A hybrid approach based on genetic algorithms to solve the problem of cutting structural beams in a metalwork company

Carlos Gracia[1], Carlos Andrés[1], Luis Gracia[2]

[1] Departamento de Organización de Empresas, [2] Instituto IDF,
Universidad Politécnica de Valencia, Camino de Vera s/n, 46022 Valencia, SPAIN
{cargraca, candres}@omp.upv.es, luigraca@isa.upv.es

Abstract. This work presents a hybrid approach based on the use of genetic algorithms to solve efficiently the problem of cutting structural beams arising in a local metalwork company. The problem belongs to the class of one-dimensional multiple stock sizes cutting stock problem, namely 1dimensional multiple stock sizes cutting stock problem. The proposed approach handles overproduction and underproduction of beams and embodies the reusability of remnants in the optimization process. Along with genetic algorithms, the approach incorporates other novel refinement algorithms that are based on different search and clustering strategies. Moreover, a new encoding with a variable number of genes is developed for cutting patterns in order to make possible the application of genetic operators. The approach is experimentally tested on a set of instances similar to those of the local metalwork company. In particular, comparative results show that the proposed approach substantially improves the performance of previous heuristics.

Keywords: Cutting stock problem. Pattern generation. Hybrid metaheuristics. GA

1 Introduction

In the context of Operations Research and under the term Cutting and Packing (C&P) is included a set of combinatorial optimization problems with a considerable variety of industrial applications. The study of C&P problems is among the first addressed by Operations Research, Kantorovich (1939) and Gilmore and Gomory (1961, 1963). The practical applications of C&P problems, along with the significant technological development of tools for modeling and implementation, have led to an increase in the number of publications in the last two decades. In particular, more than 400 publications on standard problems (i.e., excluding other variants such as multiple objectives, on line problems or stochastic problems) have been identified by Wäscher et al. (2007) between 1995 y 2005.

Within the area of C&P, this paper addresses the problem known in the literature (Wäscher et al. 2007) as multiple stock sizes cutting stock problem (1dimensional

MSSCSP) in which the property "kind of assignment" seeks the input minimization so that the availability of objects is large enough to meet items demands. The aim will be to minimize the amount of objects needed to meet all the demands on the items. Concerning the "assortment of items", different types of objects have to be considered. Recent applications for the 1dimensional MSSCSP can be found in literature (Aktin and Özdemir 2009; Poldi and Arenales 2009).

Several types of approaches have been developed in the literature to solve the 1dimensional MSSCSP: approaches based on rounding heuristics and residual problems (Holthaus 2002; Poldi and Arenales 2009), approaches based on the column generation algorithm (Gilmore and Gomory 1961, 1963), approaches based on exact methods (Alves and Valério de Carvalho 2008; Belov and Scheithauer 2002), approaches based on heuristic sequencing (Gradisar et al. 1999; Haessler 1992; Vahrenkamp 1996; Vanderbeck 2000), etc.

Evolutionary algorithms and diverse metaheuristics have been widely used (Elizondo et al., 2010; Fan and Mumford, 2010; Ghiani et al., 2010; Gonçalves and Resende, 2011) for solving combinatorial optimization problems. Compared to other techniques, few publications use evolutionary algorithms to solve the cutting stock problem (CSP), probably because of the difficulty in maintaining the essence of the concept of recombination and the frequent violation of the problem constraints. That is, in most cases, when performing a direct recombination of two parent solutions, the children solutions obtained are not feasible. However, a main advantage of using genetic algorithms (GAs) is that, due to large populations of individuals, they offer great diversity when making decisions involving multiple objectives (Wagner 1999).

Previous works that use GAs for the 1dimensional CSP (Hinterding and Kahn 1995; Wagner 1999; Liang et al. 2002) usually present certain drawbacks: they are confined to problems with low quantities of both objects and items; they ignore issues concerning overproduction and reusable remnants; and in encoding pattern oriented approaches, chromosomes are not considered as variable gene strings.

This paper presents a *High-level Relay Hybrid* (HRH) (Talbi 2002) approach based on the use of GAs to efficiently solve the 1dimensional MSSCSP arising in a local metalwork company. In contrast to previous works, the proposed approach handles over and underproduction of beams and reusability of remnants in the optimization process. For this purpose, GAs are used together with other *novel* refinement algorithms that are based on different search and clustering strategies. Moreover, a *new* encoding with a variable number of genes is developed for cutting patterns in order to make possible the application of genetic operators.

The paper is organized as follows. First, the real problem identified in the process of cutting structural metal beams is described and modeled in Section 2. Next, the proposed HRH approach to solve the 1dimensional MSSCSP is developed in Section 3. An experimental study is carried out in Section 4, followed by a comparison with previous methods in Section 5. Finally, Section 6 points out the more outstanding contributions of this research.

2 Problem description and mathematical formulation

2.1 The process of cutting structural beams in a metalwork company

The company aim of this study is dedicated primarily to the storage and sale of steel products such as structural profiles, tubes, mesh, plates and other auxiliary products for both construction and machinery. Although the main activity of the company focuses on the distribution of products, the company also makes in its facilities some transformation processes, such as cutting structural beams of different profiles (IPE, IPN, UPN, HEB, HEA) and dimensions. The 1dimensional MSSCSP appears in several workplaces of the company dedicated to cutting beams in addition to drilling, blasting and painting operations, and it is implemented through band saws prepared for both horizontal and diagonal cuts on a wide range of steel profiles.

Once the production order is placed, the operator responsible for the cutting operation selects, in light of its own experience, a certain beam to be cut from those in stock (objects). The operator selection pursues trim minimization and must prioritize low turnover beams such as those remnants generated by previous cuts. A leftover shorter than 400mm is not reusable in the future and its generation must be avoided. As the demanded beams (items) are cut, they are left in an area next to the cutting machine. This space is limited and its availability must be taken into account in order to define the cutting sequence. Finally, the product is strapped and shipped to customers. Figure 1 shows two pictures of the cutting process

Fig. 1 Cutting machines and a stack of stored beams in the local metalwork company.

As described, the responsible for production carries out the planning of cutting operations based on its experience under no clear criterion. Therefore, it is necessary to develop a procedure to plan this process automatically and taking into account all the constraints and objectives imposed by the company. Under this premise, this work develops a proposal which is explained in the following sections. As illustrative data, and in order to validate the proposed approach in further sections, the number of different lengths (objects) available for each kind of profile (shape and size) varies between four and six. Whereas, the number of different required lengths (items) of a certain profile (shape and size) for the horizon of the production program (one day) is low, around five and seldom reaching ten. These ranges are that small because of the diversity of profiles handled.

2.2 Mathematical model

The following assumptions are considered when modeling the 1dimensional MSSCSP mathematically:

- The cutting operation is made along one dimension of the beam (its length).
- All cuts are considered as if they were straight.
- Despite that the problem should consider the different types of profiles that can be found in the real case, since these are not exchangeable, the problem is separable. Therefore, each production program only considers one structural profile section (shape and size).
- More than one length is available in stock. They may be either standard lengths (std_length) or remnants generated in previous periods.
- The assignment between the required beams and those available in stock is accomplished through the use of cutting patterns.
- The delivery dates are large enough to ensure that the orders given by the production schedule are cut on time.

