Solving the Rectangle Packing Problem by an Iterative Hybrid Heuristic

David Beltrán-Cano, Belén Melián-Batista, and J. Marcos Moreno-Vega*

Dpto. Estadística, I.O. y Computación, University of La Laguna,
Avda. Astrofísico Francisco Sánchez s/n, 38271 S.C. de Tenerife, Spain
{jbeltran,mbmelian,jmmoreno}@ull.es

Abstract. In this paper we propose an iterative hybrid heuristic approach consisting of two phases to solve the Rectangle Packing Problem. In the first phase, a strip width value W is fixed and the corresponding Strip Packing Problem is solved using an efficient hybrid GRASP-VNS heuristic. In the second one, a new value W is determined. The above phases are repeated until the stopping condition is met. Then, the results obtained by this iterated heuristic are compared with the results given by a Simulated Annealing given in the literature. The comparative analysis corroborates the effectiveness of the proposed hybrid approach.

Keywords: Rectangle Packing Problem, Strip Packing Problem, GRASP, VNS, SA.

1 Introduction

Given a set of rectangular pieces, $\mathcal{R}$, where each rectangle $R_i \in \mathcal{R}$ has fixed width w_i and height h_i, a packing of $\mathcal{R}$ is a non-overlapping placement of the rectangular pieces in the plane. The Rectangle Packing Problem (RPP) consists in finding the packing with minimal area. This problem can be found in some areas of business and industry (design of chips, textile or leather industries ...).

Some researches have tackled the rectangular packing problem. Murata et al. [7] proposed the sequence-pair representation which encodes the *left-right* and *up-down* topological relations between rectangles on the plane in two permutations. Using this representation some approaches have been proposed. A Simulated Annealing is proposed in [7]. Imahori et al. [5] proposed three metaheuristic algorithms for solving the RPP with General Spatial Costs (random multi-start local search (MLS), iterated local search (ILS) and a random walk (WALK)). The computational results showed that ILS found better solutions than the other two algorithms for many instances. Recently, Imahori et al. [6] designed more efficient techniques to evaluate solutions and incorporated them into ILS. Then, ILS was compared with two Simulated Annealing algorithms (SA-BSG and SA-SP) that use the coding schemes by Nakatake et at. [8] and

* This research has been partially supported by the projects TIN2008-06872-C04-01 and PI 2007/019.

R. Moreno-Díaz et al. (Eds.): EUROCAST 2009, LNCS 5717, pp. 673–680, 2009.
© Springer-Verlag Berlin Heidelberg 2009

Murata et al [7], respectively. ILS performed better than these two algorithms for the RPP with General Spatial Cost. However, the SA procedure proposed in [8] provides better results for the RPP of minimizing the area, since the algorithm is specially tailored for this problem, which is the problem tackled in this paper. Drakidis at al. [2] presented a genetic algorithm with sequence-pair representation (GA-SP). In the computational experience, they compared the GA-SP with SA-SP and with another Genetic Algorithms previously proposed in the literature. The principal conclusion is that GA-SP is comparable to the rest of procedures.

With the purpose of solving the RPP, we propose an iterative hybrid heuristic approach which consists of two phases. In the first phase, a strip width value, W, is fixed and the corresponding Strip Packing Problem (SPP) is solved using the efficient hybrid heuristic GRASP+VNS described in [1]. In the second one, a new value W is determined. The Strip Packing Problem is defined as follows. Consider a strip of fixed width W and infinite height, and a finite set of rectangles, $\mathcal{R}$, where each rectangle $R_i \in \mathcal{R}$ has fixed width w_i and height h_i and at least one of their sides is smaller than W. The strip packing problem consists in packing the rectangles in the strip minimizing the height of the packing.

The rest of the paper is organized as follows. Section 2 describes the iterative hybrid approach proposed to solve de rectangle packing problem. The computational experience is reported in Section 3. Finally, the conclusions are summarized in Section 4.

