Available online at www.sciencedirect.com

ScienceDirect

European Journal of Operational Research 183 (2007) 1212–1229

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/ejor

A hybrid genetic algorithm-heuristic for a two-dimensional orthogonal packing problem

José Fernando Gonçalves *

Faculdade de Economia do Porto, Rua Dr. Roberto frias, 4200-464 Porto, Portugal

Received 30 September 2004; accepted 22 November 2005
Available online 13 June 2006

Abstract

In this paper we address a two-dimensional (2D) orthogonal packing problem, where a fixed set of small rectangles has to be placed on a larger stock rectangle in such a way that the amount of trim loss is minimized. The algorithm we propose hybridizes a placement procedure with a genetic algorithm based on random keys. The approach is tested on a set of instances taken from the literature and compared with other approaches. The computation results validate the quality of the solutions and the effectiveness of the proposed algorithm.

Keywords: Packing; Orthogonal packing; Cutting; Genetic algorithms

1. Introduction

The two-dimensional orthogonal packing problem addressed in this paper consists of packing rectangular pieces into a large rectangular sheet of material (stock rectangle) in order to maximize the material utilization, or equivalently, to minimize the unused area (trim loss). The problem is relevant both from a theoretical and a practical point of view because it arises in various production processes. Their applications go from the home-textile to the glass, the steel, the wood and the paper industries. These industries are mainly concerned with the cutting of rectangular figures from a large rectangular sheet of material.

The problem has been formulated as an integer program. For that approach and variations, see Beasley (1985), Tsai (1993) or Christofides (1995).

Many heuristic packing algorithms have been suggested in the literature. Surveys on solution methodologies for various types of the 2D rectangle packing problem can be found in Hinxman (1980), Sarin (1983), Hässler and Sweeney (1991) and Lodi et al. (2002).

In comparison to the great quantity of literature on heuristic algorithms to the packing problem, only a few researchers have experimented with meta-heuristic algorithms. Recently, different meta-heuristics have been applied to problems of this kind, for instance, see Leung et al. (2003), Parada et al. (1998), Lai and Chan (1997), Jakobs (1996), Glover et al. (1995), Dowsland (1993, 1996). Leung et al. (2000) carried out extensive comparisons of these

* Tel.: +351 937 061 379.
E-mail address: jfgoncal@fep.up.pt

0377-2217/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2005.11.062

heuristics based on the works of Jakobs (1996) and Lai and Chan (1997). Hopper and Turton (2000) present an empirical investigation of meta-heuristic and heuristic algorithms for the two-dimensional orthogonal packing problem.

In this paper we present a hybrid meta-heuristic algorithm for a two-dimensional orthogonal packing problem which combines a random keys based genetic algorithm with a novel fitness function and a new heuristic placement policy.

The remainder of the paper is organized as follows. Section 2 defines the problem formally. Section 3 describes the new approach: genetic algorithm; placement strategy; fitness function. Section 4 presents the experimental results. Section 5 summarizes the conclusions and proposes future work.

2. The problem

The two-dimensional orthogonal packing problem addressed in this paper is the problem of cutting from a single large planar stock rectangle (W,H), of width W and height H, smaller rectangles (W_i,H_i) each of width W_i and height H_i $(i=1,\ldots,n)$. Each rectangle i has a fixed orientation (i.e., cannot be rotated); must be cut (by infinitely thin cuts) with its edges parallel to the edges of the stock rectangle (i.e., orthogonal cuts). Each rectangle i has an associated value equal to its area and the objective is to maximize the total area of the rectangles cut, or equivalently minimize the unused area of the stock rectangle (minimize trim loss). Without significant loss of generality it is usual to assume that all dimensions (W_i,H_i) $(i=1,2,\ldots,n)$ are integers.

According to the newly developed typology by Wäscher et al. (this issue) the problem presented in this paper falls into the output maximization assignment kind and belongs to the problem type (two-dimensional, rectangular SKP).

3. New approach

The new approach combines a random key based genetic algorithm, a new placement strategy and a novel measure of solution quality, *Modified %Trim Loss*.

The genetic algorithm is responsible for evolving the chromosomes which represent the rectangle packing sequence. For each chromosome the following three phases are applied:

- *Decoding of the rectangle packing sequence*. This phase is responsible for transforming the chromosome supplied by the genetic algorithm into the sequence in which the rectangles are going to be packed in the stock rectangle.
- *Placement strategy*. This phase makes use of the rectangle packing sequence defined in the first phase and constructs a packing of the rectangles.
- *Fitness evaluation*. This phase makes use of a novel procedure which computes a Modified %Trim Loss value which is used as fitness measure (quality measure) to feedback to the genetic algorithm.

Fig. 1 illustrates the sequence of steps applied to each chromosome generated by the genetic algorithm.

The remainder of this section describes in detail the genetic algorithm, the placement strategy and the fitness function – Modified %Trim Loss.

3.1. Genetic algorithm

Genetic algorithms are adaptive methods, which may be used to solve search and optimization problems (Beasley et al., 1993). They are based on the genetic process of biological organisms. Over many generations, natural populations evolve according to the principles of natural selection, i.e., *survival of the fittest*, first clearly stated by Charles Darwin in *The Origin of Species*. By mimicking this process, genetic algorithms are able to *evolve* solutions to real world problems, if they have been suitably encoded.

Before a genetic algorithm can be run, a suitable *encoding* (or *representation*) for the problem must be devised. A *fitness function* is also required, which assigns a figure of merit to each encoded solution. During the run, parents must be *selected* for reproduction, and *recombined* to generate offspring (see Fig. 2).

It is assumed that a potential solution to a problem may be represented as a set of parameters. These parameters (known as *genes*) are joined together to form a string of values (*chromosome*). In genetic terminology, the set of parameters represented by a particular chromosome is referred to as an *individual*. The fitness of an individual depends on its chromosome and is evaluated by the fitness function.

The individuals, during the reproductive phase, are selected from the population and *recombined*, producing offspring, which comprise the next generation. Parents are randomly selected from the population using a scheme, which favors fitter

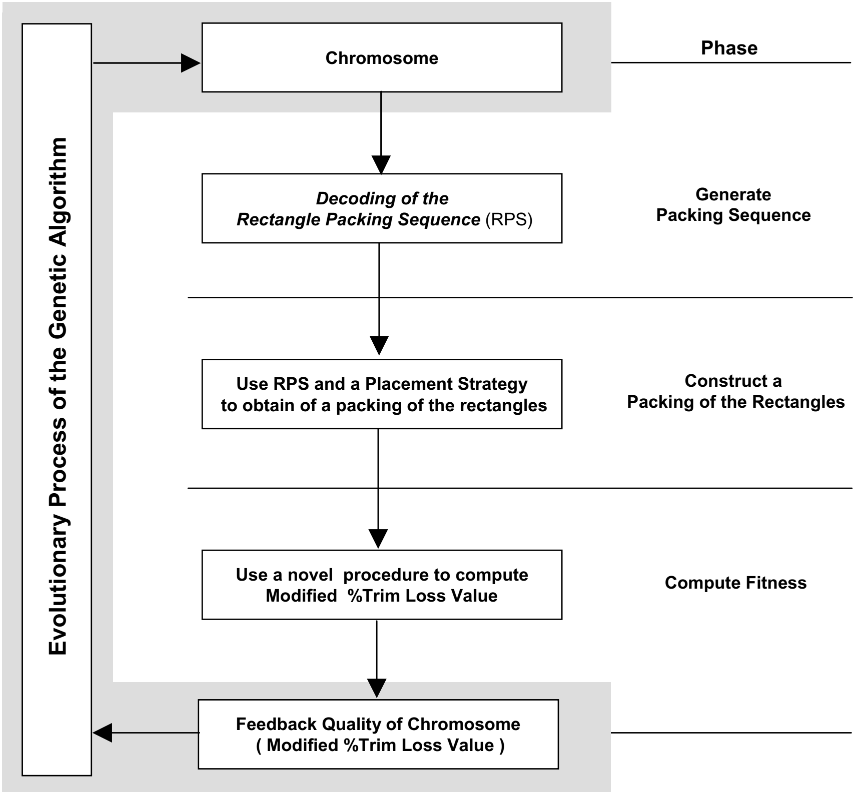

Fig. 1. Architecture of the new approach.

