Universidade do Minho
Escola de Engenharia

Semana da Escola de Engenharia
October 24 - 27, 2011

ALGORITHMS FOR INDUSTRIAL PROCESS OPTIMIZATION: An application in the automotive industry

Pedro Brás, Cláudio Alves, José Valério de Carvalho
Department of Production and Systems Engineering, University of Minho, Braga, Portugal
E-mail: {pedro.bras, claudio, vc}@dps.uminho.pt

KEYWORDS
Process optimization, constructive heuristics, meta-heuristics, leather nesting, automotive industry.

ABSTRACT

In this paper, we address a real application of a hard optimization problem related to the production of car seats in the automotive industry. In the literature, this problem is referred to as the leather nesting problem. It consists in finding the most efficient layout for a set of small irregular shapes within a large natural leather hide with holes and different quality grades. The highly complex combinatorial and geometrical aspects of this problem, associated to the high quality constraints of this industry, turns it into a very challenging problem.

Three different approaches are described. We present a set of constructive heuristics, and we show that the solutions provided by these heuristics can be improved using two different families of meta-heuristics.

INTRODUCTION

Industry is a rich source of applications for combinatorial optimization. Among the many processes that might be optimized in the industrial context, those involving the cut of a given raw material assumes a prominent position. Improving the efficiency of cutting operations is of extreme importance for the competitiveness of the companies, especially when the raw materials are so expensive as the leather hides.

Cutting and packing problems have been recently categorized by Wäscher et al. (Wäscher et al. 2007). In their work, the authors provide a new and exhaustive classification scheme for these problems. They classified the leather nesting problem that is addressed in this paper as a two-dimensional residual cutting stock problem. Despite the vitality of the research related to these problems (which is clearly reflected on the number of publications indexed by Wäscher et al.), very few attempts to solve the LNP efficiently have been reported in the literature. In fact, Heistermann et al. (Heistermann et al. 1995) were the only authors to address the same exact problem as the one described in this paper. The approach reported by these authors consists in an iterative procedure that decomposes into several key steps: the selection of a focus area where to place a shape, the selection of the next shape to be placed, the evaluation of the placement positions and the local improvement of the shape positioning. Based on several criteria, at each iteration of their algorithm, a placement area is selected from the border of the hide or along the actual layout. The selection of the next shape to be placed is done by comparing the geometries of this area with the available shapes. The positioning is evaluated by measuring the waste area generated after placing the shape. The approach proposed by Heistermann et al. (Heistermann et al. 1995) handles the existence of irregular shapes and different quality grades within the hides and shapes.

A different approach based on genetic algorithms was described by Crispin et al. (Crispin et al. 2005) for a somewhat different LNP where directionality constraints apply. These constraints apply especially to the shoe making industry. It imposes a direction for the shapes placed in given regions of the hide. The authors considered two alternative strategies to solve this particular problem. The first strategy focuses in maximizing the adjustment level between the shapes using a fitness function that measures the intersection between the offset of the shape and the actual layout. The second strategy favors the placement positions that connect the shapes in the layout. The latter relies on a fitness function that benefits the placement of the shapes that maximizes the number of contact points in the actual layout.

A more recent approach that deals with a simpler version of the general LNP was proposed by Zhang and Yang (Zhang and Yang 2009). These authors did not consider the existence of different quality grades on the hides and shapes, although they did take into account the existence of defective areas on the leather hides. The authors proposed a method that combines a genetic algorithm with a simulating annealing procedure. The first determines the sequence of the shapes, while the second controls the mutation parameter used by the genetic algorithm. The sequence of the shapes that is determined in this way are placed using a bottom-left placement strategy.

In (Alves et al. 2011a; 2011b; Brás 2011), Alves et al. and Brás proposed different approaches to solve the general LNP using a set of constructive heuristics, and two different meta-heuristics to improve the solutions generated by the former. These works are revised in this paper. In the sequel, we describe in detail the elements that characterize the general leather nesting problem, and we discuss some issues related to geometry. The three different approaches proposed to solve this problem are described in dedicated sections. At the end of the paper, we discuss the performance of these algorithms by analyzing the results of extensive computational experiments.

