Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 183 (2017) 291 – 296

Procedia Engineering

www.elsevier.com/locate/procedia

17th International Conference on Sheet Metal, SHEMET17

Meta-heuristic algorithms for nesting problem of rectangular pieces

Ernesto Lo Valvo*

Dipartimento di Architettura, Università di Palermo, Viale delle Scienze, 90128 Palermo, Italy

Abstract

Nesting problems consist of placing multiple items onto larger shapes finding a good arrangement. The goal of the nesting process is to minimize the waste of material. It is common to assume, as in the present work, that the stock sheet has fixed width and infinite height, since in the real world a company may have to cut pieces from a roll of material.
The complexity of such problems is often faced with a two-stage approach, so-called "hybrid algorithm", combining a placement routine and a meta-heuristic algorithm. Starting from a given positioning sequence, the placement routine generates a non-overlapping configuration. The encoded solution is manipulated and modified by the meta-heuristic algorithm to generate a new sequence that brings to a better value of the objective function (in this case the height of the strip).
The proposed method consists in placing the rectangles inside a strip and in combining the meta-heuristic algorithms with the No Fit Polygon algorithm. The software has been developed in Python language using proper libraries to solve the meta-heuristic techniques (*Inspyred*) and the geometric problems (*Polygon*).
The results show the effectiveness of the proposed method; moreover, with regard to problems reported in literature employed as benchmark of the nesting algorithms, the degree of occupation values (Efficiency Ratio, ER) are shown to be higher than 90%.

Keywords: Sheet metal optimisation; meta-heuristic algorithm; No Fit Polygon algorithm.

1. Introduction

In the field of nesting problems [1-2], a particular interest is focused on the frequent two-dimensional rectangular strip packing problem (2D-SPP), in which a given set of rectangular pieces have to be packed into a strip of given

* Corresponding author. Tel.: +39-091-23861845.
E-mail address: ernesto.lovalvo@unipa.it

doi:10.1016/j.proeng.2017.04.041

width and infinite length in order to minimize the required height of the packing [3]. The 2D-SPP occurs, e.g., in the cutting of rolls of paper or metal. Since the problem has NP-hard complexity, over the years many solutions have been proposed. In the literature meta-heuristics techniques have been considered more suitable for solving the 2D-SPP. These are mainly genetic algorithms (GA), but simulated annealing (SA) and other types of meta-heuristics algorithms have been also applied. Hopper et al. [4] and, recently Olivera et al. [5] and Delorme et al. [6] provide a large overview of the meta-heuristics techniques that have been developed for the different variants of the 2D-SPP.

In this work, it is propose to compare the use of different meta-heuristic methods for a 2D-SPP problem solution by the improvement of a suitable heuristic positioning technique (said Bottom Left Fill) combined with the concept of the No-Fit-Polygon previously developed [7].

The analysis was carried out by developing a suitable program in Python that, by using two libraries (*Polygon* [8] and *Inspyred* [9]), it is able to solve both the problems related to positioning and those related to the identification of the best solution for different types of meta-heuristic methods.

2. The employed methods

The nesting problem can be split into two main steps: the search of an optimal sequence of the shapes (rectangles) which have to be positioned and their allocation on the stock sheet. The placement technique is based on the intensive use of a nesting method which employ a particular polygon, called No-Fit-Polygon, shortly described in the next paragraph.

2.1. The Bottom Left Fill Algorithm

One of the placement routine often used in nesting problems is the Bottom Left Fill algorithm (BLF). This algorithm is a adaptation of the Bottom Left one proposed by Jakobs [10] that consists in placing sequentially the pieces in the position as far as possible to the bottom-left place in the examined strip, without overlapping them with those previously positioned. The routine is modified with the so-called technique Bottom Left Fill (BLF) in which, using the NFP computation, the piece can be positioned even in the existing empty spaces, in order to determine better packing, even if this increases computation complexity.

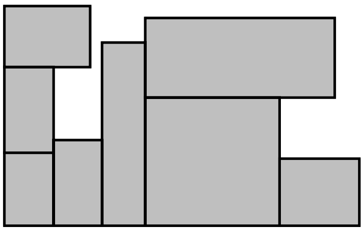

Fig. 1. An example of Bottom Left Fill positioning.

