International Journal of Production Research
Vol. 48, No. 20, 15 October 2010, 6061–6084

A hybrid approach for packing irregular patterns using evolutionary strategies and neural network

W.K. Wong* and Z.X. Guo

Institute of Textiles and Clothing, The Hong Kong Polytechnic University, Hunghom, Kowloon, Hong Kong

(*Received 2 February 2009; final version received 7 August 2009*)

Packing problems can be found in various industries. Packing regular shapes (patterns) is common in wood, glass, and paper industries while packing irregular shapes can be found in metal, clothing and leather industries. It is obvious that irregular objects packing problems are more complex than regular ones due to the geometrical complexity of irregular shapes. Relatively few scientific approaches have been developed to solve irregular objects packing problems although the effectiveness of the packing approaches determines the industrial resource utilisation. This study constructs an irregular object packing approach that integrates a grid approximation-based representation, a learning vector quantisation neural network (NN), a heuristic placement strategy and an integer representation-based $(\mu + \lambda)-$ evolutionary strategy (ES) to obtain an efficient placement of irregular objects. Real data from industry is used to demonstrate the performance of the proposed methodology through various experiments, and the results are compared with those obtained by a genetic algorithm-based packing approach, and those generated from industrial practice. The results demonstrate the effectiveness of the proposed approach.

Keywords: irregular objects packing; evolutionary strategies; neural network

1. Introduction

Packing problems are combinatorial optimisation problems that concern the allocation of multiple objects (patterns) in a large containment region without overlap, and the objective of the allocation process is to maximise the occupied space and minimise the 'wasted' space. In the literature, there are many approaches to tackling different packing problems, such as those based on the concept of 'no-fit polygons' (NFP) (Li and Milenkovic 1995, Stoyan *et al*. 1996, Oliveira *et al*. 2000, Bennell *et al*. 2001, Gomes and Oliveira 2002), methods of bottom-left (BL) placement strategy (Dowsland and Dowsland 1995, Oliveira *et al*. 2000) and those based on linear programming compaction methods (Li and Milenkovic 1995, Stoyan *et al*. 1996, Bennell and Dowsland 2001, Gomes and Oliveira 2006). In recent years, following the concept of phi-function proposed in Stoyan (1980), Stoyan *et al*. constructed mathematical models of two- or three-dimensional packing problems as problems of mathematical programming to seek their local and global optimisation solutions (Stoyan *et al*. 2002, Scheithauer *et al*. 2005, Bennell *et al*. 2008, Stoyan and Chugay 2009). It was reported that the phi-function based techniques showed

*Corresponding author. Email: tcwongca@inet.polyu.edu.hk

ISSN 0020–7543 print/ISSN 1366–588X online

DOI: 10.1080/00207540903246631
http://www.informaworld.com

superior performance to NFP-based techniques. In a landmark paper, Burke *et al.* (2006) presented a new bottom-left-fill heuristic algorithm, which integrated a geometrical definition, a new technique of primitive overlap resolution, with hill climbing and tabu local search methods, for the two-dimensional irregular stock-cutting problem. Their experimental results on a wide range of benchmark problems showed that the new bottom-left-fill heuristic algorithm outperformed the other techniques of the previous studies.

It is well known that packing problems are combinatorial optimisation problems with a very large search space. In order to search for their global optimal solutions, mathematical programming techniques as a rule search for a huge number of local extrema and it takes a lot of computational time. Various meta-heuristic algorithms have been adopted as optimisation tools to find good solutions fast. However, it very often leads to the sacrifice of high performance results. These meta-heuristic approaches include simulated annealing (Oliveira and Ferreira 1993, Heckmann and Lengauer 1995, Burke and Kendall 1999, Gomes and Oliveira 1999, 2006, Wu *et al.* 2003), tabu search (Bennell and Dowsland 1999, 2001, Blazewicz *et al.* 1993), neural networks (Han and Na 1996, Wong 2003a, Au *et al.* 2006, Wong *et al.* 2006, 2009, Yuen *et al.* 2009a,) and genetic algorithms (GAs) (Ismail and Hon 1992, Fujita *et al.* 1993, Bounsaythip *et al.* 1995, Jakobs 1996, Bounsaythip and Maouche 1997, Jain and Gea 1998, Hopper 2000, Wong *et al.* 2000, Babu and Babu 2001, Hifi and Hallah 2003, Wong 2003b, Song *et al.* 2006, Guo *et al.* 2008a, 2008b, Yuen *et al.* 2009b). Among these approaches, the genetic algorithm is the most popular technique to solve irregular objects packing problems (Hifi and Hallah 2003).

Applications of genetic algorithms to irregular objects packing problems based on geometric representation have been extensively studied. For packing approaches based on geometric representation, irregular objects are represented by polygons that are composed of a list of vertices. For instance, Fujita *et al.* (1993) developed an order-based genetic algorithm in combination with local minimisation to solve convex polygons packing problems. Jakobs (1996) also used an order-based genetic algorithm to solve polygon packing problems. Bounsaythip and Maouche (1997) provided a binary tree approach for packing problems in the textile industry. When adopting the above approaches, polygons were circumscribed by their bounding rectangles. In the packing process, low-level routines were adopted to find the smallest enclosing rectangle of the cluster using a special encoding technique (Bounsaythip *et al.* 1995), which describes the contour of a polygon relative to the enclosing rectangle by a set of integer values. Hopper (2000) proposed a genetic algorithm in combination with a bottom-left algorithm to solve both orthogonal and irregular nesting problems. Hifi and Hallah (2003) developed an approach which consists of a constructive heuristic and a hybrid genetic algorithm-based heuristic to two-dimensional layout problems for cases of regular and irregular shapes.

As reviewed in the previous paragraph, there are numerous approaches based on computational geometric description giving good performance. Nevertheless, it is hard to implement them due to their computational complexity for large and complex data sets. In order to overcome the drawback, a digitised representation approach called grid approximation (Ismail and Hon 1992) was adopted and objects were represented by two-dimensional matrices. There are two advantages over the geometric representation: the first advantage is that there is no need to introduce additional routines to identify enclosed areas in objects, and the second one is that it is easier to detect overlap.

Although grid approximation has advantages, irregular objects packing based on grid approximation is a complex task. As a result, very few attempts to develop efficient packing methods based on grid approximation for irregular objects have been reported in the literature. In Ismail and Hon's (1992) study, rectilinear shapes were digitised and represented as a two-dimensional grid array. A multi-parameter binary string including relative positions of a shape was used to indicate shape sequences. The traditional single-point crossover operator and the basic gene-alter mutation operator (Goldberg 1989) were adopted to generate new offspring. However, applying such genetic operators to the data structure may cause infeasible solutions (i.e., overlap). In view of the deficiencies of Ismail and Hon‘s (1992) method, Jain and Gea (1998) designed a new concept of a 2D genetic algorithm chromosome as a two-dimensional matrix to describe the complete layout. Crossover and mutation operators were modified to suit this 2D genetic algorithm chromosome while a new genetic operator called compaction was developed to increase the density of the layout. Nevertheless, this special encoding approach results in a very long parent chromosome and leads to a very extensive computation when it is applied to packing a large number of objects. Hence, it is impractical to implement Jain and Gea's (1998) algorithm for large-scale problems.

Although evolutionary strategy, like GAs, is also a kind of powerful evolutionary algorithm that has been used successfully in solving various engineering problems (Quagliarella *et al.* 1995) and usually shows faster convergence speed than GAs do (Hoffmeister and Bäck 1991), it has not been investigated and used to solve packing and nesting problems in the current literature.

It is desirable to investigate the performance of an evolutionary strategy based on grid approximation for the irregular packing problem. In this study, a new hybrid approach was developed which combines a $(\mu + \lambda)-$ evolutionary strategy, a learning vector quantisation neural network, a grid approximation representation and a heuristic two-stage placement strategy, to increase the usability of the stock sheet. A $(\mu + \lambda)-$ evolutionary strategy is used to determine the packing information (i.e., the packing sequence of packing cells, objects orientation, and packing rules selection), in which an integer representation is adopted to obtain higher computational efficiency than the 2D genetic chromosome in Jain and Gea's (1998) study. A learning vector quantisation neural network was also developed by a set of examples inspired by experienced packing planners to diminish the size of a search space by dividing the objects into three classes. A grid approximation representation technique was also employed to represent any shaped objects including convex and concave. In contrast to the geometric algorithms reported in previous research studies, grid approximation simplifies the calculation process, and thus it is easier to judge whether objects overlap. A two-stage placement strategy was proposed to ameliorate the shortcomings of packing approaches based on enclosing rectangles.

The remainder of the paper is organised as follows. A brief description of irregular objects packing problems is given and a new heuristic placement method is presented in detail in Section 2. A $(\mu + \lambda)-$ evolutionary strategy is used to determine the packing sequence of packing cells in Section 3. The effectiveness of the proposed methodology is illustrated in Section 4. Conclusions are summarised in Section 5.