THE LEATHER NESTING PROBLEM

The leather nesting problem is a cutting stock problem whose objective is to determine a feasible arrangement (a layout) for a set of small shapes (the pieces) on a larger surface (the leather hide), such that the total unused space (waste) is minimized. Because the leather hides are a natural product, their sizes can vary substantially and their contour may be highly irregular. Moreover, the interior of a leather hide has typically defects, holes and different levels of quality (quality zones). The first two are considered to be obstacles to the pieces placement, since a piece cannot be placed in a way that intersects a defective area or a hole. On another hand, the quality zones can be used for placing the pieces. In the specific application considered in this paper, the pieces and the hides can contain up to four different quality levels, designated by the letters A, B, C and D. The best leather quality corresponds to the quality zones A, and the worst to the quality zone D. A placement is considered to be valid if every quality zone of a piece does not overlap with any region of the hide with a quality level worse than the quality level of the piece. The quality levels define the minimum quality requirement for a given region of a piece.

The shapes that have to be placed on the leather hide are the pieces that form a car seat cover. Usually, these are highly irregular shapes with different quality zones and holes in their interior. The pieces have a high variation in size and in design configuration.

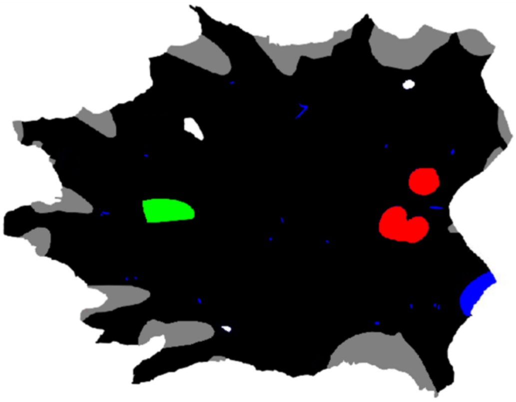

Figure 1: Leather hide

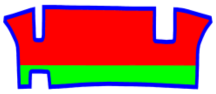

Figure 2: Piece of a car seat with quality zones

GEOMETRY

Shapes representation

All the shapes used in this work (pieces, hides, defects, holes and quality zones) are represented using polygons. The pieces are created using a CAD software, while the representation of the hides is achieved through a scanning system. This system recognizes the contour of the hide and its holes, while the quality zones are determined by specialized human operators. The number of vertices of the resulting polygons ranges from 150 to 300 for the pieces and from 500 to 1000 for the leather hides.

Since the number of vertices has a significant impact on the performance of the algorithms, we applied an additional procedure to simplify the representation of the hides and pieces. This procedure consists in removing vertices from the contour of the hides and pieces, and from their quality zones and defects. This operation was done such that any layout of these simplified shapes remain feasible when these shapes are replaced by the original ones. Figure 3 illustrates the result of this procedure applied to the contour of a piece.

a. b.

Figure 3: Representation of a piece before (a.) and after (b.) the vertex removal procedure

No-Fit-Polygon

A key issue in any nesting (and generally cutting) problem is to ensure that any two pieces never overlap in the final layout. To deal with this issue, we used the No-Fit-Polygon (NFP) that we computed using Minkowski sums. Given two polygons A and B, and given a reference point of B, the NFP between A and B ($NFP_{A,B}$) is a polygon originated by the locus of the reference point of B, when B slides around A, and such that B always touches A.

The NFP is used to distinguish a feasible from an infeasible placement. Given $NFP_{A,B}$, if the reference point of B is positioned inside the NFP, then B overlaps A, however, if it is positioned outside the NFP, then B do not even intersect A; finally, if it rests on the boundary of the NFP, then B merely touches A.

Using the same principle of the NFP, the Inner-Fit-Polygon (IFP) of two polygons A and B ($IFP_{A,B}$) corresponds to the path obtained by a reference point of polygon B, when it slides along the internal side border of polygon A. This IFP is used to ensure that one piece is placed completely inside a leather hide.

CONSTRUCTIVE ALGORITHMS

Overview

Several constructive algorithms were developed based on complementary strategies. These algorithms are described in detail in (Alves et al. 2011a). The strategies can be grouped into the following classes:

- Strategies for grouping pieces (GRP);
- Strategies for selecting the next piece to place (SEL);
- Strategies for selecting a feasible placement region inside the hide (PLAC);
- Strategies to evaluate a given placement position (EVAL).

