UNIVERSIDADE DE SÃO PAULO

An and/or-graph approach to the solution of two-dimensional non-guilletine cutting problems

MARCOS N. ARENALES

REINALDO N. MORABITO

Nº 3

NOTAS

Instituto de Ciências Matemáticas de São Carlos

Instituto de Ciências Matemáticas de São Carlos

ISSN - 0103-2577

An and/or-graph approach to the solution of two-dimensional non-guillotine cutting problems

MARCOS N. ARENALES

REINALDO N. MORABITO

Nº 3

N O T A S D O I C M S C

Série Computação

São Carlos

nov. / 1993

N O T A S D O I C M S C

Serie Computacao

Nº 002/93 - NICOLETTI, M.C.; MONARD, M.C. - Herbrand interpretation model and least model within the framework of logic programming

Nº 001/93 - MORABITO, R.; ARENALES, M.N. - An and/or graph approach to the container loading problem

AN AND/OR-GRAPH APPROACH TO THE SOLUTION OF TWO-DIMENSIONAL NON-GUILLOTINE CUTTING PROBLEMS

Marcos N. Arenales

Departamento de Computação e Estatística
Universidade de São Paulo
c.p.668, 13560.970 - São Carlos - S.P., Brasil
e. mail: arenales@icmsc.usp.br

Reinaldo N. Morabito

Departamento de Engenharia de Produção
Universidade Federal de São Carlos
13565.905 - São Carlos - S.P., Brasil
e.mail: ufscarep@brfapesp.bitnet

Abstract: This work is concerned with unconstrained two-dimensional non-guillotine cutting problems, where a rectangular plate is cut into a number of rectangular pieces in such a way as to optimise an objective function. We reduce the space of states by considering non-guillotine cutting patterns which are combinations of guillotine and simple non-guillotine cuts. These cutting patterns can be represented as complete paths in an AND/OR-graph and a branch and bound method is given. Moreover, we provide some rules and heuristics that reduce the graph search and present computational results from some example problems.

Keywords: cutting/packing problems, graph search, branch & bound method, heuristic.

1. Introdution

The *Unconstrained Two-Dimensional Cutting Problem* consists of cutting a *rectangular plate* (L,W), where L is the plate's length and W its width, in order to produce *rectangular pieces* (ℓ_i, w_i), i=1,2,...,m, which are called the *demanded pieces*. To each demanded piece *i* is associated an *utility value* v_i and there is no limit on the quantity of such pieces (if there were a limitation on the quantity of pieces the problem would be called *constrained*). The problem is to determine a *cutting pattern* that maximizes the total sum of utility values. If $v_i=\ell_i w_i$, the problem is equivalent to minimize the total waste. In addition, the cuts made on the plate are always *orthogonal* to one side of the plate.

It is possible to add a number of constraints to the cuts. For example, in several practical applications the cuts are guillotine- typed (guillotine cutting problems). This is the most studied case in the literature. In particular, a branch and bound method and heuristics based on an AND/OR graph problem representation to solve guillotine cutting problems were proposed by Morabito et al [1992] and will be extended in this work to non-guillotine cutting problems.

Bischoff and Dowsland [1982] studied the problem we are focusing in this paper for m=1, i.e., identical demanded pieces (the pallet loading problem). Their method produces a homogeneous cutting pattern which will be considered in section 5 and provides a lower bound to our branch and bound method.

Beasley [1985] dealt with constrained non-guillotine two- dimensional cutting problems. He modelled the problem as a 0-1 linear program. It is interesting to note that (at least to our knowledge) there is no similar modelling to general guillotine problems , i.e., there is no equation/inequation system to represent guillotine cutting patterns. An exception to this are two-staged guillotine cutting problems (see Gilmore and Gomory [1965]). Beasley proposed an exact tree-search method and used Lagrangean relaxation to provide bound.

There are other interesting approaches in Biró and Boros [1984] or Dowsland [1987], for example. The approach used here is more related with Morabito et al [1992] and Bischoff and Dowsland [1982].

2. Non-guillotine cutting

In this section we define the cutting types which will be used to determine cutting patterns. In all of our definitions orthogonal cuts will be implicit.

Definition 1. *(guillotine cut)* A *cut* is *guillotine*-typed if it produces two new rectangles. We call these new rectangles as *partners*. A *cutting pattern* is *guillotine*-typed if it is obtained by successive guillotine cuts.

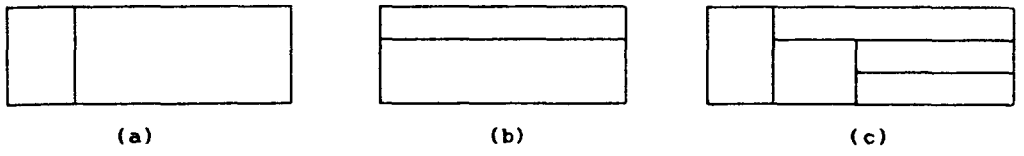


FIGURE 1 a) vertical guillotine cut
b) horizontal guillotine cut
c) guillotine cutting pattern

Defintion 2. *(1^{st}-order cut)* A *cut* is *first-order non-guillotine* - typed (or simply *1^{st}-order*) if it produces five new rectangles arranged in such a way not to form a guillotine cutting pattern. These new rectangles are also called *partners*.

The result of making a 1^{st}-order cut in a rectangle (ℓ, w) is the following five rectangles (see figure 2):

$$\{(c_1, c_3), (\ell - c_1, c_4), (\ell - c_2, w - c_4), (c_2, w - c_3), (|c_1 - c_2|, |c_3 - c_4|)\} \qquad (1)$$

with $c_1 \neq c_2$, $c_3 \neq c_4$, $c_i = 1, .., \ell - 1$, $i = 1,2$ and $c_i = 1, .. w - 1$, $i = 3,4$.

A 1^{st}-order cut is well defined by the 4-vector: (c_1, c_2, c_3, c_4), in the sense that when applied to a rectangle (ℓ, w), it produces the set in (1). The coordenates c_1 and c_2 are called *vertical cuts* and c_3 and c_4 *horizontal cuts*.

Figure 2 illustrates two possible 1^{st}-order cuts, where the rectangles A, B, C, D and E are the ones in (1), recpectively.