Three types of data are considered in the mathematical model:

- Those which have to do with the customers' and production orders (items)
  - Beams demanded: quantity and length, size and type.
- Those which have to do with the stock (objects)
  - Beams available in stock: quantity, length, size and type.
- Those which have to do with the cutting process (cutting patterns)
  - Relationship between cutting patterns and beams
  - Scrap generated by each cutting pattern

Thus, the CSP is characterized according to the following parameters

$$p,\ m,\ L=(L_1,...,L_p),\ e=(e_1,...,e_p),\ v=(v_1,...,v_p),\ l=(L_1,...,L_m),\ d=(d_1,...,d_m), \quad (1)$$

where p is the number of different lengths available in stock, and for each one $\{L, e, v\}$ represent their {length, quantity, price}. The index m is the number of different lengths demanded, and for each one $\{l, d\}$ represent their {length, quantity}.

The problem is modeled using cutting patterns. Let $n = m + p$, a cutting pattern will be defined by the n-element column vector $a=(a_1,...,a_n)^T \in \mathbb{Z}_+^n$ that satisfies the foregoing inequalities

$$\sum_{i=1}^{m} l_i a_i \le \sum_{i=1}^{p} L_i a_{i+m}\ , \qquad \sum_{i=1}^{p} a_{i+m} = 1. \quad (2)$$

The elements a_i with i between 1 and m determine how many beams of type i are cut in the cutting pattern. Whereas, the value of elements a_{k+m} corresponding to the stock length k which the cutting pattern uses will be 1, and for the other elements a_{i+m}, where $i \in \{1,\ldots,p\}\backslash k$, will be 0.

Let η be the number of cutting patterns and matrix $A=(a_{ij}) \in \mathbb{Z}_{+}^{\eta}$, where each column represents a cutting pattern. The CSP is formulated as follows:

Find a vector of frequencies of use of each cutting pattern $x=(x_1,...,x_\eta)^T$ *that minimizes the cost of materials to meet the demand of beams from those available in stock*

$$z_{\text{integer}} = \min\{cx : x \in \mathbb{Z}_{+}^{\eta}\} \tag{3}$$

subject to

$$\sum_{j=1}^{\eta} a_{ij} x_j \geq d_i, \quad 1 \leq i \leq m, \tag{4}$$

$$\sum_{j=1}^{\eta} a_{ij} x_j \leq e_{i-m}, \quad \text{m}<i \leq n. \tag{5}$$

One characteristic of the real process of cutting is the reuse of production scrap generated in previous periods. Scrap longer than 400mm is stored, becoming a part of the stock available for future production horizons. Thus, due to the reusability of trims, the model should consider the following:

- The use of patterns with remains greater than 400mm will be promoted over those with remains between 10mm and 400mm.
- The use of patterns obtained on scrap will be promoted over those that use standard lengths, since the marketability of scrap is lower.

Therefore, coefficients c_j of the objective function will depend on the length and reusability of the leftovers r_j generated by each one of the cutting patterns, and on the type of stock length used by each cutting patterns. To this end, two coefficients are introduced: γ_j penalizes the non-reusable leftovers, whereas δ_j gives a bonus on the use of scrap against the use of standard lengths (std_length). Thus, the following objective function coefficients are considered

$$c_j = \gamma_j \, \delta_j \, r_j, \tag{6}$$

where

$$r_j = \sum_{i=1}^{p} L_i a_{i+m,j} - \sum_{i=1}^{m} l_i a_{ij}, \tag{7}$$

$$\gamma_j = \begin{cases} \text{H}_1 & \text{if } 10 \leq r_j < 400 \\ 1 & \text{else,} \end{cases} \qquad \delta_j = \begin{cases} \text{H}_2 & \text{if } \sum_{i=1}^{p} L_i a_{i+m,j} \neq \text{std\_length} \\ 1 & \text{else,} \end{cases} \tag{8}$$

with $\text{H}_1 > 1$ and $\text{H}_2 < 1$.

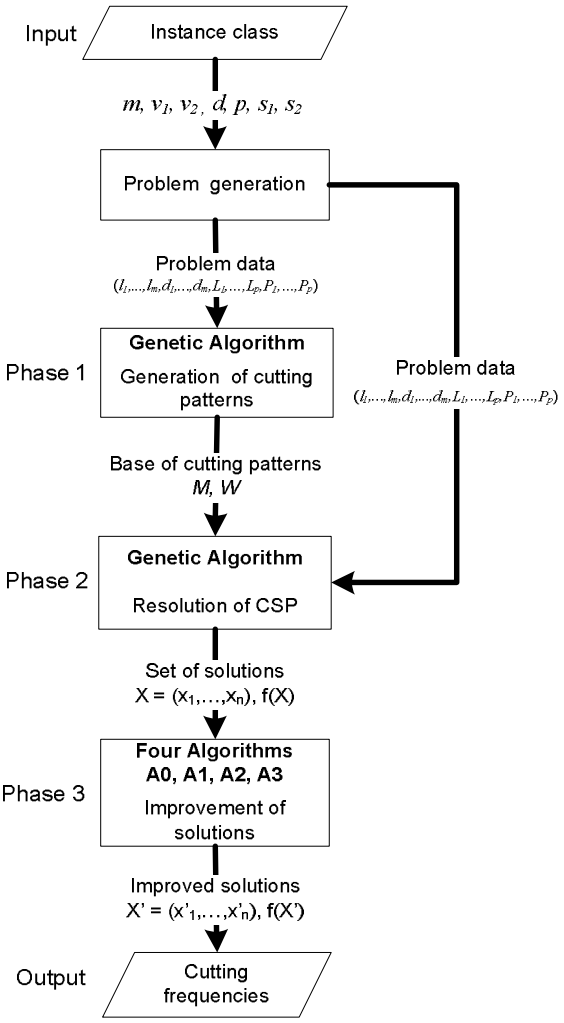

Fig. 2 Overview of the proposed HRH approach based on GAs

3 Solution approach

3.1 General overview of the proposed approach

This section describes the proposed HRH approach based on GAs for solving the 1dimensional MSSCSP identified in the company. The approach consists of a set of techniques and algorithms to be implemented sequentially over several phases, which are shown in Fig. 2. In the first stage (Phase 1) a basis of cutting patterns is generated by applying a GA (Subsection 3.1) to the data previously obtained from a problem generator (Subsection 4.2). Cutting patterns constitute the input information to solve

the CSP itself which is done in two steps: Phase 2 and Phase 3. In this sense, another GA (Subsection 3.2) is applied in Phase 2 and after several iterations a set containing the best solutions is selected. Subsequently, in Phase 3, four different algorithms are applied (Subsection 3.3) to refine and improve the performance (overproduction, dissatisfaction with demands, etc.) of the solutions obtained in the previous stage.