Genetic algorithm
{
    Generate initial population $\boldsymbol{P}_t$
    Evaluate population $\boldsymbol{P}_t$
    WHILE stopping criteria not satisfied ***REPEAT***
    {
        Select some elements from $\boldsymbol{P}_t$ to put into $\boldsymbol{P}_{t+1}$
        Crossover some elements of $\boldsymbol{P}_t$ and put them into $\boldsymbol{P}_{t+1}$
        Mutate some elements of $\boldsymbol{P}_t$ and put them into $\boldsymbol{P}_{t+1}$
        Evaluate new population $\boldsymbol{P}_{t+1}$
        $\boldsymbol{P}_t = \boldsymbol{P}_{t+1}$
    }
}

Fig. 2. Standard genetic algorithm.

individuals. Having selected two parents, their chromosomes are *recombined*, typically using mechanisms of *crossover*. Mutation is usually applied to some individuals, to guarantee population diversity.

3.1.1. Chromosome representation and decoding

The genetic algorithm described in this paper uses a random key alphabet $U(0,1)$. The evolutionary strategy used is similar to the one proposed by Bean (1994), the main difference occurring in the crossover operator. The important feature of random keys is that all offspring formed by crossover are feasible solutions. This is accomplished by moving much of the feasibility issue into the objective function evaluation. If any random key vector can be interpreted as a feasible solution, then any crossover vector is also feasible. Through the dynamics of the genetic algorithm, the system learns the relationship between random key vectors and solutions with good objective function values.

A chromosome represents a solution to the problem and is encoded as a vector of random keys (random numbers). Each solution chromosome is made of n genes where n is the number of rectangles to be packed.

$$Chromosome = (gene_1, gene_2, \ldots, gene_{n-1}, gene_n)$$

The n genes are used to obtain the Rectangle Packing Sequence, *RPS*, which is going to be used by the placement strategy. The decoding (mapping) of each chromosome into a *RPS* is accomplished by sorting the genes and packing the rectangles in ascending order of the sort (see Fig. 3).

3.1.2. Evolutionary strategy

To breed good solutions, the random key vector population is operated upon by a genetic algorithm. There are many variations of genetic algorithms obtained by altering the reproduction, crossover, and mutation operators. The reproduction and crossover operators determine which parents will have offspring, and how genetic material is exchanged between the parents to create those offspring. Mutation allows for random alteration of genetic material. Reproduction and crossover operators tend to increase the quality of the populations and force convergence. Mutation opposes convergence and replaces genetic material lost during reproduction and crossover.

Reproduction is accomplished by first copying some of the best individuals from one generation to the next, in what is called an elitist strategy (Goldberg, 1989). The advantage of an elitist strategy over traditional probabilistic reproduction is that the best solution is monotonically improving from one generation to the next. The potential downside is population convergence to a local minimum. This can, however, be overcome by high mutation rates as described below.

Parameterized uniform crossovers (Spears and Dejong, 1991) are employed in place of the traditional one-point or two-point crossover. After two parents are chosen, one chosen randomly from the best (*TOP* in Fig. 5) and the other is chosen randomly from the full old population (including chromosomes copied to the next generation in the elitist pass), at each gene we toss a biased coin to select which parent will contribute the allele. Fig. 4 presents an example of the crossover operator. It assumes that a coin toss of heads selects the gene from the first parent, a tails chooses the gene from the second parent, and that the probability of tossing a heads, crossover probability – *CProb*, is for example 0.7 (this value is determined empirically, see experimental results). Fig. 4 shows one potential crossover outcome.

Rather than using the traditional gene-by-gene mutation with very small probability at each generation, some new members of the population are randomly generated from the same distribution as the original population (see *BOT* in Fig. 5). The purpose of this process is to prevent premature convergence of the population, like in a mutation operator, and leads to a simple statement of convergence.

Chromosome 1	0.32	0.77	0.53	0.85
Chromosome 2	0.26	0.15	0.91	0.44
Rnd Number	0.58	0.89	0.68	0.25
Prob. Cross = 0.7	< 0.7	> 0.7	< 0.7	< 0.7
Offspring Chromosome	0.32	0.15	0.53	0.85

Fig. 4. Example of parameterized uniform crossover.

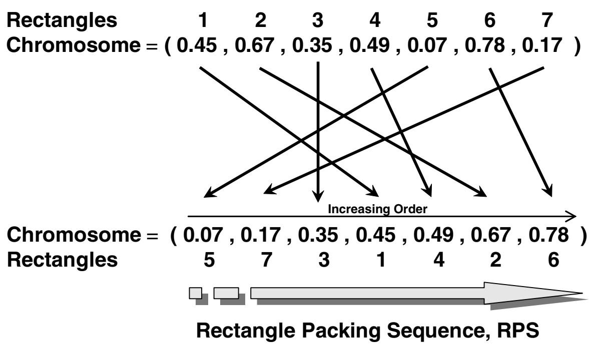

Fig. 3. Chromosome decoding procedure.

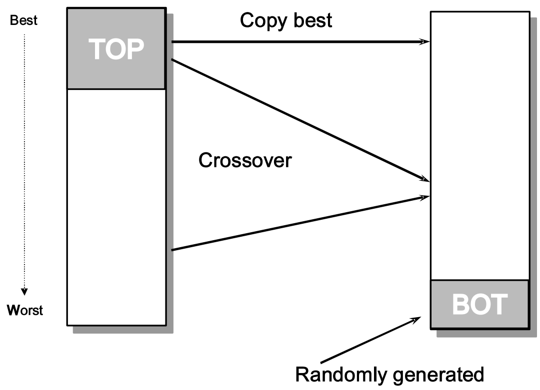

Fig. 5. Transitional process between consecutive generations.

Fig. 5 depicts the transitional process between two consecutive generations.

3.2. Placement strategy

Rectangles are placed in the stock rectangle one at a time in the order defined by the genetic algorithm (see Section 3.1.1). While trying to place a rectangle in the stock rectangle we consider only the empty rectangular spaces, with the largest possible size (ERS), in the stock rectangle. A rectangle is placed in the ERS where it fits and that has an area which is as close as possible to the area of the rectangle we are trying to place. If there are any ties we break the ties by choosing the ERS which is closest to bottom left corner of the stock rectangle. The pseudo-code for the placement strategy is presented in Fig. 6.

To generate and keep track of the empty rectangular spaces (ERS's) we make use of the Difference Process (DP) procedure developed by Lai and Chan (1997).