2.2. The No Fit Polygon Algorithm

A great number of nesting of two-dimensional irregular shapes use a particular polygon called No-Fit-Polygon (NFP) [11]. The NFP is described in the following terms (Fig. 1):

"*Given two polygons A and B such that the position of A and the orientation of B are all fixed, then the NFP of B relative to A (denoted with NFP_{AB}) completely describes all those positions where a reference point of B polygon (say orbiting polygon) can be placed in order to have B touching A polygon (say stationary polygon) without overlapping*".

The main advantage of the NFP is that it can be employed both for regular shapes and non-regular ones. The NFP is obtained by fixing the position of a reference point of orbiting polygon that moves around the perimeter of stationary polygon (fig. 2a). A relevant feature of the NFP is that the points located inside it are associated to an overlapping among the two polygons. For the points of its contour the two polygons touch each other whereas for the external points there is no contact between them. It is possible to noticed that an isle inside the NFP highlights the circumstance to insert the orbiting polygon within the stationary one.

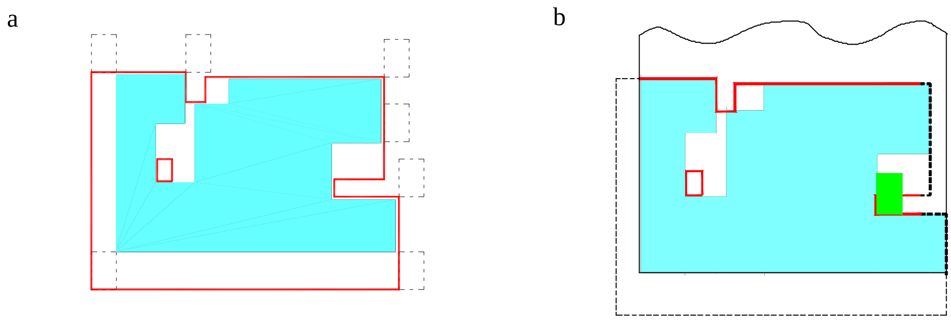

Fig. 2. (a) The NFP described from orbiting polygon during its moving around the stationary one, (b) the step 3 of proposed algorithm.

When the pieces are convex, it is very easy to calculate the No-Fit-Polygon: this computation is described in several papers (e.g. [12, 13]). In this case the NFP can be obtained from Minkowski difference from A and B. We also know that the NFP of two convex polygons is a new polygon, convex itself, whose edges are the ones of two starting polygons sorted with respect to the slope versus the horizontal line. In the case of two rectangles (A and B), the NFP_{AB} will immediately be calculated because it is a rectangle whose base is the sum of the bases and the height of the sum of the heights of the two starting rectangles. Furthermore, it can be demonstrated [12-14] that, denoting with A the Boolean-union polygon of Ka sub-polygons, $A = \bigcup_{i=1}^{Ka} A_i$, the No-Fit-Polygon NFP_{AB} can be computed with $NFP_{AB} = \bigcup_{i=1}^{Ka} NFP_{A_i B}$ in which $NFP_{A_i B}$ is the No-Fit-Polygon of polygon A_i (stationary) with respect to polygon B (orbiting). Of course, in this case, in which all the polygons are rectangles, the computation is very simple and fast.

2.3. The proposed placement routine

As already mentioned, in this work the employed placement routine makes use of a Bottom Left Fill algorithm combined with the NFP algorithm. The positioning process follows the steps below reported:

1. Starting from a given sequence of placement, the first rectangle is placed in the lower left corner of the strip and has the role of stationary polygon.
2. The following piece in the sequence is used as orbiting polygon (dashed rectangle in fig. 2a) for the generation of the NFP, using the procedure described in paragraph 2.2.
3. The point locus of possible positioning is obtained from the intersection between the obtained polygon and the rectangle of infinite height that represents the strip, taking into account the width of piece which has be positioned (the red line in fig. 2b).
4. The obtained total perimeter is explored looking for the point with smaller y coordinate; in case of equal values of y coordinate the one with smaller x coordinate is chosen.
5. The considered rectangle is positioned in the found point (fig. 2b) and the union of the previous stationary polygon with this last rectangle is defined as new stationary polygon.
6. If further pieces in the sequence are present, the procedure is iterated from step 2.
7. The H height of the necessary strip is obtained seeking in the obtained union-polygon the value of the maximum y coordinate of the points that compose it.