2. Packing method

The problem addressed in this study is to pack a set of irregular objects $\{p_1, p_2, \ldots, p_n\}$ onto a stock sheet of infinite length C_L and fixed width C_H without overlap. Hence, a general methodology which integrates a grid approximation-based heuristic placement approach, a learning vector quantisation neural network, and a $(\mu+\lambda)-$ evolutionary strategy is developed to obtain a packing pattern with the minimal length. In this case, the following assumptions are taken into consideration to construct the methodology:

(1) The stock sheet is a rectangle with a fixed width and an infinite length.
(2) Each object has only two orientations: 0° and 180° since this study focuses on marker planning process of the clothing industry. That is to say, the original object and the object obtained by a 180° anticlockwise rotation are allowed while an object is packed onto the stock sheet.
(3) The length and the width of each object are not larger than the size of the stock sheet.
(4) Each object can be placed at any position on the stock sheet.

2.1 *Object representation*

In this study, the digitised representation technique, grid approximation, proposed by Ismail and Hon (1992) was used to represent objects in any shapes including convex and concave. In contrast to geometric algorithms, the major advantage of the grid approximation is that it is easier to detect overlap. By using this technique, each object is divided into a finite number of equalised cells, and the size of a selected cell is small enough to represent the objects. $P_L^{(i)}$ and $P_H^{(i)}$ denote the length and the width of an enclosing rectangle corresponding to the object p_i. $R_x^{(i)}$ denotes the length of a cell, and $R_y^{(i)}$ denotes the height of a cell for the object p_i. (In this paper, $R_x^{(i)}=1\,\text{mm}$, and $R_y^{(i)}=1\,\text{mm}$.) The object with a two-dimensional matrix of size $A_H^{(i)}\times A_L^{(i)}$ is represented as follows:

$$A^{(i)}=\begin{pmatrix} a_{11}^{(i)} & a_{12}^{(i)} & \cdots & a_{1A_L^{(i)}}^{(i)} \\ a_{21}^{(i)} & a_{22}^{(i)} & \cdots & a_{2A_L^{(i)}}^{(i)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{A_H^{(i)}1}^{(i)} & a_{A_H^{(i)}2}^{(i)} & \cdots & a_{A_H^{(i)}A_L^{(i)}}^{(i)} \end{pmatrix},$$

where:

$$A_H^{(i)}=\frac{P_H^{(i)}}{R_y^{(i)}} \quad \text{and} \quad A_L^{(i)}=\frac{P_L^{(i)}}{R_x^{(i)}}.$$

For each entry,

$$a_{px,py}^{(i)}=\begin{cases} 1, & \text{if pixel } (px,py) \text{ is occupied} \\ 0, & \text{otherwise} \end{cases}.$$

In addition, each object examined in this study had only two orientations: 0° and 180°. The matrix representation of the rotated object (180° anticlockwise rotation) was obtained by simply modifying the matrix of the original object shown in the above-mentioned equation. Then the matrix of the rotated object becomes:

$$A^{(i)}_{\text{rotate}} = \begin{pmatrix} a^{(i)}_{A^{(i)}_H A^{(i)}_L} & \cdots & a^{(i)}_{A^{(i)}_H 2} & a^{(i)}_{A^{(i)}_H 1} \\ \vdots & \ddots & \vdots & \vdots \\ a^{(i)}_{2A^{(i)}_L} & \cdots & a^{(i)}_{22} & a^{(i)}_{21} \\ a^{(i)}_{1A^{(i)}_L} & \cdots & a^{(i)}_{12} & a^{(i)}_{11} \end{pmatrix}_{A^{(i)}_H \times A^{(i)}_L}.$$

Similar to the object representation, the stock sheet with an infinite length and a fixed width was discretised into a finite number of equi-sized cells of size $R_x \bullet R_y$. Hence, the stock sheet with the length C_L and the width C_H were characterised by a matrix $\boldsymbol{U}$ of size $U_H \times U_L$ as follows:

$$U = [u_{px,py}],$$

where:

$$U_H = \frac{C_H}{R_y} \quad \text{and} \quad U_L = \frac{C_L}{R_x}.$$

For each entry:

$$u_{px,py} = \begin{cases} 1, & \text{if pixel } (px, py) \text{ is occupied} \\ 0, & \text{otherwise} \end{cases}.$$

2.2 *Heuristic placement approach*

The architecture of the proposed heuristic placement approach is shown in Figure 1. Firstly, the grid approximation is used to represent any shaped objects in two-dimensional matrixes. Secondly, a learning vector quantisation neural network is developed as a classification heuristic to divide the objects into three classes according to their relative sizes: BIG, SMALL and OTHER. Thirdly, an evolutionary algorithm is used to determine the packing information (i.e., the packing sequence of packing cells, objects orientation, and packing rules selection). Finally, a two-stage placement strategy is proposed to the construction of a packing pattern according to packing information, which is defined by the evolutionary strategy. Objects in BIG and OTHER classes are packed onto the stock sheet according to the packing sequence of packing cells strings and packing rules selection strings defined by the evolutionary strategy. That is to say, the objects in the packing cells are placed by selecting rules from the 16 packing rules shown in Figure 2, which are acquired by pattern planning experts through in-depth interviews with experienced pattern planners on the reference sites. Objects in the SMALL class are packed onto the stock sheet according to the packing sequence of packing cells strings and objects orientation strings defined by the evolutionary strategy. In other words, the objects might be rotated (180° anticlockwise rotation).

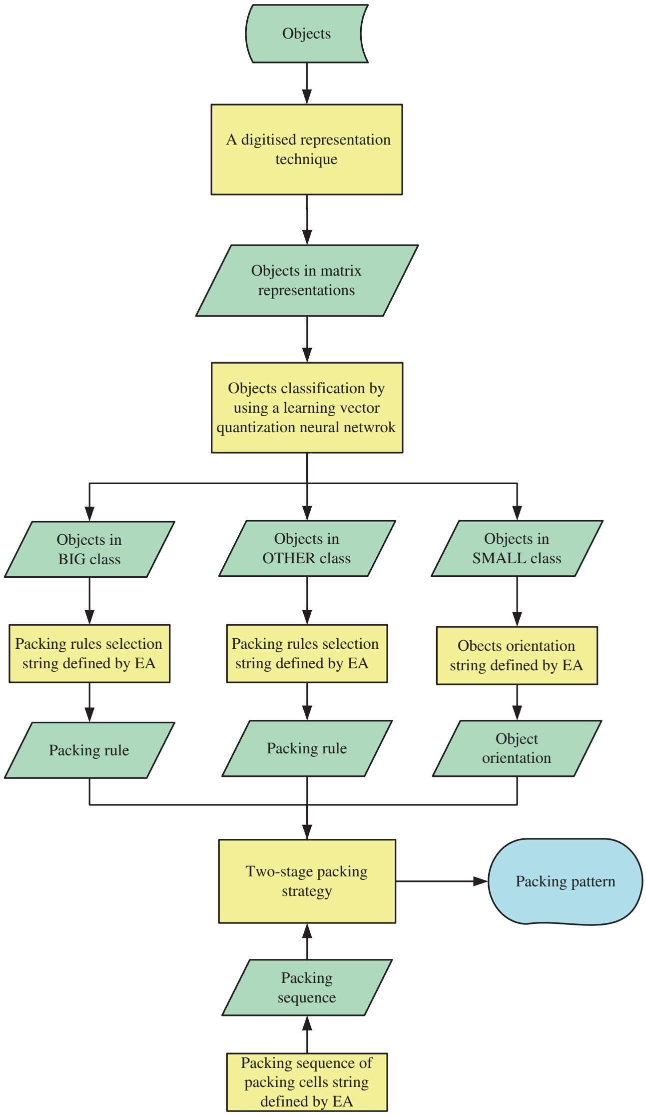


Figure 1. System architecture of packing methods.

