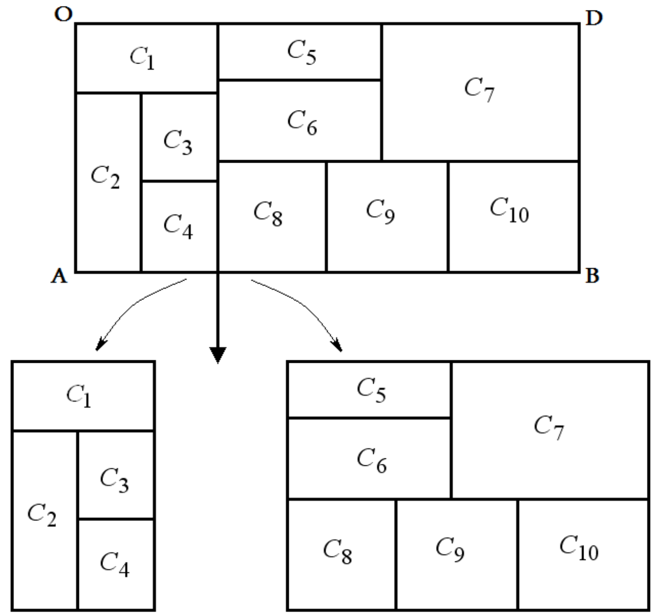

Figure 6: The first vertical cut

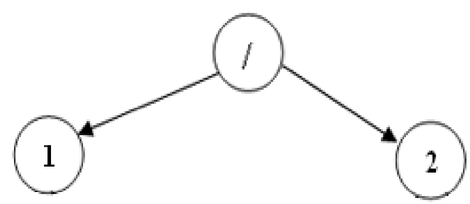

Figure 7: The first Syntax tree

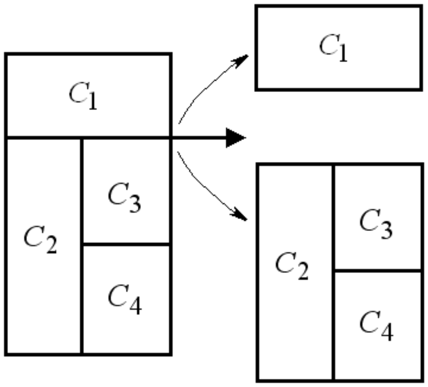

Figure 8: Step2. The first horizontal cut

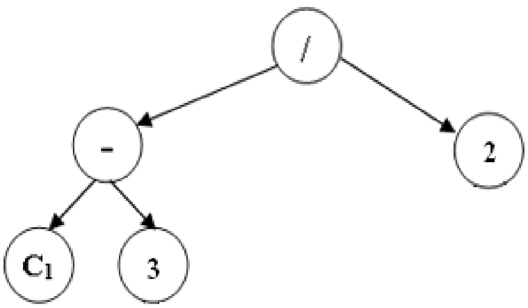

Figure 9: Step2. The syntax tree

the operation of line concatenation $\ominus$ for the horizontal cut.

The partial prefix polish form for this tree from Figure 9 is:

$$\oslash \ominus C_1.$$

On the third component we are trying to do a vertical cut, a decomposition of the set D in 2 sets like in Figure 10, so we have other two components. One of these is a leaf of the syntax tree, C_2 and another is the fourth component, let it be the set $E = \{C_3, C_4\}$, Figure 11.

The partial prefix notation associated to the syntax tree from Figure 11 is:

$$\oslash \ominus C_1 \oslash C_2.$$

The last step is the decomposition of the fourth component, the set E in two components using an