When solving of the CSP with GAs, two different encodings have been used in the literature. The first one represents the solution to the problem in terms of groups of beams (cutting patterns) and has been referred to as "group-based representation". The second one, referred to as "order-based representation", encodes the solution to the problem as an ordered list of all demanded beams. In this case, the sequence of the list makes a difference between one possible solution and another. In this work the solutions to the problem are encoded using the "group-based representation", since it makes feasible the implementation of genetic operators (Hinterding and Kahn 1995). Thus, a solution to the problem will be a chromosome defined by a sequence of n genes, i.e., cutting patterns. As shown in the Fig. 3, each gene is associated with the following information: items that are obtained, object which is cut, and quantity of waste generated by the cutting operation. The use of this encoding requires in advance the cutting patterns.

Demanded lengths:
2, 5, 6, 7, 9 y 10

Stock lengths:
12, 13 y 15

Possible cutting patterns:

Group	Items	Object	Leftover
A	(10)	12	2
B	(6 5 2)	13	0
C	(9 2)	12	1
D	(7 6)	13	0
E	(10 5)	15	0

Representation of one solution containing 10 genes

AACBDDDEEE

Fig. 3 Example of solution using the group-based representation

3.2 Phase 1: Genetic algorithm to generate cutting patterns

Pattern generation has been addressed in literature in two ways: previous to solve the CSP, and on line. Previous generation is used either in small and medium size problems where complete enumeration is possible or in large problems in which only a subset of representative patterns is generated. Online generation, however, is used to solve big problems by column generation techniques. In the case of applying metaheuristics, the cutting patterns have to be generated in advance. For example, Gilmore and Gomory (1961, 1963) use a column generation technique based on simplex method, which has been used subsequently by many researchers. Haessler (1992, 1979) develops a heuristic sequencing procedure. Christofides and Hadjiconstantinou (1995) design an enumerative procedure based on cut orders

(symmetry) to generate the columns. Suliman (2001) proposes a simple heuristic based in a solution method for the knapsack problem that uses a search tree.

A requirement for the good performance of the proposed approach is the generation of a wide variety of efficient cutting patterns, i.e., trim loss objective is achieved as long as a diverse population is provided. Thus, a GA has been developed to generate cutting patterns with the following input parameters: Population size (*Pop*); maximum number of iterations (*iter*); size of cloning proportion (*Elite*); and proportion of individuals generated by recombination and mutation (recombination and mutation rates, *Tmut* and *Trec*) according to the following expressions

$$Pop = Elite + T_{rec}(Pop - Elite) + T_{mut}(Pop - Elite), \tag{9}$$

$$1 = T_{rec} + T_{mut}. \tag{10}$$

Genetic encoding. A cutting pattern is represented by a vector whose elements indicate the ordered sequence of overlapping demanded beams over the length in stock (*k*). In order to provide more diversity to the population of cutting patterns, the number of non-zero genes for each cutting pattern will vary randomly between two integer values given by the following expressions

$$N_{\min} = \frac{L_k \sum_i d_i}{\sum_i d_i \, l_i}, \quad N_{\max} = \frac{L_k}{\min(l_i)} \quad \text{with } i = 1,...,m, \tag{11}$$

where d_i and l_i are the quantity and lengths demanded and L_k is the stock length.

Efficiency function. The efficiency of a cutting pattern is quantified in terms of the absolute amounts of scrap that it generates (r_j), which is obtained from

$$r_j = L_k - \sum_{h=1}^{N} l_{hj}, \tag{12}$$

where L_k is the length of the stock beam used by cutting pattern *j* and l_{hj} is the length vector of the beams obtained from cutting pattern *j*.

Pattern generation. The procedure used here avoids exceeding the levels of demands and is similar to that described by Annand et al. (1999) for two dimensional patterns generation. The following steps are considered to generate a cutting pattern:

1. Set a random integer value *N* between *Nmin* and *Nmax*
2. Select *N* random values $rand_h$ between 0 y 1
3. For *h*=1 to *N* do
   3.1. Find the value of i_0 satisfying expression (13)
   3.2. Set the *h*-th element of the cutting pattern to beam i_0
4. Set, if necessary, the last elements of the cutting pattern to zero until expression (14) is satisfied

$$\sum_{i<i_0} d_i < rand_h L_k \leq \sum_{i \leq i_0} d_i \tag{13}$$

$$\sum_{h=1}^{N} l_{hj} \leq L_k \tag{14}$$

Cloning procedure. At each generation solutions are ranked by efficiency and a fixed quantity (*Elite*) of the highest efficiency values is taken directly into the next generation. This process known as cloning or elitist reproduction guarantees that once a good solution is found it is not discarded until a better solution replaces it.

Recombination procedure. First, a set of individuals of the previous generation, whose size is given by the recombination rate *Trec,* is randomly selected to be the progenitors of the new individuals. The *Bernoulli crossover* procedure has been adopted for recombination with the crossover threshold fixed between 0 and 1. The crossover combination is carried out by choosing $rand_1$, ,…, $rand_N$ random numbers. If $rand_k$ is above the crossover threshold, k genes of the parents are exchanged, if it is below the threshold they are not. In this manner, two new solutions are obtained from two progenitors, the more efficient of which is placed into the next generation.

Mutation procedure. The mutation procedure ensures a larger coverage of the state space by introducing new genetic information. Thus, a portion of new cutting patterns, whose size is given by the mutation rate *Tmut*, are introduced in each generation using the procedure described above,.

The generalization of the algorithm for all stock lengths is simple; cutting patterns are generated for each length by running as many independent GAs as different lengths are available in stock; then, all them are stored in a matrix with a number of rows equal to the number of stock lengths.

3.3 Phase 2: Genetic algorithm to solve the MSSCSP

A solution to the problem consists of a sequence of patterns that meets the demand of beams without exceeding the available stock and that minimizes an objective function that weights things like: generated debris, inventory, space occupied by the work in process, etc. As before, the GA will have the following input parameters: Population size (*Pop*); maximum number of iterations (*iter*); size of cloning proportion (*Elite*); and recombination and mutation rates (T_{rec} and T_{mut}).

Genetic encoding. A solution to the CSP is encoded as a chromosome with G genes, each one of which refers to one cutting pattern. The number of genes G of each individual will vary randomly between two integer values given by:

$$G_{\max} = \left\lceil \frac{\sum_{i=1}^{m} d_i}{\min\left(\sum_{i=1}^{m} a_{ij}\right)} \right\rceil,\ G_{\min} = \left\lceil \frac{\sum_{i=1}^{m} d_i}{\max\left(\sum_{i=1}^{m} a_{ij}\right)} \right\rceil, \tag{15}$$

where d_i is the quantity demanded for beam i and $\min\left(\sum_{i=1}^{m} a_{ij}\right)$ and $\max\left(\sum_{i=1}^{m} a_{ij}\right)$ are the maximum and minimum number of demanded beams cut by pattern j, respectively.

