Waste reduction in Rectangular Figure Cutting using a Genetic Algorithm

Reducción de desperdicios en corte de figuras rectangulares usando algoritmo genético

Juan C. Rodríguez Noriega [1]
Jairo R. Coronado-Hernández [2]
Sergio Leottau [3]

Abstract

This paper introduces a genetic algorithm (GA) to minimize the waste produced during the cutting process of rectangular figures on a sheet. The chromosomes for solution codification use an object-based representation. It has the following operator: Partially Mapped Crossover (PMX), mutation based in double interchange (2-opt), and the elitism strategy for the selection process. The proposed algorithm was applied in a real case situation problem, where the numbers of items were 55 pieces. The result of this implementation was a reduction of the waste as a result of the decrease in the number of sheets used in the cutting process and at the same time an effective employment of the used area.

Key words: Genetic Algorithm, bin packing problem, metaheuristic.

Resumen

Este artículo presenta un algoritmo genético para minimizar el desperdicio producido durante el proceso de corte de figuras rectangulares en una lámina. Los cromosomas para la codificación de la solución usan una representación basada en objetivos. Este tiene el siguiente operador: Mapa de cruce por emparejamiento parcial (PMX), mutación basada en el doble intercambio (2-opc) y la estrategia de elitismo para el proceso de selección. El algoritmo propuesto fue aplicado en una situación problema de un caso real, donde el número de ítems eran 55 piezas. El resultado de esta implementación fue la reducción de desperdicio como resultado de la disminución del número de láminas usadas en los proceso de corte y al mismo tiempo el empleo efectivo del área usada.

Palabras claves: Algoritmo genético, problema de empaquetado, metaheurística.

Date Received: April 7th 2016 - *Fecha de recepción: Abril 7 de 2016*
Date Accepted: June 3rd 2016 - *Fecha de aceptación: Junio 3 de 2016*

[1] Industrial Engineer. Independent consultant. jcnoriega08@gmail.com

[2] Industrial Engineering PhD. Associate Researcher at Grupo LOGER. jaracohe@gmail.com

[3] Systems Engineer. Development Engineer at PayU América Latina. leottau7804@gmail.com

Introduction

This document presents a genetic algorithm (GA) to minimize the waste produced when cutting rectangular figures out of a sheet. In academic literature, this problem is known as the Bin Packing Problem – BPP. The problem consists in searching for the best possible distribution in a sheet format or a predetermined plate, given a set of pieces with different shapes.

Several types of industries require tools to solve this issue, that may be applied to cutting materials such as metal, wood, glass, plastic, paper, leather, among others. For example, it is a common problem in the shipbuilding industry, since rectangular shapes are cut out of metal sheets and then used to join flat and curved blocks in order to assemble vessels. There are three types of packing problems in literature: two-dimensional bin packing problem (2D-BPP); two dimensional cutting stock problem (2D-CSP); and two-dimensional strip packing problem (2D-SPP). This effort addresses the 2D-BPP problem.

This document introduces a genetic algorithm (GA) to solve the problem of cutting rectangular pieces in a rectangular area.

Binkley & Hagiwara, 2006, describe the packing issue through the four corner heuristics with an auto-adaptive genetic algorithm to find the best solution to the problem. Lee, 2008, suggests a GA a crossover operator that will enable generating several offspring from a couple of parents. Albano & Sappuro, 1980 show one of the first researches related to the problem, accepting regular or irregular-shaped pieces; it uses a heuristic as a solution-seeking method.