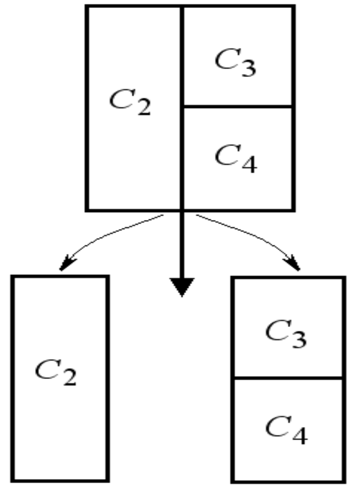

Figure 10: Step3. A vertical cut

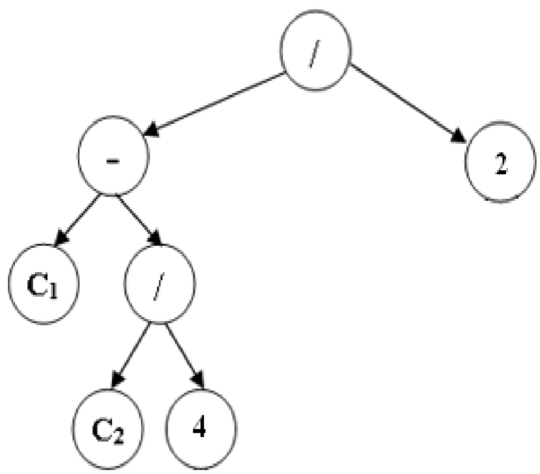

Figure 11: Step3. The syntax tree

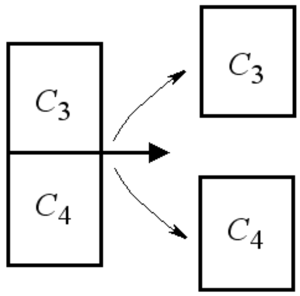

Figure 12: Step4. An horizontal cut

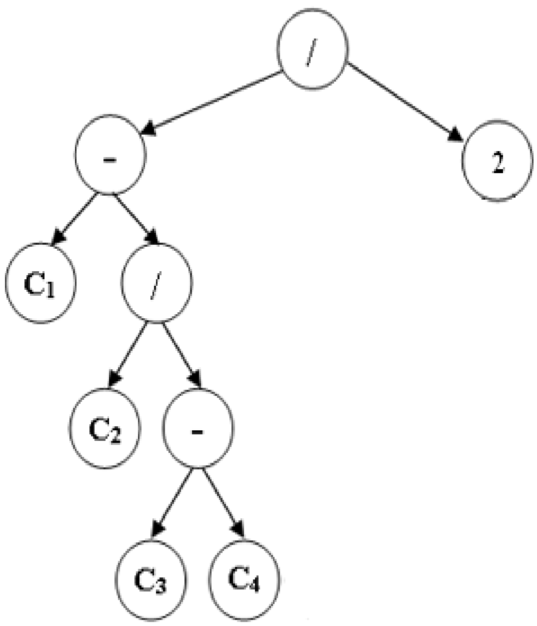

Figure 13: Step4. The syntax tree

horizontal cut Figure 12, C_3 and C_4, both leaves of the syntax tree from Figure 13.

We determinated the all left side of the syntax tree corresponding to our covering model using the two kinds of cuts. The partial polish form obtained from the left side of the syntax tree is:

$$\oslash \ominus C_1 \oslash C_2 \ominus C_3 C_4.$$

Let's consider now the right side of the syntax tree. Using the second component obtained from the first cut that we did at the beginning, we are doing an horizontal cut in Figure 14, so we have the decomposition of the set B in two sets which are corresponding to the two new components, one of them F contains the items C_5, C_6, C_7 and the other one G which contains the items C_8, C_9, C_{10}. For Step 5 we have associated the syntax tree from Figure 15 and the partial polish notation obtained for this step is:

$$\oslash \ominus C_1 \oslash C_2 \ominus C_3 C_4 \ominus .$$

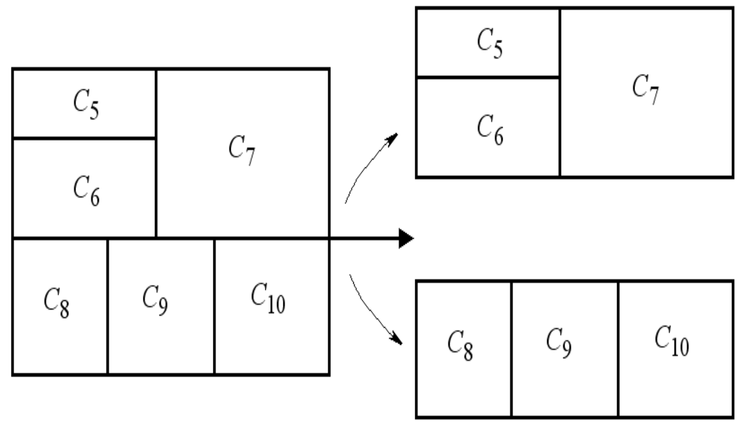

Figure 14: Step5. An horizontal cut

On the fifth component we are doing a vertical cut, a decomposition of the set F in 2 sets like in Figure 16, so we have other two components. One of these is the seventh component, let it be the set $H = \{C_5, C_6\}$ and the other one is a leaf of the syntax tree, C_7, see Figure 17. The partial prefix polish form for the tree from Figure 17 is:

$$\oslash \ominus C_1 \oslash C_2 \ominus C_3 C_4 \ominus \oslash.$$

Using the seventh component we are trying one horizontal cut, a decomposition of the set H in two items C_5 and C_6, see Figure 18, both leaves of the syntax tree from Figure 19. The partial prefix polish notation till this step is:

$$\oslash \ominus C_1 \oslash C_2 \ominus C_3 C_4 \ominus \oslash \ominus C_5 C_6 C_7.$$