3. Implementation of the nesting procedure

The nesting procedure has been developed in Python language (ver. 3.5.2) using proper libraries to solve the geometric (*Polygon*) and the meta-heuristic (*Inspyred*) problems. The parameters for the evaluation of the tests have been defined in terms of degree of occupation (Efficiency Ratio, ER) and time necessary to obtain the same one.

3.1. The Polygon library

Polygon is a Python library that simplifies the management of 2D polygons [8]. With *Polygon* it is possible manage complex polygonal areas in Python in a very intuitive way. The polygons are simple Python objects and the Boolean operations are bound to the standard operators such as +, -, |, & and ^. This package includes a large number of functions, such as that one to calculate the area, the number of points of a contour, the bounding box of a polygon, and much more.

3.2. The Inspyred library

The *Inspyred* library [9] is an Open Source framework for evolutionary computations using the Python language, based on Ken de Jong's book [15]. Evolutionary computation is a sub-field of artificial intelligence that involves combinatorial optimization problems. The philosophy that underlies it is the consideration that the approach to practical problems for which there are no direct solutions can be reformulated in terms of optimization of an objective function that represents the quality of the solution. Essentially, evolutionary computation tries to mimic the biological process of evolution of a population to solve a given problem. The goal of *Inspyred* library is to separate problem-specific computation from algorithm-specific computation, or evolutionary operators, in order to make algorithms as general as possible across a range of different problems.

The *Inspyred* library allows to use several evolutionary algorithms bio-inspired easily; in this work, has been decided to make use of this framework by comparing the following algorithms:

1. Evolutionary Computation (EC)
2. Genetic Algorithm (GA)
3. Evolution Strategy (ES)
4. Simulated Annealing (SA)
5. Estimation of Distribution Algorithm (EDA)
6. Differential Evolution Algorithm (DEA)
7. Particle Swarm Optimization (PSO)

4. Test problems

In order to compare the performances of the developed method, some problems proposed from Hopper and Turton have been taken into consideration [16]. They proposed different categories of problems (for a total of 98 problems) that differ for the dimensions of the slabs, the number and the dimensions of pieces to locate (from 16 to 197 pieces). Moreover, to be able to know the theoretical minimal length of the slab, they consider a sheet of given dimensions which is subdivided in rectangles of different size completely: finally, the so-obtained set of pieces is employed to verify the validity of their methods.

For every slab the width is given, while the height must be minimized. So, in this work, the parameters for the evaluation of the tests have been defined in terms of degree of occupation (Efficiency Ratio, ER, expressed as the ratio between the sum of the areas of the already placed pieces and the area of the strip altogether used), and time necessary to obtain the same one, taking into account that a personal a computer with Intel I5-2500 CPU at 3.3 GHz has been used. Obviously ER is equal to 1 when the solution is equal to the theoretical one.

In particular the problems denoted by the following abbreviations [17] have been analyzed: C1/P1 (16 pieces), C2/P2 (25 pieces), C3/P1 (28 pieces), C4/P3 (49 pieces), C5/P2 (73 pieces), C6/P1 (97 pieces) and C7/P3 (196 pieces). In order to compare the results with those from the literature, they have been used eight conditions. The first generates 2000 random permutations (RND), the other seven using each of the previously mentioned techniques (EC, GA, ES, SA, EDA, DEA, PSO) always with a maximum value of 2000 iterations.