Methodology

Definition of the Problem

The mathematical problem is shown in Álvarez & Toro, 2009; annex 1 where the model parameters and variables are presented. Equation (1) shows the target function which consists of minimizing the amount of wasted material. Equations (2) to (5) avoid superimposing rectangles on the sheet. Equation (6) ensures that the couple of rectangles assessed with the above equations fit within the dimensions of the sheet. Equations (7) and (8) ensure that the positions of the pieces fit within the sheet dimensions

$$Min\ L.W - \sum_{i=1}^{N} p_i \,.\, q_i \,.\, s_i \tag{1}$$

Sujeto a:

$$x_i + p_i \,.\, l_{xi} + q_i \,.\, w_{xi} \le x_k + (1 - a_{ik}) * M, \quad \forall i,\ k,\ i < k \tag{2}$$

$$x_k + p_k \,.\, l_{xk} + q_k \,.\, w_{xk} \le x_i + (1 - b_{ik}) * M, \quad \forall i,\ k,\ i < k \tag{3}$$

$$y_i + p_i \,.\, l_{yi} + q_i \,.\, w_{yi} \le y_k + (1 - c_{ik}) * M, \quad \forall i,\ k,\ i < k \tag{4}$$

$$y_k + p_k \,.\, l_{yk} + q_k \,.\, w_{yk} \le y_i + (1 - d_{ik}) * M, \quad \forall i,\ k,\ i < k \tag{5}$$

$$a_{ik} + b_{ik} + c_{ik} + d_{ik} \ge s_i + s_k - 1, \forall i,\ k,\ i < k \tag{6}$$

$$x_i + p_i \,.\, l_{xi} + q_i \,.\, w_{xi} \le L + (1 - s_i) * M, \quad \forall i \tag{7}$$

$$y_i + p_i \,.\, l_{yi} + q_i \,.\, w_{yi} \le W + (1 - s_i) * M, \quad \forall i \tag{8}$$

$$x_i + p_i \le x_k + (1 - a_{ik}) * M, \quad \forall i,\ k,\ i < k \tag{9}$$

$$x_k + p_k \le x_i + (1 - b_{ik}) * M, \quad \forall i,\ k,\ i < k \tag{10}$$

$$y_i + q_i \le y_k + (1 - c_{ik}) * M, \quad \forall i,\ k,\ i < k \tag{11}$$

$$y_k + q_k \le y_i + (1 - d_{ik}) * M, \quad \forall i,\ k,\ i < k \tag{12}$$

$$x_i + p_i + q_i \le L + (1 - s_i) * M, \quad \forall i \tag{13}$$

$$y_i + p_i + q_i \le W + (1 - s_i) * M, \quad \forall i \tag{14}$$

Packing problem models without piece rotation are obtained by replacing equations (2) to (8) for equations (9) to (14), respectively.

Genetic Algorithm Design

To solve the problem, most procedures reported in literature are based on metaheuristic techniques *(Álvarez & Toro, 2009).* Therefore, this paper addresses the problem from the genetic algorithm implementation standpoint.

Individual representation was based on objects *(Lee, 2008)* as a solution coding mechanism; this is due to the easy implementation of genetic operators. Object based coding consists of a swap of the given pieces. The values for each gene represent the pieces, whereas the sub index is only useful to indicate the packing order on the sheets, e.g.: given the following chromosome 3 2 4 5 6 1, Fig. 1 shows that pieces 3,2 are cut out of sheet 1, pieces 4, 5 are cut out of sheet 2, and pieces 6, 1, are cut out of sheet 3.

Fig. 1. Object or piece-based representation.

