UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Two-Stage and Constrained Two-Dimensional Guillotine Cutting Problems

Andréa Carla Gonçalves Vianna
Marcos Nereu Arenales
Maria Cristina Nogueira Gramani

Nº 69

NOTAS

São Carlos - SP

UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação
ISSN 0103-2577

Two-Stage and Constrained Two-Dimensional Guillotine Cutting Problems

Andréa Carla Gonçalves Vianna
Marcos Nereu Arenales
Maria Cristina Nogueira Gramani

Nº 69

NOTAS

Série Computação

São Carlos – SP
Mar./2003

Two-Stage and Constrained Two-Dimensional Guillotine Cutting Problems

Andréa Carla Gonçalves Vianna
Departamento de Computação
Faculdade de Ciências – UNESP - Bauru, SP - Brazil
vianna@fc.unesp.br

Marcos Nereu Arenales
Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo – São Carlos, SP - Brazil
arenales@icmc.sc.usp.br

Maria Cristina Nogueira Gramani
Escola de Engenharia
Universidade Presbiteriana Mackenzie – São Paulo, SP
gramani@mackenzie.com.br

Abstract

The two-dimensional cutting problem addressed in this paper consists of cutting a large rectangular plate into a number of strips (first stage) which are then cut to obtain ordered smaller rectangles (second stage), so that an objective function is maximized (the total area used, for example) and the number of ordered pieces are limited to specified numbers. We propose a nonlinear integer optimization model for this problem and some methods to solve it. One of the methods based on Lagrangian relaxation provides an upper bound that is useful to assure the optimality or ε-optimality of solutions. A computational study is also presented.

1. Introduction

Many industrial processes of goods production have an important stage that consists of cutting large pieces (here called *objects*) which can be available in stock, or be produced or bought in the market. The cutting process produces ordered items (smaller pieces) to which a demand that can be external or internal is associated (i.e., a component of a product). Shapes and sizes of objects and items are well specified as well as their quantities (availability for objects and demand for items).

A fundamental subproblem arises in the cutting stage. It consists in defining the way the items should be arranged in each single object. Such an arrangement is called *cutting plan* (high demands of items can require that a cutting plan should be used several times thus being called *cutting pattern* by Gilmore and Gomory, 1961). Note that the geometry is crucial to this subproblem since shapes and sizes of items and objects determine the feasible cutting plans. This paper is concerned with only rectangular shaped items and objects. Besides the geometry, and depending on the application, there are many rules to the cutting process, such as *guillotine* cut (i.e., each cut made on a rectangle produces exactly two new rectangles). Initially, the cuts are made parallel to one side of the object. This is called the *first stage*. Orthogonal cuts (to the cuts in the first stage) are made on the outcome rectangles from the first stage. This is called the *second stage* and so on. A *two-stage guillotine cutting plan* is one obtained within two stages at most and it was introduced by Gilmore and Gomory (1965), who proposed a very efficient solution method, called herewith *decomposition method*. Morabito and Arenales (1995) showed that optimal two-stage guillotine cutting plans can be better in terms of waste than heuristic solutions with more stages.

For problems with low demands of items, it is necessary to define a cutting plan where the quantity of items are limited, and such cutting plans are called *constrained*. Even for high demand cutting problems (called *cutting stock problems*), for which linear optimization with column generation approach is quite suitable (Gilmore and Gomory, 1961, 1965), Haessler (1980) showed that by using instances from industry, the limitation of the number of items for cutting patterns in a starting phase could improve the solution obtained with the pure Gilmore-Gomory approach. Furthermore, Wäscher and Gau (1996) made a computational study with several heuristics to obtain an integer solution when a simple rounding of the continuous solution provided by column generation does not produce a good solution to the integer cutting stock problem. In those heuristics, there is a residual problem that arises with smaller order demand making constrained cutting patterns also useful to solve problems with large demands.

Figure 1 depicts a 2-stage guillotine cutting plan. Note that two dimensions are relevant to the cutting process, so the problem is referred to as a *two-dimensional* cutting problem. In case of cutting bars or bobins, where just one dimensional is relevant the problem is called *one-dimensional*. Dyckhoff (1990) classified cutting problems using other features rather than only dimensions. Nonetheless, the aspects of the cutting problem focused on in this paper, such as the 2-stage guillotine cuts and the limitation on the quantity of items in a cutting plan, are not considered in that classification.

In order to build the best cutting plan, we associate *utility values* to the items and define a combinatorial optimization problem which consists in determining the cutting plan with the *maximum utility value* given by the sum of the utility values of each single item produced by the cutting plan. This problem is referred to as *Cutting Problem*. It is worth remarking that just one object is considered to be cut in the cutting problem, whereas the *cutting stock* problem consists of cutting a number of objects in order to supply the demand. The cutting problem is the most important subproblem that arises when solving cutting stock problems, either in column generation algorithms or in heuristics based on repeated exhaustion reduction (Hinxman, 1980). Furthermore, it is in this subproblem that the main constraints of the cutting process are included, which can vary from case to case. That is why a great effort has been made to solve the cutting problem efficiently.

This paper deals with constrained and 2-stage two-dimensional guillotine cutting problems. The unconstrained version of this problem (i.e., no limitation on the number of items in a plan) was first studied by Gilmore e Gomory (1965), who proposed an exact algorithm based on its decomposition for at most $m+1$ one-dimensional cutting problems (knapsack problems). On the other hand, the constrained cutting problem (without stage constraint) is extensively studied in the literature, see for example, Wang (1983) and subsequent modifications such as Oliveira and Ferreira (1990) and Vasko (1989). These papers use the strategy of producing cutting plans by successively building new 'items' composed of two items (new or not) respecting the sizes of the object. Daza *et al.* (1995) represented Wang's approach within a type of AND/OR-graph and proposed an exact AAO* algorithm.

If on one hand these procedures easily allow us to consider the limitation on the number of items, on the other they are at a disadvantage needing a large amount of memory, besides being difficult to consider the limitation on the number of stages. Other methods to solve the constrained two-dimensional cutting problem using alternative approaches can be found for instance in Christofides and Whitlock (1977) or Christofides and Hadijconstantinou (1995) based on dynamic programming. Methods that handle constrained and staged problems simultaneaously are rare in the literature. For this problem, Morabito and Arenales (1996) proposed a branch and bound method (using some heuristics) based on an AND/OR-graph representation of a cutting plan. Unlike Wang's approach, this method builds a cutting plan by successively cutting the given object and newly generated ones until the items are obtained. Riehme *et al.* (1996) considered the solution of two-stage guillotine cutting stock problems as having an extreme variation on the demand. They overcame the difficulty of determining two-stage and restrict guillotine cutting patterns (the focus of this paper and necessary to the classical column generation approach of Gilmore and Gomory). They aggregated some variables leading to new ones meaning the numbers of strips in a kind of "super-plate", and a final problem to assign the strips into the stock plates. In their approach just one-dimensional cutting stock problem is needed to be solved.

In section 2, the problem described above is formalized where some basic notations are introduced. Section 3 presents a mathematical model of the problem, which is an integer and non-linear optimization problem. In section 4, the methods that we investigated in this paper are given including the Lagrangian relaxation and the subgradient algorithm, a Lagrangian heuristic, and a brief review of the branch and bound method based on AND/OR-graph. Computational experiments are given in section 5.

2. *Problem Definition*

Consider a rectangular plate $L \times W$ (i.e., length L and width W) to be cut into m rectangular pieces $\ell_i \times w_i$, $i=1,2,...,m$. To each piece i, $i=1,...,m$, is associated a positive utility value v_i and a number b_i that provides the maximum quantity in any cutting plan. Non specified pieces are considered waste for which the utility values are zero. The cutting process is 2-stage and guillotine typed. Let a_i be the number of the item $\ell_i \times w_i$ in a cutting plan (note that, for any cutting plan a m-vector $(a_1, a_2, ..., a_m)$ is easily determined, but not the reverse). The constrained and 2-stage two-dimensional cutting problem can be formulated as the following:

$$\textit{Maximize } f(a_1,\dots,a_m)=\sum_{i=1}^{m} v_i a_i$$

subject to that $(a_1, a_2, ..., a_m)$ *corresponds to a 2-stage guillotine cutting plan to* $L\times W$ *and* $0\le a_i \le b_i$, $i=1,...,m$.

In section 3, we will provide the constraints that make a m-vector $(a_1, a_2, ..., a_m)$ correspond to a 2-stage cutting plan.

Note that if $v_i=(\ell_i w_i)/(LW)$, $i=1,...,m$, the problem is equivalent to minimize waste.

We should point out that the 2-stage two-dimensional guillotine cutting problem (unconstrained case) can be efficiently solved to optimality by the decomposition method proposed by

Gilmore and Gomory (1965). In this paper, we straightforwardly extend the Gilmore and Gomory method to the constrained problem, however the optimal solution can be lost. When upper bounds on the number of items in the cutting plan are imposed it can be modelled as an integer nonlinear optmization problem.