In short, the DP procedure can be described as follows. Suppose we have two small rectangles to be packed in a stock rectangle as depicted in Fig. 7a. Furthermore, assume that we will pack rectangle 1 first and then rectangle 2. At the very beginning, there is only one ERS available, ERS-1, where we can pack rectangle 1 and its bottom left corner is precisely at the origin of the stock rectangle, see Fig. 7b. After the first rectangle is packed, usually two new ERS are generated (unless the height or the width of the first rectangle is the same as that of the stock rectangle), see ERS-2 and ERS-3 in Fig. 7b. Every time a new rectangle is packed, it intersects some of the available ERS so we need to update the list of ERS available before we pack the next rectangle. From each existing ERS, which intersects with a rectangle non-trivially, it is easy to see that at most three new rectangular empty rectangular spaces are generated. From the newly generated ERS, we check if any one is entirely contained in another, and remove it in that case. This is the so-called elimination process in Lai and Chan (1997). In this manner, we update the list of ERS available whenever a rectangle is packed, see ERS-2.1, ERS-3.1 and ERS-3.2 in Fig. 7c.

3.3. Fitness function – Modified Trim Loss

The natural fitness function (measure of quality) for this type of problem is the value of the *%Trim Loss* given by

$$\%Trim\ Loss = \frac{\text{Unused area}}{\text{Stock rectangle area}} \times 100\%.$$

This measure, however, is not very good because it does not capture well the potential for improvement of a solution. Consider, for example, *Packing-1* and *Packing-2* depicted in Fig. 8. Both packings have a *%Trim Loss* of $5/16 \times 100\% = 31.25\%$. However, *Packing-2* has all the unused area concentrated in one contiguous space while *Packing-1* divides the unused area by three separated spaces. Any rectangle that can be packed in *Packing-1* can also be packed in *Packing-2*, however, the opposite is not true, i.e., we cannot pack a 3×1 or a 4×1 rectangle in *Packing-1* but we can do it in *Packing-2*. Therefore, it is easy to conclude that *Packing-2* has a

```
Let Ri the ith rectangle to be placed in the stock rectangle.

Let NR be the number of rectangles.

FOR i=1 TO NR REPEAT
{
        Let ERS*i be the ERS, in the list of available ERS, in which rectangle Ri fits and that
        has an area which is as close as possible to the area of the rectangle Ri. Break ties by
        choosing the ERS which is closer to the bottom left corner of the stock rectangle.

        IF ERS*i is not empty
        {
                Place Ri at bottom left corner of ERS*i

                Update list of available ERS's using the DP process of Lai and Chan (1997)
        }
}
```

Fig. 6. Pseudo-code for the placement strategy.

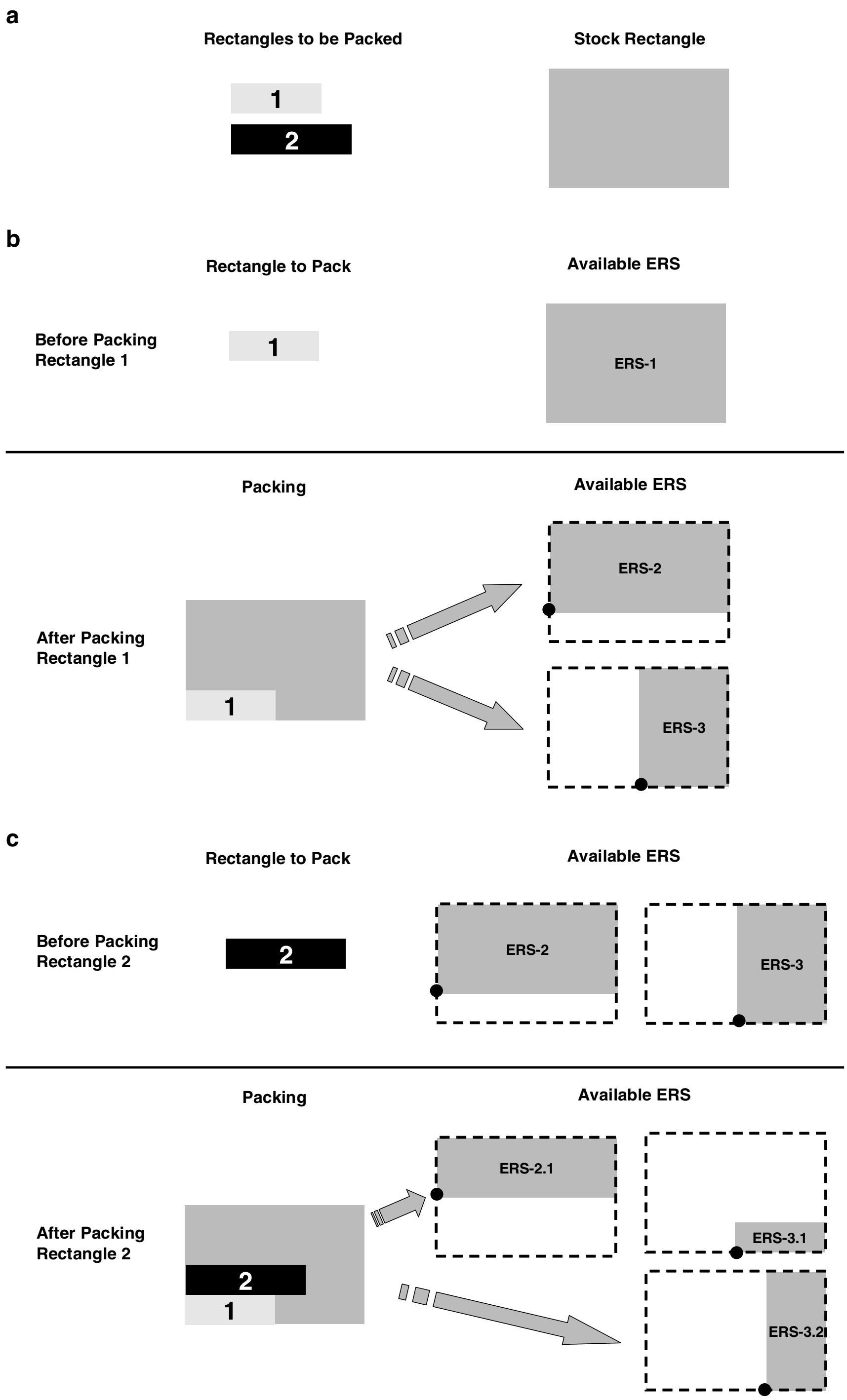

Fig. 7. Example of difference process procedure.

higher potential to have its trim loss reduced than *Packing-1*.

In order to be able to capture the improvement potential of different packings which have the same *%Trim Loss* we developed a new fitness measure which we will call *Modified %Trim Loss*.

The basic idea consists in subtracting from the *%Trim Loss* of a packing a small value which is

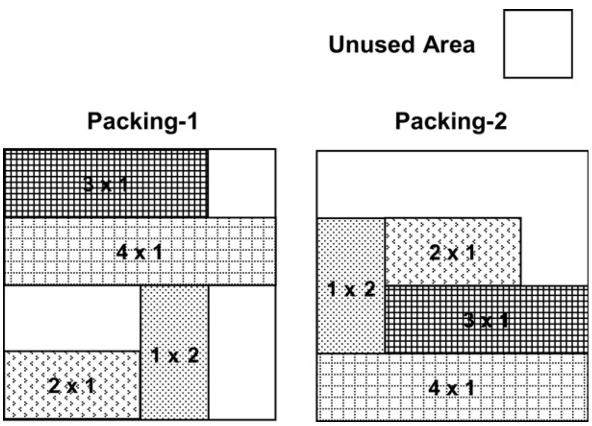

Fig. 8. Two packings with different improvement potential.

proportional to the largest empty rectangular space remaining in the packing, i.e.,

$$\textit{Modified \%Trim Loss} = \textit{\%Trim Loss} - \beta \times \frac{\text{Area of largest ERS}}{\text{Stock rectangle area}} \times 100\%,$$

where β is a factor that depends on the problem data. The factor β should be chosen carefully because the modified trim loss must be able to accommodate the following two objectives:

- For solutions with the same real trim loss we want the modified trim loss to assign better fitness values (to give a smaller trim loss) to those that have the empty space concentrated (i.e., have large empty spaces rather than a lot of small empty spaces).
- We do not want to have empty spaces in the final solutions obtained by the overall algorithm since the empty spaces do not reduce the real trim loss.

