A Hybrid Genetic Algorithm for the Cut Order Planning Problem

Ahlem Bouziri[1] and Rym M'hallah[2]

[1] Institut Supérieur des Arts Multimédia, Manouba University, 13 rue des Entrepreneurs, Charguia II, Tunisia
ahlem.bouziri@wanadoo.tn
[2] Department of Statistics and Operations Research, Kuwait University, P.O. Box 5969, Safat 13060, Kuwait
mhallah@kuc01.kuniv.edu.kw

Abstract. This paper proposes a new hybrid heuristic to a difficult but frequently occurring problem in the apparel industry: the cut order planning (COP). This problem consists of finding the combination of ordered sizes on the material layers that minimizes total material utilization. The current practice in industry solves COP independently from the two-dimensional layout (TDL) problem; i.e., COP estimates the length of the layout required to cut a particular combination of sizes instead of packing the pieces on the fabric and determining the actual length used. Evidently, this results in a build up of estimation errors; thus increased waste. Herein, COP and TDL are combined into a single problem CT. The resulting problem is modeled and solved using a hybrid heuristic which combines the advantages of population based approaches (genetic algorithms) with those of local search (simulated annealing). The experimental results show the validity of the proposed model, and the sizeable savings it induces when solved using the proposed hybrid heuristic.

Keywords: hybrid heuristics, simulated annealing, genetic algorithms.

1 Introduction

Cut order planning (COP) is a frequent problem in the apparel industry. It consists of finding the best combination of the layout of the patterns of the sizes ordered by the clients on a rectangular fabric of fixed width W. The best combination optimizes material utilization; that is, minimizes L, the total length of fabric used to generate all the pieces of the patterns included in the order. An apparel manufacturing order undergoes four stages. First, a COP problem is solved. Second, Two-Dimensional Layouts (TDL) are generated. Third, the layouts are cut and divided into lots. Fourth and last, the lots are sewn, assembled and packaged. Currently, stages 1 and 2 are undertaken independently. In stage 1, human experts or dedicated software solve COP using an *estimated* length of the layout of a combination of sizes. The estimate is generally a coarse approximation that is read from catalogs prescribing material usage of basic models.

H.G. Okuno and M. Ali (Eds.): IEA/AIE 2007, LNAI 4570, pp. 454–463, 2007.

The COP solution is then transferred to the TDL room where it is packed on rectangles of fixed width, and maximal length $\overline{l}$ where $\overline{l}$ is the length of the cutting table.

An order, in the apparel manufacturing process, consists of one or more garments (eg., a pair of pants, a shirt, a suit, etc.) in varying colors, sizes and quantities per size per color. The order is split according to garment type and color into suborders. Since different colored clothes are layered separately and different garments are treated separately (to minimize the risk of errors during assembly), hereafter, an order denotes a suborder. An order is characterized by its set S of sizes, their number n, and the order quantity q_s for size $s \in S$.

A garment is a finite set G of pieces which when assembled according to a preset procedure yield the garment. The dimensions of each piece depend on its size s. Let G_s denote the set of pieces of a garment in size $s \in S$. Each set $G_s, s \in S$, is duplicated q_s times. The duplicated sets are positioned on a fabric of known fixed width W as to minimize the total length L. The positioned pieces are cut using a computer guided cutter. Since the length of the cutting table is finite, and the set up for the cutting stage is high, producers choose to layer the cloth and to position sets of pieces on layered cloth. The automatic guided cutter works best when the number of layers or *ply height* is in the interval $[\underline{h}, \overline{h}]$, where $\underline{h}$ and $\overline{h}$ are respectively minimal and maximal ply height.

The ply height is an additional variable of the problem. For instance, if the order requires producing 20 blouses of size XP, then G_{XP} can be positioned 20 times on a fabric of a single layer, or 4 times on a fabric that has 5 plies, or only once on a fabric having 20 plies, depending on $\overline{l}, \underline{h}, \overline{h}$, and on the overall efficiency of the layout. The layered cloth is called a *section*. An order may be split into one or more sections; each having a different number of plies and different sets.