Our constructive algorithm is an iterative procedure that follows the steps given by the previous classes. Given an instance of the general LNP, the algorithm starts by sorting and grouping the pieces according to a given attribute. All the pieces and groups are ordered according to a selected attribute. The objective of this first step is to ensure that the same treatment is applied in the further steps of the algorithm to pieces that share, approximately, the same value of a given attribute. The selection of the next piece to be placed relies on the groups previously defined or instead, on the properties of a particular region of the hide. The latter implies the simultaneous selection of the piece to be placed and the corresponding placement region. The third step of the algorithm is the selection of a feasible placement region inside the hide, which is done by considering the entire hide, or by focusing on a specific region. The final step consists in evaluating several placement positions in the selected region. This is done by applying a set of criteria with the objective of minimizing the waste of the resulting layout.

Description of the strategies

Strategies for grouping the pieces (GRP)

The consequence of grouping the pieces based on the value of a given attribute is that any piece that are more or less identical in view of this attribute will have the same priority in the next steps of the algorithm. The attributes that were considered are the following:

(G1) Area;
(G2) Degree of irregularity;
(G3) Degree of concavity;
(G4) Ratio between the length and the width of the enclosing rectangle;
(G5) Value of the piece based on the quality zones and the areas of these zones;
(G6) Homogeneity of the quality zones.

These attributes rely on the geometrical characteristics of the pieces ((G1) to (G4)), and on the characteristics of their quality zones ((G5) and (G6)).

Strategies for selecting the next piece to place (SEL)

In the second step of the algorithm, the next piece to be placed on the leather hide is selected. The strategies on which this selection relies are based explicitly on the groups defined previously. They are defined as follows:

- (S1.I/D) Selection of a piece from the complete set of pieces in increasing (S1.I) or decreasing (S1.D) order of the value of the selected attribute;

- (S2.I/D) Selection of a group of pieces in increasing (respectively decreasing) order of the indexes of the groups, and selection of a piece from the selected group based on:
  - (S2.I/D.1) The characteristics of the IFPs:
    - (S2.I/D.1.a) Selection of the piece with the smallest IFP;
    - (S2.I/D.1.b) Selection of the piece with the largest IFP;
    - (S2.I/D.1.c) Selection of the piece with the largest or smallest IFP depending on the selected group of pieces.
  - (S2.I/D.2) The value provided by the function used to evaluate the placement positions.

For the subset of strategies (S2.I/D.1), the piece and the feasible placement region are selected simultaneously, while for (S2.I/D.2), the piece and its final position are selected at the same time.

Strategies for selecting a feasible placement region inside the hide (PLAC)

By limiting the placement of a piece to a given region of the hide, it is possible to decrease the computing time needed to determine the final position of the piece. For this purpose, we limited the search for a feasible placement position to the following regions:

- (P1) All the empty spaces on the hide;
- (P2) Vertical strips starting from the one with the largest x-coordinate (from the right to the left of the hide);
- (P3) Vertical strips starting from the one with the smallest x-coordinate (from the left to the right of the hide);
- (P4) The smallest IFP of the piece;
- (P5) The largest IFP of the piece;
- (P6) The largest or smallest IFP depending on the group of the selected piece;
- (P7) The smallest empty space on the hide;
- (P8) The largest empty space on the hide;
- (P9) The empty space with the lowest quality;
- (P10) The empty space with the highest quality;
- (P11) The empty space with the less irregular contour;
- (P12) The empty space with the most irregular contour.

The strategies (P2) and (P3) are inspired on the experience of human operators. In real settings, these operators use specific steel made cutting tools and they arrange them so as to fill the hides by levels, starting from their bottom side. The strategies (P4) to (P6) restrict the placement region to the IFP of a given piece relative to the hide. The placement of a piece on an empty space on the hide, considering some attributes is done using the strategies (P7) to (P12).