2.2.1 *Object classification*

A learning vector quantisation neural network (Kohonen 1990) is developed as a classification heuristic. The proposed network is trained by a set of examples inspired by experienced packing planners to diminish the size of a search space by dividing the objects into three classes according to their relative sizes: BIG, SMALL and OTHER. Once the network has been trained, it has the ability to classify various other kinds of objects that are similar to the training set, which makes the network powerful. For instance, according to the packing planners' experience, if the size of an object in the BIG class is three times larger than the size of an object in the SMALL class, and the length of an object in the OTHER class is four times larger than the width of an object in the OTHER class, BIG,

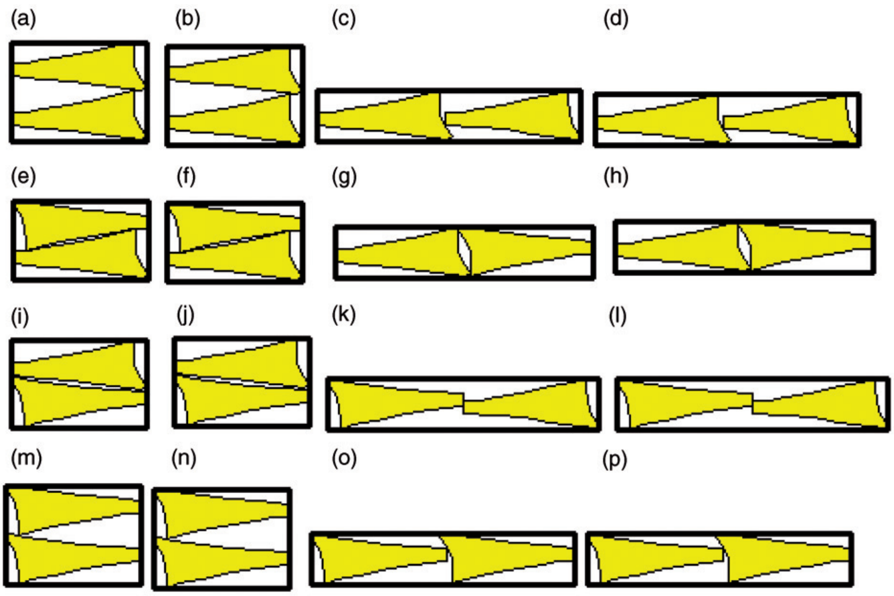


Figure 2. Packing rules: (a) object 1 top, object 2 bottom; (b) object 2 top, object 1 bottom; (c) object 1 left, object 2 right; (d) object 2 left, object 1 right; (e) object 1 top, anticlockwise rotate 180°, object 2 bottom; (f) object 2 top, anticlockwise rotate 180°, object 1 bottom; (g) object 1 left, object 2 right, anticlockwise rotate 180°; (h) object 2 left, object 1 right, anticlockwise rotate 180; (i) object 1 top, object 2 bottom, anticlockwise rotate 180°; (j) object 2 top, object 1 bottom, anticlockwise rotate 180°; (k) object 1 left, anticlockwise rotate 180°, object 2 right; (l) object 2 left, anticlockwise rotate 180°, object 1 right; (m) object 1 top, anticlockwise rotate 180°, object 2 bottom, anticlockwise rotate 180°; (n) object 2 top, anticlockwise rotate 180°, object 1 bottom, anticlockwise rotate 180°; (o) object 1 left, anticlockwise rotate 180°, object 2 right, anticlockwise rotate 180°; (p) object 2 left, anticlockwise rotate 180°, object 1 right, anticlockwise rotate 180°.

OTHER and SMALL classes are classified. Without using a neural network, the experienced parameters such as three times and four times should be input into the system manually according to the packing planners' experience. That is to say, before using a neural network, the classification is based on the analysis of a great number of objects in practice. After the network has been trained by a large number of examples, instead of using packing planners' experience, the objects can be classified by their relative sizes automatically.

BIG class is a class of bigger objects, while SMALL class is a class of smaller objects (i.e., the size of an object in the BIG class is a multiplication of the size of an object in the SMALL class). On the other hand, OTHER class is a class of objects that are very long but narrow or vice versa. Objects in BIG class and OTHER class are paired up to form packing cells. That is to say, each packing cell contains two objects that have the same or similar size. At the same time, each object in SMALL class generates a single packing cell. The objects packing sequence has thus been changed into the packing cells packing sequence, which decreases the size of the search space. For instance, it is assumed that the number of packed objects is 64 and the size of the search space is 64. However, after the procedure of object classification, if the number of objects in BIG, SMALL and OTHER classes is 20, 8 and 36, respectively, then the size of the search space is reduced to 36.

The key steps of the learning vector quantisation neural network approach are presented below:

Step 0: Initialise reference vectors, weight vectors, and learning rate $\alpha(0)$.

Step 1: While the stopping condition is false, do Steps 2–6.

Step 2: For each training input vector (i.e., the area of each piece and the narrow factor of each piece), do Steps 3–4.

Step 3: Find J so that the Euclidean distance between the input vector and the weight vector for the jth output unit is a minimum.

Step 4: Update the weight vector w_J as follows:

$$\begin{aligned} &\text{if } T = C_J, \quad \text{then } w_J(\text{new}) = w_J(\text{old}) + \alpha(X - w_J(\text{old}));\\ &\text{if } T \neq C_J, \quad \text{then } w_J(\text{new}) = w_J(\text{old}) - \alpha(X - w_J(\text{old})), \end{aligned}$$

where X denotes the training vector, T denotes the correct class for the training vector, and C_J denotes the class represented by the jth output unit.

Step 5: Reduce learning rate α.

Step 6: Test the stopping condition, which may specify a fixed number of iterations or the learning rate reaching a sufficiently small value.

2.2.2 *Two-stage placement strategy*

A two-stage placement strategy is proposed as an alternative to construct a packing pattern according to the packing information (i.e., the packing sequence of packing cells, objects orientation, and packing rules selection), which is defined by the evolutionary strategy. In this case, the enclosing rectangles of the packing cells are first examined, and then the packing cells are compacted directly. In particular, instead of implementing the compaction routine in a single step after all the enclosing rectangles of the packing cells are allocated, the compaction routine is done when each enclosing rectangle is placed. The advantage of this compaction routine is the ability to obtain a tight packing pattern providing more space for the coming packing cells. It is obvious that the two-stage placement strategy improves the packing pattern quality without compromising the computational effort. The key steps of the two-stage placement strategy are presented as follows:

Step 1: Place the coming packing cell $C_{i_{j+1}}$ at the uppermost and infinite right corner of the stock sheet. Due to the approximation of the packing cell by its enclosing rectangle at the first stage, the matrix of the stock sheet becomes:

$$U(j+1) = U(j) + \begin{pmatrix} 0 & 0 & \cdots & A^{(i_{j+1})} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}_{U_H \times U_L} = \begin{pmatrix} 0 & 0 & \cdots & A^{(i_{j+1})} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ UB^{(i_j)} & 0 & \cdots & 0 \end{pmatrix}_{U_H \times U_L},$$

with sub-matrices:

$$A^{(i_{j+1})} = \left(a^{i_{j+1}}_{px,py}\right)_{A^{(i_{j+1})}_H \times A^{(i_{j+1})}_L} \quad \text{and} \quad UB^{(i_j)} = \left(ub^{(i_j)}_{px,py}\right)_{B^{(i_j)}_H \times B^{(i_j)}_L},$$
$$\text{where} \quad a^{(i_{j+1})}_{px,py} = 1 \quad \text{and} \quad ub^{(i_j)}_{px,py} = u_{px,py}.$$

Step 2: Shift the packing cell $C_{i_{j+1}}$ leftward and downward until it meets other packing cells and cannot be moved again. In view of the property of matrices, it is convenient to shift the packing cell by counting the empty cells in the matrix. Then the matrix of the stock sheet becomes:

$$U(j+1) = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ A^{(i_{j+1})} & 0 & \ddots & 0 \\ UB^{(i_j)} & 0 & \cdots & 0 \end{pmatrix}_{U_H \times U_L}.$$

Step 3: Represent the packing cell $C_{i_{j+1}}$ at the second stage by using its enclosing rectangle without approximating it, and then the matrix of the stock sheet is:

$$U(j+1) = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ UA^{(i_{j+1})} & 0 & \ddots & 0 \\ UB^{(i_j)} & 0 & \cdots & 0 \end{pmatrix}_{U_H \times U_L},$$

with sub-matrices:

$$UA^{(i_{j+1})} = \left(ua^{i_{j+1}}_{px,py}\right)_{A^{(i_{j+1})}_H \times A^{(i_{j+1})}_L}$$
$$\text{where} \quad ua^{(i_{j+1})}_{px,py} = u_{px,py}.$$

Step 4: Compact the packing cells by removing the vacant cells between these two matrices of packing cells, then the matrix of the stock sheet becomes:

$$U(j+1) = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \ddots & 0 \\ UB^{(i_{j+1})} & 0 & \cdots & 0 \end{pmatrix}_{U_H \times U_L},$$

with sub-matrices:

$$UB^{(i_{j+1})} = \left(ub^{(i_{j+1})}_{px,py}\right)_{UB^{(i_{j+1})}_H \times UB^{(i_{j+1})}_L},$$
$$\text{for each entry} \quad ub^{(i_{j+1})}_{px,py} = u_{px,py}.$$

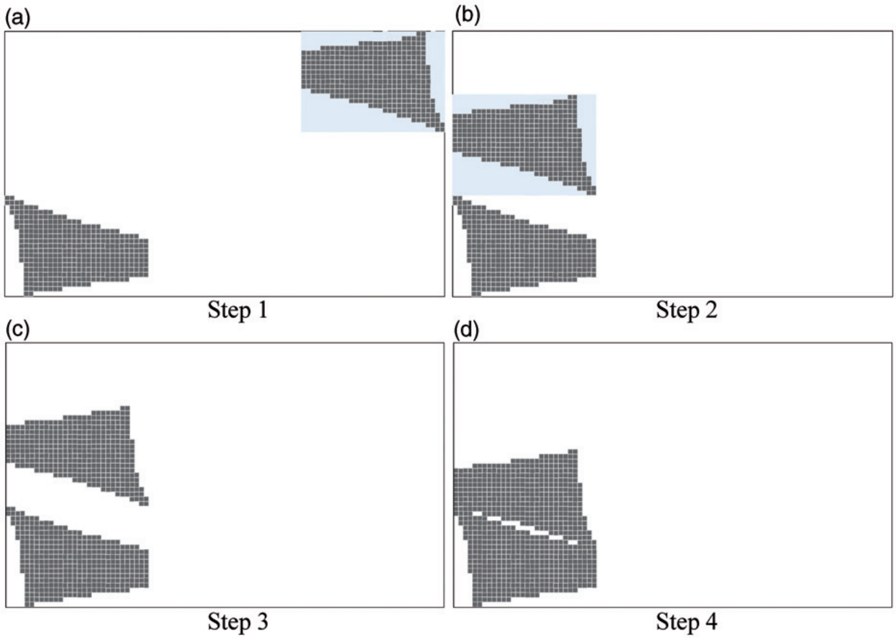


Figure 3. Procedure of two-stage placement strategy: the object at the top right in Step 1 represents the coming packing cell $C_{i_{j+1}}$.