Formally, the problem can be stated as follows. Given a set S of sizes, the sets $G_s, s \in S$ of pieces composing the garment, the fabric width W, the minimum and maximum ply height $\underline{h}$ and $\overline{h}$, and the maximum length of the cutting table $\overline{l}$, find the configuration that minimizes the *total* length L of used fabric. A configuration is completely specified by the number of sections p required to position all the pieces, where the sections need not be identical. Each section $i,\ i = 1, \ldots, p$ is characterized by its ply height h_i, the number of occurrences O_{is} of each size $s \in S$, and the shortest length layout l_i of the pieces it contains when set on a rectangle of fixed width W. The configuration is feasible if $l_i \leq \overline{l}$, and $\underline{h} \leq h_i \leq \overline{h},\ i = 1, \ldots, p$. The total length of the layout $L = \sum_{i=1}^{p} h_i l_i$.

In industry, markers solve COP using cataloged l_i values. Herein, $l_i,\ i = 1, \ldots, p$ is determined by solving a two dimensional layout problem for section i. That is, stages 1 and 2 of the apparel manufacturing process are combined into a single COP-TDL problem. The resulting problem, denoted CT, yields an overall total length that is more accurate than that estimated by markers. Fig. 1 provides an example of the input and output of CT for an order consisting of $n = 6$ different sizes: XP, P, M, T, XT, XXT with $q_{XP} = 20, q_P = 25, q_M = 35, q_T = 30, q_{XT} = 20$, and $q_{XXT} = 10$.

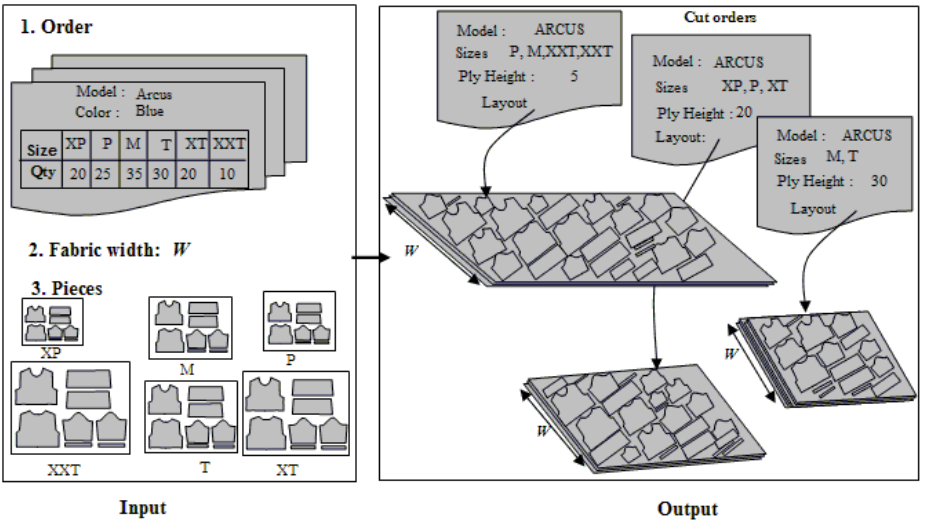

Fig. 1. Input and Output for COP

CT, which is a complex combinatorial problem, is NP-complete. It is an extension of COP, which is in turn NP-complete [7]. Herein, CT is approximately solved using genetic annealing (GAn), a hybrid heuristic that combines the advantages of local and global optimization procedures.

This paper is organized as follows. Section 2 gives a brief review of the COP and TDL literature. Section 3 describes the solution configuration. Section 4 details GAn. Section 5 assesses the performance of GAn and evaluates the potential savings induced by combining COP and TDL. Finally, Section 6 summarizes this research and indicates possible extensions.

2 Literature Review

TDL, which is a Cutting and Packing Problem, consists of finding the minimal length layout of a set of (ir)regular two-dimensional pieces on a stock sheet of finite width but infinite length. The layout must not contain any overlap. Even though NP-hard, TDL has been widely studied in the literature [1] - [4], [6], [8]. Proposed methods fit into one of two classes. In the first class, prior to being packed, irregular items are nested into regular shapes. In the second class, irregular pieces are packed using complex geometric computations [6].

COP has been addressed by ElOmri [5] and Jacobs-Blecha et al. [7] who focused on the cut order planning stage of the manufacturing process. They both assumed that the length of the layouts are catalogued. Jacobs-Blecha et al. [7] concluded that the cut order cost is the most dominant component of the cutting process. M'Hallah et al. [9] provided the example of a local industry which sells its value added for a $1/ garment, making a net profit of $.4/ garment while

its estimated cost for cloth is \$17.5/ garment. Reducing waste by 6% multiplies the original net profit by a factor of 3.5. Any saving on cloth represents a large percentage of the gross gain and a further larger percentage of the net gain. Herein, it is assumed that the cost of fabric is the dominant cost of COP and that minimizing the total COP cost minimizes the total length of the layout.