3. A Mathematical Model

We follow the same steps as the ones used to model the unconstrained problem, *i.e.*, assuming that the first cut is horizontal we first define one-dimensional cutting patterns for the strips $L\times w_j$, $j=1,\ldots,m$ (of course, if $w_j = w_k$, $j\neq k$, only one of these strips need to be considered) and then, we determine how many times each of these one-dimensional cutting patterns are repeated along the width of the plate (see Gilmore and Gomory, 1965 and Morabito and Arenales, 1995).

For the unconstrained case, it is enough to generate only the best one-dimensional cutting pattern for each strip, but for the constrained problem different cutting patterns for the same strip can be used. We consider a simple instance to make this point and the modelling clear. Let us assume that the plate is: $L\times W=100\times 100$ and $m=5$ items are specified as in Table 1 (the utility values are taken as $v_i=(\ell_i w_i)/(LW)$, $i=1,\ldots,m$):

Table 1.

Note that $w_3=w_4=43$, so that only 4 strips are needed. The optimal unconstrained two-dimensional 2-stage guillotine cutting plan is given in Fig. 1 and the total utility value equals 0.9986, but the maximum number of piece 3 is violated, since $a_3=8 > b_3=4$.

Observe that the best one-dimensional cutting for the strip could not be used a single time, if it produced more than b_i items. Of course, this kind of exceeding could be avoided when the strip is generated by considering constrained one-dimensional knapsack problems (*i.e.*, b_i is an upper bound on the quantity of item type i). However, this is not sufficient, since the overall cutting plan may repeat the strips, and might produce exceeding items (a heuristic approach is devised in section 4.1 to avoid such violation).

Fig. 1.

On the other hand, the optimal constrained and 2-stage two-dimensional guillotine cutting plan for instance 1 is given in Fig. 2, and the total utility value equals 0.9886.

Fig. 2.

Note that the piece type 3 is limited by $b_3=4$, then the most valuable one-dimensional cutting for the strip $L\times w_3 = 100\times 43$ cannot be repeated. However, the optimal solution uses a second cutting pattern for this strip. This happens only because there is a limit on the number of each piece, otherwise only the best one-dimensional cutting for each strip would appear in the overall two-dimensional cutting plan (this is why the decomposition method of Gilmore and Gomory, 1965 is valid).

Also note that the maximum number of strips of the width $w_3=43$ is $K_3=\left\lfloor \frac{W}{w_3} \right\rfloor = \left\lfloor \frac{100}{43} \right\rfloor = 2$, that is, every overall two-dimensional cutting plan uses no more than two strips of width $w_3=43$ and, therefore, no more than two one-dimensional cutting are needed for this kind of strip. For the other strips: $K_1=7$, $K_2=2$, $K_3=2$ (strip 4 is the same as strip 3), $K_5=2$.

In order to consider the first stage as a vertical cut, it suffices to take the plate $W{\times}L$ and the pieces $w_i{\times}\ell_i$, $i=1,\dots,m$ then repeating the steps described above.

The one-dimensional cutting pattern for w_j-strip can be modelled by the m-vector $(\alpha_{kj}^1 \alpha_{kj}^2 \cdots \alpha_{kj}^m)$, $j=1,\dots,r$, where r denotes the quantity of strips of different widths (e.g., in instance 1, $r=4$, although $m=5$), subject to:

$$\ell_1\alpha_{kj}^1+\ell_2\alpha_{kj}^2+\cdots+\ell_m\alpha_{kj}^m \le L,$$
$$\alpha_{kj}^i \ge 0 \text{ and integer},$$
$$\alpha_{kj}^i = 0 \text{ if } w_i > w_k,\ i=1,\cdots,m,$$

where α_{kj}^i is the number of pieces type i in the k^{th} cutting pattern of strip $L{\times}w_j$ and $k=1,\dots, K_j=\left\lfloor \frac{W}{w_j} \right\rfloor$.

The k^{th} cutting pattern of strip $L{\times}w_j$ is called *(k,j)-pattern*. For convenience, we define the set $W_j=\{i$ *such that* $w_i \le w_j\}$. Therefore, the *(k,j)-pattern* is modelled as:

$$\sum_{i\in W_j} \ell_i\alpha_{kj}^i \le L,$$
$$\alpha_{kj}^i \ge 0 \text{ and integer},\ k=1,\dots,K_j,\ j=1,\dots,r. \tag{1}$$

Now, let β_{kj} be the number of *(k,j)-patterns* in the overall two-dimensional cutting plan. Note that the number of $L{\times}w_j$ -strips in the two-dimensional cutting is given by $\sum_{k=1}^{K_j}\beta_{kj}$, then

$$\sum_{j=1}^{r} w_j \sum_{k=1}^{K_j} \beta_{kj} \le W. \tag{2}$$

Also, the total number of pieces i is $a_i=\sum_j \sum_{k=1}^{K_j}\alpha_{kj}^i\beta_{kj}$, which cannot exceed b_i - that is:

$$a_i=\sum_j \sum_{k=1}^{K_j}\alpha_{kj}^i\beta_{kj} \le b_i,\ i=1,\dots,m.$$

The index j in the summation above is such that: $i\in W_j$. In other words, let

$$\delta_{ij} = \begin{cases} 1 \text{ if } i \in W_j \\ 0 \text{ otherwise} \end{cases}.$$

Hence, the constraint above can be written as:

$$a_i = \sum_{j=1}^{r} \sum_{k=1}^{K_j} \delta_{ij} \alpha_{kj}^{i} \beta_{kj} \le b_i, \; i = 1, \ldots, m. \tag{3}$$

Therefore, we are ready to write down a mathematical model in full.

$$\text{Maximize } f(\alpha, \beta) = \sum_{j=1}^{r} \sum_{i \in W_j}^{m} \sum_{k=1}^{K_j} v_i \alpha_{kj}^{i} \beta_{kj} \tag{4}$$

Subject to: $\sum_{i \in W_j} l_i \alpha_{kj}^{i} \le L \qquad j = 1, \ldots, r, \; k = 1, \ldots, K_j$ (5)

$$\sum_{j=1}^{r} w_j \sum_{k=1}^{K_j} \beta_{kj} \le W \tag{6}$$

$$\sum_{j=1}^{r} \sum_{k=1}^{K_j} \delta_{ij} \alpha_{kj}^{i} \beta_{kj} \le b_i \qquad i = 1, \ldots, m \tag{7}$$

$$\alpha_{kj}^{i} \ge 0, \; \beta_{kj} \ge 0 \text{ and integer}, \; i = 1, \ldots m, \; j = 1, \ldots, r, \; k = 1, \ldots, K_j. \tag{8}$$

Observe that the problem (4) to (8) is an integer nonlinear optimization problem. In section 4, we devise some heuristics to solve it. Note that, in absence of constraints (7), the problem becomes the unconstrained 2-stage two-dimensional guillotine cutting problem. In the following, we show how the decomposition method proposed by Gilmore and Gomory (1965) can be deduced from the above model. Let $V_{kj} = \left(\sum_{i=1}^{m} \alpha_{kj}^{i} v_i \right)$ be the value of the *(k,j)-pattern,* then the above model (without constraints (7)) can be rewritten as:

$$\text{Maximize} \sum_{j=1}^{r}\sum_{k=1}^{K_j} V_{kj}\beta_{kj} \tag{9}$$

$$\text{subject to:} \quad \sum_{i=1}^{m}\alpha_{kj}^{i}\ell_i \le L \qquad j=1,\ldots,r,\ \ k=1,\ldots,K_j \tag{10}$$

$$V_{kj} = \sum_{i=1}^{m}\alpha_{kj}^{i}v_i \quad j=1,\ldots,r,\ \ k=1,\ldots,K_j \tag{11}$$

$$\sum_{j=1}^{r} w_j \sum_{k=1}^{K_j}\beta_{kj} \le W \tag{12}$$

$$\alpha_{kj}^{i} \ge 0,\ \beta_{kj} \ge 0 \ \text{ and integer, } i=1,\ldots,m,\ j=1,\ldots,r,\ \ k=1,\ldots,K_j \tag{13}$$

Now, observe that we need only the best pattern for the w_j-strip, because if $V_{\ell j} > V_{kj}$, for $k=2,3,\ldots, K_j$, then in the optimal solution we would have $\beta_{kj}=0$, for $k=2,3,\ldots, K_j$. Therefore, we can omit index k from the model, and we determine, in a first phase, the best plan for each w_j-strip. That is, for $j=1,\ldots,r$ let

$$V_j = \text{Maximum} \sum_{i\in W_j}\ell_i\alpha_j^i \tag{14}$$

$$\textit{Subject to:} \sum_{i\in W_j}\ell_i\alpha_j^i \le L \tag{15}$$

$$\alpha_j^i \ge 0 \text{ and integer}, \ i\in W_j. \tag{16}$$

Finally, in a second phase, a single knapsack problem is solved to determine how many times $L\times w_j$*-strips* (together with its best cutting pattern) should be repeated in the overall two-dimensional plan. That is, the integer knapsack problem remains to be solved:

$$\text{Maximize} \sum_{j=1}^{r} V_j\beta_j \tag{17}$$

$$\text{subject to:} \sum_{j=1}^{r} w_j\beta_j \le W \tag{18}$$

$$\beta_j \ge 0 \ \text{ and integer},\ j=1,2,\ldots,r. \tag{19}$$

The method of Gilmore and Gomory (1965) consists of such a decomposition (*i.e.*, the two-dimensional cutting is decomposed into $r+1$ one-dimensional knapsack problems). We heuristically extend this decomposition method to constrained cutting problems in section 4.1.