Strategies to evaluate a given placement position (EVAL)

The step that finalizes an iteration of the constructive algorithm is about finding the final position for the selected piece within the selected feasible placement region. This position is selected according to an evaluation function that are based in one of the following criteria:

- (E1) $\sum_{i=1}^{|I|} area(I_i)$: total intersection area between O, the current layout and the region outside the hide;
- (E2) $\sum_{i=1}^{|I|} area(I_i)/\big(area(O)-area(P)\big)$: relative intersection area between O, the current layout and the region outside the hide;
- (E3) $max_{i=1,\ldots,|I|} area(I_i)$: largest intersection area among the polygons resulting from the intersection between O, the current layout and the region outside the hide;
- (E4) $max_{i=1,\ldots,|I|} area(I_i)/\big(area(O)-area(P)\big)$: largest relative area among the polygons resulting from the intersection between O, the current layout and the region outside the hide;
- (E5) total intersection area between O, the current layout and the region outside the hide plus the total area of the quality zones of P placed in a similar quality zone of the hide;

(E6) relative intersection area between O, the current layout and the region outside the hide plus the relative area of the quality zones of P placed in a similar quality zone of the hide;
(E7) total intersection area between O, the current layout and the region outside the hide plus the total area of the quality zones of P placed in zones of the hide with a quality B, C or D;
(E8) relative intersection area between O, the current layout and the region outside the hide plus the relative area of the quality zones of P placed in zones of the hide with a quality B, C or D;
(E9) number of empty spaces generated when placing P;
(E10) total area of waste generated when placing P;
(E11) distance to the border of the hide;
(E12) distance to the center of the hide;
(E13) distance to the region of lowest quality of the hide;
(E14) distance to the region of highest quality of the hide.

Some of these functions ((E1) to (E8)) relies on the intersection area I between the offset O of the piece P, the current layout and the region outside the hide. This is shown in Figure 4. The piece offset is given by the NFP between the piece and a given square (represented by the light blue area, in Figure 4). Therefore $area(I_i)$ is the area of the i^{th} polygon of I. Other functions ((E2), (E4), (E6) and (E8)) consider implicitly the area of the pieces, with the purpose of avoiding the preferential selection of the largest pieces.


Figure 4: Intersection between the offset of a piece and the non-usable region of the hide

The functions (E9) to (E14) relies on simpler criteria that are computationally easier to evaluate.

A VARIABLE NEIGHBORHOOD SEARCH ALGORITHM

Overview

The Variable Neighborhood Search (VNS) meta-heuristic is based on local search methods that start with an initial solution and systematically explore different neighborhood spaces so as to improve these solutions. A shaking phase at the beginning of the local search allows the overall procedure to escape from local optima by switching among the neighborhoods. In this section, we describe a VNS algorithm for the general LNP addressed in this paper. The algorithm was proposed and described in detail in (Alves et al. 2011b).

The initial layout used by the VNS algorithm is generated by the constructive algorithms described above. The solution is constructed by using the strategies (G1), (S2.D.1.c), (P6) and (E11), corresponding to the classes GRP, SEL, PLAC and EVAL. A feasible layout can be represented as a sequence of pieces combined with the iterative application of strategies from the PLAC and EVAL classes. Based on the sequence of pieces generated using these strategies, a neighborhood of an initial solution can be obtained by applying some specific movements. In our approach, we propose a set of four distinct movements, designated by M_1, M_2, M_3, and M_4. Each one of these movements generate four different neighborhood structures, denoted by $N_i, i = 1, \dots, 4$.

Movements and neighborhood structures

Given a layout represented by the sequence of pieces

$$S = (s_1, s_2, \dots, s_{|S|})$$

the following describes the main principles of the movements used in our algorithm:

M1
- Exchange a piece p by another that is not in the sequence;
- Remove all the pieces in the sequence from this piece forward;
- Fill the hide using the constructive heuristic.

M2

- Exchange a piece p by another that is not in the sequence;
- Keep the rest of the sequence unchanged, placing the corresponding pieces using iteratively the strategies from classes PLAC and EVAL of the constructive heuristic;
- Fill the rest of the hide by applying the complete heuristic.

M3

- Swap two pieces p and p' in the sequence, if p' can be placed on the hide with a better fitness than p following the strategies from classes PLAC and EVAL of the heuristic;
- Place all the pieces of the sequence using the strategies from classes PLAC and EVAL of the heuristic;
- Fill the rest of the hide with the complete heuristic.