Very small values of β will not promote the solutions with large empty spaces and large values of β will lead the genetic algorithm into considering that all we want is finding solutions with large empty spaces.

In order to avoid having solutions with large empty spaces we imposed that the reduction of the *%Trim Loss* caused by the term

$$\beta \times \frac{\text{Area of largest ERS}}{\text{Stock rectangle area}} \times 100\%$$

should be significantly less than the reduction caused by the packing of the smallest rectangle in the problem, i.e.,

$$\beta \times \frac{\text{Area of largest ERS}}{\text{Stock rectangle area}} \ll \frac{\text{Area of smallest rectangle}}{\text{Stock rectangle area}}.$$

Note that, in the worst case, we might have

Area of largest ERS = Stock rectangle area,

which is equivalent to having

$$\beta \ll \frac{\text{Area of smallest rectangle}}{\text{Stock rectangle area}}.$$

We chose to compute β using the following expression:

$$\beta = K \times \frac{\text{Area of smallest rectangle}}{\text{Stock rectangle area}},$$

where $K \ll 1$ is a constant. Note that this expression also satisfies the previous expression.

In order to determine the appropriate value of K we performed some experimental tests (see Section 4.1. for details) with values of K between 0.01 and 0.10 in steps of 0.01 and concluded that 0.03 was the best value.

We can now fully define the *Modified %Trim Loss*, *MTL*, as

$$MTL = \textit{\%Trim Loss} - 0.03 \times \frac{\text{Area of smallest rectangle}}{\text{Stock rectangle area}} \times \frac{\text{Area of largest ERS}}{\text{Stock rectangle area}} \times 100\%.$$

The *Modified %Trim Loss* for *Packing-1* and *Packing-2* of Fig. 8 are

$$MTL_1 = \left(\frac{5}{16} - 0.03 \times \frac{2}{16} \times \frac{2}{16}\right) \times 100\% = \frac{4.88}{16} \times 100\%,$$

$$MTL_2 = \left(\frac{5}{16} - 0.03 \times \frac{2}{16} \times \frac{4}{16}\right) \times 100\% = \frac{4.76}{16} \times 100\%.$$

As desired *Packing-2* has a smaller *MTL* than *Packing-1* (small is better).

The usage of the *Modified %Trim Loss* as fitness function was responsible for improving significantly the *%Trim Loss* values obtained in the experimental tests.

4. Experimental results

In this section we present the results obtained in a set of experiments we conducted in order to evaluate the performance of the Hybrid Genetic Algorithm (HGA) proposed in this paper.

The algorithm was implemented in Visual Basic 6.0 and the experimental tests were run on a computer with a 1.333 GHz, a AMD Thunderbird CPU and the Windows XP operating system.

The experiments were devised in order to evaluate two types of performance:

- *Relative performance* – in this type of experiments we try to evaluate the performance of HGA against other meta-heuristic approaches.
- *Absolute performance* – in this type of experiments we try to evaluate how close the trim loss values obtained by the HGA are to the optimal values.

4.1. GA configuration

The present state-of-the-art theory on genetic algorithms does not provide information on how to configure them. In our past experience with genetic algorithms based on the same evolutionary strategy, see Gonçalves and Almeida (2002), Gonçalves et al. (2005) and Gonçalves and Resende (2004), we obtained good results with values of *TOP*, *BOT* and Crossover Probability (*CProb*) in the following intervals:

Parameter	Interval
TOP	0.10–0.20
BOT	0.15–0.30
Crossover Probability (CProb)	0.70–0.80

For the population size we obtained good results by indexing it to the size of the problem, i.e., use small size populations for small problems and larger populations for larger problems. Having this past experience in mind and in order to obtain a reasonable configuration we conducted a small pilot study using the following three problem instances:

Problem source	Size (number of rectangles)
Lai and Chan (1997) – Instance 3	20
Jakobs (1996) – Instance 5	30
Leung et al. (2003) – Instance 1	40

We tested all the combinations of the following values:

- $TOP = (0.10, 0.15, 0.20)$;
- $BOT = (0.15, 0.20, 0.25, 0.30)$;
- $CProb = (0.70, 0.75, 0.80)$;
- *Population sizes* with 2, 5, 10 and 15 times the number of rectangles in the problem instance.

For each of the above three instances and the 144 possible configurations we run the algorithm with three seeds and computed the average trim loss. The configuration that minimized the sum, over the three problems instances, of the average trim loss was chosen, i.e., $TOP = 15\%$, $BOT = 15\%$, $CProb = 0.7$ and *population size* $= 10 \times$ number of rectangles in the problem instance.

In order to determine the appropriate value of K for the modified trim loss we tested the algorithm using values of K between 0.01 and 0.10 in steps of 0.01 on the following four problem instances:

Problem source	Size (number of rectangles)
Jakobs (1996) – Instance 2	30
Leung et al. (2003) – Instance 1	40
Hopper and Turton (2001) – Instance C4-3	49
Hopper and Turton (2001) – Instance C5-1	73

These instances were chosen because the genetic algorithm was not obtaining good results for them using as fitness the usual trim loss. We run the algorithm, using the above best configuration, with three seeds and computed the average modified trim loss. The value of K that minimized the sum, over the four problems instances, of the average modified trim loss was chosen, i.e., $K = 0.03$. The detailed results are given in Appendix in Table 9.

The configuration presented below was held constant for all experiments and all problems instances.

Population size	Min ($10 \times$ No. rect., 1000)
Crossover Probability	0.7
TOP	The top 15% from the previous population chromosomes are copied to the next generation.

BOT	The bottom 15% of the population chromosomes are replaced with randomly generated chromosomes.
Fitness	Modified Trim Loss (see 3.3) (to minimize)
Stopping criterion	2000 Generations

The following experimental results demonstrate that this configuration provides excellent results in terms of solution quality and that it is very robust.

The next sections present the results of the experimental tests.

4.2. Relative performance

In the packing and cutting literature there are not many references presenting results for the two-dimensional orthogonal packing problem addressed in this paper. We compare the HGA against the following three approaches:

- *SGA* – Genetic Algorithm, proposed by Leung et al. (2003). This approach uses a genetic algorithm follows the basic architecture proposed by Holland (1975) and Goldberg (1989) (and is not based on random keys) to obtain the sequence in which the rectangles are placed. A bottom-left procedure is used subsequently to place the rectangles. The crossover process is accomplished by two crossover operators which are modifications of the ones proposed by Jakobs (1996) and Goldberg (1989) and which are selected with equal probabilities.
- *MSAGA* – Mixed Simulated Annealing-Genetic Algorithm, proposed by Leung et al. (2003). This approach combines a simulated annealing (SA) with a genetic algorithm. After a classical GA is applied to generate a new population the SA procedure is used to decide which parents and children remain in the population. By doing this Leung et al. (2003) hope to decrease the probability of the GA being trapped in local optima of poor quality.
- *GRASP* – Greedy Randomized Adaptive Search Procedure, proposed by Parreño (2004). This approach was developed to solve a more general problem where the rectangles can have a value that is different from their area. In spite of this we have included Parreño (2004) because he has run some experiments where the value of each rectangle is equal to the area (and that is the problem we are trying to solve in this paper).