Returning to component number 6, we are trying a vertical cut, a decomposition of the set G in two sets, Figure 20 one is $I = \{C_8, C_9\}$ and the other one is the item C_{10}, which is a leaf of the syntax tree from Figure 21. Our partial polish form till this step is:

$$\oslash \ominus C_1 \oslash C_2 \ominus C_3 C_4 \ominus \oslash \ominus C_5 C_6 C_7 \oslash .$$

Using the component 8, we are doing trying an horizontal cut first, but this is impossible so we are doing a vertical one, and this means the decomposition of set I in two items, C_8 and C_9, Figure 22, both leaves of the syntax tree. Finally, we obtain the syntax tree from Figure 23, corresponding to the covering model.

The final prefix polish form, for all the syntax tree is:

$$\oslash \ominus C_1 \oslash C_2 \ominus C_3 C_4 \ominus \oslash \ominus C_5 C_6 C_7 \oslash \oslash C_8 C_9 C_{10}.$$

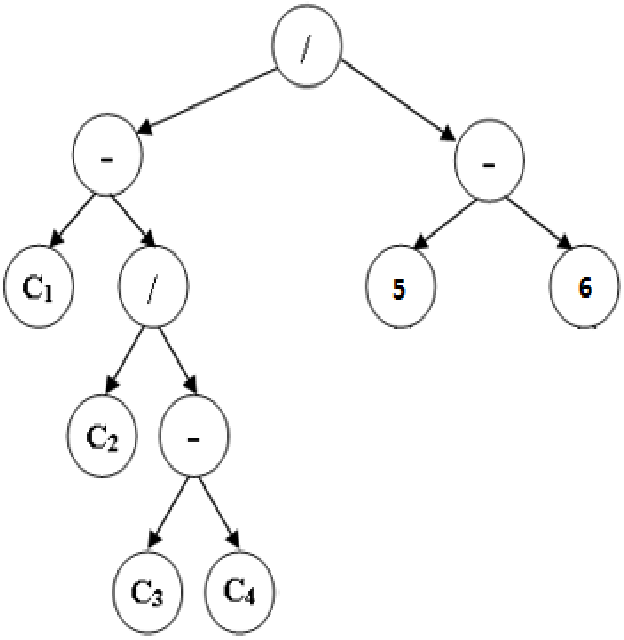

Figure 15: Step5. The syntax tree

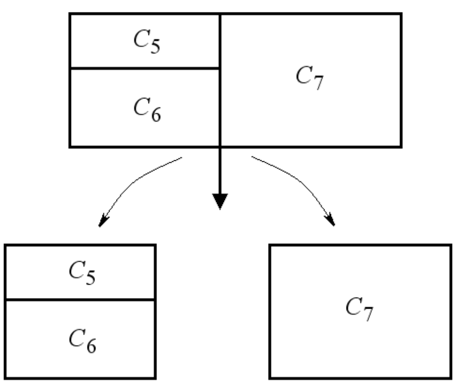

Figure 16: Step6. A vertical cut

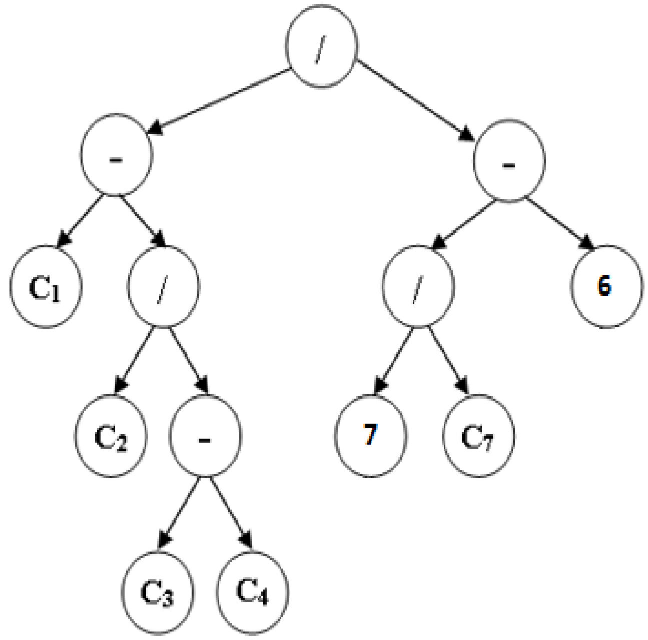

Figure 17: Step6. The syntax tree

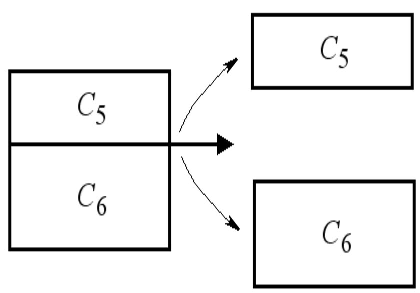

Figure 18: Step7. An horizontal cut

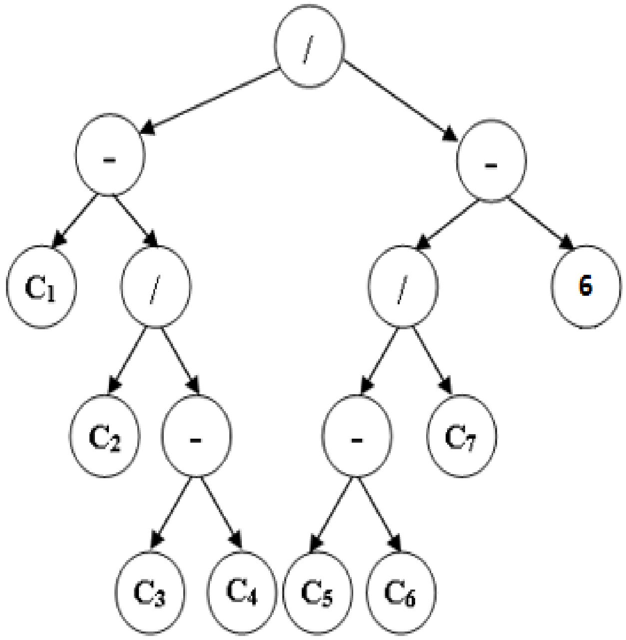

Figure 19: Step7. The syntax tree

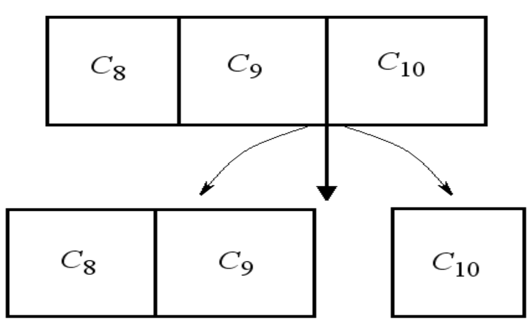

Figure 20: Step8. A vertical cut

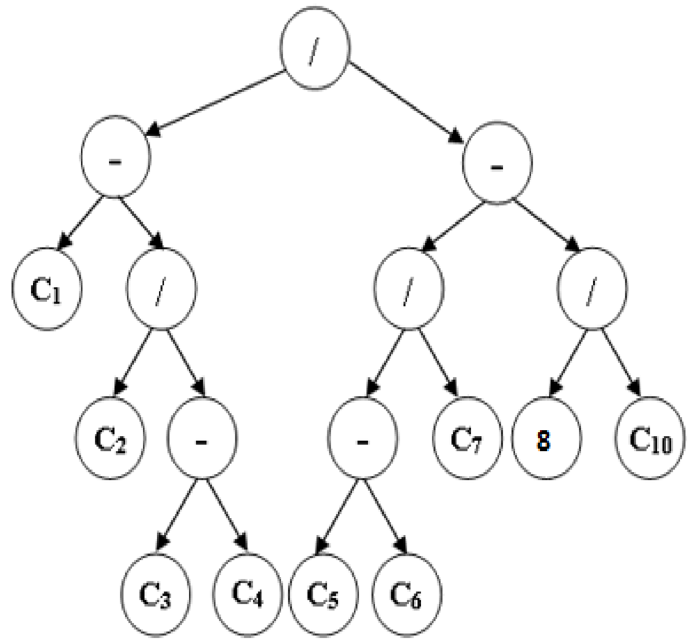

Figure 21: Step8. The syntax tree

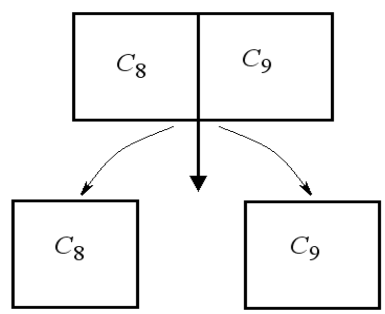

Figure 22: Step9. A vertical cut

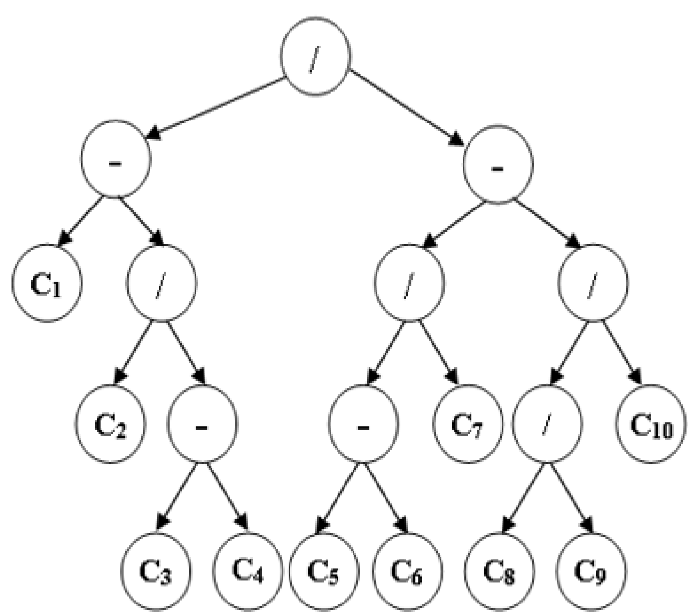

Figure 23: The syntax tree

4.2 Correctness and Complexity