M4

- Remove a piece p of the sequence;
- Place all the pieces of the resulting sequence by applying iteratively the strategies from classes PLAC and EVAL of the heuristic;
- Fill the rest of the hide with the complete heuristic.

For each movement, p is the piece that is exchanged, swapped with another piece or removed from the sequence. This piece is to be chosen from a set of candidates denoted by P. Likewise, the piece candidate to replace p is denoted by p' and belongs to a candidate set P'.

Given de sequence of pieces S, F is a vector that represents the fitness value for each piece from S. This value is given by the evaluation function from class EVAL.

$$F = \left(f_1, f_2, \dots, f_{|S|}\right)$$

with $0 \leq f_k \leq 1, k = 1, \dots, |S|$.

These movements are defined in a way that the piece p', that will replace piece p, will always have a better fitness value than the latter. Furthermore, the piece p will be selected from a restricted part of the sequence. This part is given by the percentage of material usage achieved right after a piece is placed on the hide. Given a sequence of pieces S, the evolution of the material usage, as each piece is placed on the hide, is given by the vector U:

$$U = \left(u_1, u_2, \dots, u_{|S|}\right)$$

with $u_k < u_{k+1}, k = 1, \dots, |S| - 1$.

Therefore, neighborhood structures that can be defined from these movements, depend on the following parameters:

q: number of pieces that may be exchanged, swapped with another piece or removed from sequence, i.e. $|P|$;

r: number of pieces that may substitute the piece the piece p, i.e. $|P'|$;

(u_{min}, u_{max}): the material usage interval that determines the subpart of the sequence from which the pieces of P are chosen, i.e. $(s_i, \dots, s_j)$ with

$$i = argmin_{l \in \{1,\dots,|S|\}}\{u_l : u_l \geq u_{min}, u_l \in U\},$$

and

$$j = argmax_{l \in \{1,\dots,|S|\}}\{u_l : u_l \geq u_{max}, u_l \in U\}.$$

VNS algorithm for the LNP

The implementation details of the VNS algorithm for the LNP are the following:

```
1. Input: for each $N_i, i = 1, \dots, 4$, the sets of parameters $(u_{min,i}, u_{max,i})$, $q_i$ and $r_i$ will
   define the specific neighborhood structures; A limit $t_{limit}$ of the total computing time.

2. Initialization: $L \coloneqq findInitialSolution\,()$;

3. Repeat the following steps until $cpuTime(\,) \geq t_{limit}$:
      a. $i \coloneqq 1$;
      b. Repeat the following steps until $i = 4$;
           i. $L' \coloneqq shaking\,(L, i)$;
          ii. $L'' := firstImprovement(L', i)$;
         iii. If $v(L'') < v(L)$ then
                  $L \coloneqq L''$;
                  $i \coloneqq 1$;
              Else
                  $i \coloneqq i + 1$;
```

A GREEDY RANDOMIZED ADAPTIVE SEARCH PROCEDURE

The meta-heuristic GRASP (Greedy Randomized Adaptive Search Procedure) is an iterative procedure that relies on a constructive and an improvement phase. Starting with an initial solution, given by a constructive algorithm, the improvement phase applies a local search procedure to search for a better solution.

The GRASP constructive component generates the initial solution iteratively by adding one element at a time to the actual solution. At each step of the constructive procedure, the best elements to incorporate the actual solution are represented by a Restrictive Candidate List (RCL) which is built using a greedy function. Selecting an element from the RCL is done randomly. Based on predefined neighborhood structures, the improvement phase of the GRASP algorithm consists on a local search procedure applied to the neighborhood of the initial solution. In the sequel, we describe the details of these phases and how they integrate into our GRASP algorithm for the LNP.

Constructive phase

Starting with an empty leather hide, the constructive phase of the developed GRASP algorithm consists in adding iteratively one piece at a time until no more pieces can be added. This procedure is based on the constructive heuristics described above.