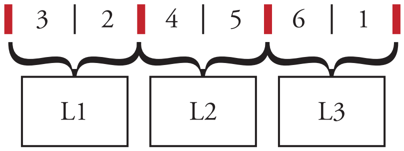


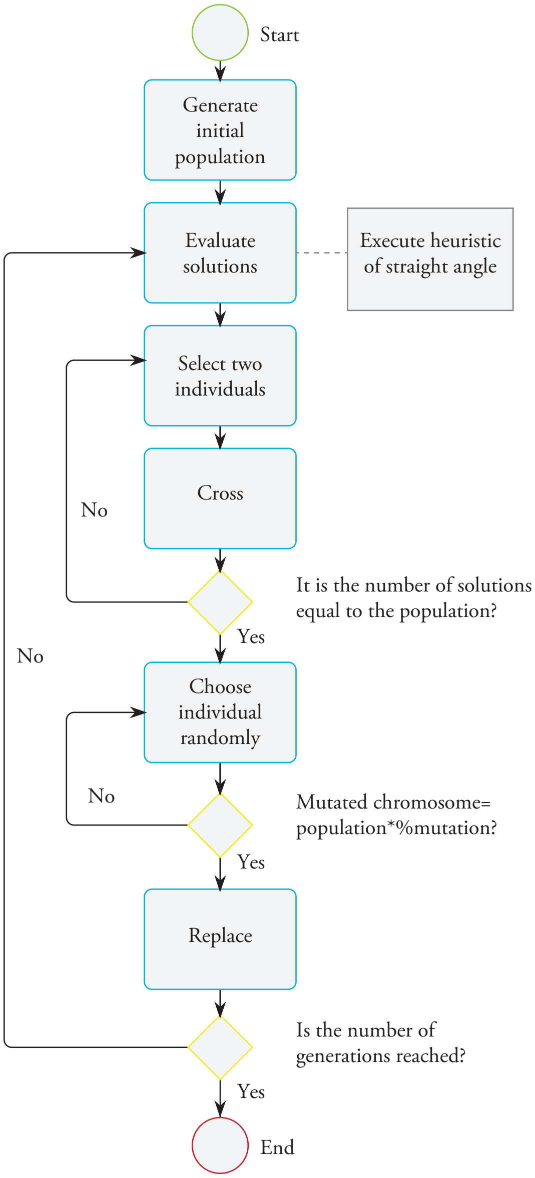
Fig. 1 shows the structure of the genetic algorithm used to solve the problem.

The starting population is generated at random, using right angle polygon heuristics (Bottom Left) proposed in *(Jakobs, 1996)*. This heuristics is used to define the way in which the pieces will be cut out of the sheet and therefore, defines the adaptation for each individual with the fitness function calculated as $F = \sum_{i=1}^{m} \sum_{j=1}^{n} (A_i - t_j)$. Where A_i is the area of the sheet *i*, and tj is the area of piece *j* packed in sheet i. In this case, the area of the last sheet is considered to be calculated with the height of the piece that stands out the most from the rest of the pack, which is known as maximum height.

Next, we carry out selection, where the individuals that will be taken to reproduction, in accordance with the results yielded by the fitness function, are defined. After several tests, the SUS-stochastic universal sampling operator was used for the selection process. In the SUS crossover operator a random number is required, and with such number all the individuals that will make part of the reproduction process are obtained.

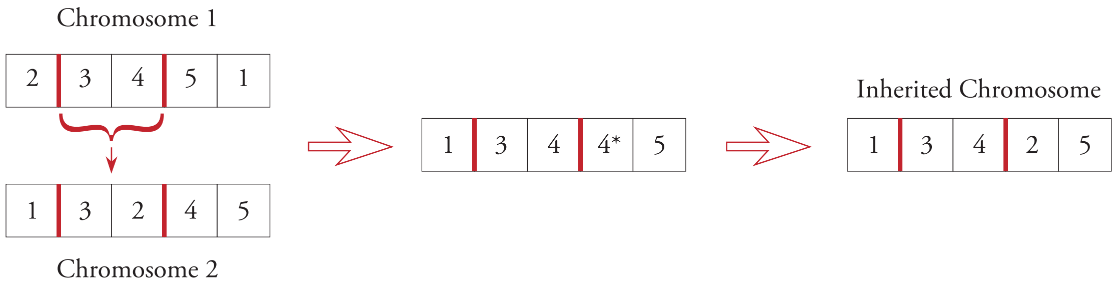
Once the fittest have been selected, a genetic code combination is performed and as a result, new chromosomes called offspring are produced through the partially matched crossover (PMX) operator. During crossover, a segment of the chromosome chains of one of the parents is copied and inserted in the offspring chromosome; consequently, the blank spaces are filled with information from the other parent, so the genes will not be repeated along the genetic code. Fig. 3 shows an example.