The correctness of the algorithm follows from the theorems 1 and 2, that make the connection between a guillotine cut and the decomposition of the graph G'_d or G'_r in conex components. For determination of the polish notation we preserve only one conex component from this decomposition.

An algorithm for determination of the conex components has the complexity $O(m)$, where m is the number of the arches [2, 3]. So the complexity of V-CUT or H-CUT is also $O(m)$. It follows that the complexity of PREORDER for a rectangular covering model of k items with guillotine restrictions is $O(km)$.

5 Conclusions

The problem, the so-called 2-dimensional guillotine problem, is a constraint on a complete partition of 2-dimensional space. Guillotine partitions were introduced in 1980ies, and they have numerous applications [1, 5] in computational geometry, computer graphics, etc.

The partitioning of 2-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 2-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.

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 model and after that, to use our algorithm for verifying the guillotine restrictons of the pattern.

Acknowledgements: The research was supported by the Transilvania University of Brasov and, in the case of the first author, it was also supported by the Grant IDEI no. 134/2007.

References:

[1] Chi-Wing Fu, Tien-Tsin Wong, Pheng-Ann Heng: Triangle-based view Interpolation without depth-buffering, *Journal of Graphics Tools*, Vol. 3, Issue 4, 1998, pp.13-31

[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] Godoy-Calderon, S., Batiz, J.D., Lazo-Cortez, M.: A non-Classical View of Coverings and its Implications in the Formalization of Pattern Recognition Problems, *Proceedings of the WSEAS Conference*, 2003, paper 459-151.