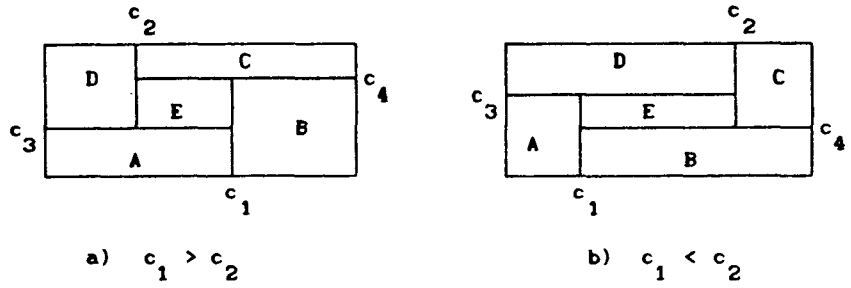


FIGURE 2. FIRST-ORDER CUTS

Observe that the condition: $c_1 > c_2$ implicates in: $c_3 < c_4$ and the degeneration $c_1 = c_2$ or $c_3 = c_4$ implicates in a guillotine cutting pattern.

Definition 3. (*1st-order cutting pattern*) A *cutting pattern* is *1st-order* -typed if it is obtained by successive guillotine or 1st-order non-guillotine cuts.

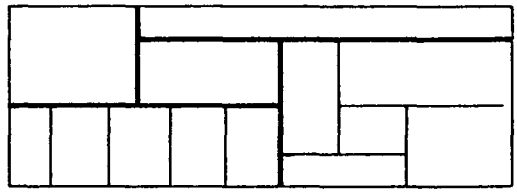


FIGURE 3. 1st-ORDER CUTTING PATTERN

In the next section we shall propose a scheme to generate all possible 1st-order cutting patterns.

It is worth noting that there are non-guillotine cutting patterns which are not 1st-order typed, such as the one illustraded in figure 4.

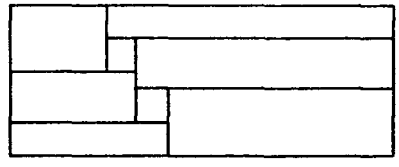


FIGURE 4. SUPERIOR ORDER NON-GUILLOTINE CUTTING

Non-guillotine cutting patterns of superior orders are not considered in this work.

3. AND/OR graph approach

3.1 The Graph Definition

In order to produce smaller rectangular pieces (ℓ_i, w_i), i=1,2,...,m, we firstly cut the plate (L,W) to obtain new rectangles. Then, the intermediary rectangles produced are successively cut in order to obtain the demanded pieces. The cutting of each rectangle is restricted to guillotine or 1^{st}-order cuts.

Suppose that in a first stage the plate (L,W) (A in figure 5a) is cut by a vertical guillotine cut producing rectangles B and C. These are called *successors* or *partner-successors* of A. The two new rectangles are then independently cut. Rectangle B is cut by a horizontal guillotine cut producing D and F, and rectangle C is cut by a 1^{st}-order non-guillotine cut producing G, H, I, J and K. Figure 5a depicts the sequence of cuts leading to the cutting pattern in figure 5b.

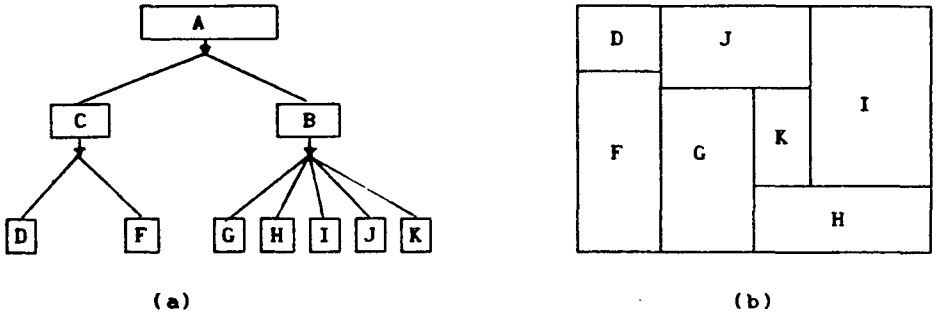


FIGURE 5. A SCHEME TO PRODUCE 1^{st}-ORDER CUTTING PATTERN

There are different manners of cutting a rectangle which indicate different cutting patterns. If we examine all possibilities of cutting, including the option of mantaining a rectangle intact, called *0-cut*, we generate all possible 1^{st}-order cutting patterns.

node definition

For each new generated rectangle we have a subproblem similar to the original problem, which consists of determining all possible cutting patterns for that rectangle. Each rectangle is represented by a node in a graph, where the plate (L,W) is represented by the initial node.

arc definition

The decision of cutting a rectangle (branching a node) is represented by an *AND-arc* which points to the successor nodes (successor-partners) generated.

A guillotine cut is represented by an AND-arc pointing to two new nodes (definition 1) whereas a 1^{st}-order cut is represented by an AND-arc pointing to five new nodes according to (1). The 0-cut is represented as an ordinary arc which points to a single successor node that represents the same rectangle at the node which it emerges from.

Different options of branching a node are indicated by *OR-arcs*. The resulting set of nodes and arcs is called an *AND/OR-graph*.

3.2 Feasibility Properties

Definition 4. (*final node*) A node obtained from a 0-cut (that is, the option of mantaining a rectangle intact) does not allow any extra arc emanating from it and is called *final*.

Definition 5. (*complete path*) Consider the following sequence of arcs in the AND/OR graph: From the initial node choose one arc (and only one) which points to its successors. From each successor node choose again one arc and so on, until all the nodes obtained are final nodes. This sequence is called a *complete path*.

We have defined a 1^{st}-order cutting pattern (see definition 3) as the successive application of guillotine or 1^{st}-order non-guillotine cuts. In addition to these basic cuts, we also introduced the 0-cut in order to stop the cutting process. This corresponds to following a path in the AND/OR-graph where each arc represents a cut.

Theorem 1: *Every 1^{st}-order cutting pattern can be represented as a complete path in the AND/OR-graph described above. And vice-versa, a complete path in the AND/OR-graph corresponds to a 1^{st}-order cutting pattern.*

The correspondence in theorem 1 is not an one-to-one correspondence, since different paths can correspond to the same cutting pattern. In the next section we shall discuss some rules to avoid duplications.

3.3 Optimality Properties

Definition 6. (*solved node*) A node is called *solved* if its optimal 1^{st}-order cutting pattern is known.

Theorem 2: *If all successors of a node are solved, then this node is also solved and the optimal 1^{st}-order cutting pattern is given by re-composing the partner-successors and choosing the most valuable one amongst them.*

To prove theorem 2 it is enough to remember that, by definition 3, a 1^{st}-order cutting pattern (in particular the optimal one) is obtained from a sequence of guillotine and/or 1^{st}-order cuts. That is, there is no chance of obtaining a cutting pattern for a rectangle whithout handling its successors.