Furthermore, $UB_H^{(i_{j+1})}$ and $UB_L^{(i_{j+1})}$ satisfy the following conditions:

$$\begin{cases} UB_H^{(i_{j+1})} \le UB_H^{(i_j)} + UA_H^{(i_{j+1})} \\ UB_L^{(i_{j+1})} = \max\left\{UB_L^{(i_j)}, UA_L^{(i_{j+1})}\right\} \end{cases}.$$

An example of how the objects are placed according to the two-stage placement strategy is shown in Figure 3.

3. Evolutionary strategy

For this study, the $(\mu + \lambda)-$ evolutionary strategy (ES) was adopted. In contrast to the elitist strategy of genetic algorithms, with the aid of the $(\mu + \lambda)-$ ES, parents survive until they are superseded by better offspring (Bäck *et al.* 1997). The following notation is used to facilitate the presentation:

$\mu =$ the population size of parents;
$\lambda =$ the population size of offspring;
$s_k = k$th individual in the individual space;
$f(s_k) =$ the fitness value of individual $s_k (k = 0, 1, 2, \ldots, \mu + \lambda - 1)$;
$t =$ generation index $(t = 0, 1, 2, \ldots)$;

It is assumed that the current generation is t and the current population is represented by **X(t)**, which is a population of μ individuals, and the general outline of the $(\mu + \lambda)-$ ES is presented below, as illustrated in the block diagram in Figure 4.

Step 1: Set $t = 0$ and generate an initial population of μ individuals randomly.

Step 2: Generate a mating pool by pre-selection (see the selection operation section for details).

Select individuals from the population according to a specified selection operation. The selected individuals are then placed into a mating pool.

Step 3: Perform recombination and mutation.

Pair up the individuals in the mating pool and generate $\lambda(\geq \mu)$ new-born offspring individuals using the operators of recombination and mutation. In this study, each chromosome consists of three portions. For the first portion of the chromosome, discrete recombination operators, repeated exchange mutation operators, and evolutionary inversion mutation operators are employed. For the second portion of the chromosome, traditional gene-alter mutation operators and traditional discrete recombination operators are developed. For the third portion of the chromosome, exchange mutation operators and traditional discrete recombination operators are developed.

Step 4: Create a new population for the next generation by post-selection (see the selection operation section for details).

Select μ best individuals from the combined population of parents (μ individuals) and offspring (λ individuals). All the selected μ individuals are then collected to form a new population known as **X(t + 1)**, which replaces **X(t)** and serves as the population of individuals for the next generation $t + 1$.

Step 5: Check the pre-specified stopping condition. In this case, the pre-specified stopping condition is satisfied when the predefined maximum number of generations is reached or no further increase in the fitness function values of the individuals is obtained. If it is satisfied, terminate the search process, and return to the best solution as the final solution. Otherwise, increase t by 1 and go to Step 2.

3.1 *Structure of the individuals*

Although there are many different representations to implement evolutionary algorithms, the most natural representation for the objects packing problem is integer representation. In this study, each chromosome as shown in Figure 5 consists of three portions: a set of bits in the first portion of the string that are a set of integer numbers to indicate the packing sequence of packing cells, which are shown as $\Omega = (i_1, i_2, \ldots, i_n)$, i: index of the packing cell C_i. The order of a gene in an individual is the order to examine the packing cell that is identified by the gene. A set of bits in the second portion of the string is a set of 0–1 binary decision variables to represent the object orientation (i.e., 0° or 180°) for each object in SMALL class, and a set of bits in the third portion of the string is a set of integer numbers contains information to select packing rules for BIG and OTHER classes.

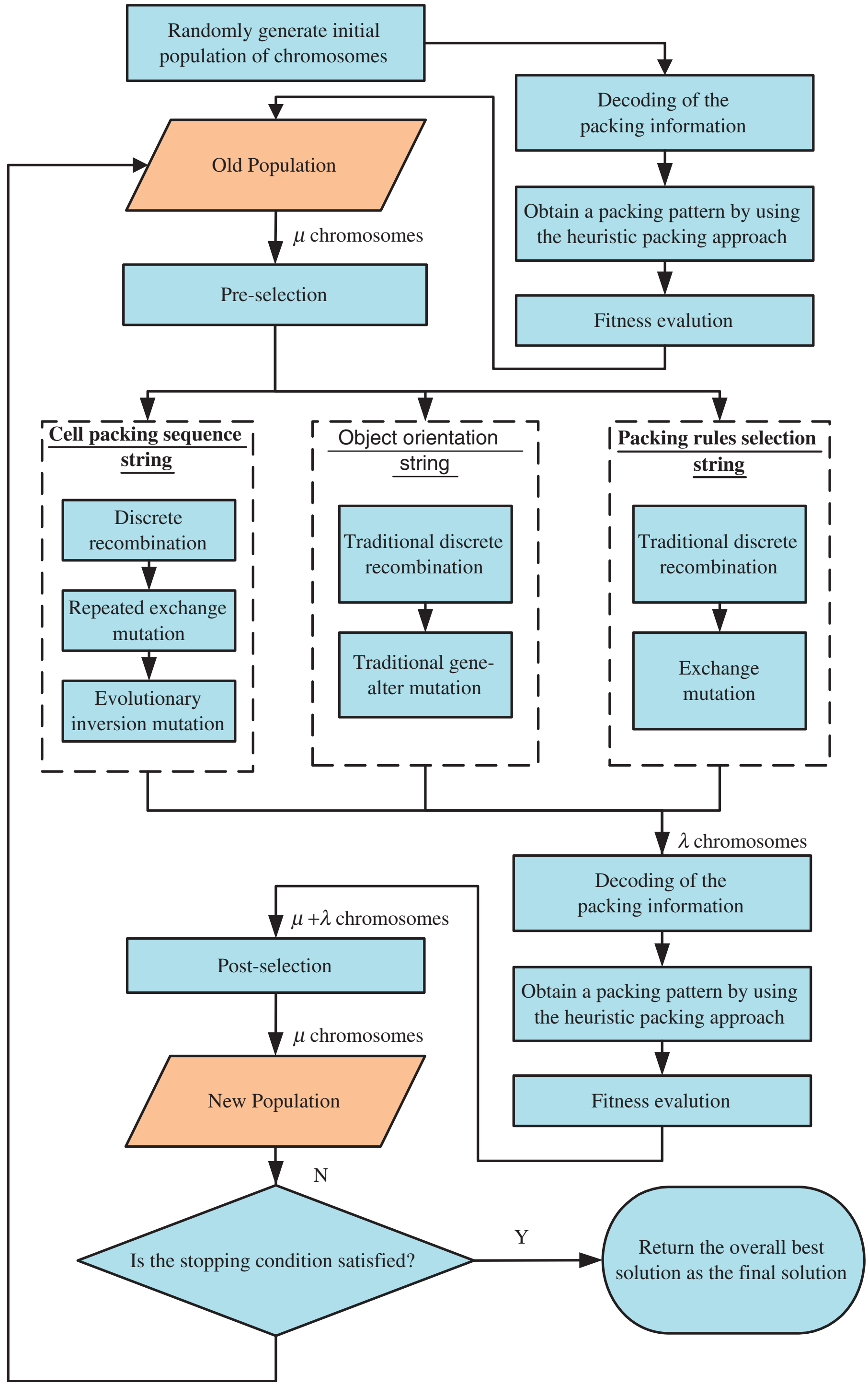


Figure 4. Block diagram of the $(\mu + \lambda)-$ ES.

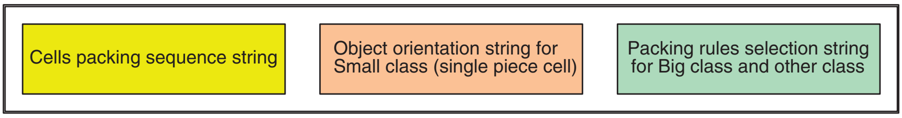


Figure 5. Chromosome structure.