The selection of the next piece to place is done randomly from a RCL composed by the ten pieces with the smallest IFP relative to the actual layout. Let P^t be the set of the pieces p_i^t available for placement at a given iteration t. Given the ordered sequence of pieces $\left\{p_1^t, p_2^t, \dots, p_{|P^t|}^t\right\}$, such that $areaIFPs(p_i^t) \leq areaIFPs(p_{i+1}^t)$, the RCL is given by:

$$RCL(t) = \{\mathrm{p}_\mathrm{i}^t : i = 1, \dots, 10\}$$

We use the information provided by the IFPs as a measure of the quality of a placement. By selecting the pieces with the smallest IFP, we try to favor the pieces with the largest probability of being placed closely to the contour of the current layout. After a piece has been selected, the selection of a feasible placement region on the hide and the evaluation of a given placement position are done by applying the strategies (P4) and (E11) of the POS and EVAL classes of our set of constructive heuristics. The strategy (P4) supports the strategy behind the definition of the RCL, since it consists in selecting the smallest IFP of the piece. Using the strategy (E11) to evaluate the placement positions allows for a fast evaluation of the whole set of feasible positions within the selected region.

Improvement phase

The improvement phase of the algorithm consists of a local search procedure that explores the neighborhood solutions of the initial layout. This neighborhood structure is given by the movement M_1 described above for the VNS algorithm. Based on the sequence of pieces representing the initial solution, this movement selects a piece p for replacement within the subsequence of pieces that leads to a material usage between 10% to 50%. This piece is replaced with piece p' that have a better fitness value than the latter. Starting with the solution given by the constructive phase, the neighborhood related to this movement is explored by the local search procedure, until a local optimum is found or a time limit is reached. Within the time limit, if a local optimum is found, the GRASP algorithm restarts from the constructive phase.

Randomization within the GRASP algorithm allows to generate an initial solution with a greater probability of being different from the previous. This feature allows the algorithm to explore the solution space more extensively.

COMPUTATIONAL RESULTS

Several computational experiments were conducted to evaluate the performance of the algorithms described above. For these experiments, we used two sets of real instances based on the actual models of cars produced by a large national company. The first set is composed by 23 different pieces, and the second by 22 pieces (see figures 5 and 6).

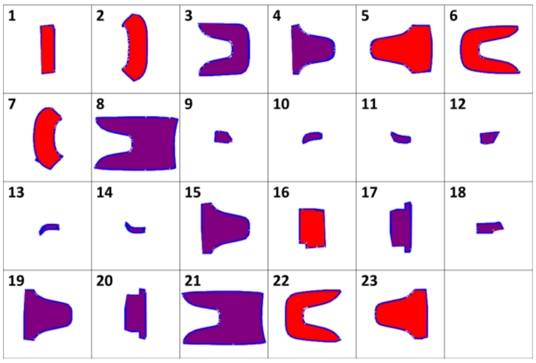

Figure 5: Set of pieces from car model 1

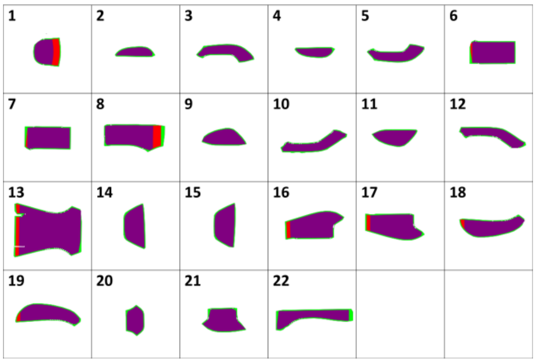

Figure 6: Set of pieces from car model 2

From the pieces of these two models, we defined a set of 7 instances from the first model and 4 instances from the second. These instances differ in the number of different pieces (*DPieces*) and the total number of pieces (*NPieces*) from each model that were considered (see Table 1).

MODEL	INSTANCE	DPieces	NPieces
1	1	23	60
	2	8	80
	3	23	80
	4	8	100
	5	23	100
	6	8	120
	7	23	120
2	1	22	60
	2	9	80
	3	22	80
	4	22	100

Table 1: Instances

The set of computational experiments can be arranged into the following groups, regarding the presented algorithms:

1. Constructive algorithms:
   a. Comparative analysis of the different strategies;
   b. Overall performance of different combinations of strategies;
2. VNS algorithms:
   a. Parameter tuning of the VNS algorithms;
   b. VNS algorithms performance analysis;
3. GRASP algorithm:
   a. GRASP algorithm performance analysis.