Up to now the possibility of rotating (by 90^0) the demanded pieces was not explicitly taken into account. In definition 6 the knowledge of an optimal solution may or not consider the possibility of rotation. But in order to state the value of a final node it is necessary to consider or not piece rotation and we shall consider it since if rotation is not permitted the process is straightforward simplified.

Definition 7. (*final node value*) If a final node represents a demanded rectangle (ℓ_i, w_i) or (w_i, ℓ_i) then its *value* is v_i, otherwise it is zero. That is, if (ℓ, w) is a final node, its value, denoted by $F_0(\ell, w)$, is defined as:

$$F_0(\ell, w)=\begin{cases} v_i \text{ , if } (\ell, w)=(\ell_i, w_i) \text{ or } (w_i, \ell_i) \\ 0 \text{ , otherwise.} \end{cases}$$

Theorem 2 suggests a recursive formula to solve the problem:

recursive formula:

let $F(\ell, w)$ be the optimal 1^{st}-order cutting pattern to the rectangle (ℓ, w). Then,

$$F(\ell, w) = \max \{ F_o(\ell, w),\ F(x, w)+F(\ell-x, w),\ F(\ell, y)+F(\ell, w-y),$$
$$F(c_1, c_3)+F(\ell-c_1, c_4)+F(\ell-c_2, w-c_4)+F(c_2, w-c_3)+F(|c_1-c_2|, |c_3-c_4|) \}$$

where x, y and c_1, c_2, c_3, c_4 assume proper values which will be discussed in the next section.

We will not emphasize this method in this work.

Note that a node representing a rectangle (ℓ, w) such that: *i)* $\ell<\ell_i$ or $w<w_i$ and *ii)* $\ell<w_i$ or $w<\ell_i$ for i=1,...,m, accepts only a 0-cut. Such node corresponds to waste on the cutting pattern.

Definition 8. *(complete path value)* The *value of a complete path* is the sum of the values of its final nodes.

Corollary 1: *The problem of determining the optimal 1st-order cutting pattern consists of determining the most valuable complete path in the AND/OR-graph. This path is determined as soon as the initial node is solved.*

A search strategy is a particular way of traversing the graph, or a way of enumerating its nodes that defines a method to solve the problem.

The complete enumeration (or generation) of the nodes is, in general, computationally unfeasible. In practice it is enough to generate a few of them. Several rules can be designed in order to avoid unnecessary branchings, without loss of generality. The rules designed for guillotine cuttings (see Herz 1972, Christofides and Whitlock 1977 or Morabito and Arenales 1992) can also be used here. In section 4 we describe an additional rule to avoid symmetric 1st-order cuts.

The search can also be reduced with the use of bounds to avoid nonpromising paths, implicitly enumerating the nodes (a branch and bound method). The 'guillotine branch and bound method' described in Morabito and Arenales (1992) can be straightforwardly extended to the non-guillotine case and the lower and upper bounds presented there, as well as the heuristics they provided are also valid. Bischoff and Dowsland (1982) described a specific non-guillotine homogeneous cutting pattern which can also be used as a lower bound. Their work is reviewed in section 5.

In section 6 we shall extend the search strategy described in Morabito and Arenales (1992) to deal with 1st-order non-guillotine cuts.

4. Avoiding Equivalent Patterns

Definition 8. (*equivalent cutting patterns*) Different arrangements of the same set of demanded pieces on a rectangle produce *equivalent cutting patterns*.

Herz (1972) showed that, without loss of generality, the guillotine cuts can be considered in the following **discretization sets**:

vertical cut: $X = \{ x \mid x = \sum_{i=1}^{m} \alpha_i \ell_i,\ 1 \le x \le L-\ell_o,\ \alpha_i \ge 0 \text{ and integer} \}$

horizontal cut: $Y = \{ y \mid y = \sum_{i=1}^{m} \beta_i w_i,\ 1 \le y \le W-w_o,\ \beta_i \ge 0 \text{ and integer} \}$

where $\ell_o = \min\{\ell_i, i=1,..,m\}$ and $w_o = \min\{w_i, i=1,..m\}$.

That is, if any guillotine cutting pattern has cuts taken out of the discretization sets, there is an equivalent guillotine cutting pattern with all cuts belonging to the discretization sets. Christofides and Whitlock (1977) gave recursive formulae to generate these sets (see also Morabito and Arenales, 1992).

The same assertion is still valid for the cutting patterns considered in this work. As we are considering the possibility of rotating the pieces, the cuts can be taken as integer linear combinations of length and width of the pieces, that is, in the discretization sets above X and Y

$$x = y = \sum_{i=1}^{m} \alpha_i \ell_i + \sum_{i=1}^{m} \beta_i w_i$$

Theorem 3: *Consider any 1st-order cutting pattern. There is an equivalent 1st-order cutting pattern so that all cuts belong to the discretization sets.*

proof: We give a sketch of the proof. Consider that the first cut is a 1st-order cut with $c_1 > c_2$ (see figure 2a). By not overlaying the pieces in the northwest rectangle (D in figure 2a), pull them to the left in such a way that the left side of any rectangular piece always touches either a right side of another piece or the left side of the plate. Then, pull c_2 to the left up to

touching a right side of the most eastern rectangular piece in rectangle D. The new value for c_2 is a number in $\mathbb{X}$. This process enlarges the northeast and the inner rectangles (C and E respectively in figure 2a), so that an equivalent cutting pattern is obtained but with a new c_2 in $\mathbb{X}$. Similar processes can be applied to to bring c_1 into $\mathbb{X}$, c_3 and c_4 into $\mathbb{Y}$. ∎

Actually, this theorem is valid for any orthogonal cutting pattern.

In order to avoid equivalent cutting patterns the discretization sets can be more reduced. Here we give a rule to avoid symmetric 1st-order cuts.

Figure 6 shows four symmetric equivalent non-guillotine cuttings.

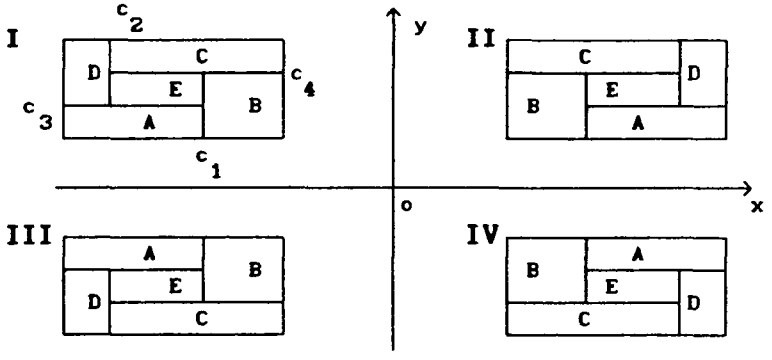


FIGURE 6. SYMMETRIC EQUIVALENT 1st-ORDER CUTS

If the I-cut in figure 6 is characterized by (c_1,c_2,c_3,c_4), then II-cut will be $(L\text{-}c_1,L\text{-}c_2,c_4,c_3)$, III-cut will be $(c_2,c_1,W\text{-}c_3,W\text{-}c_4)$ and IV-cut will be $(L\text{-}c_2,L\text{-}c_1,W\text{-}c_4,W\text{-}c_3)$.