Fig. 2. Flow chart of the proposed genetic algorithm.


Start
Generate initial population
Evaluate solutions
Execute heuristic of straight angle
Select two individuals
Cross
No
It is the number of solutions equal to the population?
Yes
No
Choose individual randomly
No
Mutated chromosome= population*%mutation?
Yes
Replace
Is the number of generations reached?
Yes
End

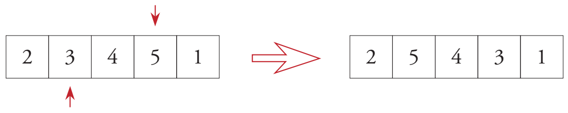
After crossover, mutation comes to have a broader search area space. Mutation consists of modifying certain genes at random. For the case of the object or piece-based representation, the proposed (2-

Fig. 3. PMX operator for GA crossover.



opt) operator was used. This operator consists of selecting two random objects and then swapping positions between them provided both objects do not have the same random number. Fig. 4 shows an example.

Fig. 4. Mutation with a 2-opt operator.



The replacement operator consists of substituting some members of the population with newly generated individuals. The most renowned replacement operator is elitism, which is based on the adaptation function. It prioritizes the individuals with the highest adaptation function to remain in future generations, thus and just as in the natural evolution process, the least feasible solutions are replaced by the best ones.

Genetic algorithm parameters

In order to select the genetic algorithm parameters, an experiment in which its values varied was carried out; for the case of the number of generations, values of 100, 150, 200 were selected; in mutation, 1%, 5% and 10% values were used; and for the population size values ranging from 10 to 50 were chosen. The response variable is the average waste percentage after five replications of the experiment. Table 1 shows the results of the experiment carried out for parameter selection.

For each parameter combination the genetic algorithm was performed five times, using the case study; then, the parameter combination was determined which presented the best results; in this regard, the best performance was achieved with a number of generations with 200 individuals, a 50 individual population size and a 5% mutation percentage.

Table 1. Experiment for parameter selection.

Mutation Rate	Number of Generations	Population Size			
		20	30	40	50
1%	100	0,2477	0,1963	0,2192	0,2201
	150	0,2955	0,2192	0,1963	0,1944
	200	0,2533	0,2432	0,2192	0,1963
5%	100	0,2432	0,2399	0,2210	0,2210
	150	0,2730	0,2201	0,2210	0,2210
	200	0,1988	0,2210	0,2210	0,1899
10%	100	0,2418	0,2477	0,2174	0,1944
	150	0,2174	0,2477	0,2192	0,2192
	200	0,2174	0,2152	0,2150	0,2150

Application of the genetic algorithm in a case study of cutting pieces out of sheets

I order to test the benefits of the genetic algorithm, a specific case related to the distribution of pieces of a product in a predefined format was set for solving. Some of these pieces are shown in Fig. 5. The sheets used in the manufacturing process are 2.4 meters wide, 1.2 meters high, and 12 mm thick; the total product consists of 20 different components. Since some pieces are repeated several times, there is a problem consisting of cutting a total of 55 pieces.

Even though most of these pieces are rectangles of several dimensions, there are also pieces known as “right angle polygons” (having more than four sides), such as the one shown in Fig. 6a.

Fig. 5. Types of pieces to be cut out of the sheet.