For this type of experiments we have used the set of problem instances used by Leung et al. (2003) which are composed by the following 21 instances:

- 3 instances from Lai and Chan (1997) (instances 2 and 10 of the paper of Leung et al. (2003) are not correct, in instance 2 rectangle 10 should be 100×70 and in instance 10 rectangle 12 should be 16×14 and rectangle 20 should be 32×8);
- 5 instances from Jakobs (1996);
- 2 instances from Leung et al. (2003);
- 9 instances from Categories C5–C7 of Hopper and Turton (2001).

All these problem instances have known optimal solution where the trim loss is zero.

Leung et al. (2003) present results for 3 versions of their SGA and MSAGA algorithms corresponding to the usage of 0.3, 0.7 and 1.0 for the mutation ratio. For ach version they present the minimum and the average *%Trim Loss* value obtained over 15 runs. Each run consists of 30,000 generations of a population with 100 chromosomes.

Parreño (2004) runs a GRASP approach for 10,000 iterations and only provides the minimum value obtained for the first 10 instances in Leung et al. (2003).

In order to make comparisons fair and meaningful we replicated our HGA a number of times such that the number of solutions evaluated by it, for each problem instance, is close to the number used by Leung et al. (2003), i.e., $15 \times 30{,}000 \times 100 = 45{,}000{,}000$. We chose to replicated our HGA, for each problem, 20 times, which, in the worst case, is equivalent to evaluating $20 \times 2000 \times 1000 = 40{,}000{,}000$ solutions (<45,000,000).

Table 1 summarizes the results. It lists problem source, instance number, stock rectangle dimension (width $\times$ height), no. of rectangles, optimal value for the trim loss (opt.) and minimum and average trim loss obtained over the no. of replications of each algorithm/version.

As can be seen from Table 1 the HGA clearly outperforms, in terms of solution quality, all of the other seven algorithm/versions.

The HGA obtained the best average values for all of the 19 problem instances and obtained the best

Table 1
Relative performance of HGA

Problem source	Instance	Dimensions $W \times H$	No. of rectangles	Opt.	Leung et al. (2003) SGA			Leung et al. (2003) Mixed SAGA			Parreño (2004) GRASP	HGA
					Mutation rate			Mutation rate				
					0.3	0.7	1	0.3	0.7	1		
					Minimum Average			Minimum Average			Minimum	Minimum Average
Lai and Chan (1997)	1	400×200	10	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
					0	**0**	**0**	**0**	**0**	**0**		**0**
	2	400×200	15	0	**0**	**0**	**0**	**0**	**0**	**0**	0.01250000	**0**
					0	**0**	**0**	**0**	**0**	**0**		**0**
	3	400×400	20	0	**0**	**0**	**0**	**0**	**0**	**0**	0.03375000	**0**
					0.02250000	0.01350000	0.01462500	0.01687500	0.01237500	0.02400000		**0.00084352**
Jakobs (1996)	1	70×80	20	0	**0**	**0**	**0**	**0**	**0**	**0**	0.02732143	**0**
					0.03673810	0.02909520	0.02313100	0.00182143	0.01204760	0.02122620		**0**
	2	70×80	25	0	0.02142860	0.02589290	0.02428570	0.01571430	0.02142860	0.02142860	0.02589286	**0.01071252**
					0.03705950	0.03819050	0.03369050	0.02660710	0.03111900	0.03279760		**0.02543516**
	3	120×45	25	0	**0**	**0**	**0**	**0**	**0**	**0**	0.01333333	**0**
					0.01288890	0.00911111	0.00611111	0.00177778	0.00266667	0.00400000		**0.00133298**
	4	90×45	30	0	0.01777780	**0**	**0**	**0**	**0**	0.01777780	0.01777778	**0**
					0.01896300	0.01777780	0.01718520	0.00948148	0.01185190	0.01777780		**0.00355452**
	5	65×45	30	0	0.01230770	**0**	**0**	**0**	**0**	0.01230770	0.01846154	**0**
					0.02140170	0.02222220	0.01736750	0.01312820	0.00929915	0.01996580		**0.00830548**
Leung et al. (2003)	1	150×110	40	0	0.02496970	0.02278790	0.02036360	0.01890910	0.01527270	**0.01090910**	0.03903030	0.01527238
					0.03335760	0.02946260	0.02799190	0.03107880	0.02614950	0.02555150		**0.02299344**
	2	160×120	50	0	0.02208330	0.02250000	0.01895830	0.02270830	0.01958330	0.02083330	0.02979167	**0.01249960**
					0.02980560	0.02765280	0.02641670	0.02666670	0.02325000	0.02756940		**0.02047848**
Hopper and Turton (2001)	C5-1	60×90	73	0	0.00666667	0.00388889	0.00388889	0.00444444	0.00462963	0.00648148	–	**0.00222178**
					0.00993827	0.00971605	0.00839506	0.00979012	0.00833333	0.01024690		**0.00543467**
	C5-2	60×90	73	0	**0.00111111**	0.00259259	0.00407407	0.00148148	0.00259259	0.00370370	–	0.00222183
					0.00753086	0.00672840	0.00764198	0.00543210	0.00600000	0.00677778		**0.00527702**
	C5-3	60×90	73	0	0.00740741	0.00814815	0.00518519	0.00518519	0.00333333	0.00677777	–	**0.00148126**
					0.01308640	0.01116050	0.00998765	0.00918519	0.00924691	0.00962963		**0.00449955**
	C6-1	80×120	97	0	0.00729167	0.00645833	0.00687500	0.00562500	0.00583333	0.00656250	–	**0.00458302**
					0.01202080	0.01086810	0.01079170	0.00835417	0.00968750	0.01034720		**0.00653092**
	C6-2	80×120	97	0	0.00458333	0.00437500	0.00333333	0.00208333	0.00260417	0.00395833	–	**0.00104154**
					0.00706250	0.00675000	0.00695833	0.00438889	0.00530556	0.00733333		**0.00308812**
	C6-3	80×120	97	0	0.00854167	0.00583333	0.00447917	0.00468750	0.00447917	0.00427083	–	**0.00374969**
					0.01069440	0.00917361	0.00793056	0.00759722	0.00695139	0.00857639		**0.00587981**
	C7-1	160×240	196	0	0.01429690	0.01388020	0.01171880	0.01190100	0.01119790	0.01226560	–	**0.00869767**
					0.01643750	0.01545310	0.01411110	0.01506770	0.01312670	0.01487850		**0.01082784**
	C7-2	160×240	197	0	0.00375000	0.00484375	0.00375000	0.00283854	0.00442708	0.00453125	–	**0.00265625**
					0.00652083	0.00603472	0.00601562	0.00664757	0.00569271	0.00627778		**0.00371224**
	C7-3	160×240	196	0	0.00794271	0.00804688	0.00776042	0.00791667	0.00776042	0.00723958	–	**0.00513021**
					0.01076040	0.01034030	0.00962847	0.00982118	0.00920313	0.00959549		**0.00728906**

Bold cells mean best minimum or best average.

minimum Trim Loss values for 17 of the problem instances. MSAGA-1 obtained the best minimum Trim Loss for instance 2 of Leung et al. (2003) and SGA-0.3 obtained the best minimum Trim Loss for instance C5-2 of Hopper and Turton (2001).