4. Solution Methods

Here, we present four methods to solve the constrained and 2-stage two-dimensional guillotine cutting problem. The first three methods are based on the model (4) to (8). The first one is a straightforward extension of the exact decomposition method described in section 3, but now the optimal solution may be lost. The second approach uses the first one repeatedly trying to improve the solution. The third method is the subgradient method applied to solve the dual Lagrangian problem. Then, alternatively to methods based on model (4)-(8), we review a branch and bound method based on AND/OR-graph representation of the cutting plans (Morabito and Arenales, 1996).

4.1. Decomposition Heuristic Methods

First of all, find the best one-dimensional cutting pattern for each w_j*-strip*, but slightly modified to take into account b_i - the maximum number of piece *i*.

First Step: For $j=1,\dots,r$ (*i.e.,* for each strip), solve the constrained knapsack problem:

$$V_j = \text{Maximum} \sum_{i \in W_j} \ell_i \alpha_j^i \tag{20}$$

$$\textit{subject to:} \quad \sum_{i \in W_j} \ell_i \alpha_j^i \leq L \tag{21}$$

$$0 \leq \alpha_j^i \leq b_i \text{ and integer, } i \in W_j = \{i \text{ such that } w_i \leq w_j\}. \tag{22}$$

Now, with these cutting patterns for each strip at hand, we arrange them along the width.

Second Step: Solve the integer linear optimization problem:

$$\text{Maximize} \sum_{j=1}^{r} V_j \beta_j \tag{23}$$

$$\textit{subject to:} \quad \sum_{j=1}^{r} w_j \beta_j \leq W \tag{24}$$

$$\sum_{j=1}^{r} \alpha_j^i \beta_j \leq b_i, \quad i = 1,\dots,m \tag{25}$$

$$\beta_j \geq 0 \text{ and integer, } j = 1,\dots,r. \tag{26}$$

Note that the problem in this second step is no longer a knapsack problem because of the maximum number of piece *i* (compare with (17)-(19)). The coefficients α_j^i of variable β_j in constraints (25) have already been determined by problems (20)-(22). Also, this is a heuristic procedure since it considers only one (the best) cutting for each strip and, as illustrated by instance 1, the optimal solution could be made up of different cuttings for a strip. Moreover, Step 2 is not an easy problem to be solved, requiring a more advanced code than one to solve a knapsack problem. Then, instead of solving problem (23)-(26) to optimality, we use a relaxation of it (*i.e.,* a surrogate constraint instead of (24) and (25)) just to identify the most valuable strip among those produced by Step 1.

Step 2': Summing (24) multiplied with γ and each constraint in (25) multiplied with $(1-\gamma)$, and relaxing the integer condition, we have the continuous knapsack problem:

$$\text{Maximize } \sum_{j=1}^{r} V_j \beta_j \tag{27}$$

$$\textit{subject to: } \sum_{j=1}^{r} [\gamma w_j + (1-\gamma)\sum_{i=1}^{m} \alpha_j^i]\beta_j \le \gamma W + (1-\gamma)\sum_{i=1}^{m} b_i \tag{28}$$

$$\beta_j \ge 0,\ j=1,\dots,r, \tag{29}$$

which is a one constraint linear optimization problem, and the only index k of the basic optimal variable is given by:

$$\frac{V_k}{\gamma w_k + (1-\gamma)\sum_{i=1}^{m} \alpha_k^i} = \text{Maximum}\left\{ \frac{V_j}{\gamma w_j + (1-\gamma)\sum_{i=1}^{m} \alpha_j^i}, j = 1,\dots, r \right\} \tag{30}$$

Then, use w_k-*strip*, with its best plan obtained in Step 1, which can be made in two ways:

- *only once (i.e.,* $\beta_k=1,\ \beta_j=0,\ j\neq k$*), or*
- *several times exhaustively (i.e.,* $\beta_k = \text{minimum}\left\{ \left\lfloor \frac{W}{w_k} \right\rfloor, \left\lfloor \frac{b_i}{\alpha_k^i} \right\rfloor, i = 1,\dots, m \right\}$, $\beta_j=0,\ j\neq k$ *).*

(this second way was used in computational experiments)

Then, update $W \leftarrow W - w_k\,\beta_k$ and $b_i \leftarrow b_i - \alpha_k^i\,\beta_k$ and repeat Step 1.

Note that the first way gives the chance of more one-dimensional cuttings for $L \times w_k$ to be included in the overall 2-dimensional cutting and it does not inhibit the same strip with the same cutting to be repeated.

4.2. A truncated AND/OR-tree

Consider the plate $L \times W$ and assume that in a first stage a horizontal guillotine cut is made at p $(0<p<L)$, producing two new rectangles: $p \times W$ and $(L-p) \times W$. If one of these rectangles is, in sequence, cut vertically (producing two new rectangles), then no extra horizontal cuts should be made in advance,

otherwise a 3-stage cutting plan would be produced. This sequence of cuts together with intermediate rectangles can be formalized as an AND/OR-graph (see section 4.4). Here, instead of completely searching for the AND/OR-graph, we stop it just after the first guillotine cut (horizontal or vertical) and apply the decomposition heuristic of section 4.1 on both resulting rectangles. Similar ideas were used by Fayard *et al.* (1998). The heuristic is summarized as the following.

1. For all p such that $p=\sum_{i=1}^{m}\alpha_i w_i < W$, where $\alpha_i \geq 0$ *and integer*
   1.1 Apply the decomposition method (section 4.1) to the rectangle $p \times W$, obtaining a_i^I type-*i* pieces.
   1.2 Apply the decomposition method to the rectangle $(L-p) \times W$, but with the upper limits b_i updated to $b_i - a_i^I$.
2. For all q such that $q=\sum_{i=1}^{m}\beta_i w_i < W$, where $\beta_i \geq 0$ *and integer*
   2.1 Apply the decomposition method (section 4.1) to the rectangle $L \times q$, obtaining a_i^I type-*i* pieces.
   2.2 Apply the decomposition method to the rectangle $L \times (W-q)$, but with the upper limits b_i updated to $b_i - a_i^I$.
3. Take the best obtained solution from steps 1 and 2.

Observe that the discretization set: $\{p=\sum_{i=1}^{m}\alpha_i w_i < W$, where $\alpha_i \geq 0$ *and integer*$\}$ can be previously determined by formulae given in Christofides and Whitlock (1977) and cuts taken from this set do not exclude any optimal solution.

4.3. The Lagrangian Relaxation and Subgradient Optimization

In this section, we apply the subgradient algorithm to the Lagrangian dual problem obtained by dualizing the nonlinear constraints (7). There is no guarantee that the optimal solution will be found with this method since duality gap can occur. But, upper and lower bounds on the objective function are now determined, which makes it possible to assure the optimality or ε-optimality of a solution. Note that for the early methods nothing could be said about the quality of the obtained solutions.

We define the Lagrangian problem by relaxing the constraints (7) and adding them to the objective function by using the dual multipliers λ_i, $i=1,\dots,m$. Because the constraints (7) are those which limit the number of pieces in the cutting plan, the Lagrangian problem is an unconstrained two-dimensional 2-stage guillotine cutting problem, that is solved to optimality by the decomposition method of Gilmore and Gomory. The Lagrangian problem is given by (h is called the *Lagrangian dual function*):

$$h(\lambda) = \text{Maximum} \quad \sum_{i=1}^{m}\sum_{j=1}^{r}\sum_{k=1}^{K_j}(v_i - \lambda_i)\alpha_{kj}^{i}\beta_{kj} + \sum_{i=1}^{m}\lambda_i b_i \tag{31}$$

$$\text{subject to:} \quad \sum_{i=1}^{m}\alpha_{kj}^{i}\ell_i \le L \quad j = 1,\dots,r, \quad k = 1,\dots,K_j \tag{32}$$

$$\sum_{j=1}^{r} w_j \sum_{k=1}^{K_j}\beta_{kj} \le W \tag{33}$$

$$\alpha_{kj}^{i} \ge 0, \quad \beta_{kj} \ge 0 \quad \text{and integers, } i = 1,\dots,m, j = 1,\dots,r, \ k = 1,\dots,K_j. \tag{34}$$

Let $\alpha = (\alpha_{kj}^{i})_{j=1,\cdots,r,\ k=1,\cdots,K_j}^{i=1,\cdots,m}$ and $\beta = (\beta_{kj})_{j=1,\cdots,r,\ k=1,\cdots,K_j}$ denote the problem variables in matrix notation. For every feasible solution (α,β) (*i.e.*, (α,β) that satisfies (5) to (8) constraints) and $\lambda \ge 0$, it follows that:

$$h(\lambda) \geq f(\alpha,\beta), \tag{35}$$

that is, $h(\lambda)$ is an upper bound to the objective function $f(\alpha,\beta)$. Then, the Lagrangian dual problem arises from determining the minimum upper bound:

$$\mathit{Minimize}_{\lambda\ge 0} \ h(\lambda). \tag{36}$$

The dual problem (36) is then solved by the subgradient algorithm.