2 Solution Approach: Iterative Hybrid Heuristic

In order to solve the rectangle packing problem, we propose an iterative hybrid heuristic that iteratively solves a strip packing problem. Since in the strip packing problem the strip width is fixed, the iterative heuristic fixes a strip width value W and then solves the corresponding SPP using the efficient hybrid heuristic GRASP+VNS described in [1]. Once this problem is solved, a new strip width value is selected.

The hybrid heuristic used to solve the SPP combines GRASP as constructive method and Variable Neighborhood Search (VNS) as a postprocessing step. GRASP (Greedy Randomized Adaptive Search Procedure) [3] is a constructive metaheuristic that consists of two phases; a constructive phase and a post-processing phase. In the constructive phase, a solution is iteratively constructed by selecting at random one element of a restricted candidate list. The goal of the post-processing phase is to improve this solution by running an improvement method. These steps are repeated until a stopping criterion is met. The best solution found along the search procedure is provided as the result.

The GRASP procedure uses the notion of *contour*. Once a new rectangle is added to the partial solution, it determines a new upper contour and new wasted areas. Given a partial solution, the upper contour consists of a series of horizontal segments taken from left to right. The contour C is represented by a series of values as follows:

$$C = [(y^1, x_1^1, x_2^1), (y^2, x_1^2, x_2^2), ..., (y^c, x_1^c, x_2^c)],$$

where y^i is the y-coordinate (height) of the i-segment and $[x_1^i, x_2^i]$ is the interval of x-coordinates of its points, for $i = 1, 2, ..., c, c \geq 1$. The contour C verifies $x_1^1 = 0$, $x_2^c = W$ and, for $1 \leq i < c$, $x_2^i = x_1^{i+1}$. It also verifies $y^i \neq y^{i+1}, for 1 \leq i < c$.

Let $C(t)$ be the contour at iteration t that is determined by the partial solution at iteration t. All the rectangles that do not belong to the solution are evaluated by means of their adjustment to the lowest segment of $C(t)$; i.e. the segment of the contour with minimum height. Given a parameter $\alpha \in [0, 1]$, the Restricted Candidate List (RCL), which consists of a set of rectangles, is constructed as follows.

$$RCL = \{R(w_i, h_i) \in R_2 : w_i \leq l \leq w_i + \alpha\},$$

where R_2 is the set of rectangles not in the partial solution and l is the length of the lowest segment. Only the rectangles that fit the best to the length of the lowest segment of the contour are in the restricted candidate list. If the RCL is empty, the rectangles that fit the best to the lowest segment are selected. If there are not rectangles that fit in the lowest segment, then the contour has to be rebuilt.

In order to carry out the post-processing phase of the above mentioned GRASP procedure, a Variable Neighborhood Search (VNS) [4] is considered. The k (a parameter) last rectangles of the solution are extracted and the VNS is applied over the solution space obtained by all possible permutations of these k rectangles.

The iterative procedure begins with a strip width and increases it until a maximum value using a step parameter. For each strip width, the corresponding SPP is solved by running the proposed heuristic.

3 Computational Experience

This section summarizes the computational experience carried out in this paper. The results provided by the iterative hybrid heuristic proposed in this paper to solve the RPP are compared with the best algorithm provided in the literature for this problem. All the codes were implemented in C++ and run in a Pentium processor with 2GB of memory RAM.

In order to corroborate the effectiveness of the proposed approach, we have implemented a Simulated Annealing as proposed by Nakatake, et al. [8], which, to the best of our knowledge, is the best algorithm proposed in the literature to solve the RPP. Nakatake, et al. proposed a new method of packing the rectangles based on the bounded-sliceline grid (BSG) structure. This structure has been used in this paper to implement the Simulated Annealing as proposed by these authors. According to these authors, the grid size, r, should not be close to the square root of the number of objects in order to guarantee the good performance of the method. Moreover, different grid sizes provide different results.