The purpose of the computational experiments performed in 1.a. was to compare the performance results of each strategies within the each classes defined for the constructive algorithm. The results were analyzed considering the efficiencies of the layouts and the time needed to generate them. From these experiments, it was possible to verify that both the area and the quality zones of the pieces were the best attributes to consider within the set of grouping strategies. Furthermore, the strategies that rely on the IFPs of the pieces (strategies SEL and PLAC) leads usually to better results. Finally, the strategies that relies on the computation of the offset of the pieces to evaluate the quality of the fitness, despite an increased computational time, leads to the most efficient layouts layouts.

Based on the computational experiments conducted in 1.a., we defined a set of 20 constructive algorithms based on different combinations of strategies. These algorithms can be divided into two groups: one with the fastest combinations of strategies and another with the strategies that produce the best layouts. The computational tests performed in 1.b. were conducted from the 11 sets of instances reported in Table 1, and using 3 leather hides for each instance. Considering the results obtained from these experiments, we can distinguish between two different sets of constructive algorithms (see Table 2).

The combinations 5 and 12 exhibit an interesting compromise between the efficiency of the layouts and computational time required to compute them. On average, for the first two hides, these combinations reached efficiencies of 72% and 62%, respectively. Even with the most computationally expensive combination (12), the time required is approximately 200 seconds.

The results achieved by the combinations 6 to 10 shows a good performance in terms of the computational times. They can generate a layout 4 times faster than the previous combinations. Even if the efficiency of the corresponding layouts is slightly worse (69% and 60%, on average, for the first two hides), these combinations can still be considered as a good starting point for the improvement methods.

COMB	GRP	SEL	PLAC	EVAL
5	(G5)	(S2.D.1.c)	-	(E1)
12	(G5)	(S1.D)	(P6)	(E1)
6	(G5)	(S2.D.1.c)	-	(E12.MAX)
7	(G5)	(S2.D.1.c)	-	(E14.MIN)
8	(G5)	(S2.D.1.c)	-	(E14.MAX)
9	(G5)	(S2.D.1.c)	-	(E13.MIN)
10	(G5)	(S2.D.1.c)	-	(E13.MAX)

Table 2: Best combination strategies

On 2.a., an extensive set of computational experiments were conducted with the purpose of tuning the VNS algorithms parameters and compare different strategies. These objectives can be summarized as follows:

1. Evaluate the impact of each neighborhood on the quality of the generated layouts;
2. Compare the impact on performance of different strategies obtained by setting $(u_{min,i}, u_{max,i})$ parameter;
3. Evaluate the impact on performance by setting various values for the parameters q_i and r_i.

From these experiments, it was possible to verify that the best results were achieved when all the neighborhoods were used. This happens because of the diversity of solutions obtained when the VNS algorithm considers the neighborhoods from all the described movements. Different settings for the parameters $(u_{min,i}, u_{max,i})$, q_i and r_i shows that the best results are achieved when these parameters are set, respectively, to $(10\%, 50\%)$, 3 and 3.

Using the parameterization defined previously, an extensive set of computational experiences was conducted in 2.b. In addition there was set a computational time limit to 600 seconds. This limit is equivalent to the average time needed to construct the layout manually by a team of u 2 human operators. The results obtained shows that the VNS algorithm was able to improve the initial solution given by the constructive heuristic in 96% of the tested instances. Furthermore, for almost half of the instances, the results show an improvement on efficiency after the first 200 seconds. On average, the average improvement on efficiency reaches approximately 3%.

The last set of computational experiments 3.a. was conducted with the objective of evaluating the performance of the GRASP algorithm. For these experiments, we used a time limit of 600 seconds. The results show a good performance of this algorithm. On average, the initial result were improved by 5%. Besides, every single solution given by the constructive heuristic is improved by the GRASP algorithm. The efficiency of approximately half of the layouts was improved by 2% only after 200 seconds of computational time.