3 Solution Configuration

A solution to CT is represented by a $(p \times n + 2)$ matrix. The first n columns contain $O_{is}, i = 1, \ldots, p, s \in S$. Columns $n + 1$ and $n + 2$ contain respectively h_i and $l_i, i = 1, \ldots, p$. The number of columns of the matrix is constant but the number of sections is not necessarily identical for all feasible solutions. A solution to CT is feasible if it satisfies the demand constraints; i.e., if $\sum_{i=1}^{p} h_i O_{is} = q_s,\ s \in S$. The fitness of a feasible solution is the total length of fabric needed to cut all its sections: $fitness = \sum_{i=1}^{p} h_i l_i$. The solution corresponding to the output of Fig. 1 is given by the 3×8 matrix

$$\begin{pmatrix} 0\,0\,1\,1\,0\,0\,30\,l_1 \\ 1\,1\,0\,0\,1\,0\,20\,l_2 \\ 0\,1\,1\,0\,0\,2\ \ 5\,l_3 \end{pmatrix}.$$

The first row of the matrix represents section 1, which includes two sizes, and has a ply height equal to 30 and a length l_1. The third column shows that size M occurs once on section 1 (i.e., $O_{1M} = 1$) but does not appear on section 2 (i.e., $O_{2M} = 0$) and appears once on section 3 (i.e., $O_{3M} = 1$). The sets of pieces assigned to any section $i, i = 1, \ldots, p$, are packed using the sequential best local position (BLP) layout procedure of [6].

4 The Hybrid Approach

CT is solved using a hybrid heuristic: genetic annealing, which combines the advantages of global and local optimization. It guides the diversification / intensification of the search of the solution space. During the first generations, it favors diversification, but gradually increases intensification by imposing stricter criteria for the inclusion of offsprings into the population as time evolves.

GAn is implemented using a low-level teamwork hybridization [10]. It explores several regions of the solution space using a population based meta heuristic, which in this case is GA. An offspring is immediately admitted into the population if its fitness is lower than that of its parent. However, a non-improving offspring is allowed into the population if and only if its fitness is less than its parent's threshold level. Initially, the threshold level of an individual is its fitness. As it proliferates, an individual is either replaced by its offspring -if it produces a fitter offspring- or has its threshold decreased; that is, its endurance is increased allowing it to further proliferate. At each iteration, a set composed of the fittest chromosomes is subject to intensification around its immediate neighborhood using SA. Formally, the adopted GAn proceeds as follows.

Initialization.

1. Create an initial population that has $popsize$ random chromosomes.
2. Compute the fitness L_j of each chromosome j and set its threshold level $\theta_j = L_j,\ j = 1, \ldots, popsize$.

Iterative Step.

For $generation = 1, n_g$,

1. For $j = 1, popsize$,
   (a) Let chromosome j be Parent1 and let Child be the offspring obtained by mutation or crossover of Parent1.
   (b) Compute L_c, the fitness of Child.
   (c) If $L_c < L_j$, then replace chromosome j by Child, set $\theta_j = L_c$,
   (d) Else,
   - Set $\tau_0 = 0.6$, and $\theta_j = \theta_j * \alpha$, where $\alpha \in [.95, .99]$.
   - If $\exp(-\theta_j)$ > Rand[0,1], replace chromosome j by Child, and set $\theta_j = L_c$.
2. Select the $n_{SA}/2$ fittest chromosomes of the current population
3. Set $\tau_0 = 0.2$.
4. For $j = 1, n_{SA}/2$,
   -- Apply SA with chromosome j being the initial solution and T_0 such that $\tau_0 = e^{-\frac{|\overline{\Delta}|}{T_0}}$, where $\overline{\Delta}$ is the average variation of the fitness of the chromosomes of the current population from their threshold levels.
5. Randomly pick $n_{SA}/2$ chromosomes from the current population
6. Set $\tau_0 = 0.6$.
7. For $j = 1, n_{SA}/2$,
   -- Apply SA with chromosome j being the initial solution and T_0 such that $\tau_0 = e^{-\frac{|\overline{\Delta}|}{T_0}}$, where $\overline{\Delta}$ is the average variation of the fitness of the chromosomes of the current population from their threshold levels.