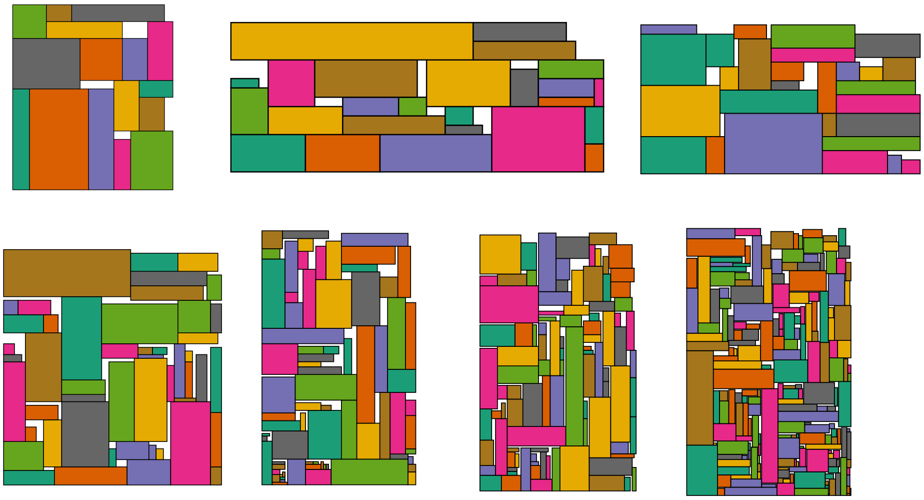

Fig. 3. Best layout found for the different analyzed configurations.

5. Result analysis

The analysis of the simulation results of the seven examined problems has shown good efficiency; in fact, for all methods used, the found value of ER has been often greater than 0.9. Moreover it has been possible to observe (Table 1) that the positioning technique used is very effective, since the solutions obtained by a random choice of the positioning sequence never fall below 0.88.

Table 1. Comparison between Efficiency Ratios.

Problem	Pieces	RND	EC	GA	ES	SA	EDA	DEA	PSO	Best
C1/P1	16	0.909	0.877	0.952	0.952	0.909	0.952	0.957	0.909	0.957
C2/P2	25	0.882	0.882	0.938	0.938	0.882	0.938	0.938	0.938	0.938
C3/P1	28	0.909	0.882	0.909	0.938	0.909	0.938	0.968	0.938	0.968
C4/P3	49	0.870	0.882	0.923	0.938	0.896	0.909	0.952	0.923	0.952
C5/P2	73	0.857	0.882	0.923	0.918	0.891	0.900	0.938	0.928	0.938
C6/P1	97	0.882	0.882	0.916	0.909	0.882	0.902	0.916	0.902	0.916
C7/P3	196	0.882	0.882	0.916	0.916	0.909	0.909	0.923	0.909	0.923

About the comparison between the different heuristic methods used, which typically require a very similar calculation time and, while being quite similar values among them, some methods (in particular always DEA) reach the best results. This result is in agreement with the literature; see Burke and Kendall [18] where they compare three different meta-heuristic algorithms. Considering the work of Leung and al. [19], the ER values are better, probably because in this work a more effective positioning algorithm has been employed. Actually increasing the number of pieces, the times of computation grow in considerable way (Table 2), although the positioning time prevails on the computation time, so that the time required random arrangement are similar to those that use meta-heuristics techniques. As far as computation time is regarded, it grows exponentially with the number of pieces while remaining acceptable also for problems of considerable magnitude, thus allowing to increase the number of iterations to try to get some further improvement.

Table 2. Computation time (in seconds).

Problem	Pieces	RND	EC	GA	ES	SA	EDA	DEA	PSO
C1/P1	16	4.1	4.5	4.2	4.3	4.7	4.3	4.6	4.3
C2/P2	25	10.4	25.2	26.5	25.6	25.2	26.1	26.4	25.6
C3/P1	28	13.6	32.0	33.1	33.3	32.2	33.7	33.5	32.3
C4/P3	49	43.6	104.4	107.2	106.3	103.2	107.2	106.2	108.1
C5/P2	73	115.3	258.5	267.9	262.4	250.7	261.6	241.3	332.0
C6/P1	97	186.4	180.0	178.5	180.5	178.3	176.0	183.7	183.1
C7/P3	196	930.2	934.7	932.3	942.6	971.6	990.1	1040.4	1058.1

6. Conclusions

In this paper the problem of the optimal pieces placement of rectangular shape on strip of given width has been deal. The developed algorithm uses a hybrid system based on a positioning method of type Bottom Left Fill, based on the employ of a nesting system that use the concept of "No Fit Polygon", coupled with a meta-heuristic optimization method. Many different evaluation functions have been tested using a proper library and the performance and efficiency of these meta-heuristic algorithms have been compared.