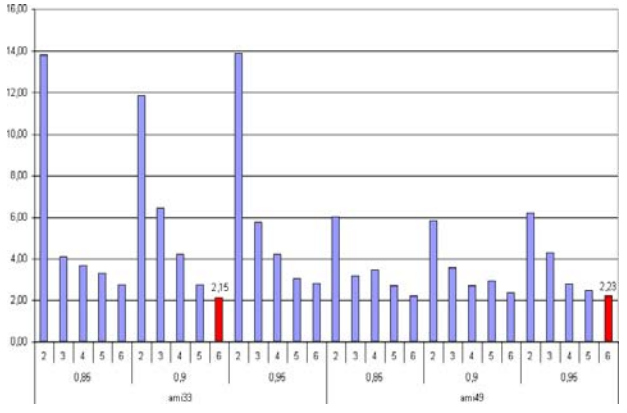

Fig. 1. Minimun Deviation for the BSG-SA algorithm: ami33, ami49

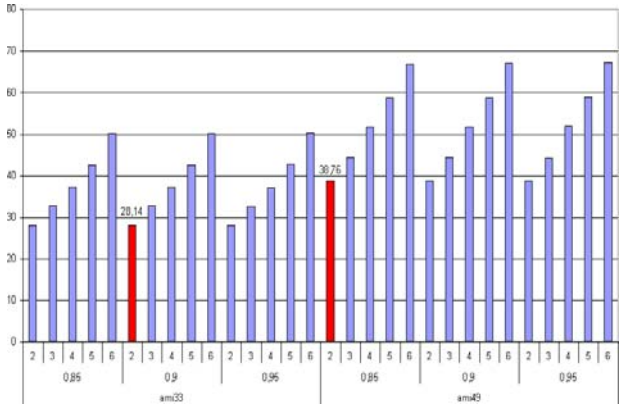

Fig. 2. Minimun CPU Time for the BSG-SA algorithm: ami33, ami49

With the purpose of evaluating the effect of the parameters α and r involved in the Simulated Annealing procedure, the problems $ami33$, $ami49$, $rp100$, $pcb146$ and $rp200$ were solved using different combinations of these two parameters. The algorithm was run 10 times for each parameters combination. Figures 1 and 2 show the minimum relative percentage deviations with respect to the lower bound, which is obtained by adding up the areas of all the rectangles to be packed, and the CPU times for the problems $ami33$ and $ami49$, respectively. Figures 3 and 4 show the same information for the problems $rp100$, $pcb146$ and $rp200$.

For the same value of α, the deviation values decrease as the grid size increases. In the small problems, the best deviation values are obtained for the grid size value $r = square\_root(number\_of\_objects) + 6$.

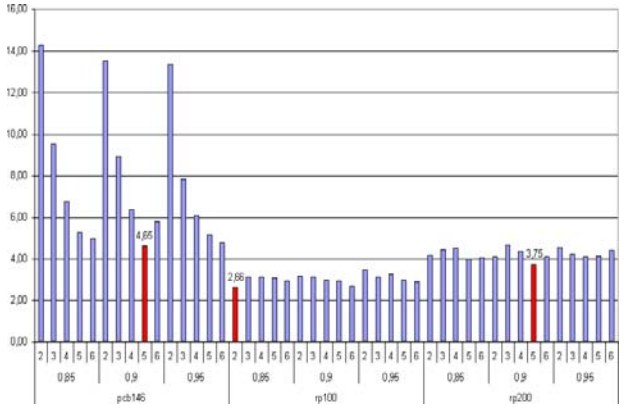

Fig. 3. Minimun Deviation for the BSG-SA algorithm: pcb146, rp100, rp200

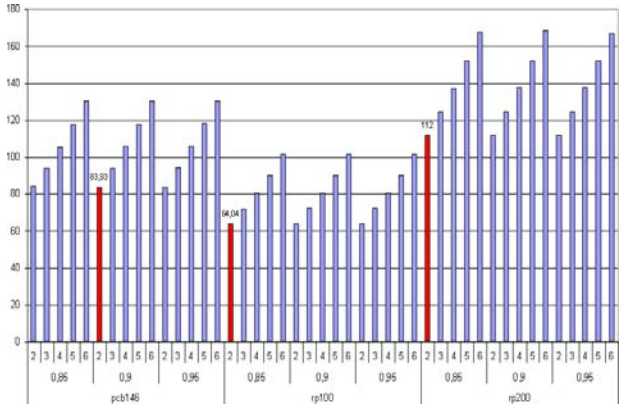

Fig. 4. Minimun CPU Time for the BSG-SA algorithm: pcb146, rp100, rp200

For the hybrid algorithm GRASP-VNS proposed in this paper, the computational experience was carried out in two phases. In the first phase we analyze the behavior that the value of the different parameters have on the efficiency and effectiveness of the proposed method. Different values for these parameters were set and the CPU time and the relative percentage deviation between the objective value and the lower bound were obtained. The small problems ($ami33$, $ami49$ and $rp100$) were used during this phase.