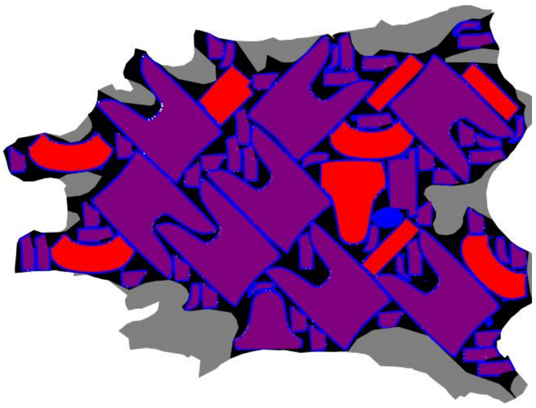

Figure 7: Cutting pattern (Efficiency: 76.85%; Strategies: (G1), (S1.D), (P2) and (E1))

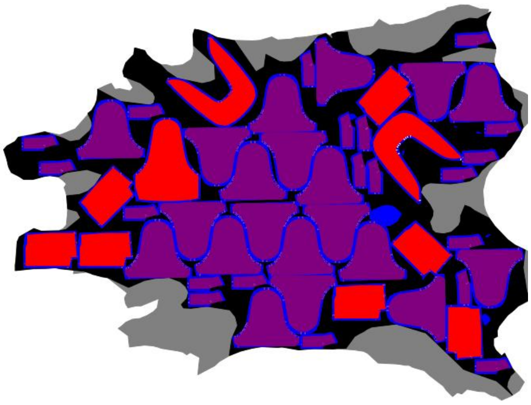

Figure 8: Cutting pattern (Efficiency: 69.32%; Strategies: (G4), (S2.D.1.a) and (E1))

CONCLUSIONS

In this paper, we described three different approaches to solve leather nesting problems in the context of the automotive industry. The first set of algorithms, based on a constructive heuristic, uses an extensive collection of strategies with the purpose of taking advantage of the singularities of this problem: the high irregularity of the geometric shapes, the high variety of shape sizes, and the existence of defects on the hides and quality zones. A VNS based algorithm was also presented. A set of 4 different neighborhood structures, based on original movements, allows the exploration of a variety of solution in order to improve the efficiency results. The third and last approach consists in a GRASP algorithm. The simplicity and efficiency of this method allows to explore an high range of solution in a limited amount of time. Finally, we summarized the results of an extensive set of computational experiments that were conducted to evaluate the performance of the algorithms. We used several instances derived from real car models produced by a large national company. The results of these experiments illustrate the quality of the presented algorithms both in terms of material usage and computational time.

REFERENCES

Heistermann, J. and Lengauer, T. The nesting problem in the leather manufacturing industry. *Annals of Operations Research 57*, 1 (1995), 147-173.

Alves, C., Brás, P., Valério de Carvalho, J., and Pinto, T. New constructive algorithms for leather nesting in the automotive industry. *Computers & Operations Research, in press*, (2011a).

Alves, C., Brás, P., Valério de Carvalho, J., and Pinto, T. A variable neighborhood search algorithm for the leather nesting problem. submitted, (2011b).

Brás, P. Algoritmos de posicionamento de figuras irregulares: aplicação a um caso real da indústria automóvel. *PhD Thesis, University of Minho,* (2011).

Crispin, A., Clay, P., Taylor, G., Bayes, T., e Reedman, D. Genetic Algorithm Coding Methods for Leather Nesting. *Applied Intelligence 23*, 1 (2005), 9-20.

Wäscher, G., Haußner, H., and Schumann, H. An improved typology of cutting and packing problems. *European Journal of Operational Research* 183, 3 (2007), 1109-1130.

Zhang, Y. and Yang, C. A Generic Approach for Leather Nesting. *Fifth International Conference on Natural Computation 5*, (2009), 303-307

AUTHORS' ACKNOWLEDGMENTS

This work is a result of my PhD thesis, supervised by professors Cláudio Manuel Martins Alves and José Manuel Vasconcelos Valério de Carvalho.

BRÁS, PEDRO
Email: pedro.bras@dps.uminho.pt

Graduation degree: Mathematics;
MSc degree: Industrial Engineering;
PhD degree: Production and Systems Engineering

ALVES, CLÁUDIO
Email: claudio@dps.uminho.pt

Graduation degree: Systems and Informatics Engineering;
MSc degree: Industrial Engineering;
PhD degree: Production and Systems Engineering

VALÉRIO DE CARVALHO, JOSÉ
Email: vc@dps.uminho.pt

Graduation degree: Electrotechnical engineering;
MSc degree: Industrial Engineering;
PhD degree: Production and Systems Engineering