Computational results show the great effectiveness of the proposed method, actually it has shown values of the efficiency ratio greater than 90% with regard to a set of problems reported in literature, employed as benchmark of the nesting algorithms; they show that the DEA (Differential Evolution Algorithm) gives better results.

Furthermore, computation times are very low even in case of large problems. The research work, considered the capability of the developed method, will be extended to the problem of the positioning of non-regular shapes.

References

[1] H. Dyckoff, A typology of cutting and packing problem, European Journal of Operations Research 44 (1990) 145-159.

[2] G. Wascher, H. Hauβner, H. Schumann, An improved typology of cutting and packing problem, European Journal of Operational Research (2007) 1109-1130.

[3] A. Bortfeldt, A genetic algorithm for the two-dimensional strip packing problem with rectangular pieces, European Journal of Operational Research 172(3) (2006) 814-837.

[4] E. Hopper, B.C.H. Turton, A review of the application of meta-heuristic algorithms to 2D strip packing problems, Artificial Intelligence Review,16.4 (2001) 257-300.

[5] J.F. Oliveira, A. Neuenfeldt Júnior, E. Silva, M.A. Carravilla, A survey on heuristics for the two-dimensional rectangular strip packing problem, Pesquisa Operacional 36(2) (2016) 197-226.

[6] M. Delorme, M. Iori, S. Martello, Bin packing and cutting stock problems: Mathematical models and exact algorithms, European Journal of Operational Research 255-1 (2016) 1-20.

[7] E. Lo Valvo, C. Borsellino, V. F. Ruisi, The nesting problem of rectangular pieces: A new approach using meta-heuristic algorithms with No Fit Polygon, 3th CIRP Int. Conf. on Intelligent Computation in Manufacturing Engineering (2002) 163-167.

[8] https://www.j-raedler.de/projects/polygon/ [accessed 2016 December 1]

[9] http://pythonhosted.org/inspyred [accessed 2016 December 1]

[10] S. Jakobs, On genetic algorithms for the packing of polygons, European Journal of Operational Research 88(1) (1996) 165-181.

[11] M. Adamowicz, A. Albano, Nesting two-dimensional shapes in rectangular modules, Computer-Aided Design 8(1) (1976) 27-33.

[12] E. Lo Valvo, R. Licari, A More Efficient Method for Clustering Sheet Metal Shapes, Key Engineering Materials 344 (2007) 921-927.

[13] R. Licari, E. Lo Valvo, Optimal positioning of irregular shapes in stamping die strip, The International Journal of Advanced Manufacturing Technology 52(5) (2011) 497-505.

[14] T. Lozano-Perez, Spatial Planning: A configuration space approach, IEEE Transactions on Computers c-327 (1983) 108-120.

[15] K.A. De Jong, Evolutionary Computation: A Unified Approach, MIT Press, 2006.

[16] E. Hopper, B.C.H. Turton, An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. European Journal of Operational Research 128 (1) (2001) 34-57.

[17] http://dip.sun.ac.za/~vuuren/repositories/levelpaper/HopperAndTurtonData[1].htm [accessed 2016 December 1]

[18] E. Burke, G. Kendall, Applying Simulated Annealing and No Fit Polygon to the Nesting Problem, Proceedings of WMC '99: World Manufacturing Congress (1999) 70-76.

[19] T.W. Leung, C.H. Yung, M. D. Troutt, Applications of genetic search and simulated annealing to the two-dimensional non-guillotine cutting stock problem, Computers & Industrial Engineering 40(3) (2001) 201-214.