Efficiency function. Three terms are considered: cost associated to the cutting patterns, ending inventory generated by the solution, and unmet demands. Thus, the function to be minimized is given by

$$w_{cost}\sum_{j}(X_j c_j) + w_{inv}\sum_{i}\left(I_i^{1+}\right) + w_{unmet}\sum_{i}\left(I_i^{1-}\right)^2. \tag{16}$$

where $\{w_{cost}, w_{inv}, w_{unmet}\}$ are the weights for each term; X represents a solution, and the element X_j indicates how many cutting patterns of type j are used by the solution; c_j represents the cost of using cutting pattern j that according to (6), depends on the scrap generated and its reusability and on the type of stock length used by each cutting patterns; and I_i^{1+} and I_i^{1-} represent the ending inventories and the unmet demands for demanded beam i, respectively, which are obtained from

$$I_i^{1+} = \max\left(I_i^0 + \sum_j (X_j a_{ij}) - d_i, 0\right),\ I_i^{1-} = \max\left(d_i - I_i^0 - \sum_j (X_j a_{ij}), 0\right), \tag{17}$$

where I_i^0 represents initial inventory for beam i (if any).

Generation of the initial population. The following steps are applied:

1. The number of genes G of each individual is randomly computed as an integer value between G_{min} and G_{max}.
2. The value of each gene of each individual is randomly assigned from the available cutting patterns (if one stock length runs out, the cutting patterns associated with it become unavailable).
3. Set to zero remaining genes until G_{max}.

Recombination procedure. The selection of parents is made randomly. For each gene, one of the cutting patterns is selected randomly from the parents. Afterwards, null genes are sent to the end of the chromosome. Finally, it is checked if the generated solution is feasible according to the availability of stock lengths. Figure 4 shows an example.

Cloning and mutation procedures. Both operations are performed like those in the GA of Phase 1 (Subsection 3.2).

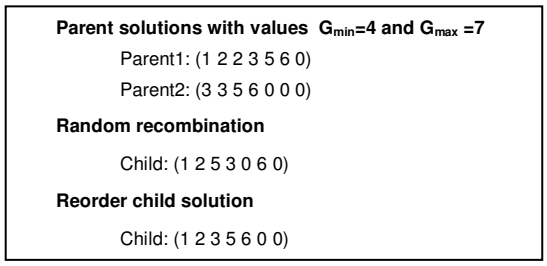

Fig. 4 Example of the recombination procedure for the GA in Phase 2

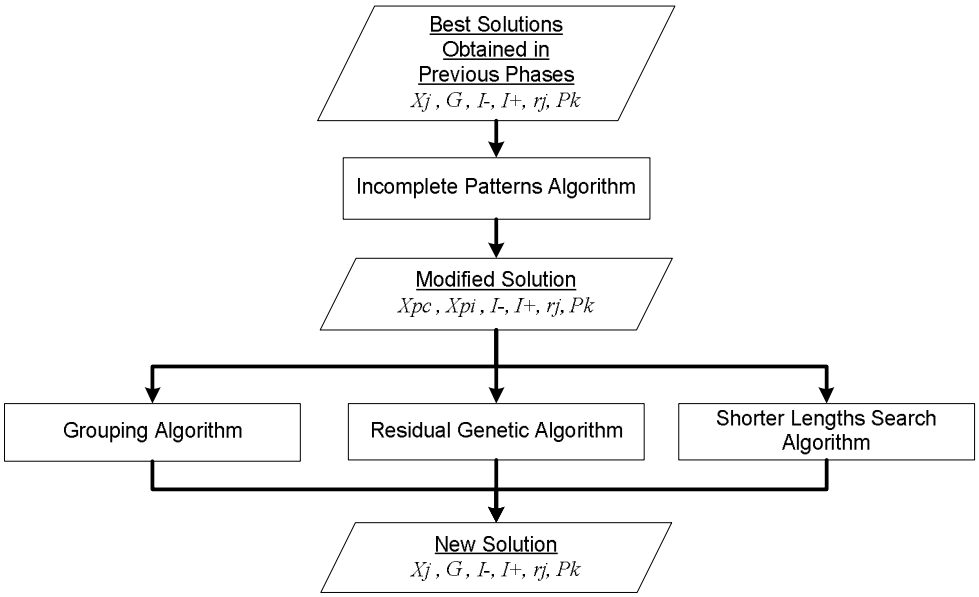

Fig. 5. Overview of the refinement algorithms in Phase 3

1- Initialization

1.1 Select one solution to the MSSCSP $X=(X_1,\ldots,X_N)$

1.2 Calculate the initial inventories for each demanded beam i (I_i^{0+})

2- while $(I_i^+ \neq 0)$

2.1 Calculate the contribution (C_{ij}) *of each pattern j to inventory of beam i* (I_i^+)

if $I_i^+ > X_{ij}$ *then* $C_{ij}=X_{ij}$

if $I_i^+ \leq X_{ij}$ *then* $C_{ij}=I_i^+$

2.2 Select pattern $J / \sum_i C_{iJ} = max (\sum_i C_{ij})$, $j=1,\ldots N$ *as an "Incomplete Pattern"*

If more than one value is possible for J, choose the cutting pattern with the smallest r_J

2.3 Recalculate Ii^+

Fig. 6. Pseudo code for incomplete patterns algorithm

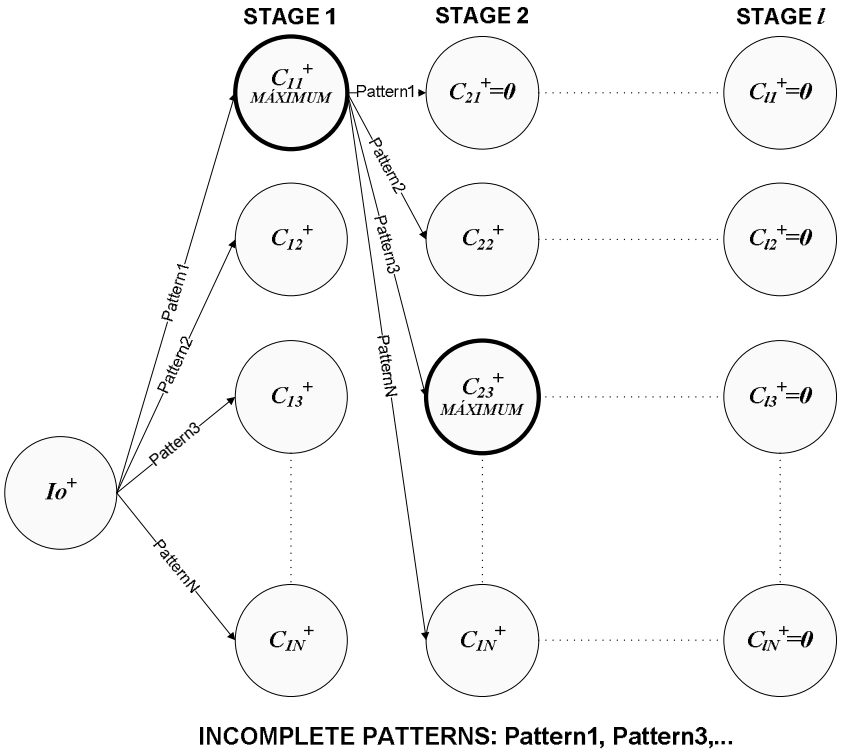

Fig. 7 Graphical representation of the search for incomplete patterns

3.4 Phase 3: Refinement algorithms

Perhaps, the main limitation of an approach to solve the MSSCSP based solely on the use of GAs lies in the non-complete satisfaction of all demands, as well as the generation of excess inventories. Although the efficiency function used by the GA in the previous section takes into account such considerations, see expression (16), it appears that under certain circumstances (a high number of beams, low demands, etc.) the results get higher levels of overproduction and even partially unmet demands, which violates of the original constraints. To solve this situation, a sequence of *novel* algorithms is proposed here to improve the overproduction and to eliminate the unmet demands. The method, which is based on clustering patterns and search for shorter lengths solutions, consists of four algorithms, see Fig. 5.