The parameters used to obtain these graphs are the following in reverse order of their appearance on the x-axis of the graphs: $W\_initial$, that determines the width of the lowest strip used in the iterative process (it is a percentage on the side of the lowest square that would ideally contain all the boxes that you want to pack); W_final, that determines the width of the widest strip used in the iterative process; $Step$, that represents the step value with which the width

of the strip iteratively increases; N_iter_GRASP, that indicates the number of iterations for the GRASP procedure; and N_iter_VNS, that indicates the number of iterations for the VNS procedure (if this value is equal to zero, VNS is not used as a postprocessing phase of the GRASP). These results are shown in Figures 5 and 6.

For the *ami*33 problem, the best objective values are obtained with a wider window in which the width of the strip iteratively varies ($w\_inicial = 5$, $w\_final = 5$). For this window, both the hybrid approach and the GRASP procedure achieve the best results. There is not a clear conclusion about the effect of using VNS as a postprocessing method. In some cases, increasing the number of iterations of VNS let us obtain better results, but in other cases, the

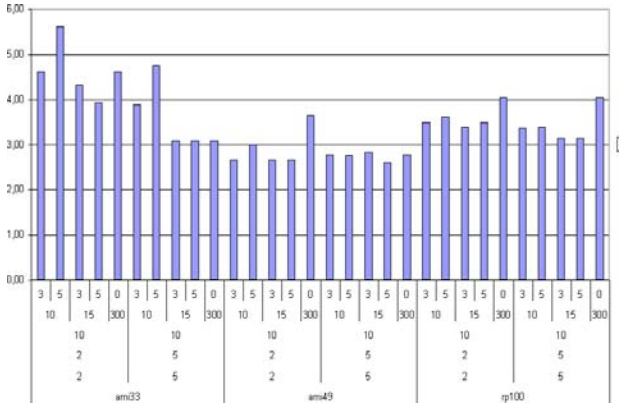

Fig. 5. Minimun Deviation for the GRASP-VNS algorithm: ami33, ami49

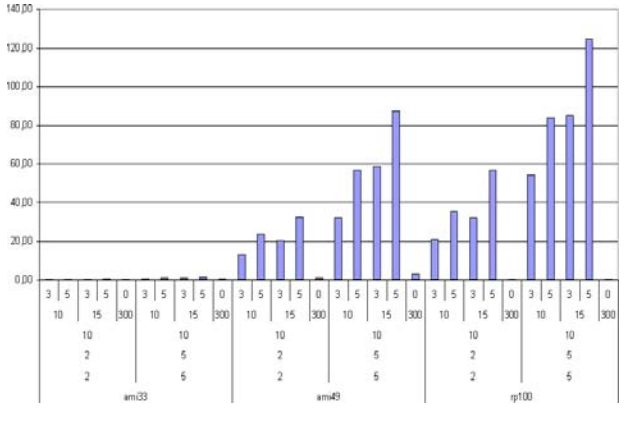

Fig. 6. Minimun CPU Time for the GRASP-VNS algorithm: ami33, ami49

Table 1. BSG-SA and GRASP-VNS best results

	BSG-SA		GRASP-VNS	
Problem	Deviation	CPU Time	Deviation	CPU Time
*ami*33	2.15	50.15	3.10	0.49
*ami*49	2.23	67.23	2.61	87.36
*pcb*146	4.79	130.69	1.88	308.59
*rp*100	2.66	64.04	3.14	85.13
*rp*200	3.75	152.12	3.01	433.64

opposite happens. There is a clear trend in the CPU time; if the number of iterations of VNS increases, the time of the hybrid procedure also increases.