Note that if $c_1>c_2$ then $c_1^{II} = L\text{-}c_1 < c_2^{II} = L\text{-}c_2$, and $c_1^{III} = c_2 < c_2^{III} = c_1$ where c_1^{II}, c_2^{II} and c_1^{III}, c_2^{III} are the vertical cuts in II-cut and III-cut respectively. Then if we restrict vertical cuts such that $c_1>c_2$, II and III cuts will be avoided.

Let $c_1^{IV} = L\text{-}c_2$, $c_2^{IV} = L\text{-}c_1$ be the vertical cuts in IV-cut. In order to avoid the duplication IV-cut, note that

$$c_1^{IV} + c_2^{IV} = L\text{-}c_2+L\text{-}c_1 = 2L\text{-}(c_1+c_2)$$

Then, $(c_1+c_2)<L$ if and only if $c_1^{IV}+ c_2^{IV}>L$. Therefore, by avoiding 1st-order cuts where the sum of vertical cuts is greater than L we avoid the duplication

in IV-cut. However, if $c_1+c_2=L$ (that is, rectangles B and D in I-cut have the same length) it implicates that $c_1^{IV} + c_2^{IV} = L$. In this case, the duplication can be avoided by imposing $c_3 + c_4 \leq W$.

These rules can be summarised as the following:

1st-order symmetry rule:

Consider (c_1, c_2, c_3, c_4) such that

i) $c_1, c_2 \in X$ and $c_3, c_4 \in Y$

ii) $c_1 > c_2$ (then $c_3 < c_4$)

iii) $(c_1 + c_2 < L)$ or $(c_1 + c_2 = L$ and $c_3 + c_4 \leq W)$.

From the above remarks and theorem 3 follows:

Theorem 4: *The 1st-order simmetry rule does not lose generality.*

All the rules which avoid equivalent patterns should be included in the graph search.

5. Bounds and heuristics

Definition 9. (*homogeneous guillotine cutting*) A guillotine cutting pattern that produces the maximum number of one type of piece, without rotating it, is called *homogeneous guillotine cutting*.

The homogeneous guillotine cutting for a rectangle (x,y) using the piece i: (l_i, w_i) produces $\lfloor x/l_i \rfloor \lfloor y/w_i \rfloor$ of such pieces. If the piece i is rotated, that is, the piece (w_i, l_i) is considered, then the homogeneous guillotine cutting produces $\lfloor x/w_i \rfloor \lfloor y/l_i \rfloor$ of such pieces. See figure 7a.

Definition 10. (*homogeneous 1st-order cutting*) Consider a 1st-order cut and its five rectangles in the order given in (1). Fill up each of the rectangles with just one type of piece using homogeneous guillotine cuttings in such a way the piece is successively rotated. To the fifth

rectangle choose the best possibility of rotating or not the piece. The result is a *homogeneous 1st-order cutting*. (See figure 7b).

(a) (b)

FIGURE 7 - HOMOGENEOUS CUTTINGS

The definition 10 produces patterns used by Bischoff and Dowsland. Both homogeneous cutting in definitions 9 and 10 produce easy feasible solutions to the problem and then lower bounds.

Consider a node N representing the rectangle (x,y) and let

$$\mathbb{M}(N) = \{\ i \mid \text{either } l_i \le x \text{ and } w_i \le y,\ \text{or } w_i \le x \text{ and } l_i \le y,\ i=1,\ldots,m\ \}$$

be the set of the pieces that can be produced at node N. Of course, if $\mathbb{M}(N) = \emptyset$ then the only possible cut is the 0-cut indicating a final node whose value is zero (waste).

Lower bounds based on homogeneous guillotine cutting

A homogeneous guillotine cutting using the piece (l_i, w_i) gives:

$$\mathcal{L}_i^+(N) = \{\ v_i \lfloor x/l_i \rfloor \lfloor y/w_i \rfloor\ \}$$

Similarly, if we rotate the piece (l_i, w_i) to obtain (w_i, l_i) the homogeneous cutting provides:

$$\mathcal{L}_i^-(N) = \{\ v_i \lfloor x/w_i \rfloor \lfloor y/l_i \rfloor\ \}$$

Besides, we can define an improved lower bound as:

$$\mathcal{L}^0(N) = \max_{i \in \mathbb{M}(N)} \{\ \mathcal{L}_i^+(N),\ \mathcal{L}_i^-(N)\ \}$$

Suppose a branching from node N leading to N_1 and N_2 (representing a guillotine cut). Then the composition of homogeneous guillotine cutting of N_1 and N_2 provides $\mathcal{L}^0(N_1)+\mathcal{L}^0(N_2)$ which is also a lower bound to node N.

Lower bounds based on homogeneous 1st-order cutting

Consider a 1st-order cut as in figure 2a. The A-rectangle is (c_1,c_3), B-rectangle is $(x-c_1,c_4)$ and so on. The value of the homogeneous 1st-order cutting, according to definition 10, using piece (l_i,w_i) is given by:

$$\mathcal{L}_i^1(N) = \max_{(c_1,c_2,c_3,c_4)} \{ \mathcal{L}_i^+(A)+\mathcal{L}_i^-(B)+\mathcal{L}_i^+(C)+\mathcal{L}_i^-(D)+\max\{\mathcal{L}_i^+(E),\mathcal{L}_i^-(E)\},$$

$$\mathcal{L}_i^-(A)+\mathcal{L}_i^+(B)+\mathcal{L}_i^-(C)+\mathcal{L}_i^+(D)+\max\{\mathcal{L}_i^+(E),\mathcal{L}_i^-(E)\} \}$$

where $\mathcal{L}_i^+(A)$ is obtained similarly to the definition of $\mathcal{L}_i^+(N)$ by changing (x,y) with (c_1,c_3) and analogously to $\mathcal{L}_i^-(A)$, $\mathcal{L}_i^+(B)$, etc. Moreover, the vertical cuts c_1, c_2 and horizontal cuts c_3, c_4 no more need to be in the complete discretization sets $\mathcal{X}$ and $\mathcal{Y}$ respectively. It is enough consider them as positive multiples of l_i and w_i.