[6] Iacob, P., Marinescu, D., Luca, C.: L-Shape room. *Proceeding of WMSCI*, Orlando Florida USA 2005 Vol III pp. 175-179

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

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

[9] Iacob, P.,Marinescu, D. and Kiss-Jakab, K: Some algorithms for generating receipts in the rectangular cutting-covering problem, *NAUN, Journ. of Math. Models and Methods in Applied Science*, Issue 3, Vol. 1., 2007, pp. 182-187.

[10] Jin, Z. and Ito, T.: The three Dimensional Bin Packing Problem and its Practical Algorithm, *JSME*, Series , Vol. 46, No.1, 2003.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] Marinescu, D.: Some Turing machines for the determination of the cuts in a cutting-stock model, *Proceeding of the scientific symposion with the contribution of teachers and researchers from the Republic of Moldova*, Brasov 1991, pp 85-96.

[17] 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.

[18] 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.

[19] 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.

[20] D. Marinescu and A. Băicoianu, The determination of the guillotine restrictions for a rectangular covering model, *Proc. of the 13-th WSEAS International Conference on Computers*, Crete, 2009, pp. 307-312.

[21] T. Papke, A. Bortfeldt and H. Gehring, Software Programs for Solving a Cutting Problem in the Wood-Working Industry A Case Study, *WSEAS Transactions on Information Science & Applications*, Issue 5, Volume 4, May 2007, pp. 932-938.

[22] K. Prasanna and D. Khemani, Applying Set Covering Problem in Instance Set Reduction for Machine Learning Algorithms, *WSEAS Transactions on Computers*, Issue 3, Volume 3, July 2004, pp. 661-666.

[23] 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.