Remarks:

1. Sufficient conditions for optimality are:
   - the solution obtained by the Lagrangian problem satisfies the relaxed constrained, and
   - $h(\lambda)=f(\alpha,\beta)$.

However, the above conditions should not be expected to be achieved for every instance, since nonconvex problems can present a positive *dual gap*.

2. A feasible solution (α,β) that satisfies: $h(\lambda) \le (1+\varepsilon) f(\alpha,\beta)$ is called *ε-optimal*, since it holds: $f(\alpha^*,\beta^*)-f(\alpha,\beta) \le \varepsilon f(\alpha^*,\beta^*)$, where (α^*,β^*) is an optimal solution to the problem (9)-(13). In the presence of a *dual gap* (unkown quantity) we may have optimal solutions at hand without knowing it.
3. The most used stopping criteria for the subgradient algorithm are:
   - a maximum number of iterations;
   - small step lengths, producing slight modifications to the solution;
   - lower bound close enough to upper bound, that is, $h(\lambda) - f(\alpha,\beta)$ small enough.

The Subgradient Method

The incumbent solution, that is, the best feasible solution, is denoted by $(\alpha^{best}, \beta^{best})$ and $Z_{LB}^{best}=f(\alpha^{best}, \beta^{best})$ is a lower bound to the objective function (4). Initially a feasible solution is obtained by using the decomposition heuristic of section 4.1. The best generated upper bound is given by Z_{UB}^{best}. Initially, $Z_{UB}^{best}=\sum_{i=1}^{m} v_i b_i$.

Choose $\lambda^0 \geq 0$, an initial Lagrangian multiplier, $\varepsilon>0$, a small number and *it_max*, the maximum number of iterations for the subgradient algorithm. Let $t=0$ and *STOP=FALSE*.

While *STOP=FALSE* and $t \leq it\_max$ do

1. *{upper bound}*
   Solve the Lagrangian problem (31)-(34) to obtain the (possibly unfeasible) solution (α^t, β^t) and the upper bound $Z_{UB}=h(\lambda^t)$.
   If $Z_{UB} < Z_{UB}^{best}$ then update $Z_{UB}^{best} = Z_{UB}$.
2. *{subgradient}*
   Denote the subgradient at iteration t by the *m-vector* $\mathbf{a}^t$ where each component i is given by:
   $$a_i^t = b_i - \sum_{j=1}^{m}\sum_{k=1}^{K_j} \alpha_{kj}^{i}\beta_{kj} \;, i=1,\ldots,m.$$
3. *{optimality test or new iteration}*
   If $\mathbf{a}^t=\mathbf{0}$ then

   STOP=TRUE. The current solution (α^t, β^t) is optimal.

   else

   If $\mathbf{a}^t \geq \mathbf{0}$ (but $\mathbf{a}^t \neq \mathbf{0}$) then

   the current solution is feasible. Let $Z_{LB}=f(\alpha^t, \beta^t)$

   else

   ($a_i^t<0$, for some i) then the current solution is unfeasible. If no improvement has been made during the last iterations, then use a *Lagrangian heuristic* (see below) to obtain a feasible solution, which provides a lower bound Z_{LB}.

   If $Z_{LB} > Z_{LB}^{best}$ then update $Z_{LB}^{best}=Z_{LB}$ and hold the corresponding solution.

   If $Z_{UB}^{best} - Z_{LB}^{best} < \varepsilon \cdot Z_{LB}^{best}$ then *STOP=TRUE*. The incumbent solution is *ε-optimal.*

   Else let $\lambda_i^{t+1} = \max\{0, \lambda_i^t + \theta_t a_i^t\}$, $i=1,\ldots,m$, *where* θ_t *is the steplength.*

   Let $t=t+1$.

A Lagrangian Heuristic

After solving the Lagrangian problem (*i.e.*, dual objective function evaluated), a solution is determined which is in general unfeasible to the original problem. One can devise some procedures to perturb it in order to obtain a feasible solution. These kind of procedures are very dependent on specific problems. Here we used the following procedure.

1. Identify an item exceeding its upper bound. Discard the strip that contains the largest number of such an item. Repeat this step until there is no exceeding items (Fig. 3a),
2. Move up the remaining strips (Fig. 3b),
3. Apply a decomposition heuristic method (section 4.1) to the empty rectangle, but taking into account the items already in the other strips (Fig. 3c).

Table 2.

Fig. 3.

4.4. The AND/OR-Graph Approach

A variety of Cutting Problems can be solved by searching for special graphs, called *AND/OR-graphs*, where a *node* represents a rectangle (the initial plate, or an intermediary rectangle, or an ordered piece) and an *AND-arc*, linking a node N to two other N_1 and N_2 represents a guillhotine cut made on the rectangle in N producing the rectangles in N_1, N_2. *OR-arcs* represent alternative manners of cutting a rectangle. For example, if node N represents a rectangle $x \times y$ and a horizontal guillotine cut, say cut C_1, is made at the point y_1 then two new rectangles are created: $x \times y_1$ and $x \times (y-y_1)$, which are then represented by nodes N_1 and N_2. The point y_1 can be taken without loss of generality from a finite set, called *discretization set* (Herz, 1972, Chirstofides and Whitlock, 1977, Morabito and Arenales, 1996). If an optimal solution (*i.e.*, an optimal cutting pattern) is known by a node then this node is called *solved* and it has no more arc emanating from it.

Therefore, every cutting pattern on a plate $L \times W$ (similarly, on boxes for three dimensional cutting problems and bobbins for one dimensional) is represented as a complete path in the *AND/OR-Graph* (i.e., from the initial node representing the plate $L \times W$ to final nodes which are solved nodes) and one can define different strategies to search it (Morabito and Arenales, 1994, 1996). Fig.4 shows a cutting plan and its corresponding complete path in the *AND/OR-graph*. Note that the whole *AND/OR-graph* is not represented in the figure, but only one complete path. *OR-arcs* (alternative cuts) would represent different paths for different cutting plans. In Fig. 4a, C_1 is the first guillotine cut made on the plate which corresponds to the first *AND-arc* in the complete path of Fig. 4b.

Fig. 4.

This data structure, that is the AND/OR-graph, has been shown to be quite efficient to represent cutting plans whatever the problem dimension (1-dimensional, 2-dimensional, 3-dimensional) and it permits to include along the search other nontrivial constraints. Furthermore, two-dimensional nonguillotine cuttings as well as guillotine cuttings can be represented (Arenales and Morabito, 1995). Next, we show how to include limitations in the number of stage and in the number of items in the cutting plan.

k-stage Problem

Along the search in the *AND/OR-graph*, one can control the number of stages and the number of items (Morabito and Arenales, 1996), that is, such constraints are imbedded in the searching process. In order to control the number of stages, let N be a node of the AND/OR-graph, and *CUT* and *STAGE* be functions of N, defined as:

$$CUT(N) = \begin{cases} 0 & \text{if } N \text{ was obtained by a vertical cut} \\ 1 & \text{otherwise,} \end{cases}$$

STAGE(N) = the number of stages already performed to get node N.

Let N_1 and N_2 be a pair of successors of N, obtained from a guillotine cut (nonguillotine cuts can be represented by an AND-arcs pointing to more than two successors, see Arenales and Morabito, 1995). Of course, $CUT(N_1)= CUT(N_2)$, which are defined by the decision of a vertical or horizontal cut on N. The *STAGE* function is defined for the new nodes as:

For s=1,2 do
If $CUT(N_s) = CUT(N)$
then: $STAGE(N_s) = STAGE(N)$,
else: $STAGE(N_s) = STAGE(N)+1$.

The initial plate $L \times W$ is represented in node $N=0$, for which $STAGE(0)=0$ and $CUT(0)=-1$. Let us assume that k is the maximum number of stages. So, if $CUT(N)=0$ and $STAGE(N)=k$ (*i.e.*, the k^{th} stage is attained to node N after a vertical cut has been made), from now on only vertical cuts will be accepted on N until final nodes are met.

Constrained Problem

In order to control the number of items in the pattern, Morabito and Arenales (1996) proposed a heuristic method. Again, consider a node N and its successors N_1 and N_2 obtained by a guillotine cut. The search is now much more complicated since the decision of including items type i in the node N_1 has an influence on the number of items type-i in the N_2. The heuristic method is summarized in the following. Let $\bar{b}_i(N)$ be the maximum quantity of type-i items that can be produced from the rectangle in N. For $N=0$ (the initial plate), let $\bar{b}_i(0) = b_i$. Initially consider the successor node N_1 with $\bar{b}_i(N_1) = \bar{b}_i(N)$, $i=1,\dots,m$. Only after node N_1 is solved (*i.e.*, all successors of node N_1 have already been solved), consider node N_2 with $\bar{b}_i(N_2) = \bar{b}_i(N) - a_i(N_1)$ where $a_i(N_1)$ is the number of type-i items in node N_1. Then, the same procedure is repeated in the reverse order, that is, initiate with N_2 and then go to N_1.