Since factors such as object orientation and packing rules selection in the second and third portions of the string complicate the packing problem, this new chromosome structure could prevent potential or even detrimental squashing of the solution space. The length of the new chromosome is $3N$, where N is the number of cells to be packed.

3.2 *Selection operation*

In this study, two selection schemes, pre-selection and post-selection, are implemented. The pre-selection scheme is stochastic, whilst the post-selection scheme is deterministic. For the pre-selection operation shown in Figure 4, one of the most well-known selection schemes called 'biased roulette wheel scheme' (Goldberg 1989) was used. The probability of selecting an individual s_k from the current population $\mathbf{X(t)}$ is given by the following equation:

$$P_{\text{select}} = \frac{f(s_k)}{\sum_{k=0}^{\mu-1} f(s_k)}.$$

In any generation, the individuals are selected by their respective selection probabilities governed by the above-mentioned equation. If the individual s_k represents a candidate solution, then the fitness function is $f(s_k) = 1/C_L$. Therefore, the candidate solutions with lower objective function values have higher selection probabilities. Through this connection, the optimal objective function value can be obtained by maximising the fitness function values of the individuals. When the pre-selection process is completed, the individuals in the mating pool will then be paired up to generate λ new offspring by recombination and mutation operations.

In the case of the post-selection operation in Figure 4, the combined population of parents (μ individuals) and offspring (λ individuals) are sorted by the fitness function values. The μ best individuals with higher fitness function values will survive whereas the λ remaining individuals with lower fitness function values will be discarded.

3.3 *Recombination operation*

The discrete recombination operator was used in this study. The procedure of the discrete recombination operator for the first portion of the chromosome is presented below:

(1) Select two parents randomly from the mating pool.
(2) Randomly generate a decision string with the same length as the parent chromosomes. Each bit in the decision string can take a value of '1' or '2'. A value of '1' indicates that the corresponding components of the offspring chromosome are copied from the first parent chromosome; otherwise, '2' represents

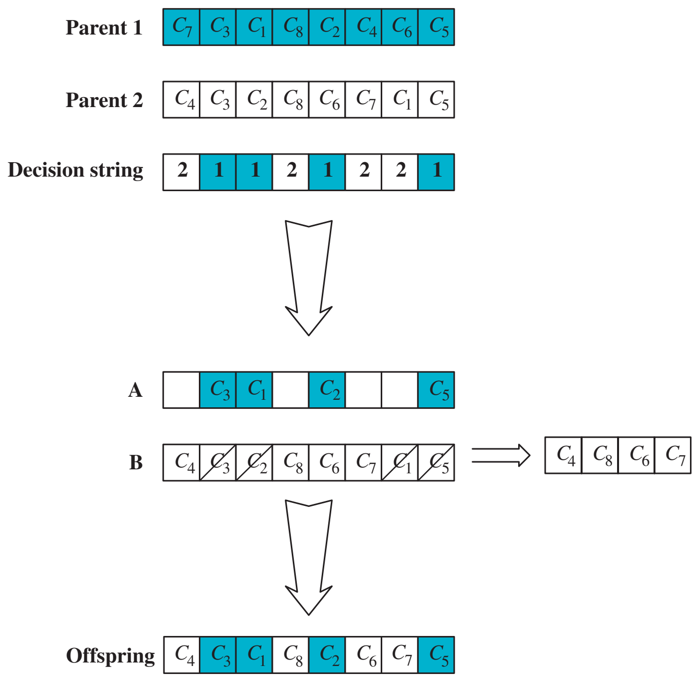


Figure 6. Discrete recombination operator.

that the positions in the offspring chromosome are filled with the elements of the second parent chromosome.

(3) Fill some positions with the offspring chromosome by copying corresponding elements of the first parent chromosome associated with a '1' in the decision string. That is to say, the same components appear in the same positions in the offspring chromosome as they do in the first parent chromosome.
(4) With reference to the second parent chromosome, the components present in the offspring chromosome are omitted; otherwise, the remaining part is reserved.
(5) The remaining positions in the offspring chromosome are filled with the reserved elements of the second parent chromosome in the same order whenever the decision string contains a '2'.

Consequently, each offspring chromosome consists of two portions: a set of bits in the first portion of the string preserves information from the first parent chromosome, and a set of bits in the second portion of the string incorporates information from the second parent chromosome. Figure 6 illustrates the mechanism of the recombination process graphically.

For the second and third portions of the chromosome, the traditional discrete recombination operator is employed, where each bit is randomly copied from either the first or second parent chromosome.

3.4 *Mutation operation*

After the recombination process is completed, instead of using the traditional gene-alter mutation operation (Goldberg 1989), for the first portion of the chromosome the repeated exchange mutation operation and the evolutionary inversion mutation operation are employed to prevent infeasible solutions in this study. In contrast to the recombination operator, the mutation operator is always regarded as a background operator. However, Bäck *et al*. (1997) suggested that the mutation operator becomes more productive as the ES converges. The repeated exchange mutation operator is used to introduce new schemata into the population in order to prevent premature convergence of the population whilst the evolutionary inversion mutation operator is adopted to manipulate the local search process over the solution space like an uphill-climbing technique to improve the capability of the local search process. The algorithm regulates a balance between the exploration and exploitation of the solution space. The repeated exchange mutation operator has the following procedure:

Step 1: Generate a random integer ω within a range of $[1, l]$ (where l is the length of the chromosome) to determine the number of exchanges.

Step 2: Randomly choose two bits along the string and the two selected bits are exchanged.

Step 3: Iteratively implement Step 2 ω times.

Figure 7(a) shows an illustration of Step 1 of the above-mentioned procedure.

The procedure of the evolutionary inversion mutation operation is outlined below:

Step 1: Set $Loop\_num = 0$. Generate a random integer: θ within a range of $[1, l]$ (where l is the length of the chromosome) to determine the number of loops.

Step 2: Two cutting points are selected randomly along the length of the chromosome. The sub-string between these two cutting points is reversed and the remaining part of the chromosome is preserved.

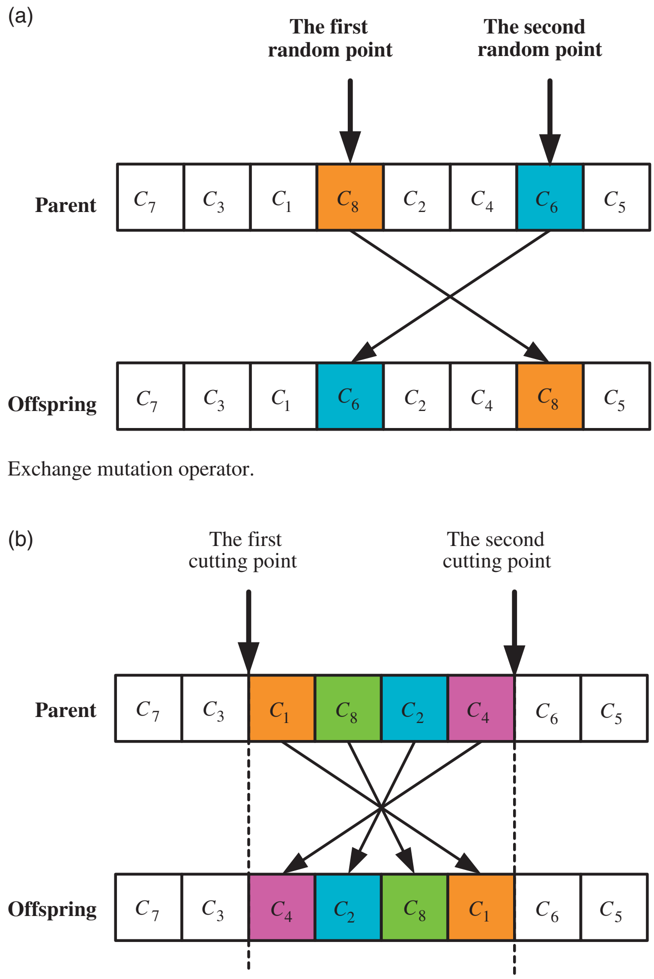
Step 3: If the fitness function value of the newly generated individual is higher than the original one, then the inversion operation in the above-mentioned step is implemented; otherwise, go back to the first step.

Step 4: If $Loop\_num \geq \theta$ is satisfied, terminate the process; otherwise, increase $Loop\_num$ by 1, then go to Step 2.

Figure 7(b) illustrates an example of a simple inversion mutation process presented in Step 2.

For the second portion of the chromosome, the traditional gene-alter mutation operator [14] was adopted. For instance, if an offspring individual is encoded by the binary representation, (0 1 1 0 0 1), then six random numbers ranging from 0.00 to 1.00 are drawn: (0.653, 0.231, **0.007**, 0.014, **0.003**, 0.024). If the mutation rate is 0.01, two random numbers in the above-mentioned array have their values smaller than the mutation rate. These two numbers will trigger the mutation operation to take place in the third and fifth bits of the string. The mutation operator causes the bits to change from 1 to 0 or 0 to 1 whenever the mutation operations are triggered. The resulting individual becomes (0 1 **0** 0 **1** 1).