Crossover produces a new chromosome (offspring or Child) from two parents: Parent1 and Parent2. The Child inherits part of its genes from the p_1 genes of Parent1 and is completed according to the p_2 genes of Parent2, as explained below.

1. Pick two integer random numbers i_1 and i_2 from the discrete uniform $[1, p_1]$.
2. Copy sections i_1 through i_2 from Parent1 to the first $i_2 - i_1 + 1$ genes of Child.
3. Set the current number of sections of Child $p^{\text{Child}} = i_2 - i_1 + 1$, and set $i = 1$.
4. Calculate the residual demand for Child: $r_s = q_s - \sum_{i=1}^{p^{\text{Child}}} O_{is} h_i, s \in S$.
5. If $\max_{s \in S}\{r_s\} = 0$, stop.
6. Generate for Child a new gene $i' = p^{\text{Child}} + i$, whose $h_{i'} = \min\{h_i^{\text{Parent2}}, LCD(r_s : r_s > 0, O_{is}^{\text{Parent2}} > 0, s \in S)\}$, with $O_{i's} = \frac{r_s}{h_{i'}},\ s \in S$, where h_i^{Parent2} is the ply height of section i of Parent2, and O_{is}^{Parent2} is the number of occurrences of size s in section i of Parent2.
7. Increment i by 1. Goto step 4.

To illustrate how crossover proceeds, consider the example of crossing `Parent1`

$$\begin{pmatrix} 1\,0\,0\,0\,0\,1\;10\;l_1 \\ 0\,1\,0\,0\,1\,0\;20\;l_2 \\ 0\,0\,1\,1\,0\,0\;30\;l_3 \\ 2\,1\,1\,0\,0\,0\;\;5\;l_4 \end{pmatrix}$$

and `Parent2`

$$\begin{pmatrix} 2\,2\,3\,3\,2\,1\;10\;l_1 \\ 0\,0\,1\,1\,0\,0\;\;5\;\;l_2 \end{pmatrix}.$$

Let i_1 and i_2, generated from the discrete Uniform $[1, 4]$, be 2 and 3, respectively. `Child` inherits genes 2 and 3 of `Parent1`:

$$\begin{pmatrix} 0\,1\,0\,0\,1\,0\;20\;l_2 \\ 0\,0\,1\,1\,0\,0\;30\;l_3 \end{pmatrix}.$$

Computing the residual demand yields: $r_{XP} = 20,\ r_P = 25 - 20 = 5,\ r_M = 35 - 30 = 5,\ r_T = 30 - 30 = 0,\ r_{XT} = 20 - 20 = 0$, and $r_{XXT} = 10 - 0 = 10$. Since $\max_{s \in S}\{r_s\} = 20 > 0$, then `Child` is completed using the genes of `Parent2`. The third section of `Child` has $h_3 = \min\{20, 5\}$, where 20 is the ply height of section 1 of `Parent2`, and $5 = LCD(r_s : r_s > 0, O_{is} > 0, s \in S) = LCD(r_{XP}, r_P, r_M, r_{XXT}) = LCD(20, 5, 5, 10)$.

Computing $O_{3s} = \frac{r_s}{h_3}$ yields $O_{3XP} = 4, O_{3P} = 1, O_{3M} = 1,\ O_{3XXT} = 2$, and $r_s = 0, s \in S$. Therefore, `Child` has only 3 sections:

$$\begin{pmatrix} 0\,1\,0\,0\,1\,0\;20\;l_2 \\ 0\,0\,1\,1\,0\,0\;30\;l_3 \\ 4\,1\,1\,0\,0\,2\;\;5\;\;l_3' \end{pmatrix}.$$

Mutation is applied to some of the offspring population to make them better adapted to their new environment. It is not intended to greatly disturb the current structure of an offspring. As such, mutation moves a size from its original section to another section. A formal statement of the mutation operator follows.