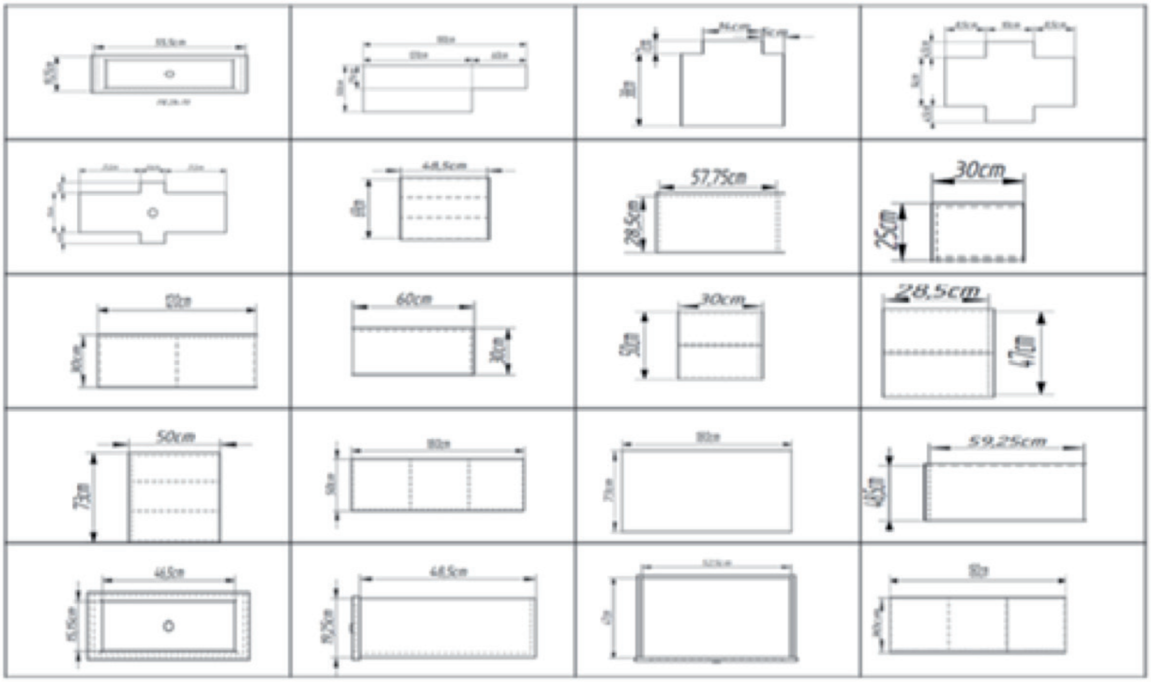
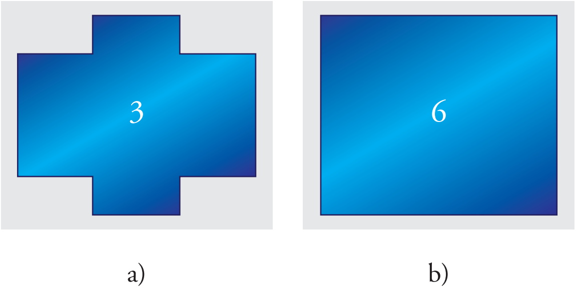


Fig. 6. Piece shapes.



This right angle polygon name is due to the fact that the angle formed by every two adjacent sides is a 90 degree angle. In case of the rectangle (Fig. 6b), the same principle applies, considering that not all right angle polygons are rectangles, which affects the heuristics construction that allows placing the pieces in the sheets.

Before solving the case study using the GA, an evaluation of the process of distributing the pieces on the sheets was carried out, as done in the company (manually); for such purpose, the average waste generated in manufacturing 10 products was carried out, with the aid of Equation (15).