For the third portion of the chromosome, exchange mutation operator [3] is employed. The procedure of the exchange mutation operator is to randomly choose two bits along the string, and then the two selected bits were exchanged.



Exchange mutation operator.

Inversion mutation operator.

Figure 7. Mutation operators.

4. Experiments and discussion

In this section, eight real examples are used to evaluate the performance of the proposed methodology. First of all, the results of the proposed methodology are compared with those obtained by the genetic algorithm (GA) with the elitist strategy and the heuristic placement (HP) approach (GA + HP approach). The GA + HP approach is the same as the proposed approach except that a GA with elitist strategy is used to replace the ES so that the performance of GA and ES can be compared in the problem investigated. Then the results are also compared with those derived by industrial practice (IP) in order to demonstrate the effectiveness of the proposed methodology.

Table 1 lists six real examples taken from a marker planning process of the clothing industry. In all experiments, the parameters adopted for the evolutionary strategy after testing were $\mu = 50$, $\lambda = 100$, recombination rate = 0.7, mutation rate = 0.03, and maximum number of generations = 500. In addition, the GA with the elitist strategy was also

Table 1. Data sets used in the illustrative examples.

Problem name	Number of objects	Sheet width (inch)
SHIRT1	48	48
SHIRT2	64	48
SHIRT3	80	48
SWIM1	60	60
SWIM2	78	60
SWIM3	108	60

Table 2. Comparison of the offline performance by the proposed approach and the genetic algorithm.

	CGA	ES
Overall best solution among five runs	141.74	138.64
Average of the best solution among five runs	142.17	138.85
Best offline performance among five runs	143.45	138.98
Average offline performance among five runs	144.05	139.57

used to solve the examples for comparison purposes, and the genetic parameters adopted for the GA after testing were population size = 100, crossover rate = 0.7, mutation rate = 0.003, and maximum number of generations = 500. Due to space limitation, only example SWIM3 was used to evaluate the performance of the evolutionary strategy by the offline performance measure:

$$\text{Offline performance measure} = \frac{1}{T}\sum_{t=1}^{T} z_t^*,$$

where z_t is the best objective function value among the candidate solutions in generation t, z_t^* is defined by the equation:

$$z_t^* = \min\{z_1, z_2, \ldots, z_t\}.$$

Table 2 shows that the average objective function value of the final solutions among the five runs, for example SWIM3 is 138.85, which is less than the best solutions obtained by the GA. Table 2 also shows that the proposed algorithm has better offline performance than those of the GA and also outperforms the GA in terms of quality of the final solution. The proposed algorithm is superior to the GA as a function optimiser.

Each example listed in Table 1 was run five times by the ES and the GA while five trials were conducted by five marker planners. Table 3 summarises the best results of the six packed stock sheets, and the results obtained by the proposed approach are marked in bold. The efficiencies of the packing pattern for the proposed approach, the GA + HP approach, and the IP are shown in the third, fourth and fifth columns of Table 3. The efficiency was measured as a quotient between the area of packed objects and the used rectangle area of the stock sheet (Gomes and Oliveira 2006). The results indicate that the proposed methodology improves the efficiency of the packing pattern and shortens the length of the packing pattern. Table 4 shows the details of the improvement percentage of

Table 3. A summary of the results for the eight illustrative examples.

Problem name	Number of objects	Proposed methodology (ES+HP) Sheet length (inch)	Efficiency (%)	GA+HP Sheet length (inch)	Efficiency (%)	IP Sheet length (inch)	Efficiency (%)
SHIRT1	48	**146.60**	**75.91**	146.94	75.74	151.78	73.21
SHIRT2	64	**193.72**	**76.61**	201.32	73.71	203.21	73.03
SHIRT3	80	**243.44**	**76.20**	248.71	74.58	260.69	71.16
SWIM1	60	**92.60**	**58.62**	94.94	57.18	100.30	54.13
SWIM2	78	**122.49**	**58.65**	126.64	56.73	133.06	53.99
SWIM3	108	**138.64**	**57.84**	141.74	56.54	147.85	54.20

Table 4. Method comparisons.

Problem name	Improvement (proposed methodology vs GA+HP) (%)	Improvement (proposed methodology vs IP) (%)
SHIRT1	0.23	3.4
SHIRT2	3.78	4.67
SHIRT3	2.12	6.62
SWIM1	2.46	7.67
SWIM2	3.28	7.94
SWIM3	2.18	6.23

each example. It reveals that the average improvement of the examples is 1.92% for the first comparison in column 2, and 9.99% for the second comparison in column 3. Finally, the packing patterns for each example generated by the proposed methodology, the GA+HP approach, and the IP approach are presented in Figures 8–10, respectively.

5. Conclusion

In this study, a heuristic placement approach based on grid approximation, a learning vector quantisation neural network, and an integer representation-based evolutionary strategy were proposed to establish an effective methodology for solving irregular objects packing problems. This approach has many advantages. Firstly, with the placement approach based on grid approximation, it provides the system designers with an easier way to detect whether overlap occurs. Secondly, the two-stage placement strategy improves the packing pattern quality without compromising the computational effort. Thirdly, the formulation of optimal packing information can be accomplished easily by manipulating the composition of the integer string format. Fourthly, a learning vector quantisation neural network was developed as a classification heuristic to reduce the size of the search space. Fifthly, adding factors such as object orientation and packing rules selection in the second and third portions of the string could prevent potential or even detrimental squashing of the solution space. Finally, the proposed evolutionary strategy can maintain a

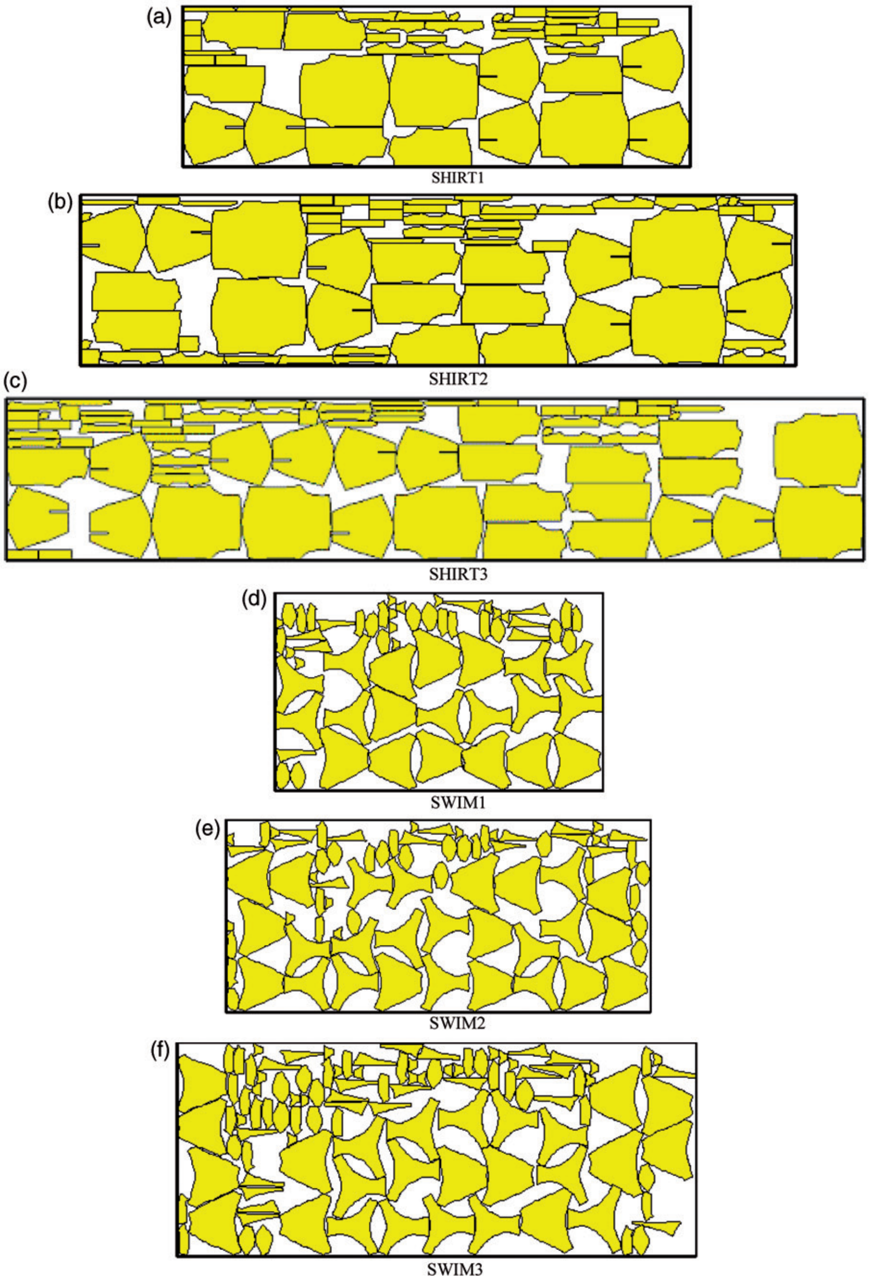


Figure 8. The best packing pattern generated by the proposed approach for the illustrative examples.