1. Randomly choose a section i of the p sections of Parent.
2. Randomly choose a size s from set $S' = \{s \in S : O_{is} > 0\}$.
3. Compute the residual demand resulting from moving s from section i, i.e., $r_s = h_i O_{is}$.
4. Find a section $j,\ j = 1, \ldots, p, j \neq i$, whose h_j is a divider of r_s.
   - If only one such section exists, set $O_{js} = O_{js} + \frac{r_s}{h_j}$.
   - Else if more than one section exist say j and j', then add size s to the section whose total length will be smallest when s is added.
   - Else if no such section exists, set $J = \{j = 1, \ldots, p,\ j \neq i : h_j \leq r_s\}$.
     * If $J \neq \emptyset$, dispatch r_s among the sections of J such that $\sum_{j \in J} \Delta O_{js} h_j = r_s$, where ΔO_{js} is the additional number of occurrences of size s in section j.
     * If $J = \emptyset$, create a new section $p + 1$ whose $h_{p+1} = h_i, O_{p+1s} = O_{is}$, and $O_{p+1s'} = 0, s' \in S, s' \neq s$.

The **Genetic parameters** are the population size *popsize*, the number of generations n_g, and the probability of mutation P_m. A large population size insures the diversification of the population while a large number of generations gives chromosomes a higher chance of strengthening their better genes. However, the increase of these two parameters increases runtime. A good tradeoff between diversification and intensification is therefore needed. We opted for a moderately large population size and a relatively small number of generations; thus, opting for diversification of the search in the solution space. Intensification is achieved by allowing the best chromosomes to proliferate at least once and to be mutated more frequently than lower quality chromosomes. Indeed, P_m is set inversely proportional to the rank of the chromosome in the population.

Simulated Annealing (SA) is a meta strategy for optimization by local improvements. It is a local search heuristic that allows occasional uphill moves. It allows moves from a current solution to a not necessarily improving one in hope that it leads to a minimal cost one. The process becomes more selective and accepts fewer non-improving solutions as it ages or gains experience. A description of SA follows.

1. Let Sol be the initial solution and f_{Sol}. its fitness
2. Let T_0 be the initial temperature
3. Set $Sol^* = Sol$, and $f* = f_{Sol}$.
4. Fix the size of the neighborhood M (i.e., M is the length of the plateau).
5. Set $k = 0$.
6. Repeat M times.
   - Obtain a neighboring solution N_{Sol}, and evaluate its fitness $f_{N_{Sol}}$.
   - If $\Delta = f_{N_{Sol}} - f_{Sol} \leq 0$ then
     -- $Sol = N_{Sol}$, and $f_{Sol} = f_{N_{Sol}}$;
     -- if $f_{N_{Sol}} < f^*$, then $Sol^* = N_{Sol}$, and $f^* = f_{N_{Sol}}$;
   - else if $\exp(\frac{-\Delta}{T_k}) > Uniform[0,1]$, then
     -- $Sol = N_{Sol}$, and $f_{Sol} = f_{N_{Sol}}$.
7. Increment$k$ by 1, and compute T_k, the temperature at plateau k.
8. If the stopping criterion is not satisfied, go to Step 5.

In Step 5, the acceptance of an uphill move is controlled by the probability $\exp(\frac{-\Delta}{T_k})$. The probability that an uphill move of size Δ is accepted diminishes as the temperature declines. For a fixed T_k, small uphill moves have higher probabilities of acceptance than large ones. In this implementation of SA, the **temperature** of the annealing process is decreased geometrically; that is, at plateau k, $T_k = 0.9T_{k-1}$.

The **initial temperature** T_0 is a function of the probability of acceptance $\tau_0 = e^{-\frac{|\overline{\Delta}|}{T_0}}$, where $\overline{\Delta}$ is the average variation of the fitness of 100 randomly generated solutions. τ_0 is, in turn, a function of the quality of the initial solution. $\tau_0 = 0.2$ when the initial solution is "good", and $\tau_0 = 0.6$ otherwise. The initial solution is "good"if its percent deviation from a computed lower bound is less than 25%. Setting $\tau_0 = 0.2$ makes the algorithm very selective starting its first