In terms of computational times we cannot make any fair and meaningful comparisons since no times are reported in the case of Leung et al. (2003) and since all the approaches were implemented with different programming languages and tested in computers with different computing power. Instead we report the average computation run times the HGA approach took for each problem instance and for 2000 generations, see Table 2.

4.3. Absolute performance

In this section we present the results from the experimental tests we conducted in order to evaluate how close the trim loss values obtained by the HGA are to the optimal values.

For the experimental tests of this section the following four problem set instances were used:

Set	Problem source	No. of instances	No. of rectangles
I	Hifi and Zissimopoulos (1998) ftp://panoramix.univ-paris1.fr/pub/CERMSEM/hifi/Strip-cutting/	25	7–22
II	Beasley (1985), Hadjiconstantinou and Christofides (1995), Wang (1983), Christofides and Whitlock (1977) and Fekete and Schepers (1997)	21	7–97
III	Hopper and Turton (2001) http://home.t-online.de/home/villahost/eva/data.htm#R1	21	16–197
IV	Hopper and Turton (2000) http://home.t-online.de/home/villahost/eva/data.htm#R1	35	17–197

Table 2
Average computational run times

Problem source	Instance	Dimensions $W \times H$	No. of rectangles	Time (seconds)
Lai and Chan (1997)	1	400×200	10	12.82
	2	400×200	15	25.24
	3	400×400	20	45.75
Jakobs (1996)	1	70×80	20	55.25
	2	70×80	25	87.87
	3	120×45	25	84.13
	4	90×45	30	105.63
	5	65×45	30	107.62
Leung et al. (2003)	1	150×110	40	218.50
	2	160×120	50	412.54
Hopper and Turton (2001)	C5-1	60×90	73	936.00
	C5-2	60×90	73	937.12
	C5-3	60×90	73	963.75
	C6-1	80×120	97	1977.88
	C6-2	80×120	97	1922.75
	C6-3	80×120	97	2049.25
	C7-1	160×240	196	6570.00
	C7-2	160×240	197	6310.00
	C7-3	160×240	196	6587.99

These problem set instances have been used by other authors for experimental tests of algorithms for the Strip Cutting Problem. They were chosen because they come from various sources, they have instances with small and large number of rectangles and because they have a known value for the optimal trim loss.

For these experimental tests, and since we were not comparing with other approaches, we replicated the HGA 10 times for each problem instance.

Tables 3–6 present the minimum and average Trim Loss values obtained for each instance of the problem sets I, II, III and IV, respectively.

For problem set I (see Table 3) the HGA obtained the optimal trim loss for all the 25 instances and for all the 10 replications (average = minimum). Since the problem instances of this set have a small number of rectangles (7–22) obtaining the optimal value is not as relevant as the fact that the optimal value was obtained on all the 10 replications and therefore, confirming the robustness of the proposed configuration.

For problem set II (see Table 4) the optimal or best known trim loss values were obtained from Oliveira (2004). For this set the HGA obtained the optimal Trim Loss values for all the 19 instances with known optimal value, obtained 3 Trim Loss values equal to best known Trim Loss values and was able to improve the best known Trim Loss for

Table 3
Experimental results for problem set I

Problem source	Instance	Dimensions $W \times H$	No. of rectangles	Optimum	Minimum Average
Hifi and Zissimopoulos (1998)	SCP1	13 × 5	10	0	**0** 0
	SCP2	40 × 4	11	0.01250000	**0.01250000** 0.01250000
	SCP3	14 × 6	15	0	**0** 0
	SCP4	20 × 6	11	0.07500000	**0.07500000** 0.07500000
	SCP5	20 × 20	8	0.56250000	**0.56250000** 0.56250000
	SCP6	38 × 30	7	0.16140351	**0.16140351** 0.16140351
	SCP7	14 × 15	8	0.18095238	**0.18095238** 0.18095238
	SCP8	17 × 15	12	0.05098039	**0.05098039** 0.05098039
	SCP9	68 × 27	12	0	**0** 0
	SCP10	80 × 50	8	0.02500000	**0.02500000** 0.02500000
	SCP11	48 × 27	10	0.02083333	**0.02083333** 0.02083333
	SCP12	34 × 81	18	0	**0** 0
	SCP13	50 × 70	7	0.16971429	**0.16971429** 0.16971429
	SCP14	69 × 100	10	0.13463768	**0.13463768** 0.13463768
	SCP15	34 × 45	14	0	**0** 0
	SCP16	33 × 6	14	0.03535354	**0.03535354** 0.03535354
	SCP17	39 × 42	9	0.12820513	**0.12820513** 0.12820513
	SCP18	101 × 70	10	0.12404526	**0.12404526** 0.12404526
	SCP19	26 × 5	12	0.06923077	**0.06923077** 0.06923077
	SCP20	21 × 15	10	0.10158730	**0.10158730** 0.10158730
	SCP21	145 × 30	11	0.07471264	**0.07471264** 0.07471264
	SCP22	34 × 90	22	0.00457516	**0.00457516** 0.00457516
	SCP23	35 × 15	12	0.20380952	**0.20380952** 0.20380952
	SCP24	114 × 50	10	0.09947368	**0.09947368** 0.09947368
	SCP25	36 × 25	15	0.03666667	**0.03666667** 0.03666667

Bold cells mean minimum value = optimal.

instance 2 of Fekete and Schepers (1997). Also, it is important to point out the fact that for 18 of the problem instances the optimal/best known value was obtained on all the 10 replications.

For problem set III (see Table 5) the HGA obtained the optimal trim loss values for 8 of the 21 problem instances. For all the other instances the minimum Trim Loss value was always less

Table 4
Experimental results for problem set II

Problem source	Instance	Dimensions $W \times H$	No. of rectangles	Optimum or best known	Optimality?	Minimum Average
Beasley (1985)	1	10×10	10	0.05000000	Yes	**0.05000000** 0.05000000
	2	10×10	17	0.03000000	Yes	**0.03000000** 0.03000000
	3	10×10	21	0	Yes	**0** 0
	4	15×10	7	0.08000000	Yes	**0.08000000** 0.08000000
	5	15×10	14	0.06666667	Yes	**0.06666667** 0.06666667
	6	15×10	15	0	Yes	**0** 0
	7	20×20	8	0.56250000	Yes	**0.56250000** 0.56250000
	8	20×20	13	0.05000000	Yes	**0.05000000** 0.05000000
	9	20×20	18	0.02500000	Yes	**0.02500000** 0.02500000
	10	30×30	13	0.02333333	Yes	**0.02333333** 0.02333333
	11	30×30	15	0.06444444	Yes	**0.06444444** 0.06444444
	12	30×30	22	0.00222222	Yes	**0.00222222** 0.00222222
Hadjiconstantinou and Christofides (1995)	1	30×30	7	0.15444444	Yes	**0.15444444** 0.15444444
	2	30×30	15	0.10333333	?	**0.10333333** 0.10333333
Wang (1983)	1	70×40	42	0.02642857	Yes	**0.02642857** 0.02642857
Christofides and Whitlock (1977)	12	40×70	62	0.02642857	Yes	**0.02642857** 0.02642857
Fekete and Schepers (1997)	1	100×100	50	0.00260000	Yes	**0.00260000** 0.00260000
	2	100×100	30	0.02270000	?	**0.01240000** 0.01987979
	3	100×100	30	0.01230000	Yes	**0.01230000** 0.01385642
	4	100×100	61	0.00240000	?	**0.00240000** 0.00240000
	5	100×100	97	0.00180000	?	**0.00180000** 0.00253996

Bold cells mean minimum value = optimal value or minimum value $\leqslant$ best known value.

than 1%. Table 6 summarizes the Trim Loss values obtained for each instance category. For comparison purposes, we include in Table 6, a column with the best height deviation values reported by Hopper and Turton (2001) in Table 5 of their experimental tests for the Strip Cutting Problem (SCP).