better balance between exploitation and exploration of the solution space by generating the evolution of the populations. The effectiveness of the proposed methodology was demonstrated through various experiments, and the results of this methodology are compared with those of the genetic algorithm using the heuristic placement approach and

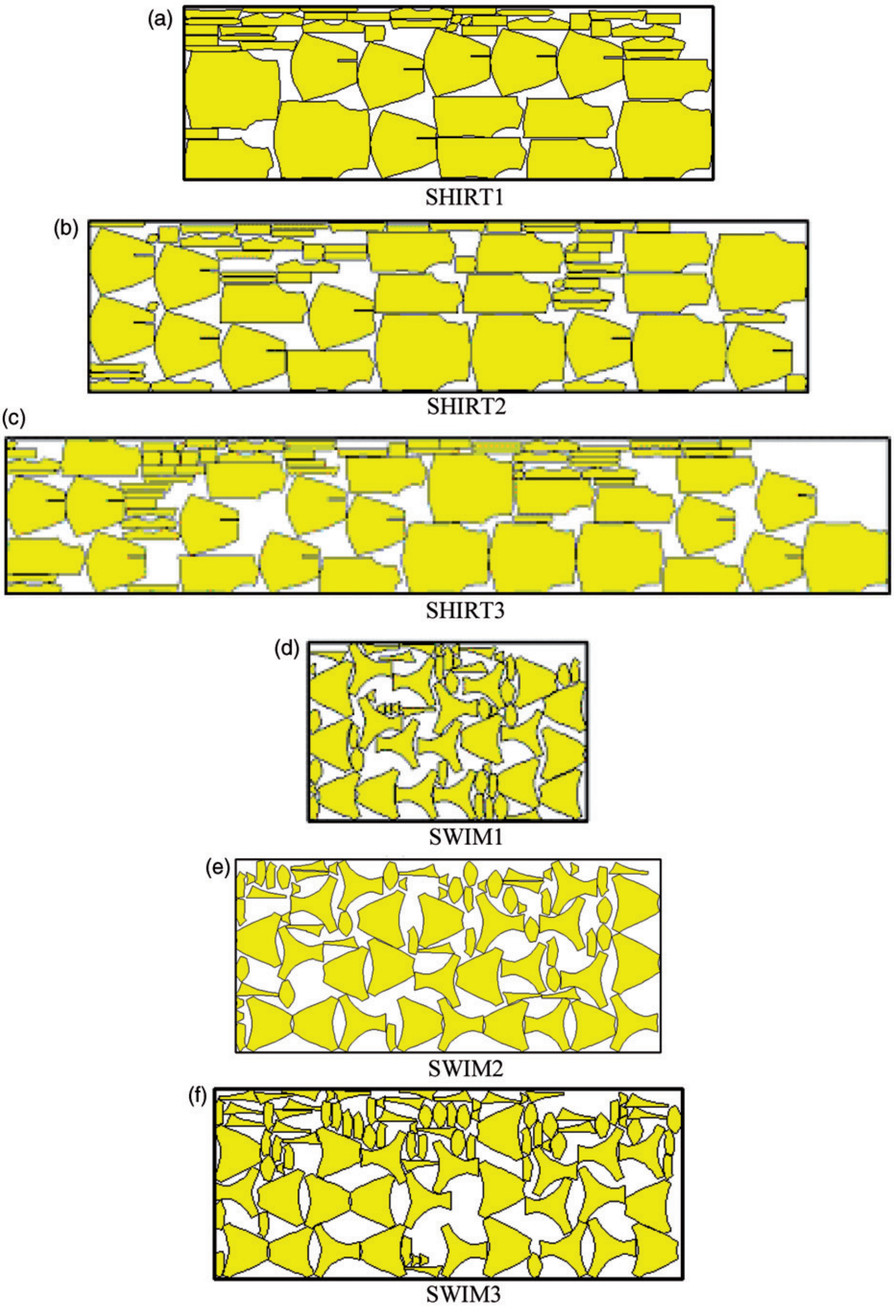


Figure 9. The packing pattern generated by the GA+HP approach for the illustrative examples.

the results derived from marker planners in the industry. The results show that the proposed methodology provides an effective means to increase the usability of the stock sheet.

The proposed methodology can handle convex and concave shapes well and obtain global optimisation solutions. However, this study has not compared the performance of

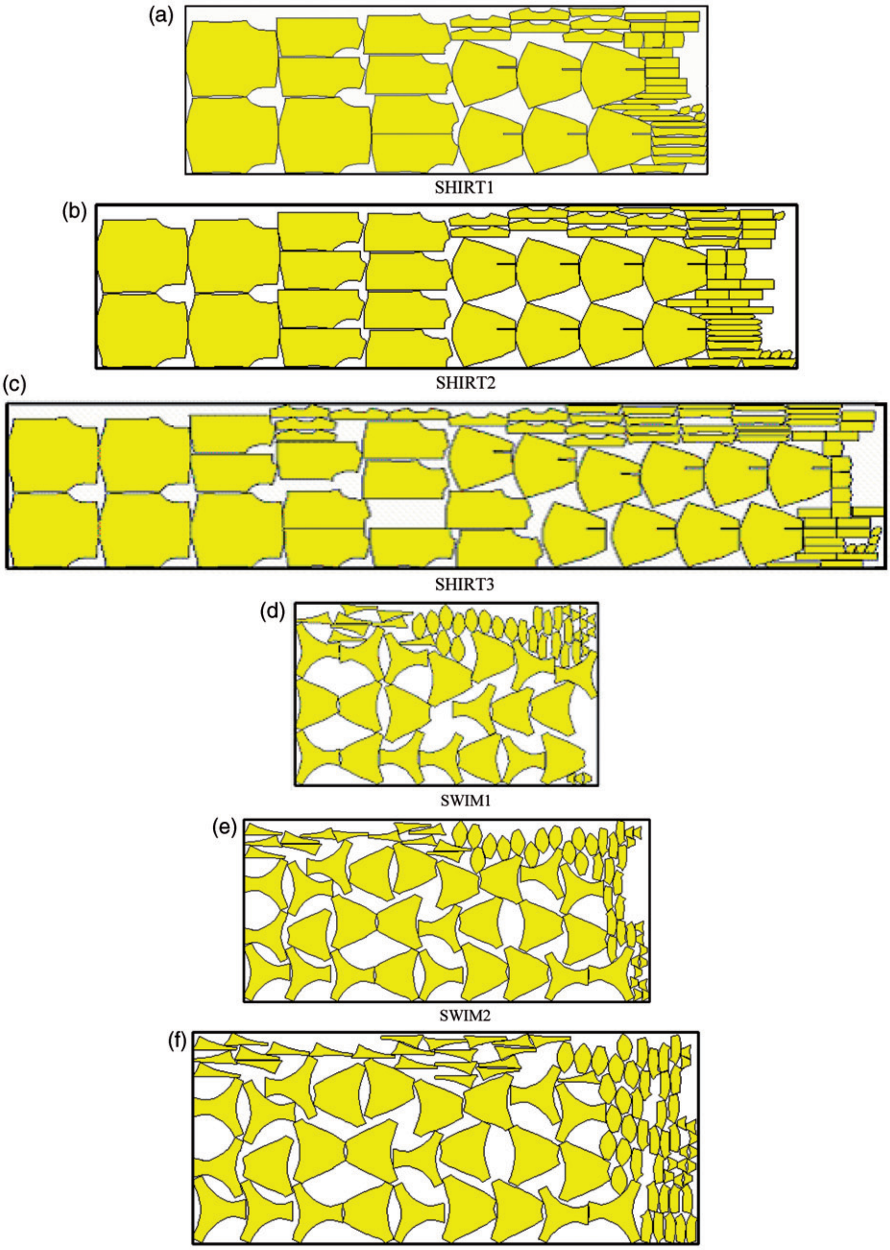


Figure 10. The packing pattern derived from the marker planner in the clothing industry for the illustrative examples.

the proposed approach with the existing approaches in the literature. Based on various benchmark problems in the open literature, future work will aim at the performance comparison of the proposed approach with various existing approaches, such as NFP techniques, phi-function techniques and Burke *et al.*'s (2006) new bottom-left-fill heuristic algorithm. Moreover, the proposed approach will also be fine-tuned particularly in the parameter setting which influences the optimisation performance.

Acknowledgements

The authors would like to thank Earmarked Research Grant of Research Grants Council of Hong Kong and The Hong Kong Polytechnic University for the financial support in this research project (Project No. 529005).

References

Au, K.F., Wong, W.K., and Zeng, X.H., 2006. Decision model for country site selection of overseas clothing plants. *International Journal of Advanced Manufacturing Technology*, 29 (3–4), 408–417.

Babu, A.R. and Babu, N.R., 2001. A generic approach for nesting of 2-D parts in 2-D sheets using genetic and heuristic algorithms. *Computer-Aided Design*, 33 (12), 879–891.