$$I_d = \frac{A_{sheet} - A_{AP.sheet}}{A_{sheet}} \times 100\% \qquad (15)$$

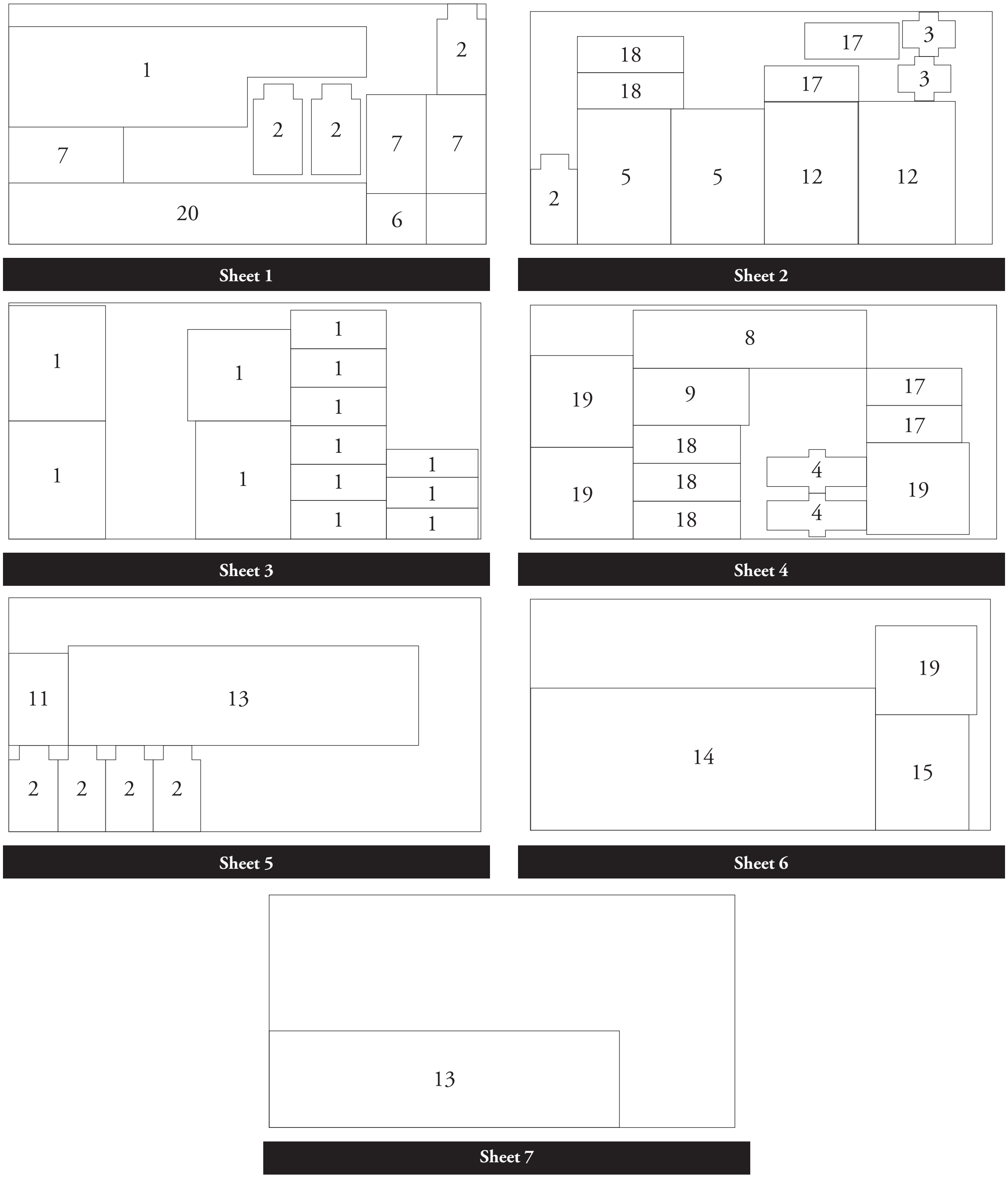
Where I_d represents the waste indicator for a product distribution, A_{sheet} is the total area of the sheets used in distributing a product, considering that the area of the last packed sheet is calculated with the maximum, and $A_{AP.sheet}$ is the total area of the pieces used in distributing a product, or the sum of the areas of all the pieces.

When performing the manual piece distribution based on experience, seven sheets are required, resulting in a total of 18.42 m^2 and 31.84% of waste, which is equivalent to 5.88 m^2 of the total sheets. Fig. 7 shows the distribution of the pieces over the total sheets for cutting, performed manually and based on experience. The average time associated to distributing the pieces on the formats was 32,5 min.