For the *ami*49 problem, the best objective values are obtained with a wider window in which the width of the strip iteratively changes ($w\_inicial = 5$, $w\_final = 5$). There are not significant differences in the best deviations when the values of the parameters are changed. However, there is one combination of parameters that makes the hybrid approach GRASP + VNS perform better the the GRASP. The CPU time trends are the same as the observed in the ami33 problem. GRASP is more efficient than GRASP + VNS. If N_iter_GRASP is kept constant, the best values are obtained with the widest window.

For the *rp*100 problem, the best objective values are achieved with a wide window with $N\_iter\_GRASP = 15$ and $N\_iter\_VNS = 3$. Note that increasing the number of iterations of VNS has no effect on the quality of the solutions obtained. Therefore, it is required that the GRASP procedure uses VNS as a postprocessing method to improve the solutions, but it is not necessary to perform too many iterations of VNS to achieve good quality solutions. GRASP is the most efficient method. It runs 300 iterations in less than a second.

From the previous observations, we conclude that it is recommended to use a wide window in which to vary the width of the strip ($w\_inicial = w\_final = 5$). Despite the efficiency reached by the GRASP procedure, this method alone is not able to find the best objective function values in all instances. Therefore, VNS has to be used with a small value of iterations, as the postprocessing method in GRASP.

In the second phase of the computational experience carried out in this paper for the GRASP-VNS method, the larger problems (*pcb*146 and *rp*200) were solved following the parameters setting obtained from the above analysis. Each problem was solved 10 times.

Table 1 shows the best objective function values obtained by BSG-SA and GRASP-VNS. From the results reported in this table, we observe that the iterative hybrid method proposed in this paper performs better than the SA over the instances *pcb*146 and *rp*200, while the SA is better over the other three instances. Moreover, for some instances, the GRASP, which is most efficient than the SA, is able to reach the best results without the use of the VNS as a postprocessing step.

4 Conclusions

In this work, we propose an iterative hybrid heuristic to solve the Rectangle Packing Problem, which combines GRASP and VNS as postprocessing step. The results obtained by this method are compared with the results given by a Simulated Annealing proposed in the literature. The hybrid method performs better than the SA over two instances, while the SA is better over three instances. However, for some instances, the GRASP, which is more efficient than the SA, is able to reach the best results without the use of the VNS as a postprocessing step in some instances.

References

1. Beltrán, J.D., Calderón, J.E., Jorge, R., Moreno-Pérez, J.A., Moreno-Vega, J.M.: GRASP-VNS hybrid for the Strip Packing Problem. In: Proceedings of the First International Workshop in Hybrid Metaheuristics (HM 2004) at ECCAI, pp. 79–90 (2004)
2. Drakidis, A., Mack, R.J., Massara, R.E.: Packing-based VLSI module placement using genetic algorithm with sequence-pair representation. In: IEE Proc.-Circuits Devices Syst., vol. 153, pp. 545–551 (2006)
3. Feo, T.A., Resende, M.G.C.: Greedy randomized adaptive search procedures. Journal of Global Optimization 6, 109–133 (1995)
4. Hansen, P., Mladenovic, N.: Variable neighborhood search: principles and applications. European Journal of Operations Research 130, 449–467 (2001)
5. Imahori, S., Yagiura, M., Ibaraki, T.: Local search algorithms for the rectangle packing problem with general spatial costs. Mathematical Programming 97, 543–569 (2003)
6. Imahori, S., Yagiura, M., Ibaraki, T.: Improved local search algorithms for the rectangle packing problem with general spatial costs. European Journal of Operational Research 167, 48–67 (2005)
7. Murata, H., Fujiyoshi, K., Nakatake, S., Kajitani, Y.: VLSI module placement based on rectangle packing by the sequence pair. IEEE Transactions on Camputer Aided Design 15, 1518–1524 (1996)
8. Nakatake, S., Fujiyoshi, K., Murata, H., Kajitani, Y.: Module placement on BSG-structure and IC layout applications. In: Proc. Intl. Conf. Comput. Adided Des., pp. 484–491 (1996)