The trim loss values obtained by the HGA are 3.6–13.6 times less than the values reported by Hopper and Turton (2001) in Table 5. Even though the SCP is not identical to the problem addressed in this paper (but is quite similar), we think it fair to say that, the Trim Loss values obtained by the HGA, for this problem set, are excellent in terms of solution quality.

For problem set IV (see Table 7) the HGA obtained the optimal trim loss values for 5 of the 35 problem instances. For all the other instances the minimum Trim Loss value was always less than 3.17%.

Table 5
Experimental results for set III

Problem source	Instance	Dimensions $W \times H$	No. of rectangles	Optimum	Minimum Average
Hopper and Turton (2001)	C1-1	20×20	16	0	**0** 0
	C1-2	20×20	17	0	**0** 0.01099797
	C1-3	20×20	16	0	**0** 0
	C2-1	40×15	25	0	**0** 0.00166647
	C2-2	40×15	25	0	**0** 0.00099975
	C2-3	40×15	25	0	**0** 0
	C3-1	60×30	29	0	**0** 0.00483255
	C3-2	60×30	29	0	0.00777661 0.01160954
	C3-3	60×30	29	0	**0** 0.00805407
	C4-1	60×60	49	0	0.00555506 0.00952694
	C4-2	60×60	49	0	0.00833233 0.01027662
	C4-3	60×60	49	0	0.00333292 0.00494386
	C5-1	60×90	73	0	0.00222178 0.00512912
	C5-2	60×90	73	0	0.00222183 0.00496216
	C5-3	60×90	73	0	0.00148126 0.00399957
	C6-1	80×120	97	0	0.00458302 0.00621842
	C6-2	80×120	97	0	0.00124987 0.00328081
	C6-3	80×120	97	0	0.00374969 0.00583294
	C7-1	160×240	196	0	0.00937476 0.01073933
	C7-2	160×240	197	0	0.00265625 0.00379948
	C7-3	160×240	196	0	0.00528646 0.00774739

Bold cells mean minimum value = optimal value.

Table 6
Problem set III summary of minimum trim loss for categories C1–C7 of Hopper and Turton (2001)

Problem source	Instance	No. of rectangles	Best height deviation from optimal (Table 5 – Hopper and Turton (2001))	HGA minimum trim loss
Hopper and Turton (2001)	C1	16–17	0.04	0
	C2	25	0.06	0
	C3	29	0.05	0–0.0078
	C4	49	0.03	0.0033–0.0083
	C5	73	0.03	0.0015–0.0022
	C6	97	0.03	0.0013–0.0046
	C7	196–197	0.04	0.0027–0.0094

Table 7
Experimental results for problem set IV

Problem source	Instance	Dimensions $W \times H$	No. of rectangles	Optimum	Minimum	Average
Hopper and Turton (2000)	N1a	200×200	17	0	**0**	0.021906
	N1b	200×200	17	0	**0**	0.012306
	N1c	200×200	17	0	**0**	0.001260
	N1d	200×200	17	0	**0**	0.016433
	N1e	200×200	17	0	**0**	0.012866
	N2a	200×200	25	0	0.029700	0.033267
	N2b	200×200	25	0	0.023650	0.033209
	N2c	200×200	25	0	0.027350	0.039615
	N2d	200×200	25	0	0.025650	0.030202
	N2e	200×200	25	0	0.028200	0.033089
	N3a	200×200	29	0	0.028675	0.033262
	N3b	200×200	29	0	0.022625	0.035355
	N3c	200×200	29	0	0.020575	0.031782
	N3d	200×200	29	0	0.031700	0.039237
	N3e	200×200	29	0	0.028050	0.032867
	N4a	200×200	49	0	0.017100	0.027362
	N4b	200×200	49	0	0.018700	0.024460
	N4c	200×200	49	0	0.017350	0.021212
	N4d	200×200	49	0	0.017400	0.022285
	N4e	200×200	49	0	0.015250	0.024575
	N5a	200×200	73	0	0.013600	0.017462
	N5b	200×200	73	0	0.013500	0.015795
	N5c	200×200	73	0	0.015025	0.018785
	N5d	200×200	73	0	0.020150	0.023877
	N5e	200×200	73	0	0.016375	0.018845
	N6a	200×200	97	0	0.009250	0.012152
	N6b	200×200	97	0	0.008125	0.010147
	N6c	200×200	97	0	0.008500	0.010735

Table 7 (*continued*)

Problem source	Instance	Dimensions $W \times H$	No. of rectangles	Optimum	Minimum Average
	N6d	200×200	97	0	0.014100 0.016777
	N6e	200×200	97	0	0.010350 0.012587
	N7a	200×200	197	0	0.004325 0.005222
	N7b	200×200	197	0	0.002000 0.004867
	N7c	200×200	197	0	0.003150 0.005177
	N7d	200×200	197	0	0.002925 0.004927
	N7e	200×200	197	0	0.004850 0.005990

Bold cells mean minimum value = optimal value.

Table 8
Problem set IV summary of minimum trim loss for categories N1–N7 of Hopper and Turton (2000)

Problem source	Instance	No. of rectangles	Best height deviation from optimal (Table F.26 – Hopper and Turton (2000))	HGA minimum trim loss
Hopper and Turton (2000)	N1	17	0.06	0
	N2	25	0.06	0.0237–0.0297
	N3	29	0.06	0.0206–0.0317
	N4	49	0.07	0.0153–0.0187
	N5	73	0.07	0.0135–0.0202
	N6	97	0.07	0.0081–0.0141
	N7	197	0.05	0.0020–0.0049

Table 8 summarizes the Trim Loss values obtained for each instance category. For comparison purposes, we include in Table 8, a column with the best height deviation values reported by Hopper and Turton (2000) in Table F.26 of the experimental tests for the Strip Cutting Problem (SCP). The trim loss values obtained the HGA are 1.9 to 10.2 times less than the values reported by Hopper and Turton (2000) in Table F.26. Even though the SCP is not identical to the problem addressed in this paper (but is quite similar), we think it fair to say that, the Trim Loss values obtained by the HGA, for this problem set, are excellent in terms of solution quality.

5. Conclusions and future work

In this paper we addressed a two-dimensional (2D) orthogonal packing problem, where a fixed set of small rectangles has to be placed on a larger stock rectangle in such a way that the amount of trim loss is minimized. An algorithm which hybridizes a placement procedure with a genetic algorithm based on random keys was proposed. The approach was tested on a set of instances taken from the literature and compared with other approaches. The experimental results demonstrate the excellent quality of the proposed algorithm when compared with other meta-heuristics and in absolute terms.

Future work can be done by extending the approach to the strip cutting problem and the weighted version of the orthogonal packing problem.

Appendix A

See Table 9.