Table 1. First problem set

Instance	S	$\|G\|$	W (m)	q_s	L^*
a_1	$\{P, M, T, XT\}$	7	1.0	$q_P = 2, q_M = 5, q_T = 2, q_{XT} = 1$	1.40
a_2	$\{P, M, T, XT\}$	7	1.0	$q_P = 10, q_M = 25, q_T = 10, q_{XT} = 5$	7.00
a_3	$\{P, M, T, XT\}$	7	1.0	$q_P = 20, q_M = 50, q_T = 20, q_{XT} = 30$	22.00
a_4	$\{P, M, T, XT\}$	7	1.0	$q_P = 10, q_M = 25, q_T = 10, q_{XT} = 15$	11.00
b_1	$\{P, M, T, XT\}$	11	1.6	$q_P = 6, q_M = 6, q_T = 2, q_{XT} = 2$	4.00
b_2	$\{P, M, T, XT\}$	11	1.2	$q_P = 20, q_M = 30, q_T = 20, q_{XT} = 5$	26.67
b_3	$\{P, M, T, XT\}$	11	1.6	$q_P = 12, q_M = 12, q_T = 18, q_{XT} = 18$	25.5
b_4	$\{P, M, T, XT\}$	11	1.2	$q_P = 20, q_M = 30, q_T = 20, q_{XT} = 15$	37.34
c_1	$\{XP, P, M, T, XT, XXT\}$	20	2.0	$q_{XP} = 18, q_P = 30, q_M = 12, q_T = 18, q_{XT} = 6, q_{XXT} = 6$	99.00
c_2	$\{XP, P, M, T, XT\}$	20	2.5	$q_{XP} = 24, q_P = 24, q_M = 42, q_T = 24, q_{XT} = 6$	90.00
c_3	$\{XP, P, M, T, XT, XXT\}$	20	2.0	$q_{XP} = 18, q_P = 30, q_M = 12, q_T = 18, q_{XT} = 12, q_{XXT} = 12$	136.5
c_4	$\{XP, P, M, T, XT\}$	20	2.5	$q_{XP} = 24, q_P = 24, q_M = 42, q_T = 24, q_{XT} = 12$	100.8

Table 2. Computational Results for first problem set

Instance	$L*$	Δ_{SA}	Δ_{GA}	Δ_{GAn}
a_1	1.40	3.57	3.57	3.57
a_2	7.00	1.07	1.07	1.07
a_3	22.00	0.68	0.45	0.11
a_4	11.00	0.91	0.68	0.45
b_1	4.00	1.88	2.50	1.87
b_2	26.66	0.50	0.87	0.50
b_3	25.50	6.18	2.05	0.29
b_4	37.33	1.92	1.72	1.58
c_1	99.00	4.95	3.03	2.42
c_2	90.00	5.81	2.66	0.69
c_3	136.50	4.29	2.96	2.96
c_4	100.80	6.67	2.77	2.38

iterations whereas setting $\tau_0 = 0.6$ makes the algorithm accept uphill moves more often during the first iterations.

A **neighbor** is obtained by randomly moving a size $s \in S$ from a random section $i,\ i = 1, \ldots, p$ in the current solution Sol to a different existing section $j,\ j = 1, \ldots, p,\ i \neq j$ while maintaining demand feasibility. Removing a size s from its section i creates a residual demand $r_s = h_i * O_{is}$. Restoring demand feasibility without creating an additional section requires moving size s to a section j whose h_j is a divider of r_s. When such a section j is identified, its corresponding O_{js} is updated; i.e., $O_{js} = O_{js} + \frac{r_s}{h_j}$. If no such section exists, a different size from a different section is considered for a move. The **size** M of the **plateau** or of the **neighborhood** is set to 12 accepted solutions. The algorithm is **stopped** if the best current solution is not improved for 3 consecutive plateaus.

5 Computational Results

The purpose of the computational results is twofold. First, we evaluate the performance of GAn. Second, we assess the need for combining the COP and TDL problems into a single CT problem. GAn and the sequential TDL algorithm are coded in `Fortran` and run on a `Pentium IV,` 1.7 GHz and 256 Mb of RAM.

The computational results for the instances of Table 1 are displayed in Table 2. Columns 1 and 2 display the instance and its corresponding optimal length. Columns 3 - 5 report the deviation from L^* of the SA, GA and GAn solutions. Δ_{SA} is computed based on the best solution obtained when replicating SA six times with each replication initiated with a randomly generated solution.

Table 3. Industrial problems

Instance	S	$\|G\|$	W (m)	q_s	$L*$	$\underline{h}$	$\overline{h}$	$\overline{l}$
I_1	4	21	1.5	10-20-20-10	144.00	5	30	12.00
I_2	3	21	1.5	6-12-12	78.00	5	30	12.00
I_3	4	18	1.1	6-24-24-12	87.8	5	20	8.00

Table 4. Computational results for the industrial problems

Instance	$L*$	Δ_{SA}	Δ_{GA}	Δ_{GAn}
I_1	144.00	0	0	5
I_2	78.00	5	7	11
I_3	87.00	0	1	4

Our experimentation shows that SA's solution is not very sensitive to the initial solution. Yet, starting SA from a very bad solution will not lead to a global optimum. Furthermore, increasing GA's population size does not necessarily improve the quality of GA's solution. GA's local optimum is generally obtained randomly during the initial population or is the result of crossover; i.e., it is the result of diversification rather than of intensification. Starting GA with a population whose individuals are of high quality leads to premature stagnation; hindering the escape from local optima. These conclusions highlight the need for diversification and intensification of the search as in GAn.