Incomplete Patterns Algorithm (A0). This novel algorithm searches among all the patterns in a solution j those which contribute most to beam overproduction. These patterns, referred to as *incomplete patterns*, are removed from the solution. The pseudo code for this algorithm is shown in Fig. 6, whereas Fig. 7 represents the way in which incomplete patterns are obtained, where C_{lj}^{+} represents the contribution to the ending inventories of pattern j during stage l and I_0^{+} is the initial inventory given by the solution.

1- *Unify all beams of the residual problem in a unique cutting pattern*

2- *Seek, among all the available stock lengths, the one that fits all beams with minimum waste*

for k=1 to K, if $L_k \geq \sum_{ij} C_{ij}$ *then select* $k' / L_{k'} = min(L_k)$

3- *Update stock levels* $P_{k'}=P_{k'}^{0}-1$;

4- *Calculate the total scrap of the new solution*

Fig. 8. Pseudo code for the Grouping Algorithm

1- *Define the instance parameters obtained from the data of the residual problem*

$(l_1,...,l_m,d_1,...,d_m,L_1,\ldots,L_p,P_1,\ldots,P_p)$

2- *Apply the pattern generation GA to obtain a new array of cutting patterns* m_{it}, w_{kt}.

3- *Apply the GA to solve the residual CSP*

4- *Select the best solution (minimum waste)*

5- *Calculate the total scrap of the new solution*

Fig. 9. Pseudo code for the Residual Genetic Algorithm

1- *Initialization*

1.1 *Calculate the total length* L_m *of each Incomplete Pattern (m),* $L_m=\sum_i l_{im}$, *where* l_{im} *are the lengths of all beams i cut with m*

1.2 *Obtain the scrap* (r_{mk}) *generated if the cutting pattern m uses the stock length* L_k $r_{mk} = L_k - L_m$

2- *while* $I^{-} \neq 0$

2.1 *Obtain all C combinations to group the Q beams of* I^{-} *(groups from 1 to Q elements)*

2.2 *Calculate for each combination the sum of all the demanded beams that it contains, that is* $L_{c=}=\sum_i l_{ic}$

2.3 *Calculate* $T_{mkc}= r_{mk} - L_c$

2.4 *if* $T_{mkc} < 0$ *then* $T_{mkc} = 10000$

2.5 *Select the smaller* T_{mkc} *(if tie, take any of them)*

2.6 *Assign the beams of combination c to Incomplete Pattern m*

2.7 *Update* $I^{-} = I_0^{-} - l_{ic}$

3- *Assign to each of the new "Incomplete Patterns" the smaller stock length which gives smaller scrap.*

4- *Calculate the total scrap of the new solution*

Fig. 10. Pseudo code for Shorter Lengths Search Algorithm

Both unmet demands and the demanded beams belonging to incomplete patterns constitute the so called *residual problem*, which is subsequently solved independently

by the three novel algorithms described below. All three are based on different strategies to cover as many possibilities as possible. Each one solves independently the *residual problem* and among the solutions provided by all three, the one with smaller scrap is chosen.

Grouping Algorithm (A1). Unmet demands and all the incomplete patterns are simply grouped in a unique cutting pattern. Figure 8 provides the pseudo code for this algorithm.

Residual Genetic Algorithm (A2). Phase 1 and Phase 2 are applied to the residual problem. Since demands and stocks are different from the original problem, new cutting patterns are generated. The pseudo code for this algorithm is shown in Fig. 9.

Shorter Lengths Search Algorithm (A3). This algorithm maintains the incomplete patterns and seeks for each one of them an available stock length that fits in it with minimum scrap. The application of the algorithm is simple for solutions that meet all demands ($I^{-}=0$); if not ($I^{-}\neq 0$), it has to be determined which one of the incomplete patterns will contain the unmet demands I^{-}. For this purpose, the algorithm raises all possible combinations of grouping $I^{-}$and selects the one that gives less waste. Figure 10 provides the pseudo code for this algorithm.

4 Experimental study

4.1 Introduction

In this section, several issues concerning the proposed approach (instance generation, parameter characterization, performance of each phase, computational costs, etc.) are experimentally analyzed. The proposed algorithms have been implemented in commercial software MATLAB® release R2007b. The set of input parameters for the GAs in Phase 1 and Phase 2 are selected according to those used by Wagner (1999) and Annand et al. (1999):

- $\{Pop=50, Iter=100, Elite=10, T_{rec}=0.8, T_{mut}=0.2\}$ for the GA in Phase 1
- $\{Pop=100, Iter=500, Elite=10\}$ for the GA in Phase 2

In addition, the values of the recombination/mutation rate and the weights in the efficiency function for the GA in Phase 2 are optimized in Subsection 4.4. For comparison purposes in the experimental study, parameters δ_j and γ_j in expression (6) are set equal to 1 and, therefore, coefficients c_j and r_j coincide. Moreover, when the number of possible patterns is less than 250, Phase 1 is replaced by complete enumeration and the best 12 patterns are selected. Furthermore, refinement algorithms in Phase 3 are applied to the best solutions obtained in Phase 2. In particular the first 20 solutions are considered for A1 and A3 algorithms while only the first 10 solutions for A2 due to computational reasons, see Subsection 4.7

4.2 Instances generation

For testing the proposed algorithms, it is required to implement a *problem generator*. For example, Gau and Wäsher (1995) develop a problem generator for the 1dimensional SSSCSP which only considers one type of stock length. For this work each *instance* has the following parameters: m, types of different demanded lengths; $\{v_1, v_2\}$, upper and lower limits for demanded lengths; d, average demand for each demanded length; p, types of different stock lengths; e, average quantities in stock; $\{s_1, s_2\}$, upper and lower limits for stock lengths. Thus, a test problem is defined as a 2m+2p-dimensional vector generated randomly: $(l_1,...,l_m,d_1,...,d_m,L_1,\ldots,L_p,e_1,\ldots,e_p)$, where lengths and quantities are specified for the demanded and stock beams.