Table 9
Detailed results of tests to determine best value of K

K	Problem source	Average best value (over 3 seeds)	Sum of average best value (over 4 instances)
0.01	Hopper and Turton (2001) – Instance C4-3	0.007222022	
0.01	Hopper and Turton (2001) – Instance C5-1	0.006049342	
0.01	Jakobs (1996) – Instance 2	0.027558497	
0.01	Leung et al. (2003) – Instance 1	0.024969146	0.065799007
0.02	Hopper and Turton (2001) – Instance C4-3	0.007222022	
0.02	Hopper and Turton (2001) – Instance C5-1	0.006049342	
0.02	Jakobs (1996) – Instance 2	0.022617632	
0.02	Leung et al. (2003) – Instance 1	0.024322333	0.060211329
0.03	Hopper and Turton (2001) – Instance C4-3	0.00574071	
0.03	Hopper and Turton (2001) – Instance C5-1	0.005617264	
0.03	Jakobs (1996) – Instance 2	0.022612733	
0.03	Leung et al. (2003) – Instance 1	0.022626043	0.056596751
0.04	Hopper and Turton (2001) – Instance C4-3	0.005740679	
0.04	Hopper and Turton (2001) – Instance C5-1	0.005617264	
0.04	Jakobs (1996) – Instance 2	0.022613996	
0.04	Leung et al. (2003) – Instance 1	0.022626043	0.056597983
0.05	Hopper and Turton (2001) – Instance C4-3	0.006759	
0.05	Hopper and Turton (2001) – Instance C5-1	0.006049348	
0.05	Jakobs (1996) – Instance 2	0.022615259	
0.05	Leung et al. (2003) – Instance 1	0.024159312	0.059582919
0.06	Hopper and Turton (2001) – Instance C4-3	0.006758957	
0.06	Hopper and Turton (2001) – Instance C5-1	0.006049348	
0.06	Jakobs (1996) – Instance 2	0.022615259	
0.06	Leung et al. (2003) – Instance 1	0.024160234	0.059583797
0.07	Hopper and Turton (2001) – Instance C4-3	0.006758914	
0.07	Hopper and Turton (2001) – Instance C5-1	0.006049348	
0.07	Jakobs (1996) – Instance 2	0.022614628	
0.07	Leung et al. (2003) – Instance 1	0.024160004	0.059582892
0.08	Hopper and Turton (2001) – Instance C4-3	0.006758914	
0.08	Hopper and Turton (2001) – Instance C5-1	0.006049342	
0.08	Jakobs (1996) – Instance 2	0.022615891	
0.08	Leung et al. (2003) – Instance 1	0.024159773	0.05958392
0.09	Hopper and Turton (2001) – Instance C4-3	0.006758914	
0.09	Hopper and Turton (2001) – Instance C5-1	0.006049337	
0.09	Jakobs (1996) – Instance 2	0.022615891	
0.09	Leung et al. (2003) – Instance 1	0.024159543	0.059583684
0.10	Hopper and Turton (2001) – Instance C4-3	0.006758914	
0.10	Hopper and Turton (2001) – Instance C5-1	0.006049332	
0.10	Jakobs (1996) – Instance 2	0.022615891	
0.10	Leung et al. (2003) – Instance 1	0.024159543	0.059583679

References

Bean, J.C., 1994. Genetics and random keys for sequencing and optimization. ORSA Journal on Computing 6, 154–160.

Beasley, J.E., 1985. An exact two-dimensional non-guillotine cutting tree search procedure. Operations Research 33, 49–64.

Beasley, D., Bull, D.R., Martin, R.R., 1993. An overview of genetic algorithms: Part 1, fundamentals. University Computing 15, 58–69.

Christofides, N., 1995. An exact algorithm for general, orthogonal two-dimensional knapsack problems. European Journal of Operational Research 83, 39–56.

Christofides, N., Whitlock, C., 1977. An algorithm for two dimensional cutting problems. Operations Research 25, 31–44.

Dowsland, K.A., 1993. Some experiments with simulated annealing techniques for packing problems. European Journal of Operational Research 68, 389–399.

Dowsland, K.A., 1996. Genetic algorithms – a tool for OR?. Journal of the Operational Research Society 47 550–561.

Fekete, S.P., Schepers, J., 1997. A new exact algorithm for general orthogonal d-dimensional knapsack problems. Springer Lecture Notes in Computer Science 1284, 144–156.

Glover, F., Kelly, J.P., Laguna, M., 1995. Genetic algorithms and tabu search: Hybrids for optimization. Computers and Operations Research 22, 111–134.

Goldberg, D.E., 1989. Genetic Algorithms in Search Optimization and Machine Learning. Addison-Wesley.

Gonçalves, J.F., Almeida, J.R., 2002. A hybrid genetic algorithm for assembly line balancing. Journal of Heuristics 8, 629–642.

Gonçalves, J.F., Resende, M.G.C., 2004. An evolutionary algorithm for manufacturing cell formation. Computers and Industrial Engineering 47, 247–273.

Gonçalves, J.F., Mendes, J.J.M., Resende, M.G.C., 2005. A hybrid genetic algorithm for the job shop scheduling problem. European Journal of Operational Research 167, 77–95.

Hadjiconstantinou, E., Christofides, N., 1995. An exact algorithm for general, orthogonal, two dimensional knapsack problems. European Journal of Operations Research 83, 39–56.

Hässler, R.W., Sweeney, P.E., 1991. Cutting stock problems and solution procedures. European Journal of Operational Research 54, 141–150.

Hifi, M., Zissimopoulos, V., 1998. Exact algorithms for the guillotine strip cutting/packing problem. Computers and Operations Research 25, 925–940.

Hinxman, A.I., 1980. The trim loss and assortment problems. European Journal of Operational Research 88, 8–18.

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor, Michigan.

Hopper, E., Turton, B.C.H., 2000. An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. European Journal of Operational Research 128, 34–57.

Hopper, E., Turton, B.C.H., 2001. An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. European Journal of Operational Research 128, 34–57.

Jakobs, S., 1996. On genetic algorithms for the packing of polygons. European Journal of Operational Research 88, 165–181.

Lai, K.K., Chan, W.M., 1997. An evolutionary algorithm for the rectangular cutting stock problem. International Journal of Industrial Engineering 4, 130–139.

Leung, T.W., Chan, C.K., Troutt, M., 2000. Comparison of meta-heuristics for the two dimensional orthogonal packing problem. Working Paper, Department of Applied Mathematics, The Hong Kong Polytechnic University, Hong Kong.

Leung, T.W., Chan, C.K., Troutt, M.D., 2003. Application of a mixed simulated annealing-genetic algorithm heuristic for the two-dimensional orthogonal packing problem. European Journal of Operational Research 141, 241–252.

Lodi, A., Martello, S., Monaci, M., 2002. Two-dimensional packing problems: A survey. European Journal of Operational Research 141, 241–252.

Oliveira, J.F., 2004. A computational evaluation of Wang's algorithm and its variants for the 2D constrained rectangular cutting problem. Presented at the EURO XX Conference, Rhodes, 4–7 July.

Parada, V., Sepulveda, M., Solar, M., Gomes, A., 1998. Solution for constrained guillotine cutting problem by simulated annealing. Computers and Operations Research 25, 37–47.

Parreño, F. 2004. Algoritmos heurísticos y exactos para problemas de corte no guillotina en dos dimensiones, Ph.D. thesis.

Sarin, S.C., 1983. Two-dimensional stock cutting problems and solution methodologies. ASME Transactions Journal of Engineering for Industry 104, 155–160.

Spears, W.M., Dejong, K.A., 1991. On the virtues of parameterized uniform crossover. In: Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 230–236.

Tsai, R.D., 1993. Three dimensional palletization of mixed box sizes. IIE Transactions 25, 64–75.

Wang, P.Y., 1983. Two algorithms for constrained two-dimensional cutting stock problems. Operations Research 31, 573–586.

Wäscher, G., Haussner, H., Schumann, H., this issue. An Improved Typology of Cutting and Packing Problems. European Journal of Operational Research.