Bäck, T., Fogel, D., and Michalewicz, Z., 1997. *Handbook of evolutionary computation*. New York: IOP Publishing and Oxford University Press.

Bennell, J.A. and Dowsland, K.A., 1999. A tabu thresholding implementation for the irregular stock cutting problem. *International Journal of Production Research*, 37 (18), 4259–4275.

Bennell, J.A. and Dowsland, K.A., 2001. Hybridising tabu search with optimisation techniques for irregular stock cutting. *Management Science*, 47 (8), 1160–1172.

Bennell, J.A., Dowsland, K.A., and Dowsland, W.B., 2001. The irregular cutting-stock problem – a new procedure for deriving the no-fit polygon. *Computers and Operations Research*, 8 (3), 271–287.

Bennell, J., *et al.*, 2008. Tools of mathematical modelling of arbitrary object packing problems. *Annals of Operations Research*, doi: 10.1007/s10479-008-0456-5. [In Press].

Blazewicz, J., Hawryluk, P., and Walkowiak, R., 1993. Using a tabu search approach for solving the two-dimensional irregular cutting problem. *Annals of Operations Research*, 41 (1–4), 313–325.

Bounsaythip, C., Maouche, S. and Neus, M., (1995). Evolutionary search techniques application in automated lay-planning optimization problem. *In: Proceedings of the IEEE conference on SMC*, 22–25 October 1995, Vancouver, BC, Canada, 4497–4502.

Bounsaythip, C. and Maouche, S., 1997. Irregular shape nesting and placing with evolutionary approach. *In*: *Proceedings of the IEEE international conference on systems, man and cybernetics*, Vol. 4, 12–15 October 1997, Orlando, FL, USA, 3425–3430.

Burke, E. and Kendall, G., 1999. Applying simulated annealing and the no fit polygon to the nesting problem. *In: Proceedings of the world manufacturing congress*, 27–30 September 1999, Durham, UK, 27–30.

Burke, E., *et al.*, 2006. A new bottom-left-fill heuristic algorithm for the two-dimensional irregular packing problem. *Operations Research*, 54 (3), 587–601.

Dowsland, K.A. and Dowsland, W.B., 1995. Solution approaches to irregular nesting problems. *European Journal of Operational Research*, 84 (3), 506–521.

Fujita, K., Akagji, S., and Kirokawa, N., 1993. Hybrid approach for optimal nesting using a genetic algorithm and a local minimisation algorithm. *In: Proceedings of the 19th annual ASME design automation conference*, 16–18 September 1993, Albuquerque, NM, USA, 477–484.

Goldberg, D.E., 1989. *Genetic algorithms in search, optimization and machine learning*. Reading, MA: Addison Wesley.

Gomes, A.M. and Oliveira, J.F., 1999. Nesting irregular shapes with simulated annealing. *In: Proceedings of 3rd metaheuristics international conference*, 19–23 July 1999, Angra dos Reis, Rio de Janeiro, Brazil, 235–239.

Gomes, A.M. and Oliveira, J.F., 2002. A 2-exchange heuristic for nesting problems. *European Journal of Operational Research*, 141 (2), 359–370.

Gomes, A.M. and Oliveira, J.F., 2006. Solving irregular strip packing problems by hybridising simulated annealing and linear programming. *European Journal of Operational Research*, 171 (3), 811–829.

Guo, Z.X., Wong, W.K., and Leung, S.Y.S., 2008a. A genetic-algorithm-based optimization model for scheduling flexible assembly lines. *International Journal of Advanced Manufacturing Technology*, 36 (1–2), 156–168.

Guo, Z.X., *et al.*, 2008b. A genetic algorithm based optimization model for solving the flexible assembly line balancing problem with work-sharing and workstation revisiting. *IEEE Transactions on Systems Man and Cybernetics Part C: Applications and Reviews*, 38 (2), 218–228.

Han, G.C. and Na, S.J., 1996. Two-stage approach for nesting in two-dimensional cutting problems using neural network and simulated annealing. *Journal of Engineering Manufacture*, 210 (6), 509–519.

Heckmann, R. and Lengauer, T., 1995. A simulated annealing approach to the nesting problem in the textile manufacturing industry. *Annals of Operations Research*, 57 (1), 103–133.

Hifi, M. and Hallah, R.M., 2003. A hybrid algorithm for the two-dimensional layout problem: the cases of regular and irregular shapes. *International Transactions in Operational Research*, 10 (3), 195–216.

Hoffmeister, F. and Bäck, T., 1991. Genetic algorithms and evolution strategies: similarities and differences. *In*: *Parallel problem solving from nature*. Book series: Lecture notes in computer science, vol. 496/(1991). Berlin: Springer, 455–469.

Hopper, E., 2000. *Two-dimensional packing utilising evolutionary algorithms and other meta-heuristic methods*. Thesis (PhD). School of Engineering, University College Cardiff, Wales.

Ismail, H.S. and Hon, K.K.B., 1992. New approaches for the nesting of two-dimensional shapes for press tool design. *International Journal of Production Research*, 30 (4), 825–837.

Jain, S. and Gea, H., 1998. Two-dimensional packing problems using genetic algorithms. *Engineering with Computers*, 14 (3), 206–213.

Jakobs, S., 1996. On genetic algorithms for the packing of polygons. *European Journal of Operational Research*, 88 (1), 165–181.

Kohonen, T., 1990. Improved version of learning vector quantization. *In*: *Proceedings of international joint conference on neural networks*, 17–21 June San Diego, CA, 545–550.

Li, Z.Y. and Milenkovic, V., 1995. Compaction and separation algorithms for non-convex polygons and their applications. *European Journal of Operational Research*, 84 (3), 539–561.

Oliveira, J.F. and Ferreira, J.S., 1993. Algorithms for nesting problems. *In*: R.V.V. Vidal, ed. *Applied simulated annealing*. Book series: Lecture notes in economics and mathematical systems, vol. 396. Berlin: Springer, 255–273.

Oliveira, J.F., Gomes, A.M., and Ferreira, J.S., 2000. TOPOS – a new constructive algorithm for nesting problems. *OR Spektrum*, 22 (2), 263–284.

Quagliarella, D., *et al.*, 1995. *Genetic algorithms and evolution strategies in engineering and computer science: recent advances in industrial applications*. Chichester, UK: John Wiley and Sons.

Scheithauer, G., Stoyan, Y.G., and Romanova, T., 2005. Mathematical modeling of interactions of primary geometric 3D objects. *Cybernetics and Systems Analysis*, 41 (3), 332–342.

Song, B.L., *et al.*, 2006. A recursive operator allocation approach for assembly-line optimization problem with the consideration of operator efficiency. *Computers and Industrial Engineering*, 51 (4), 585–608.

Stoyan, Y.G., 1980. Concept generalization of the tight placement function. *Reports of NAS of USSR*, 8, 70–74.

Stoyan, Y.G., Novozhilova, M.V., and Kartashov, A.V., 1996. Mathematical model and method of searching for a local extremum for the non-convex oriented polygons allocation problem. *European Journal of Operational Research*, 92 (1), 193–210.

Stoyan, Y., *et al.*, 2002. Phi functions for primary 2D-objects. *Studia Informatica Universalis*, 2 (1), 1–32.

Stoyan, Y. and Chugay, A., 2009. Packing cylinders and rectangular parallelepipeds with distances between them into a given region. *European Journal of Operational Research*, 197 (2), 446–455.

Wong, W.K., Chan, C.K., and Ip, W.H., 2000. Optimization of spreading and cutting sequencing model in garment manufacturing. *Computers in Industry*, 43 (1), 1–10.

Wong, W.K., 2003a. A fuzzy capacity-allocation model for computerised fabric-cutting systems. *International Journal of Advanced Manufacturing Technology*, 21 (9), 699–711.

Wong, W.K., 2003b. Optimisation of apparel manufacturing resource allocation using a generic optimised table-planning model. *International Journal of Advanced Manufacturing Technology*, 21 (12), 935–944.

Wong, W.K., Au, K.F., and Zeng, X.H., 2006. Classification decision model for Hong Kong clothing manufacturing plant locations using artificial neural network. *International Journal of Advanced Manufacturing Technology*, 28 (3–4), 428–434.

Wong, W.K., Zeng, X.H., and Au, W.M.R., 2009. A decision support tool for apparel coordination through integrating the knowledge-based attribute evaluation expert system and the T-S fuzzy neural network. *Expert Systems with Applications*, 36 (2), 2377–2390.

Wu, T.H., Low, C.Y., and Tang, P.T., 2003. Nesting of two-dimensional parts in multiple plates using hybrid algorithm. *International Journal of Production Research*, 41 (16), 3883–3900.

Yuen, C.W.M., *et al.*, 2009a. Fabric stitching inspection using segmented window technique and BP neural network. *Textile Research Journal*, 79 (1), 24–35.

Yuen, C.W.M., *et al.*, 2009b. A hybrid model using genetic algorithm and neural network for classifying garment defects. *Expert Systems with Applications*, 36 (2), 2037–2047.