In order to evaluate the performance of the approach, instances obtained from the ten classes of Table 2, which fit the data handled by the company described in Section 2, are considered. These instance classes have taken from literature and they have recently been used by Poldi and Arenales (2009) to evaluate their residual heuristics when solving the 1dimensional MSSCSP for low demands. The following expressions, which are taken from Poldi and Arenales (2009), are used to generate the test problem vector:

- for $k=1$ *to* p, L_k is randomly generated between 100 and 1000 length units,
  e_k is randomly generated between 1 to $(100 \cdot m/2)$ units;
- for $i=1$ *to* m, l_i is randomly generated between $(v_1 \cdot L)$ and $(v_2 \cdot L)$ length units,
  d_i is randomly generated between 1 and 10 units;

where L is the average value of $\{L_1,...,L_p\}$.

Table 1 Instance classes used in the experimentation

	Parameters			
Instance class	*p*	*m*	v_1	v_2
1	3	5	0,01	0,2
2	5	10	0,01	0,2
3	7	10	0,01	0,2
4	3	5	0,1	0,8
5	5	10	0,1	0,8
6	7	10	0,1	0,8
7	3	5	0,01	0,8
8	5	10	0,01	0,8
9	7	10	0,01	0,8
10	7	20	0,01	0,8

4.3 Performance of the algorithm to generate cutting patterns

According to Haesler and Sweeny (1991), one key ingredient for the success of a heuristic procedure applied to the CSP is the efficiency of its cutting patterns. Thus,

the performance of our pattern generation algorithm is evaluated in this subsection. In this sense, Fig. 11 shows the average results obtained for 150 test problems from the instance classes of Table 1. Each bin of the histogram groups the cutting patterns depending on their waste: 0%, 0.1%–2%, 2.1%–6%... 25.1%–31%, 31.1%–100%. Note that, 17% of the cutting patterns have no waste and 31% of them have a maximum waste of 2%. Therefore, it is conclude that the GA in Phase 1 is able to provide many good cutting patterns, which will represent a good base to solve the SCP in subsequent phases.

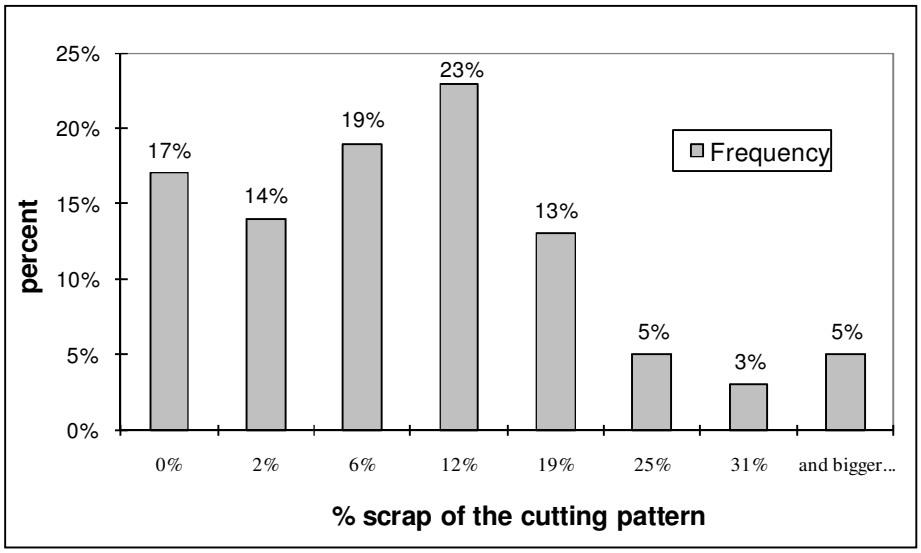

Fig. 11 Efficiency histogram of the cutting patterns generated by the GA in Phase 1

4.4 Sensitivity analysis to different algorithmic configurations in Phase 2

In order to get a good parameter setting, we analyze here the effects on the quality of the solutions obtained by the GA in Phase 2 when the recombination/mutation rate and the efficiency function weights are modified. Table 2 displays the average results obtained after applying Phase 1 and Phase 2 to 20 different instances from classes 1 and 2 for different values of $\{T_{rec}, w_{umet}, w_{inv}\}$ and setting $\{w_{cost}=1\}$ without loss of generality. In light of these results, it seems reasonable to choose $\{T_{rec}=0.8, w_{inv}=3, w_{unmet}=10\}$ as the values for GA in Phase 2. Note that recombination and mutation rates are related by equation (10) and the value for *Elite* is given in Subsection 4.1, i.e. $T_{rec}=1-T_{mut}$ and *Elite*=10. Furthermore, Table 3 shows the Pearson's correlation coefficients between the algorithm parameters $\{T_{rec}, w_{inv}\}$ and the output indexes in Table 2. These coefficients assess from -1 to 1 how sensitive the output indexes are to the algorithm parameters. In this case, the correlations are medium and low.

Table 2 Sensistivity analisys for the GA in Phase 1

Instance	*Trec*	*Winv*	*Wunmet*	*Solutions all demands met*	*Minimum overproduced beams*	*Scrap in the best solution*
1	0.8	1	10	25,9	11,2	5,5
1	0.4	1	10	27,2	17,8	26,4
1	0.8	3	10	23,6	10,9	6,6
1	0.4	3	10	28,4	14,6	58,3
1	0.8	10	10	15,9	12,8	37,2
1	0.4	10	10	15,1	18,5	91,7
2	0.8	1	10	46,1	4,4	9,2
2	0.4	1	10	54,1	6,1	23,3
2	0.8	3	10	38,6	3,4	14,9
2	0.4	3	10	37,6	3	9,5
2	0.8	10	10	27,1	2,1	6,7
2	0.4	10	10	36,3	2,9	8,9
1	0.8	1	1	8,3	5,0	2,3
1	0.4	1	1	10,8	5,4	5,4
1	0.8	3	1	6,8	4,7	1,3
1	0.4	3	1	9,7	5,9	13,9
1	0.8	10	1	10,0	3,9	6,9
1	0.4	10	1	14,0	3,6	20,6
2	0.8	1	1	2,9	8,5	2,3
2	0.4	1	1	4,2	11,8	6,1
2	0.8	3	1	2,2	7,4	3,3
2	0.4	3	1	1,8	8,0	1,9
2	0.8	10	1	1,4	6,6	1,8
2	0.4	10	1	1,8	8,6	2,2

4.5 Sensitivity of the approach to the quality of cutting patterns

Here we focus on analyzing how each phase of the proposed approach, contributes to the achievement of its performance. To this end, Phase 2 and Phase 3 are applied to 40 test problems from 4 instance classes (to cover a wide range of situations) using three different sets of cutting patterns: one set of good cutting patterns obtained from Phase 1; and two more sets with less efficient patterns. Therefore we consider three different qualitative situations, see Figure 12 where the efficiency histograms for each set are shown. Thus, evaluating the improvement obtained applying Phase 2 and 3 for

these three cases, we can determine the necessity of Phase 1 in the proposed approach. Table 4 compares the average scrap results for each instance class when applying both Phase 2 and Phase 3 to the above mentioned sets of cutting patterns. First observe that, the better the quality of the input patterns, the better the average scrap result. Table 4 also indicates the average number of cutting patterns in the solution and the percentage of those generated in Phase 3 (new patterns). Comparing these values with the average scrap values we conclude that as the number of patterns in the solution increases, the contribution of Phase 3 decreases and the quality of the input patterns is less relevant.