Upper and lower bounds can easily be formulated to design a branch and bound method. They are also useful to devise a number of other heuristics in order to avoid nonpromissor paths. For example, if $V(N)$ denotes the best current solution for node N and, $L(N)$ and $U(N)$ are lower and upper bounds respectively (as usual in branch and bound methods, $L(N)$ is based on trivial solutions defined for the rectangle in N and $U(N)$ is based on a relaxation of the problem), then the branching from N towards N_1 and N_2 is abandoned if:

i) $(1+\lambda_1)V(N) \geq U(N_1)+U(N_2)$ (here we used λ_1=0.001); or

ii) $\lambda_2 L(N) \geq L(N_1)+L(N_2)$ (here we used λ_2=0.95).

For details, see Morabito and Arenales (1996).

5. Empirical Analysis and conclusions

In order to analyse the computational performance of the methods described in the previous section, we applied them to a number of randomly generated instances, which are grouped into six classes. Each class is determined by combining two parameters: instance size (small and big) and demand level (low, medium and high).

Definition 1. We say that an instance is small or large if, for $i=1,2,...,m$:

Small: $\ell_i \in [0.25L, 0.75L]$ and $w_i \in [0.25W, 0.75W]$ (i.e., $0.25L \leq \ell_i \leq 0.75L$ and $0.25W \leq w_i \leq 0.75W$)

Big: $\ell_i \in [0.10L, 0.50L]$ and $w_i \in [0.10W, 0.50W]$.

Note that for small problems there is less possibility of combining items in the plate than for big problems.

Definition 2. Let $\theta_i = \dfrac{L*W}{\ell_i * w_i}$, which is an approximate number of how many items type i is necessary to fill the whole plate. We say the level of a demand is low, medium or high if, for $i=1,2,...,m$:

Low: $b_i \in [\,0.2\lfloor\theta_i\rfloor,\ 0.4\lceil\theta_i\rceil\,]$

Medium: $b_i \in [\,0.4\lfloor\theta_i\rfloor,\ 0.8\lceil\theta_i\rceil\,]$

High: $b_i \in [\,0.8\lfloor\theta_i\rfloor,\ \lceil\theta_i\rceil\,]$,

where $\lfloor x \rfloor$, $\lceil x \rceil$ means respectively the largest integer less than or equals x and the smallest integer greater than or equals x.

Note that for high level demand instances, a single item may be sufficient, or almost sufficient, to fill in the whole plate, whilst for low level demand at most 40% of the plate might be filled using a single item.

Therefore we have 6 different classes of instances. For short notation, let (S,L)=(small instance, low demand), (S,M)=(small instance, medium demand), (S,H)=(small instance, high demand), (B,L)=(big instance, low demand), (B,M)=(big instance, medium demand), and (B,H)=(big instance, high demand).

We also denote the methods as: DH=Decomposition Heuristic (section 4.1), T-AO=Truncated AND/OR-graph approach (section 4.2), LR=Lagrangian relaxation/subgradient method (section 4.3) and AO=AND/OR-graph approach (section 4.4). The maximum number of iterations in the subgradient method is set at 100 and the Lagrangian heuristic is only performed after 10 iterations without improving the lower bound.

For each class, we vary the number of ordered items: *m=5, 10, 15* and *20*, and randomly generated 100 instances for each value of *m.* The results obtained firstly by cutting horizontally or vertically are presented separately.

The following graphs (Fig. 5-10) summarise the average running time for each method.

Fig. 5-10.

There are some lessons that we can learn by analysing these graphs. Firstly, the lower the demand, the harder it is to solve the instance, and the bigger the instance the harder it is to solve it: the hardest class is (B,L). The subgradient method (LR) is slower than others and it is the most sensitive when *m* increases. Also, more sophisticated heuristics, such as T-AO or AO, run as quickly as DH. This is an important lesson because simple heuristics may perform badly in a number of instances and the argument to use them in practice because they are quick is not valid. In the next part (Fig. 11-16), we will compare the best objective function values (ZLB) obtained by the methods (ZLB in Y-axis).

Fig. 11 – 16.

These graphs show that the AO heuristic performed as good as the LR. Curiously, all the methods applied to the instances in the class (B,L), (i.e., big instances with low demand, the hardest instances in terms of time consuming) tend to produce good solutions as *m* increases. On the contrary, the instances in the class (S,H) (i.e., small instances with high demand, the easiest instances in terms of time), tend to emphasize the quality of solutions. We can see that small instances, independent of demand level, are more complicated to be solved. This happens because an exchange of a single item by any other may completely destroy the solution. For instance, if we combine items 1, 2 and 3 in order to produce a good cutting plan and we change item 3 for item 4, for instance, the cutting plan using items 1, 2 and 4 can be of unacceptably low quality, contrary to big instances for which it is easier to combine items to produce good solutions. Therefore, simple heuristics, such as DH (a greedy type heuristic) tend to produce poor solutions.

In the next part we will analyse the quality of the solutions by comparing them with the upper bound given by Lagrangian relaxation and using the ε-optimal solution definition given in the following.

Definition 3. Let Z_{UB} and Z_{LB}^{method} be upper and lower bounds to the objective function (4) respectively. Z_{LB}^{method} equals the objective function value to a *feasible solution* obtained by the method. We say that such a feasible solution is *ε%-optimal* if $\frac{Z_{UB} - Z_{LB}^{method}}{Z_{UB}} * 100 \leq \varepsilon$.

Note that a *0%-optimal* solution is certainly an optimal solution, however we may have at hands an optimal solution without knowing it. That is, if $Z_{UB} > Z^*$ (where Z^* is the optimal objective function value) and $Z_{LB}^{method} = Z^*$ (i.e., the feasible solution is optimal), the solution is computed as an *ε-optimal, $\varepsilon > 0$.*

The upper bound Z_{UB} is calculated by the subgradient method (see section 4.3) and the lower bound Z_{LB}^{method} is the best objective function value to a feasible solution obtained by a method which can be: Decomposition Heuristic (DH), Truncated AND/OR-graph (T-AO), Lagrangian Relaxation/Subgradient Method (LR), or AND/OR-graph approach (AO).

The Figures 17 to 28 show the results only for *m=5 and 20* (the conclusions are similar for other values of *m*) by depicting the number of *ε-optimal* solutions (*ε = 0%, 0.1%, 0.5%, 1%, 2%* and *5%*) obtained by each heuristic. In the horizontal X-*axis* are the values for *ε* and in the vertical Y-*axis* are the number of *ε-optimal* solutions for each method (e.g., in 100 instances with *m*=5 and the first cut made is Horizontal, *DH* was able to find 19 optimal solutions for the class (B,H), while *AO* found 77 optimal solutions). It is worth noting that *ε-optimal* solutions are possible to be established here because of the given Lagrangian relaxation upper bound.

Fig. 17 – 28.

It is possible to see the influence of the demand level concerning the quality of solutions. For instances with higher demands (classes (B,H) and (S,H)), the best methods, LR and AO, were able to find more than 80% of optimal solutions (*0*-optimal solutions). However, instances with low demands, only fewer optimal solutions were obtained (this influence is higher for big instances). Although the class (B,L) is the worst one to find optimal solutions for both T-AO and AO heuristics were able to find 100% of *5%*-optimal solutions. Note also the good performance of the AO heuristic for all classes, mainly for smaller instances.

To summarize, the demand level has a great influence in the quality of solutions, mainly for big instances. Simple heuristics, such as the greedy-typed, although simple to be designed and implemented are not quicker than others which are more sophisticated, however they may produce low quality solutions. In general, the first cut, horizontal or vertical, has less importance for the quality of solutions.

Acknowledgements

This research was partially supported by FAPESP and CNPq. The authors wish to thank Dr. Reinaldo Morabito, Dr. Horácio Hideki Yanassi and Dr. Luiz Antonio Nogueira Lorena for their useful suggestions.

6. References

ARENALES, M. AND MORABITO, R. (1995), *An AND/OR-graph approach to the solution of two-dimensional non-guillotine cutting problems,* European Journal of Operational Research 84, 599-617.

CHRISTOFIDES, N. AND HADJICONSTANTINOU, E. (1995), *An exact algorithm for orthogonal 2-D cutting problems using guillotine cuts,* European Journal of Operational Research 83, 21-38.

CHRISTOFIDES, N., WHITLOCK, C. (1977), *An algorithm for two-dimensional cutting problems.* Operations Research 25, 30-44.

DAZA, V. P., ALVARENGA, A. G. and DIEGO, J. (1995), *Exact solutions for constrained two-dimensional cutting problems*, European Journal of Operational Research 84, 633-644.

DYCKHOFF, H. (1990), *A Typology of cutting and packing problems.* European Journal of Operational Research. 44, 145-159.

FAYARD, D., HIFI, M. and ZISSIMOPOULOS, V. (1998), *An efficient approach for large-scale two-dimensional guillotine cutting stock problems*, Journal of the Operational Research Society, 49, 1270-1277.

GILMORE, P., and GOMORY, R. (1963), *A linear programming approach to the cutting stock problem - part II,* Operations Research 11, 863-888.