The lower bound $\mathcal{L}_i^1(N)$ is the method in Bischoff and Dowsland where just one type of piece is considered.

Improved lower bounds can also be defined as:

$$\mathcal{L}^1(N) = \max_{i\in\mathcal{M}(N)} \{ \mathcal{L}_i^1(N) \}$$

or

$$\mathcal{L}^2(N) = \max_{(c_1,c_2,c_3,c_4)} \{ \mathcal{L}^0(A)+\mathcal{L}^0(B)+\mathcal{L}^0(C)+\mathcal{L}^0(D)+\mathcal{L}^0(E) \}$$

(note that $\mathcal{L}^2$ can produce a non-homogeneous 1st-order cutting pattern)

Upper bounds based on area

A simple upper bound used by Beasley(1985) and Morabito and Arenales (1992) can also be useful here. Consider again a node N representing a rectangle (x,y). Consider a relaxation by taking only the constraint involving area. So, we have an upper bound:

$$\mathcal{U}(N) = \text{maximum} \sum_{i \in \mathbb{M}(N)} v_i a_i$$

$$\text{subject to:} \quad \sum_{i \in \mathbb{M}(N)} (l_i w_i) a_i \leq (xy) \qquad (10)$$

$$a_i \geq 0 \ , \ i \in \mathbb{M}(N).$$

The solution of the above problem is:

$$\mathcal{U}(x,y) = max \{ v_i (x/l_i)(y/w_i), \ i \in \mathbb{M}(N) \} \qquad (11)$$

with $\mathcal{U}(x,y)=0$ if $\mathbb{M}(N) = \varnothing$. Note that this upper bound is not dependent on rotation since $l_i w_i = w_i l_i$.

A Branch & Bound Method

The previous lower and upper bounds can be used to implicitly enumerate a number of nodes in the graph.

Let $V(N)$ be the current best value attached to node N (it is given by a lower bound or by a known complete path emanating from N). It is updated when a better solution is obtained from the successors of N. For example, consider a branching from N, representing a guillotine cut, pointing to N_1, N_2 and suppose that

$$V(N) < \max\{ \mathcal{L}^0(N_1)+\mathcal{L}^0(N_2), \ \mathcal{L}^1(N_1)+\mathcal{L}^1(N_2) \}$$

then $V(N)$ is updated to:

$$V(N) \leftarrow \max\{ \mathcal{L}^0(N_1)+\mathcal{L}^0(N_2), \ \mathcal{L}^1(N_1)+\mathcal{L}^1(N_2) \} \ .$$

Similarly if a 1^{st}-order cut is made pointing to $N'_1, N'_2, N'_3, N'_4, N'_5$ and

$V(N)<$

$$\max\{\mathcal{L}^0(N'_1)+\mathcal{L}^0(N'_2)+\mathcal{L}^0(N'_3)+\mathcal{L}^0(N'_4)+\mathcal{L}^0(N'_5), \ \mathcal{L}^1(N'_1)+\mathcal{L}^1(N'_2)+\mathcal{L}^1(N'_3)+\mathcal{L}^1(N'_4)+\mathcal{L}^1(N'_5)\}$$

then $V(N)$ is updated to:

$V(N)\leftarrow$

$$\max\{\mathcal{L}^0(N'_1)+\mathcal{L}^0(N'_2)+\mathcal{L}^0(N'_3)+\mathcal{L}^0(N'_4)+\mathcal{L}^0(N'_5), \quad \mathcal{L}^1(N'_1)+\mathcal{L}^1(N'_2)+\mathcal{L}^1(N'_3)+\mathcal{L}^1(N'_4)+\mathcal{L}^1(N'_5)\}$$

(of course $\mathcal{L}^2$ or any other lower bound can be considered).

Note that if

$$V(N) \geq U(N_1) + U(N_2)$$

then N_1 and N_2 need not be explicitly considered.

Analogously, if

$$V(N) \geq U(N'_1) + U(N'_2) + U(N'_3) + U(N'_4) + U(N'_5)$$

then N'_1, N'_2, N'_3, N'_4, N'_5 need not be also considered.

In addition, observe that if

$$V(N) = U(N)$$

then $V(N)$ provides the best value to node N, and then the node is *solved*.

When the initial node is solved, we will have found the optimum cutting pattern to plate (L,W).

heuristics

The bounds defined above can also be used in order to produce heuristics that reduce the search space. These heuristics bet that some branchings do not lead to an optimal solution, and so they are discarded.

Heuristic 1 (H1)

Consider a node N and the partner-successors N_1, N_2 (or analogously N'_1, N'_2, N'_3, N'_4, N'_5). We expect $U(N_1)+U(N_2)$ (or $U(N'_1) + U(N'_2) + U(N'_3) + U(N'_4) + U(N'_5)$) to be "substantially" larger than $V(N)$. Otherwise, it might be a sign that $V(N)$ will not be overcome on the branching leading to N_1 and N_2 (or to N'_1, N'_2, N'_3, N'_4, N'_5).

Let λ_1 a fraction defined a priori. Supposing that a guillotine cut is made (if a 1^{st}-order cut is made the heuristic is analogous) we define heuristic *H1* as:

If: $(1+\lambda_1)V(N) \geq U(N_1) + U(N_2)$

Then: abandon the branching leading to N_1 and N_2.

Note that if $\lambda_1=0$, the previous procedure to discard branchings is not a heuristic anymore.

Heuristic 2 (H2)

Consider a node N and the partner- successors N_1, N_2 (or N'_1, N'_2, N'_3, N'_4, N'_5 if the branching corresponds to a 1^{st}-order cut). Let $\mathcal{L}$ be one of the lower bounds defined previously which can be applied to any node. Observe that the current value $\mathcal{L}(N)$ can be greater, smaller or equal to $\mathcal{L}(N_1)+\mathcal{L}(N_2)$. However, if $\mathcal{L}(N)$ is "substantially" greater than $\mathcal{L}(N_1)+\mathcal{L}(N_2)$, it might be a sign that this branching will not produce better values.

Let λ_2 be a fraction previously defined. Supposing that a guillotine cut is made (it is similar if the cut is of 1^{st}-order- typed) the heuristic *H2* is defined as:

If: $\lambda_2\mathcal{L}(N) \geq \mathcal{L}(N_1) + \mathcal{L}(N_2)$

Then: abandon the branching leading to N_1 and N_2.

Note that if $\lambda_2=0$, the above procedure to discard branchings is not a heuristic anymore.