Results and Discussion

The Genetic Algorithm was implemented in JAVA and ran on a PC with WIN 7, AMD Turion

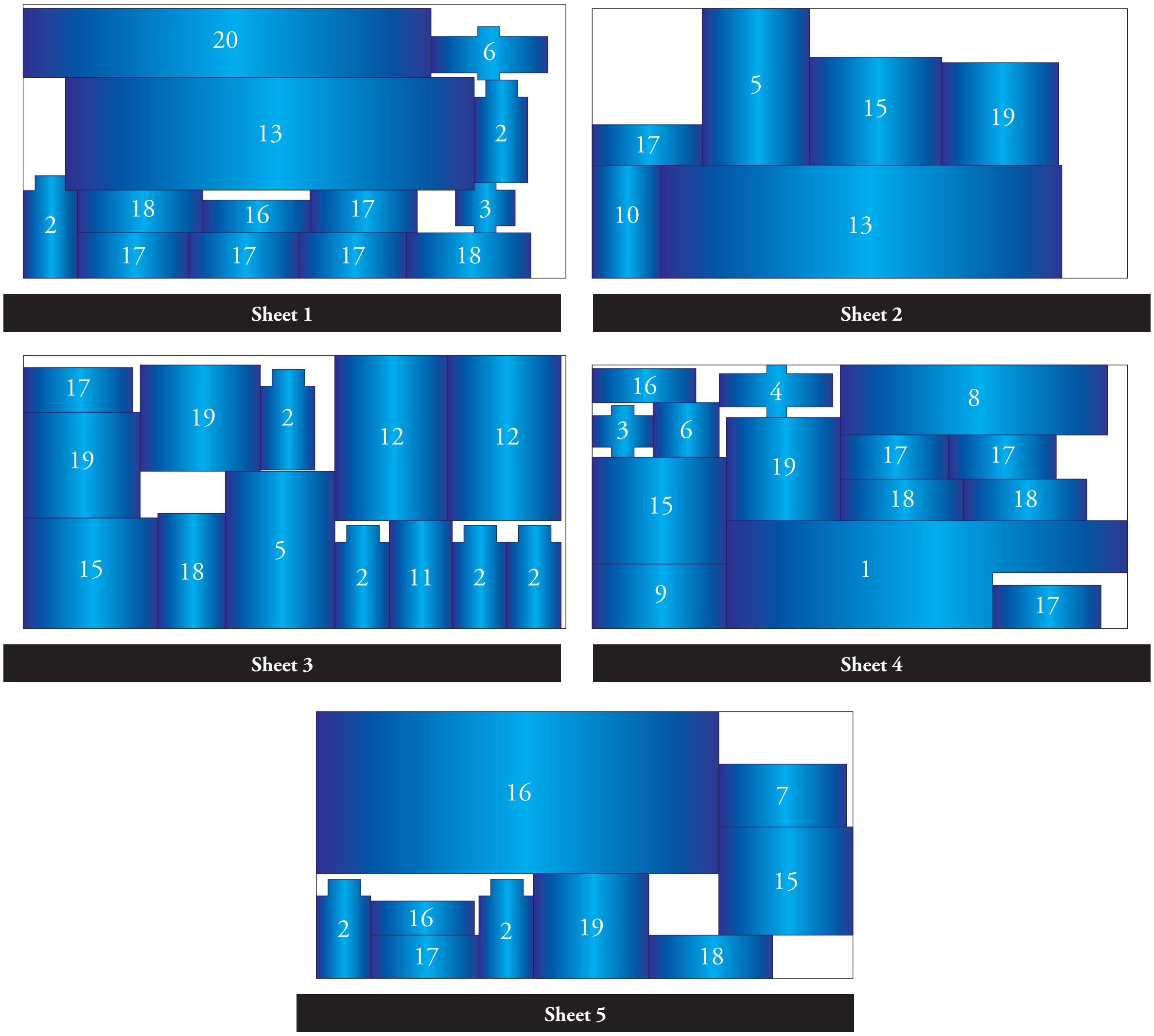
Fig. 7. Distribution of the pieces on the sheets using an experience-based design.