The results of Column 4 are consistently better than those reported in Column 3, demonstrating the need for the intensification search. (Note that both GA and GAn were run with an identical population size and number of generations.)

Second, we assess the performance of the proposed approach on real life industrial problems where the garments are a collection of non-convex irregular pieces. Problems $I1-I3$, summarized in Table 3, were obtained from a medium-sized local apparel manufacturer along with the "adopted" cut order plan and the corresponding layout for each of its sections. Column 1 of Table 3 displays the instance number. Columns 2-4 indicate the number of ordered sizes n, the number of pieces per garment $|G|$, and the width of the fabric W. Columns 5 and 6 display respectively the ordered quantity per size $q_s, s \in S$, and the best known solution L^*. Finally, columns 7-9 display respectively $\underline{h}, \overline{h}$, and $\overline{l}$ the minimum and maximum ply height and the maximum length of a section.

The computational results for the industrial instances are reported in Table 4. Column 2 indicates the best known length L^*. Column 3 tallies Δ_{SA}, the minimum percent improvement of total length when SA is replicated six times starting each replication with a random initial solution. Column 4 gives Δ_{GA}, the percent improvement of total length when GA is applied without any intensification strategy. Finally, Column 5 displays Δ_{GAn}, the percent improvement of the total length when GAn is applied. The results show that the solutions

obtained by human experts/specialized software can be improved. They further demonstrate the need to combine COP and TDL into a single problem. Finally, they emphasize the role of hybridization in refining the search.

6 Conclusion

This paper tackles a real life problem that is of sizeable importance to the apparel manufacturing industries. This problem consists of finding the optimal cut order plan for a given order subject to the industrial set of constraints such as maximum and minimum ply height, maximum length of a section, etc. The problem is solved using a hybrid heuristic which diversifies the search by undertaking a global search via genetic algorithms and intensifies the search using local search via simulated annealing. The computational results highlight the need for hybridization, and assesses the percent improvements that industry can incur by adopting the proposed approach.

References

1. Blazewicz, J., Hawryluk, P., Walkowiak, R.: Using a tabu search approach for solving the two dimensional irregular cutting problem, Annals of Operations Research 41, 313–325 (1994)
2. Dowsland, K.A., Dowsland, W.: Solution approaches to irregular nesting problems. European Journal of Operational Research 84, 506–521 (1995)
3. Li, Z., Milenkovic, V.: Compaction and separation algorithms for non-convex polygons and their applications. European Journal of Operational Research 84, 539–561 (1995)
4. Dighe, R., Jakiela, M.J.: Solving pattern nesting problems with GA employing task decomposition and contact detection. Evolutionary Computation 3, 239–266 (1996)
5. ElOmri, Méthode d'optimisation dans un contexte productique, Ph.D. Dissertation, Université de Bordeaux1 (1992)
6. Hifi, M., M'Hallah, R.: A best-local position procedure-based heuristic for the two-dimensional layout problem. Studia Informatica Universalis. International Journal on Informatics (Special Issue on Cutting, Packing and Knapsacking) 2(1), 33–56 (2002)
7. Jacobs-Blecha, C., Ammons, J.C., Schutte, A., Smith, T.: Cut order planning for apparel manufacturing. IIE Transactions 30, 79–90 (1996)
8. Jakobs, S.: On the genetic algorithms for the packing of polygons. European Journal of Operational Research 88, 165–181 (1996)
9. M'Hallah, R., Bouziri, A., Jilani, W.A.: Layout of Two Dimensional Shapes Using Genetic Algorithms. In: Del Pobil, A.P., Mira, J., Ali, M. (eds.) Lecture Notes in Artificial Intelligence, subseries of Lecture Notes in Computer Science, pp. 403–411 (2001)
10. Talbi, E.G.: A Taxonomy of Hybrid Metaheuristics. Journal of Heuristics 8, 541–564 (2002)