If the number of elements in $\mathbb{X}$ and $\mathbb{Y}$ is large, then the graph's size is computationally untreatable. The following two heuristics discard a number of elements in the discretization sets before the search begins. The first one was suggested by Beasley(1985a), where smaller pieces are not considered in building the discretization sets. The second heuristic imposes a kind of symmetry in the 1^{st} order cutting pattern.

Heuristic 3 (H3)

This heuristic redefines $\mathbb{X}$ and $\mathbb{Y}$ until $|\mathbb{X}| \leq M$ and $|\mathbb{Y}| \leq M$, where M is a number previously chosen. The redefinition of $\mathbb{X}$ is described as (the set $\mathbb{Y}$ can be redefined in a similar way):

first step:

Let $\mathbb{N}=\{1,\ldots,m\}$ be the set of the pieces to be used in determing $\mathbb{X}$.

Build the set $\mathbb{X}$:

$$\mathbb{X} = \{x \text{ such that } x=\sum_{i\in\mathbb{N}}\alpha_i\ell_i \leq \text{L-min}_{j\in\mathbb{N}}(\ell_j),\ \alpha_i\geq 0 \text{ and integer}\}$$

second step:

While ($|\mathbb{X}| > M$) do:

1. Determine $\ell_j = \min\{\ell_i,\ i \in \mathbb{N}\}$
2. Redefine $\mathbb{N} \leftarrow \mathbb{N} - \{j\}$
3. Rebuild $\mathbb{X}$ with the new $\mathbb{N}$.

The above procedure removes from $\mathbb{N}$ the pieces with the smallest length, until set $\mathbb{X}$ reaches the desired size. However, with $\mathbb{X}$ and $\mathbb{Y}$ reduced in this way, we cannot assure optimality, since the optimal solution may depend on the discarded cuts. The reduced sets can even exclude very good ease to find guillotine cutting patterns (see Morabito and Arenales(1992)).

Heuristic 4 (H4)

The cutting pattern in figure 8a is called *self-symmetric* in opposition to the one in figure 8b.

FUGURE 8

Another heuristic to reduce the state space consists in avoiding patterns like the one in figure 8b, restricting the search to self-symmetric 1^{st} order cutting patterns. The heuristic *H4* is described as:

Restrict c_1, c_2, c_3, c_4 in (1) to:

$$c_1 \geq L - L/K$$
$$c_2 \leq L/K$$
$$c_3 \leq W/K$$
$$c_4 \geq W - W/K$$

where $K \geq 1$.

Of course if $K=1$ then the above procedure of restricting cuts is not a heuristic anymore.

6. A search strategy

In this section we present the search strategy used to traverse the graph described in section 3. It is a hybrid strategy combining two basic strategies: BackTracking and HillClimbing.

BackTracking (*BT*) is an important implementation of *depth-first search*. It always chooses to explore the newly not final generated node.

HillClimbing (*HC*) is a strategy based upon local optimization that after expanding a node (i.e. generating all its successors) chooses the best successor to be futher expanded and discards the remaining ones.

If a depth bound is imposed to *BT*, both strategies can be combined by firstly generating all nodes up to the depth bound (using the backtracking strategy) and then choosing the best path whose not final nodes are again expanded up to the depth bound and so on.

It is the same strategy implemented by Morabito and Arenales [1992] who obtained there very good results when apllied to guillotine cutting problems.

Algorithm BT-HC

1. Let *ROOT* be a list which contains initially only the initial node (A node in ROOT is called *root-node*). Define *DB* the depth bound for each expanding from a root node.

2. While *ROOT* is not empty, do:

    3. Let *s* be the first node in *ROOT*. Generate all the successors of the root-node *s*, using the backtracking strategy and respecting *DB*. Take *s* out of *ROOT*.
    4. Choose the most valuable path from *s* and discard the remaining paths (*hillclimbing* strategy). If there are nodes in this path whose depth is equal to *DB* and are not final, put them in *ROOT*.

Remarks

i) In step 3 the generation of the successors from the root-node *s* should take into account the rules discussed in section 4, the branch and bound method and heuristics discussed in section 5.

ii) For simplicity it was implemented an AND/OR-tree instead of AND/OR-graph, that is, there is no checking to test whether a newly generated node (a intermediary rectangle) has already been generated. Indeed, only the most valuable path is kept in memory. This turns the BT-HC algorithm almost independent of memory limitation which is very convenient for micro-computers, quite the opposite of dynamic programming. Note also that, in this implementation, step 4 is actually implicitly considered in step 3. It was specified to make clear of the use of the HillClibing strategy.

iii) In step 4 each chosen path from s corresponds to a section of the complete path from the initial to the final nodes with depth at most equal to DB. Observe that the *HC* strategy, based upon local optmization in each section, does not ensure that the most valuable path (that is, the optimum cutting pattern to the problem) is obtained even in the absence of the heuristics presented in section 5. That is, the algorithm BT-HC is a heuristic search itself.

7. Computational results

The algorithm was implemented in turbo-PASCAL 5.5 and executed on an IBM-PC 286 compatible, with numeric co-processor, 20 MHertz and 640 Kbytes RAM.

Firstly we present the results obtained in a number of examples taken from Dowsland(1984). She studied the Pallet Loading Problem where non-guillotine cutting pattern should be generated for just one type of piece, with rotation allowed. As the implementation of the algorithm does not consider rotation, then m=2 and the dimensions of the demanded pieces are (ℓ_1, w_1) and (w_1, ℓ_1). As the objective is to minimize waste, we define $v_i=\ell_i w_i/LW$ (since $v_1=v_2$, the objective is equivalent to maximizing the number of pieces in the plate). Table 1 summarizes data for 8 examples.

example	piece	plate	example	piece	plate
D1:	(5, 3)	(22, 16)	D5:	(9, 7)	(53, 51)
D2:	(7, 4)	(30, 22)	D6:	(11, 8)	(63, 60)
D3:	(11, 6)	(46, 34)	D7:	(13, 10)	(76, 73)
D4:	(11, 7)	(50, 36)	D8:	(15, 11)	(86, 82)

Table 1 - Dowsland's examples for the Pallet Loading Problem

Experiments with constrained guillotine cutting problems (see Morabito and Arenales 1992) suggest we should to take λ_1=0.01, λ_2=0.9 and *DB*=3. Here, these values proved not to be a good choice. In fact, *DB*=3 makes the graph's size unbearably large and we used *DB*=2 for all examples that were run. Table 2 shows the performance of our algorithm on the data of table 1.

In table 2 the parameters of heuristics *H1* and *H2*, λ_1 and λ_2 were initially taken as 0.01 and 0.9 respectively. The parameter *M* in heuristic *H3* was large enough to consider all discretization points. The parameter *K* in heuristic *H4* was assumed to be 2. Then, the parameters λ_1 and λ_2 were chosen depending on data in the following way: $\lambda_1 = v_1$ and $\lambda_2 = 1 - 0.5\ v_1$ (in this case, $v_1 = v_2 = \ell_1 w_1 / LW$).