2.00GHz processor, 3 GB RAM and 500 GB HD. The distribution of the cuts generated by the genetic algorithm is shown in Fig. 8. This solution used only five sheets for a total of 15.42 m^2 and 21.25% of waste, which is equivalent to a 3.28 m^2 area. The average execution time was 8.8 minutes. When this numbers are compared with the average result of the manual distributions, a considerable reduction of 23.67 minutes on average can bee identified.

Considering the diagnosis results and the ones yielded by the computerized tool, Table 2 shows a comparison of the results obtained by the genetic

Fig. 8. Distribution for cutting pieces on sheets using the Genetic Algorithm.



algorithm vs the results of the experience-based manual distribution.

Table 2. Comparative results chart.

Description	Wasted per product		
	%	Área (m^2)	Cost
Genetic Algorithm	21.29%	3.286	$75,314.84
Manual Distribution	31.84%	5.88	$134,842.40
SAVINGS	10.55%	2.594	$59,527.56

The implementation of genetic algorithms yields savings of 10.55 percentile points of the total waste area, which is equivalent to 2.594 m^2, adding up to $59,527.56 Colombian Pesos saved.

Conclusions

The issue of distributing pieces on sheets considering right-angled pieces, known in literature as the 2D-BPP problem was solved by implementing a genetic algorithm. Two benefits were accomplished through this work: first, regarding waste percentage, by reducing 1.55 percentage points, which leads to cost reduction.

Also, a reduction in the administrative workload used to generate the distributions of the cuts on the sheets, which translates into an increase in productivity was observed.

For future works, including 90° rotations will be considered, in order to seek higher waste reduction indexes in sheet cuts. Also, the possibility of allowing the algorithm to cut curved pieces is suggested.

References

ALBANO, A., & SAPUPPO, G. (1980). Optimal Allocation of Two Dimensional Irregular Shapes Using Heuristic Search Methods. *Systems, Man and Cybernetics, IEEE Transactions*, 10(5), 242–248.

ÁLVAREZ, D., & TORO, E. (2009). Solution to the two-dimensional strip packing problem using a hybrid algorithm. *Scientiaet Technica*, 15(42), 205–210.

BINKLEY, K., & HAGIWARA, M. (2006). Applying self-adaptive evolutionary algorithms to two-dimensional packing problema asusing a fourcorners. *European Journal of Operational Research*, 183, 1230–1248.

JAKOBS, S. (1996). On genetic algorithms for the packing of polygons. *European Journal of Operational Research*. Retrieved from http://www.sciencedirect.com/science/article/pii/0377221794001669

LEE, L. (2008). A genetic algorithm for two-dimensional bin packing problem. *Math Digest : Research Bulletin Institute for Mathematical Research*, 2(1), 34–39. Retrieved from http://psasir.upm.edu.my/12464/

Annex 1. Model parameters and variables

Parameters

- *N*: Number of rectangles available to place on the sheet of material.
- *L*: Sheet length.
- *W*: Sheet width
- p_i: Length of rectangle *i*.
- q_i: Width of rectangle *i*.

Variables

- (x_i , y_i): Variable indicating the position of rectangle *i*, with the bottom left corner of the sheet as reference.
- s_i: Binary variable that indicates if rectangle *i* was located on the sheet.
- (l_{xi} , l_{yi}) : Binary variable that indicates the axis of the sheet of material. The p_i side of rectangle *i* is parallel to it.
- (w_{xi} , w_{yi}): Binary variable that indicates the axis of the sheet of material. The q_i side of rectangle *i* is parallel to it.
- a_{ik}: If its value is 1, indicates that the rectangle is on the left side of rectangle *k*.
- b_{ik}: If its value is 1, indicates that the rectangle is on the right side of rectangle *k*.
- c_{ik}: If its value is 1, indicates that the rectangle i is behind rectangle *k*.
- d_{ik}: If its value is 1, indicates that the rectangle i is in front of rectangle *k*.
- *M*: A very large integer.