In short: an efficient basis of patterns becomes more necessary as the number of patterns of the solution increases; and the contribution of refinement algorithms is greater for sets of patterns with low and medium efficiency, but this is not enough to neutralize effects of using those basis of patterns.

Table 3 Pearson's correlation coefficients for the variables in Table 2

Variables	Pearson's Coefficient
T_{rec} / Minimum overproduced beams	-0,26685041
T_{rec} / Solutions all demands met	-0,1944211
T_{rec} / Scrap in the best solution	-0,6489404
W_{unmet} / Minimum overproduced beams	0,3031187
W_{unmet} / Solutions all demands met	0,88586149
W_{unmet} / Scrap in the best solution	0,74583432
W_{inv} / Minimum overproduced beams	-0,01710628
W_{inv} / Solutions all demands met	-0,54228093
W_{inv} / Scrap in the best solution	0,45635913

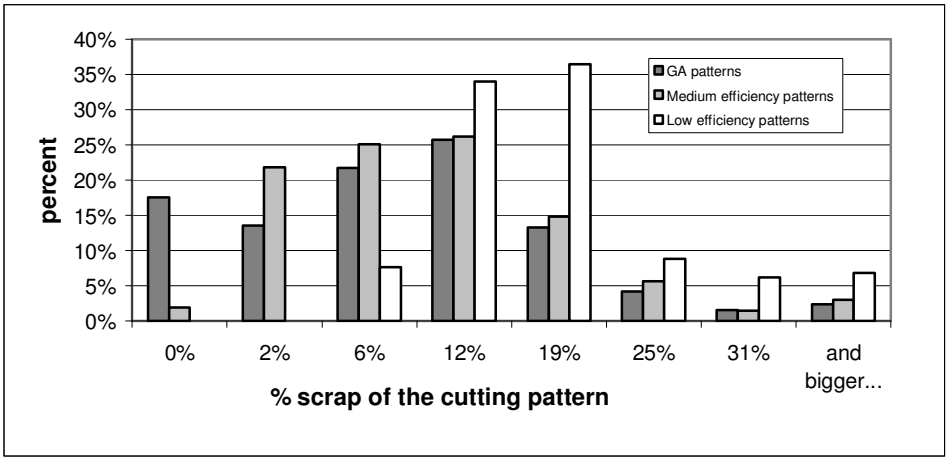

Fig. 12 Efficiency histogram of three different sets of cutting patterns

Table 4 Impact of the cutting patterns basis on solutions

	Genetic algorithm patterns			Medium efficiency patterns			Low efficiency patterns		
Instance class	Avg. scrap	Patterns from Phase 3 (%)	Avg. number of patterns in solution	Avg. scrap	Patterns from Phase 3 (%)	Avg. number of patterns in solution	Avg. scrap	Patterns from Phase 3 (%)	Avg. number of patterns in solution
1	85	50	2	85	50	2	116	50	2
2	27	27	4,75	50	27	4,75	104	39	4,75
3	12	18	6,4	34	27	5,6	43	34	5
4	257	27	6,5	463	34	6,5	1378	41	6

4.6 Contribution of each refinement algorithm to the performance of the proposed approach

In Section 3, three algorithms {A1, A2, A3} were proposed to be run in parallel in Phase 3 to improve the solutions obtained by the GA of Phase 2. Therefore, it is necessary to check how far all they contribute to get better solutions in Phase 3 or if, on the contrary, one stands out from the other. In this sense, Table 5 shows the contribution of the three refinement algorithms {A1, A2, A3} to the new solutions obtained in Phase 3 as mentioned in Subsection 4.1, for 400 test problems. It can be conclude that there is no clear superiority of one algorithm over the others and, therefore, the use of all three is appropriate.

Table 5 Contribution of each refinement algorithm in Phase 3

	Percentage of new solutions
Grouping algorithm (A1)	29%
Residual genetic algorithm (A2)	28%
Shorter lengths search algorithm (A3)	43%

4.7 Computational costs

In principle, the 1dimensional MSSCSP here studied can be solved *off-line* and, therefore, the computational cost of the method used to solve it would not be critical. However, Table 6 shows average computing times (over all generated test problems) of each phase of the proposed approach. Algorithms were implemented using commercial software *MATLAB*® (release 2007b) in a PC with *Intel® Core™ 2 Duo @2.00GHz* processor. The execution of all the algorithms is quite fast, around 77 seconds. Similarly, Poldi and Arenales (2009) obtained a computation average time of 37seconds for their residual heuristics algorithms (RGH1, RGH2, RGH3). Note that, the residual GA (A2) has a longer running time because it applies both GAs of Phases 1 and Phase 2 over 10 solutions. Since algorithms A1, A2 and A3 are independent they could be implemented in parallel to reduce the computation time. However, it is considered that the improvement in the computation time obtained, i.e.

100*(0,2859+0,0037)/77,5899=0.3732% (see Table 6), does not worth such parallel implementation.

Table 6 Computational costs of each phase of the proposed approach

	time (s)	time per iteration (ms)
Previous phase- Problem generator	**0,1218**	
Phase 1- GA to generate cutting patter	**3,356**	33,56
Phase 2- GA to solve MSSCSP	**5,4325**	10,865
Phase 3.1- Incomplete patterns algorithm	0,6384	
Phase 3.2- Grouping algorithm (A1)	0,0037	
Phase 3.2- Residual GA (A2)	67,749*	
Phase 3.2- Shorter lengths search algorithm (A3)	0,2859	
Phase 3.3- Selection of the best refinement algorithm	0,0026	
Phase 3- Refinement algorithms	**68,6796**	
Total	**77,5899**	

*Note: The computation time of A2 algorithm is not ten times that of Phase1 plus Phase2 because usually the number patterns for residual problems are less than 250 and, therefore, in Phase1 complete enumeration is made.

5 Results and comparison with other approaches

In this section, the parameter values for the proposed algorithms are the same as those used in previous section. Table 7 shows the results obtained with the proposed approach for 400 instances obtained from the ten classes of Table 2. These results are compared to those obtained with residual and constructive heuristics of type First Fit Decreasing (FFD) and Greedy and with those obtained by Poldi and Arenales (2009) using their residual heuristics (RGH1, RGH2, RGH3). Best results appear in boldface numbers in Table 7. In particular, the hybrid approach obtains the best result in 70% of instances. The remaining 30% of instances are those in which p=7 and m={10,20}, but according to data in Subsection 2.1 this is not frequent in the local metalwork company. Therefore, it is conclude that the proposed approach is appropriate at least for instances similar to real data found in the metalwork company.