Example 1 (D1) was the only one for which the algorithm with the parameters as described above failed to find the optimal solution. In this case, a pattern with 23 pieces (an optimal solution) was obtained by the algorithm only with *K*=1, that is, *H4* was switched off. Assuming *K*=1, the generated graph from D1 with λ_1=0.01 and λ_2=0.9 had 46519 nodes and the running time was 928 seconds; on the other hand, with $\lambda_1 = v_1$ and $\lambda_2 = 1 - 0.5\ v_1$ the graph had 4201 nodes generated and the running time was 173 seconds. Just to illustrate the effect of heuristics, we switched off *H1, H2, H3 and H4*, that is, we took $\lambda_1 = \lambda_2 = 0$, *K*=1 and *M* large enough. Then, the example D1 generated a graph with 167063 nodes in 1610 seconds.

	K=2					
	λ_1 = 0.01 and λ_2 = 0.9			$\lambda_1 = v_1$ and $\lambda_2 = 1-0.5\ v_2$		
example	objective[1]	time (sec)	number of nodes	objective[1]	time (sec)	number of nodes
D1	22[2]	20	3543	22[2]	9	1031
D2	23	24	3855	23	8	907
D3	23	24	3875	23	7	701
D4	23	22	3759	23	8	905
D5	42	1411	155701	42	62	5625
D6	42	1348	152895	42	70	6221
D7	42	1393	153957	42	58	5479
D8	42	1339	154305	42	396	42955

[1] NUMBER OF PIECES IN THE CUTTING PATTERN

[2] THE PARAMETER K=2 HAS LIMITED THE ALGORITHM TO THIS SOLUTION

TABLE 2 - ALGORITHM PERFORMANCE ON THE DOWSLAND'S EXAMPLES

It is worth of note that the cutting pattern presented by Dowsland(1984, pp.899) for the example D8 is not a 1^{st}-order cutting pattern. But there is another one which is of 1^{st}-order type and produces 42 pieces. Figure 9 shows the path in the AND/OR-graph which produces that cutting pattern (homogenous cutting patterns are used in the leaves).

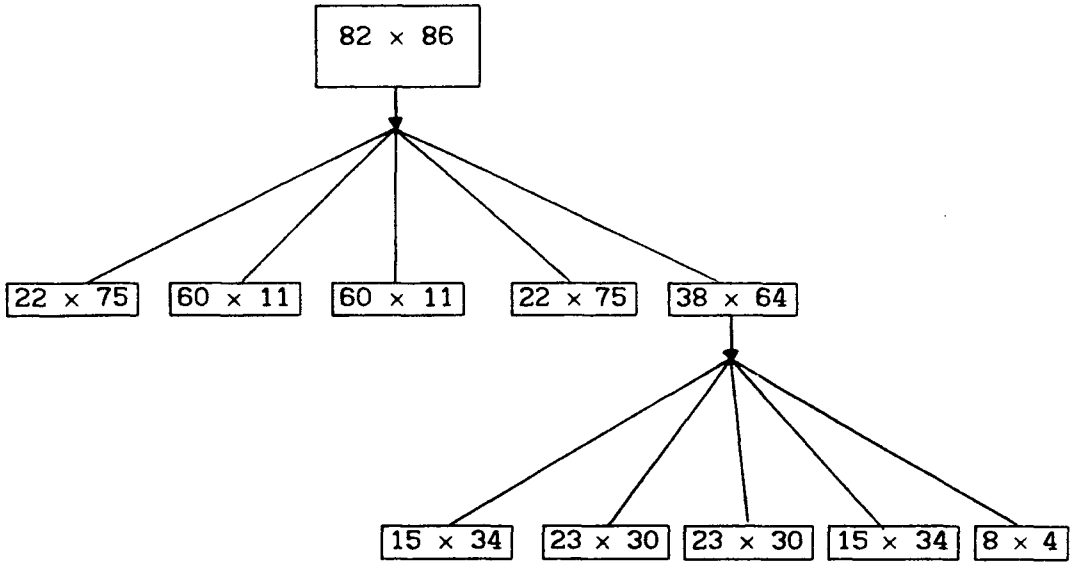


FIGURE 9 - PATH IN THE AND/OR-GRAPH WHICH PRODUCES A SOLUTION TO THE EXAMPLE D8

In addition, 10 randomly generated examples with different dimensions of pieces were run with varying values for λ_1 and λ_2.

These examples were generated as following: with m=5 (that is, 5 different pieces), L=100, W=100, values ℓ_i and w_i, i=1,..,m are uniformly randomly generated such that ℓ_i/L and w_i/W belong to [0.2, 0.5]. The values are defined as $v_i=\ell_i w_i/LW$, i=1,..,m.

Heuristics *H3* and *H4* had their parameters fixed as M=50 (large enough to generate all the discretization points) and K=2 (K=1 makes the algorithm computationally too expensive). The combination K=1 and lower values of M produced poor heuristic solutions, most of them worse than guillotine solutions.

Table 3 summarizes the objective function average values (column Value) and running times in seconds (column Time) for the 10 random examples and for guillotine (row G) and non-guillotine (row NG) cutting patterns. The upper row

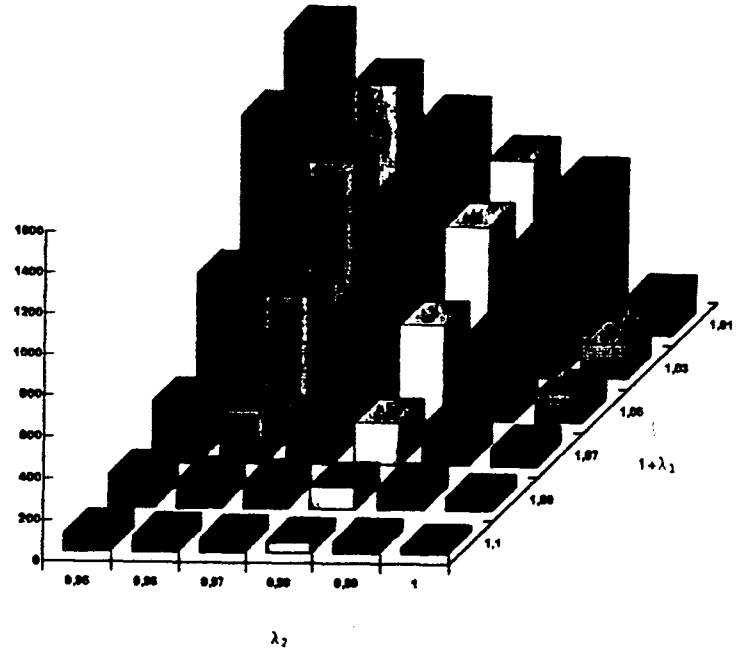
corresponds to values taken by λ_1, whereas the left column corresponds to values taken by λ_2. The algorithm used for guillotine cutting was the same as described in section 6 but the successors in step 3 are only due to guillotine cuttings.