GILMORE, P., and GOMORY, R. (1965), *Multi-stage cutting stock problems of two and more dimensions,* Operations Research 14, 1045-1074.

HAESSLER, R. W. (1980), *A Note on Computational Modifications to the Gilmore-Gomory Cutting Stock Algorithm,* Operations Research 28, 1001-1005.

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

HINXMAN, A. I. (1980), *The Trim-Loss and Assortment Problems: A Survey,* European Journal of Operational Research 5, 8-18.

MORABITO, R., and ARENALES, M. (1995), *Performance of two heuristics for solving large scale two-dimensional guillotine cutting problems,* INFOR 33/2, 145-155.

MORABITO, R., and ARENALES, M. (1996), *Staged and Constrained two-dimensional guillotine cutting problems: An AND/OR-graph approach,* European Journal of Operational Research, 94, 548-560.

OLIVEIRA, J. and FERREIRA, J. (1990), *An improved version of Wang's algorithm for two-dimensional cutting problems,* European Journal of Operational Research 44, 256-266.

RIEHME, J., SHEITHAUER, G. and TERNO, J. (1996), *The Solution of Two-Stage Guillotine Cutting Stock Problems Having Extremely Varying Order Demands,* European Journal of Operational Research 91, 543-552.

VASKO, F. J. (1989), *A Computational Improvement to Wang's Two-Dimensional Cutting Stock Algorithm,* Computer and Industrial Engineering 16, 109-115.

WANG, P. (1983), *Two algorithms for constrained two-dimensional cutting stock problems,* Operations Research 31, 573-587.

WÄSCHER, G. and GAU, T. (1996), *Heuristics for the Integer One-dimensional Cutting Stock Problem: a computational study,* OR Spektrum 18, 131-144.

item	*1*	*2*	*3*	*4*	*5*
$\ell_i \times w_i$	33×14	20×38	15×43	40×43	31×44
b_i	8	4	4	3	3
v_i	0.0462	0.0760	0.0645	0.1720	0.1364

Table 1. Data for instance 1.

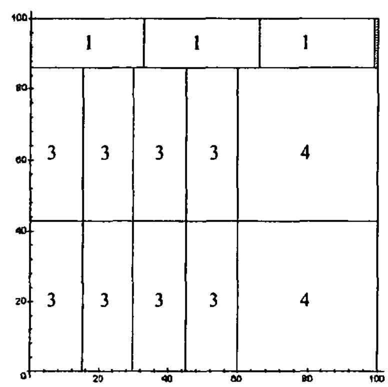


Figure 1. An optimal unconstrained 2-stage two-dimensional guillotine cutting plan for instance 1.

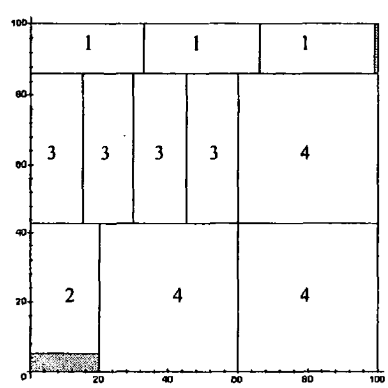
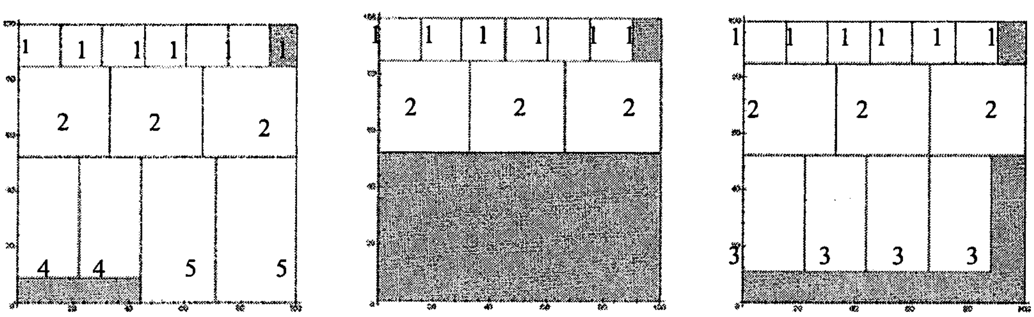


Figure 2. An optimal constrained two-dimensional 2-stage guillotine cutting for instance 1.

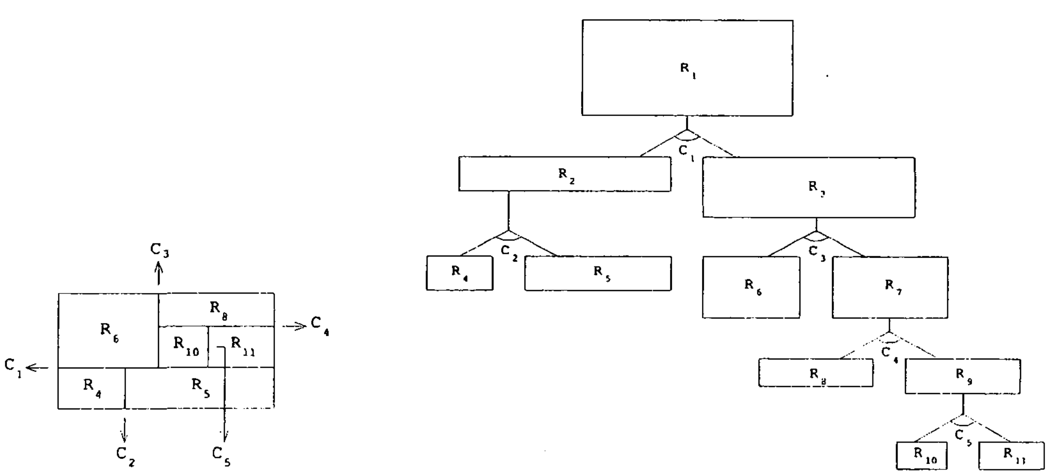
Item	*1*	*2*	*3*	*4*	*5*
$\ell_i \times w_i$	15×15	33×36	22×38	22×43	27×49
b_i	8	3	4	7	1
v_i	0.0225	0.1188	0.0836	0.0946	0.1323

Table 2. Data for instance 2.



(a) (b) (c)

Figure 3. Steps of the Lagrangian Heuristic for data in Table 2.



(a) (b)

Figure 4. (a) A Cutting Plan; (b) A Complete Path in the AND/OR-graph.