6 Conclusions

In this paper, a hybrid solution based on genetic algorithms is developed to solve efficiently the problem of cutting structural beams arising in a local metalwork company. In contrast to previous works, the proposed approach handles over and underproduction of beams and reusability of remnants in the optimization process. For this purpose, several novel refinement algorithms based on different search and clustering strategies have been introduced and a new encoding with a variable number

of genes has been developed for cutting patterns. An analysis of the performance of different algorithmic configurations allowed us to properly adjust several input parameters of the GAs. In addition, statistical data evidenced the adequacy and convenience of the proposed algorithms, since all them contribute to the good performance of the hybrid solution. Furthermore, it has been shown that the execution of all the proposed algorithms is quite fast. Finally, comparative results have shown that the proposed approach competes favourably with other known heuristics.

We consider that the proposed high-level relay hybrid approach could be useful for other multiple sizes CSP with higher dimension. For this purpose, the pattern generation phase should obviously be redefined to the particular conditions of the problem but the rest of the methodology would remain basically the same. This issue is an interesting field for future work.

Table 7 The average waste result comparison with other approaches

	Total scrap							
	Constructive		Residual		Residual Heuristics			Hybrid
Instance	FFD	Greedy	FFD	Greedy	RGH1	RGH2	RGH3	Approach
1	118,75	113,35	123,25	116,25	113,05	114,40	108,7	**85,40**
2	161,05	152,30	146,80	165,20	137,85	129,20	138,60	**27,00**
3	179,45	146,05	147,95	142,75	117,65	116,05	115,90	**11,60**
4	71319,35	31493,55	36648,90	46662,30	625,55	646,85	617,20	**257,00**
5	96475,80	101711,50	57310,90	60145,25	785,10	808,40	812,75	**339,60**
6	118647,90	134773,10	52624,25	51184,85	247,75	255,15	**247,65**	476,40
7	42429,35	52808,85	7551,20	8762,75	709,05	683,05	709,30	**594,68**
8	74434,35	73920,20	45636,20	47154,75	545,00	552,85	576,30	**540,00**
9	87379,30	107851,45	47095,25	35678,10	**241,35**	258,50	247,90	303,20
10	92130,75	91638,20	42535,50	39284,45	135,95	140,85	**132,60**	242,00
Being best	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**10%**	**0%**	**20%**	**70%**

References

Aktin, T., Özdemir, R. G.: An integrated approach to the one dimensional cutting stock problem in coronary stent manufacturing. Eur. J. Oper. Res. 196, 737–743 (2009)

Alves, C., Valério de Carvalho, J. M.: A Stabilized Branch-and-Price-and-Cut Algorithm for the Multiple Length Cutting Stock Problem. Comput. Oper. Res. 35, 1315–1328 (2008)

Anand, S., McCord, C., Sharma, R. et al.: An Integrated Machine Vision Based System for Solving the Nonconvex Cutting Stock Problem using Genetic Algorithms. J. Manuf. Syst. 18, 396–415 (1999)

Belov, G., Scheithauer, G.: A Cutting Plane Algorithm for the One-Dimensional Cutting Stock Problem with Multiple Stock Lengths. Eur. J. Oper. Res. 141, 274–294 (2002)

Christofides, N., Hadjiconstantinou, E.: An Exact Algorithm for Orthogonal 2-D Cutting Problems using Guillotine Cuts. Eur. J. Oper. Res. 83, 21–38 (1995)

Elizondo, R., Parada, V., Pradenas, L., Artigues, C.: An evolutionary and constructive approach to a crew scheduling problem in underground passenger transport. J. Heur. 16, 575–591 (2010)
Fan, L., Mumford, C. L: A metaheuristic approach to the urban transit routing problem. J. Heur. 16, 353–372 (2010)
Gau, T., Wäscher, G.: CUTGEN1: A Problem Generator for the Standard One-Dimensional Cutting Stock Problem. Eur. J. Oper. Res. 84, 572–579 (1995)
Gilmore, P. C., Gomory, R. E.: A linear programming approach to the cutting stock problem. Oper. Res. 9, 849–859 (1961)
Gilmore, P. C., Gomory, R. E.: A linear programming approach to the cutting stock problem, PartII. Oper. Res. 11, 863–888 (1963)
Ghiani, G., Laganà, G., Laporte, G., Mari, F.: Ant colony optimization for the arc routing problem with intermediate facilities under capacity and length restrictions. J. Heur. 16, 211–233 (2010)
Gonçalves, J. F., Resende, G. C.: Biased random-key genetic algorithms for combinatorial optimization. J. Heur. In Press (DOI: 10.1007/s10732-010-9143-1) (2011)
Gradisar, M., Kljajic, M., Resinovic, G. et al.: A Sequential Heuristic Procedure for One-Dimensional Cutting. European Eur. J. Oper. Res. 114, 557–568 (1999)
Haessler, R. W.: One-Dimensional Cutting Stock Problems and Solution Procedures. Math. Comput. Model. 16, 1–8 (1992)
Haessler, R. W.: Solving the Two-Stage Cutting Stock Problem. Omega 7, 145–151 (1979)
Hinterding, R., Khan, L.: Genetic algorithms for cutting stock problems: with and without contiguity. In: Yao, X., (ed.) Progress in evolutionary computation. LNAI, vol. 956, pp. 166-186. Berlin: Springer, Berlin (1995)
Holthaus, O.: Decomposition Approaches for Solving the Integer One-Dimensional Cutting Stock Problem with Different Types of Standard Lengths. Eur. J. Oper. Res. 141, 295–312 (2002)
Kantorovich, L.V.: Mathematical methods of organizing and planning production (1939) (Translation to English in Manage. Sci. 6, 366–422 (1960))
Liang, K., Yao, X., Newton, C. et al.: A New Evolutionary Approach to Cutting Stock Problems with and without Contiguity. Comput. Oper. Res. 29, 1641–1659 (2002)
Poldi, K., Arenales, M.: Heuristics for the One-Dimensional Cutting Stock Problem with Limited Multiple Stock Lengths. Comput. Oper. Res. 36, 2074–2081 (2009)
Suliman, S. M. A.: Pattern Generating Procedure for the Cutting Stock Problem. Int. J. Prod. Econ. 74, 293–301 (2001)
Talbi, E.-G.: A Taxonomy of Hybrid Metaheuristics. J. Heur. 8, 541–564 (2002)
Vahrenkamp, R.: Random Search in the One-Dimensional Cutting Stock Problem. Eur. J. Oper. Res. 95, 191–200 (1996)
Vanderbeck, F.: Exact Algorithm for Minimizing the Number of Set Ups in the One Dimensional Cutting Stock Problems. Oper. Res. 48, 915–926 (2000)
Wagner, B. J.: A Genetic Algorithm Solution for One-Dimensional Bundled Stock Cutting. Eur. J. Oper. Res. 117, 368–381 (1999)
Wäscher, G., Haußner, H., Schumann, H.: An Improved Typology of Cutting and Packing Problems. Eur. J. Oper. Res. 183, 1109–1130 (2007)