	.01		.03		.05		.07		.09		.1	
	Value	Time	Value	Time	Value	Time	Value	Time	Value	Time	Value	Time
.95	G: .9474	.7	.9474	.9	.9474	.7	.9427	.3	.9333	.1	.9294	.2
	NG: .9486	32.4	.9482	28.3	.9477	13.2	.9390	5.9	.9291	2.9	.9260	2.3
.96	G: .9474	.8	.9474	.7	.9474	.4	.9427	.5	.9333	.2	.9274	.3
	NG: .9486	29.7	.9482	26.4	.9477	12.5	.9390	5.5	.9291	2.9	.9260	2.3
.97	G: .9471	.7	.9471	.7	.9471	.8	.9424	.4	.9333	.2	.9294	.1
	NG: .9483	28.5	.9479	25.2	.9474	12.2	.9387	5.5	.9291	2.6	.9260	2.4
.98	G: .9471	.7	.9471	.6	.9471	.4	.9444	.2	.9333	.2	.9294	.1
	NG: .9483	25.6	.9479	23.0	.9474	11.1	.9387	5.2	.9291	2.9	.9245	2.1
.99	G: .9471	.8	.9471	.7	.9471	.5	.9444	.4	.9333	.1	.9294	.4
	NG: .9483	22.5	.9479	20.0	.9474	10.7	.9387	5.2	.9291	2.9	.9245	2.1
1.0	G: .9471	.3	.9471	.4	.9471	.3	.9444	.3	.9367	.1	.9307	.3
	NG: .9483	17.9	.9479	15.9	.9474	9.1	.9387	4.8	.9333	3.1	.9245	2.4

TABLE 3 - COMPUTATIONAL RESULTS

Table 3 confirms the expected results, namely; if different piece dimensions are at disposal to be combined, then simpler cutting patterns (guillotine, for example) provide very good solutions and sometimes the heuristics (unavoidable) prune those simpler solutions, yielding worse results. Similar remark was made in Morabito and Arenales (1992) where large unconstrained two dimensional guillotine cutting problem was focused. There 2-staged guillotine cutting patterns proved to be better, in the sense of objective function and running time than heuristic solutions to multi-staged guillotine cuttings when the discretization sets were large.

Graphic 1 shows the variation in the number of nodes with λ_1 and λ_2. Each colunm in the graphic gives the average number of generated nodes in the graph for the 10 randomly generated examples.



GRAPHIC 1. VARIATION IN THE NUMBER OF NODES

8. Conclusions

In this paper we presented a new approach to the solution of Two-Dimensional Non- Guillotine Cutting Problems, where a rectangular plate must be cut into an arbitrary number of rectangular-shaped pieces and all the cuts have to be made orthogonal to one side of the plate. We restricted the solutions to first-order cutting patterns, that is, a combination of guillotine and the simplest non-guillotine cuts. A branch and bound method was described to solve the problem. Branchings (represented as AND-arcs pointing to two or five new nodes) are defined by possible cuts on rectangles (represented as nodes) which altogether produce an AND/OR-graph where complete paths correspond to 1^{st} order non-guillotine cutting patterns. A specific rule based on first-order non-guillotine cutting was developed to eliminate simmetric cutting patterns. In order to search the graph, the Backtracking and HillClimbing strategies were combined. Lower bounds were defined based on simple solutions and an upper bound was defined by considering only the area constraint. These bounds were used to design heuristics in order to reduce the state-space. The algorithm, implemented in PASCAL and run on a 286 micro-computer, was able to generate the optimal solutions for a number of known examples from the literature.

References

BEASLEY, J.,"Algorithms for Unconstrained Two Dimensional Guillotine Cutting", *Journal of Operational Research Society* 4, pp. 297-306, 1985a.

BEASLEY, J., "An Exact Two-Dimensional Non Guillotine Cutting Tree Search Procedure", *Operations Research* 33, pp.49-64, 1985b.

BIRÓ, M. and BOROS, E., "Network Flows and Non-Guillotine Cutting Patterns", *European Journal of Operational Research* 16, pp.215-221, 1984.

BISCHOFF, E. and DOWSLAND, W., "An Application of the Micro to Product Design and Distribution", *Journal of the Operational Research Society* 33, pp.271-280, 1992.

CHRISTOFIDES, N. and WHITLOCK, C., "An Algorithm for Two Dimensional Cutting Problems", *Operations Research* 25, pp.30-44, 1977.

DOWSLAND, K., "The Three-Dimensional Pallet Chart: An Analysis of the Factors Affecting the Set of Feasible Layouts for a Class of Two-Dimensional Packing Problems", *Journal of the Operational Research Society* v. 35, n. 10, pp. 895-905, 1984.

DOWSLAND, K., "An Exact Algorithm for the Pallet Loading Problem", *European Journal of Operational Research* 31, pp.78-84, 1987.

DOWSLAND, K., "Efficient Automated Pallet Loading", *European Journal of Operational Research* 44, pp.232-238, 1990.

GILMORE, P. and GOMORY, R. "MultiStage Cutting Stock Problems of Two and More Dimensions", *Operations Research 14,* pp. 1045-1074, 1965.

HERZ, J., "Recursive Computational Procedure for Two Dimensional Stock Cutting", *IBM Journal of Research and Development* 16, pp.462-469, 1972.

MORÁBITO, R. and ARENALES, M., "On Solving Large Two Dimensional Guillotine Cutting Problems", Technical Report: *Notas do ICMSC*, n. 85 Universidade de São Paulo, 1992a (*presented at the ORSA/TIMS Conference*, San Francisco, November 1992).

MORÁBITO, R. and ARENALES, M., "Staged and Constrained Two-Dimensional Guillotine Cutting Problems: A New Approach", Technical Report: *Notas do ICMSC* n. 126, Universidade de São Paulo, 1992b (*presented at the ORSA/TIMS Conference*, San Francisco, November 1992).

MORÁBITO, R., ARENALES, M. and ARCARO, V., "An And/Or-Graph Approach for Two-Dimensional Cutting Problems", *European Journal of Operational Research* 58(2), pp.263-271, 1992.