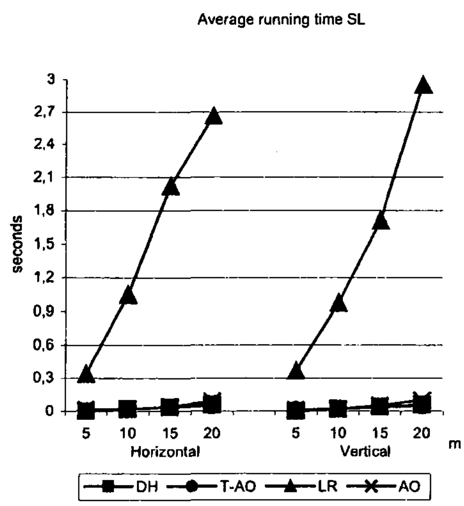


Figure 5

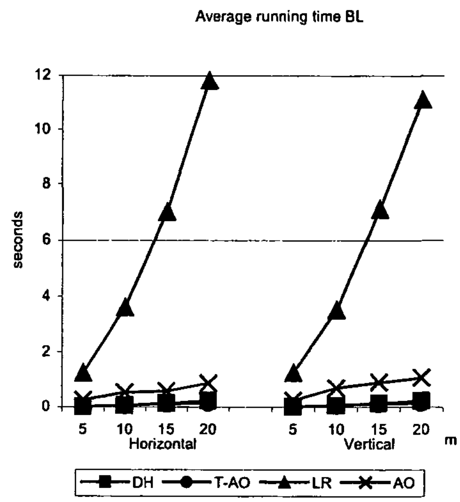


Figure 6

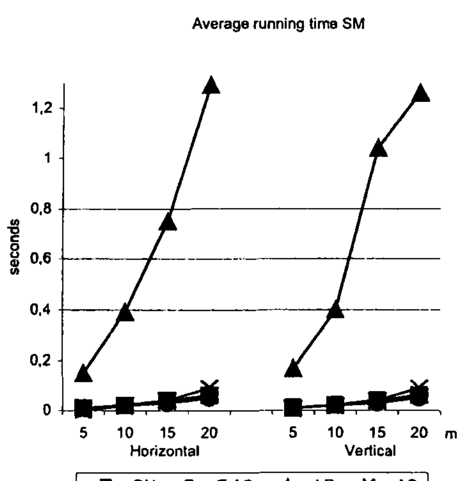


Figure 7

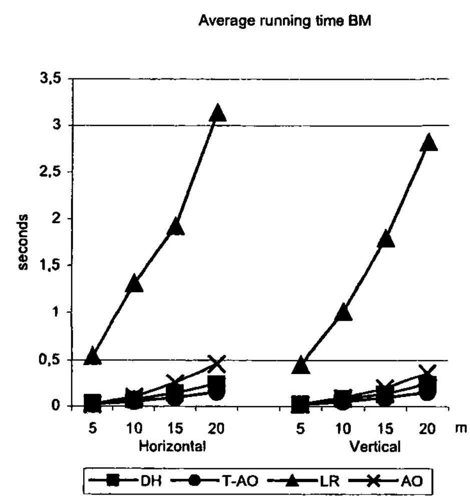


Figure 8

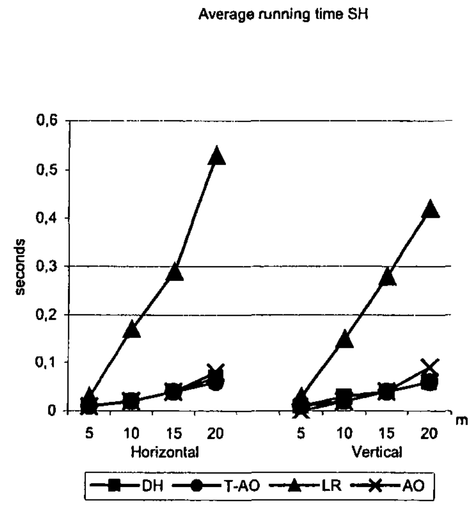


Figure 9

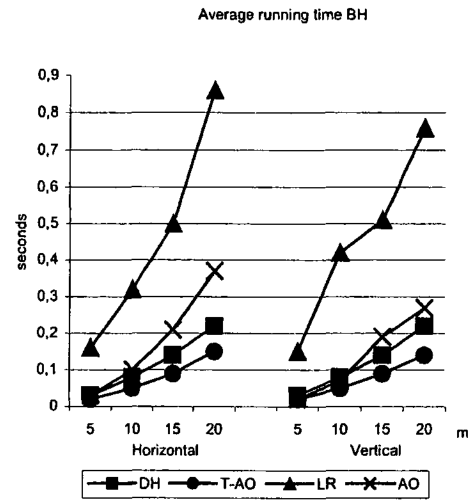


Figure 10

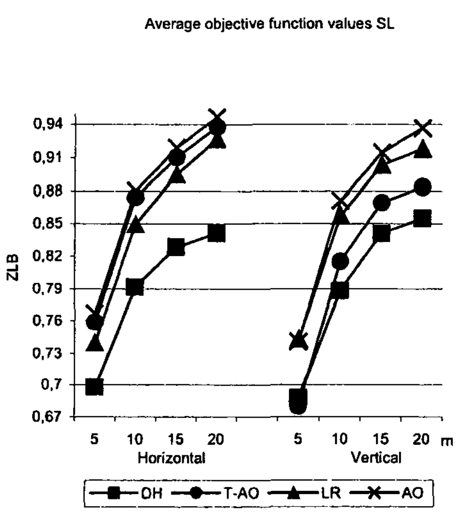


Figure 11

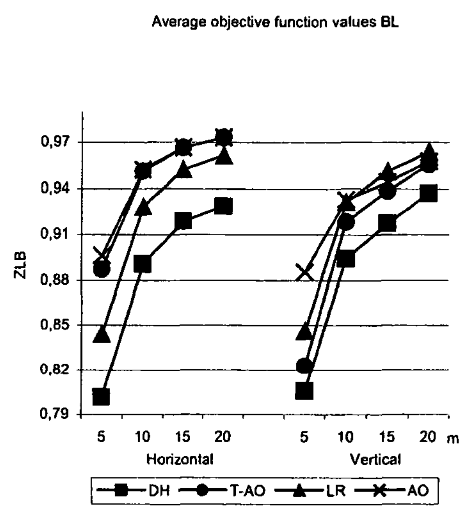


Figure 12

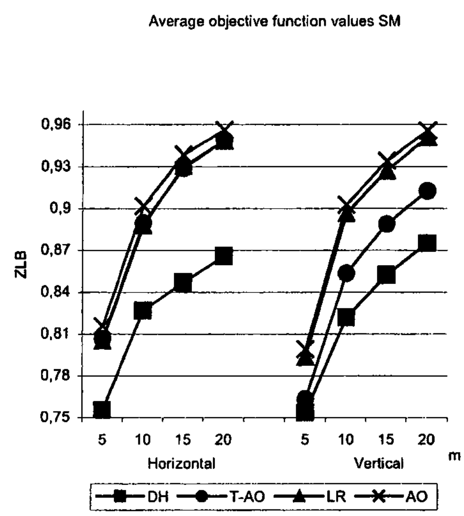


Figure 13

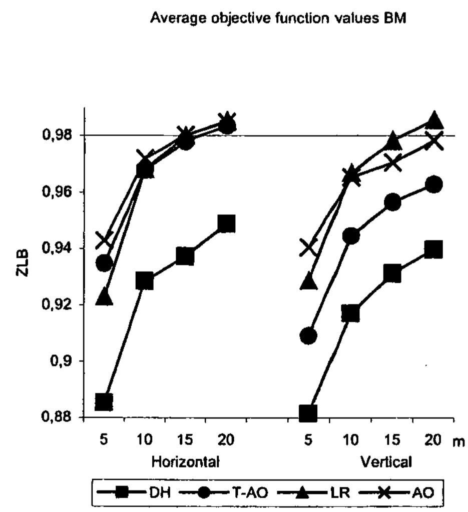


Figure 14

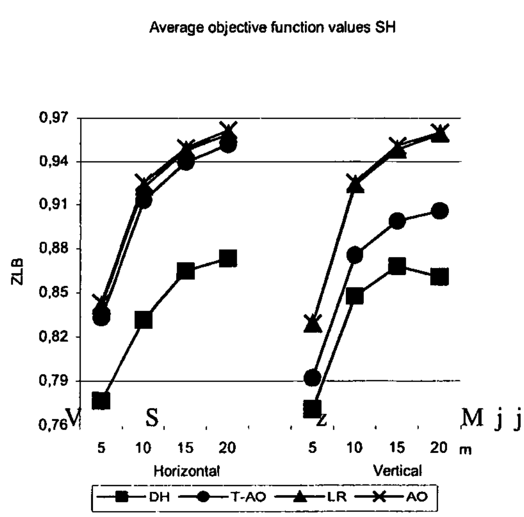


Figure 15

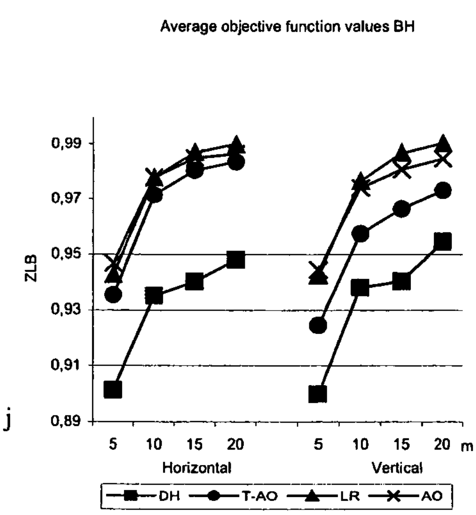


Figure 16

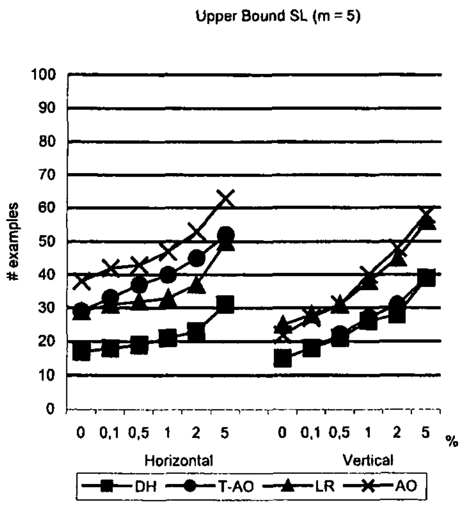


Figure 17

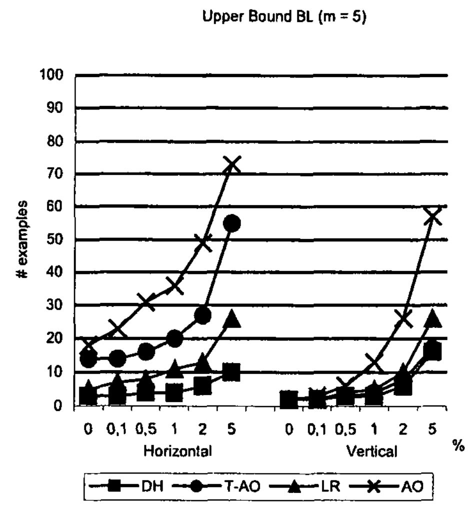


Figure 18

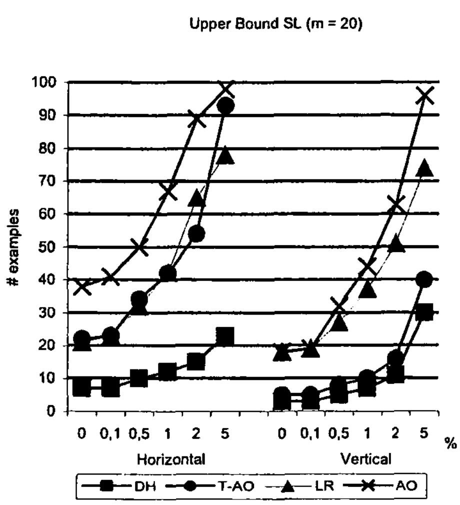


Figure 19

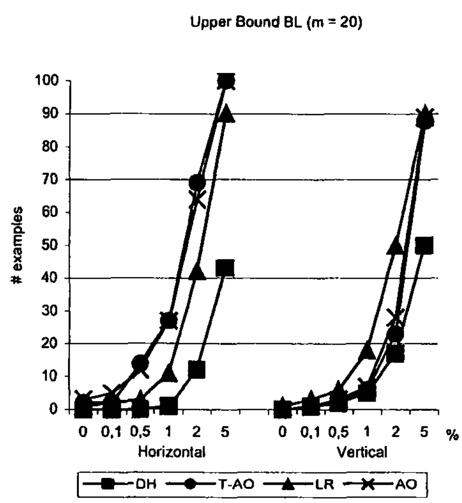


Figure 20

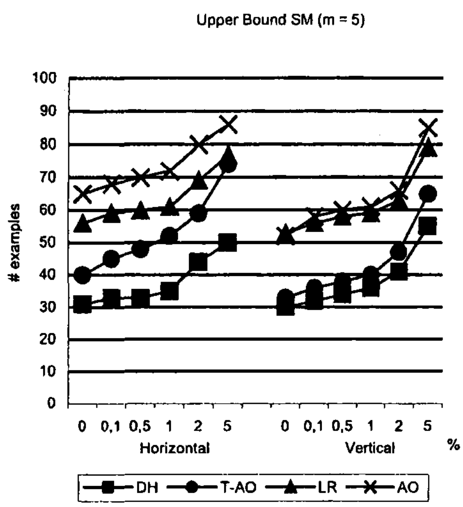


Figure 21

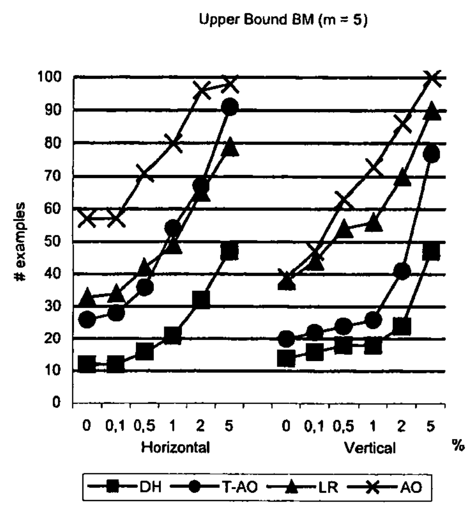


Figure 22

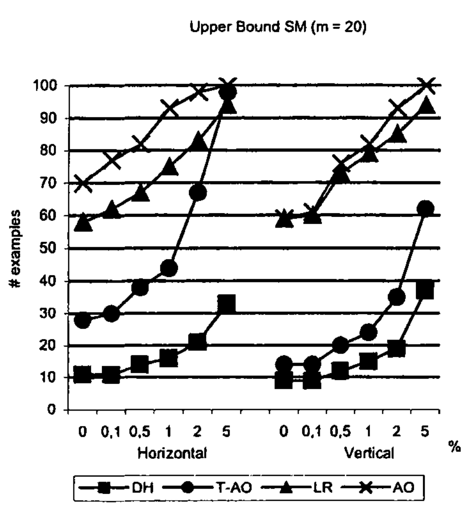


Figure 23

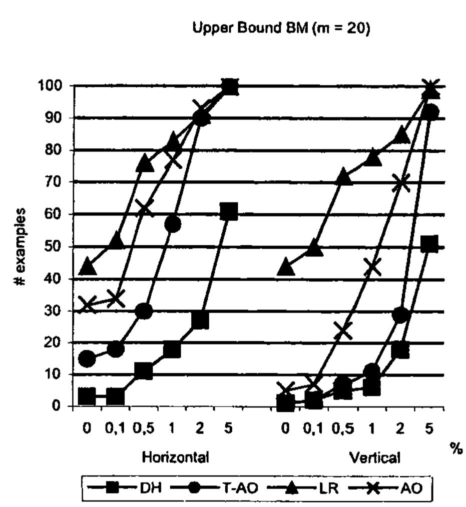


Figure 24

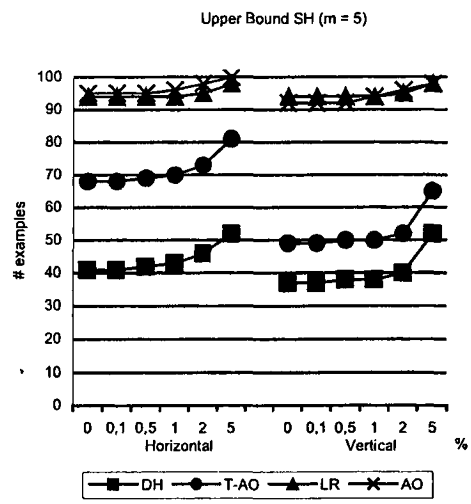


Figure 25

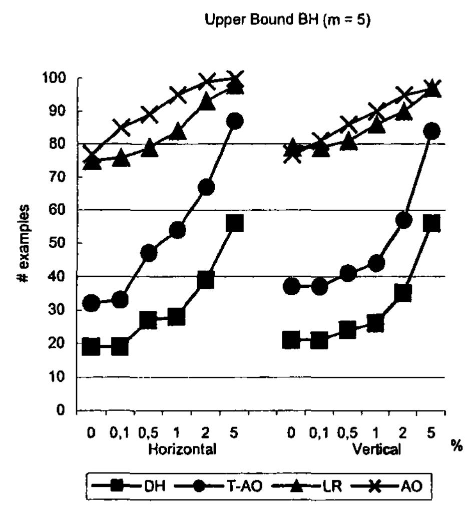


Figure 26

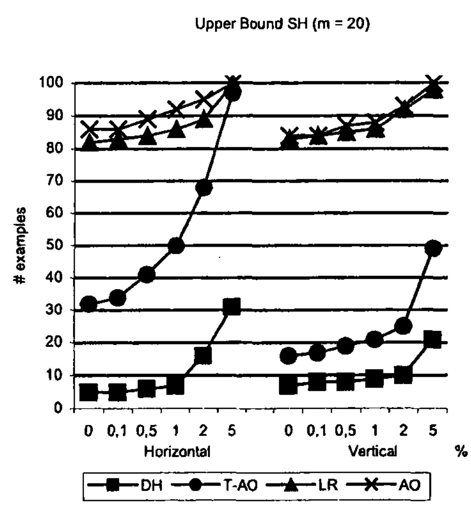


Figure 27

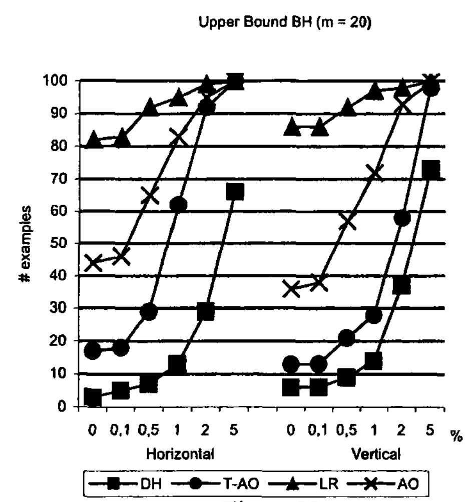


Figure 28

NOTAS DO ICMC

SÉRIE COMPUTAÇÃO

068/2003 ARAUJO, S.A.; ARENALES, M.N.; CLARK, A.R. – A lot-sizing and scheduling problem in a foundry.

067/2003 ARAUJO, S.A.; ARENALES, M.N. – Dimensionamento de lotes e programação do forno numa fundição automatizada de porte médio.

066/2002 VALERIO NETTO, A.; OLIVEIRA, M.C.F. – Industrial application trends and market perspectives for virtual reality and visual simulation.

065/2002 VALERIO NETTO, A.; OLIVEIRA, M.C.F. – Desenvolvimento de um protótipo de um torno CNC utilizando realidade virtual.

064/2002 MARQUES, F.P.; ARENALES, M.N. - O problema da mochila compartimentada e aplicações.

063/2001 TOMÉ, M F; MANGIAVACHI, N; CUMINATO, J A; CASTELO, A – A marker-and-cell technique for simulating unsteady viscoelastic free surface flows.

062/2001 VARGAS, A J C; NONATO, L G. – β-conexão: uma família de objetos tridimensionais reconstruídos a partir de seções planares.

061/2001 OLIVEIRA JR., O N; MARTINS, R T; RINO, L H M ; NUNES, M G V – O uso de interlínguas para comunicação via internet: o projeto UNL/Brasil.

060/2001 SILVA, E Q ; MOREIRA, D A – Use of software agents to the management of distance education courses over the internet.

059/2001 OLIVEIRA, M C F; LEVKOWITZ, H – Visual data exploration and mining: a survey.