A review of the application of meta-heuristic algorithms to 2D regular and irregular strip packing problems , Artificial Intelligence Review, vol. 16, 257 – 300, 2001.

# A Review of the Application of Meta-Heuristic Algorithms to 2D Strip Packing Problems

**E. Hopper and B. C. H. Turton**

School of Engineering, Cardiff University, The Parade, PO Box 689, Cardiff CF24 3TF, UK
HopperE@gmx.net, Turton@cf.ac.uk

**Abstract**

This paper is a review of the approaches developed to solve 2D packing problems with meta-heuristic algorithms. As packing tasks are combinatorial problems with very large search spaces, the recent literature encourages the use of meta-heuristic search methods, in particular genetic algorithms. The objective of this paper is to present and categorise the solution approaches in the literature for 2D regular and irregular strip packing problems. The focus is hereby on the analysis of the methods involving genetic algorithms. An overview of the methods applying other meta-heuristic algorithms including simulated annealing, tabu search, and artificial neural networks is also given.

*Keywords: artificial neural networks, genetic algorithms, irregular, meta-heuristics, packing problem, regular, simulated annealing, tabu search, two-dimensional*

## 1. Introduction

Packing problems are optimisation problems concerned with finding a good arrangement of multiple items in larger containing regions. This type of problem is encountered in many areas of business and industry. The objective of the packing process is to maximise the utilisation of material.

High material utilisation is of particular interest to mass production industries since small improvements of the layout can result in large savings of material and considerably reduce production cost. The manual generation of layouts is costly in terms of man-power hence methods for the automation of packing are being sought.

This paper reviews the application of meta-heuristic methods to 2D regular and irregular strip packing. Particular emphasis is put on solutions involving genetic algorithms.

## 2. Packing Problems

### 2.1 Definitions

Cutting and packing problems describe patterns consisting of geometric combinations of large *objects* (e.g. stock) and small *items* (e.g. order book, order list). In the case of packing problems the large objects (e.g. container, bin) are defined as empty and need to be filled with small items (e.g. boxes). Cutting problems are characterised by large objects (e.g. sheet, roll) that need to be cut up into small items (e.g. 2D shapes). The residual objects, that occur in the pattern and do not belong to the order list, are called *trim loss*. The objective of most cutting and packing problems is to minimise the trim loss or wastage.

Dyckhoff (1990) emphasises the strong relationship between cutting and packing problems, which results from the duality of material and space. In this sense cutting stock problems can be seen as packing the space occupied by the small items into the large objects. Vice versa packing problems can be seen as cutting the large objects into small items.

### 2.2 Classification of Packing Problems

Over the past 30 years research in cutting and packing problems have been widely described in the literature. These problems arise in many industries and are not restricted to the manufacturing sector. Packing problems for instance are encountered in operational research and the financial sector in a more abstract form.

Due to this diversity of problems and application areas similar packing problems appear under different names in the literature. Analysing packing problems shows that many of them have the same basic logical structure, although they are encountered in different application areas. In order to facilitate the information exchange across different disciplines Dyckhoff (1990) identified common characteristics and properties and proposed a classification system. He distinguishes between packing problems involving spatial dimensions and those involving non-spatial dimensions. The first group consists of cutting and packing or loading problems that are defined by up to three dimensions in Euclidean space (e.g. cutting stock problems, vehicle loading and pallet loading). The other group covers abstract 'cutting and packing' problems including non-spatial dimensions such as weight, time or financial dimensions (e.g. memory allocation, capital budgeting, coin change and line balancing).

Dyckhoff's classification system describes four important characteristics of packing and cutting problems (Dyckhoff, 1990; Dyckhoff and Finke, 1992):

- The most important characteristic is the *dimensionality* defining the minimum number of dimensions necessary to describe the geometry of the pattern. Problems with more than three dimensions are obtained when they expand to non-spatial dimensions, e.g. time or weight.
- The *kind of assignment* describes whether all objects and items or only a selection have to be assigned.
- *Assortment of the objects:* This characteristic distinguishes between problems, which have objects of identical or different shape.
- The *assortment of the items* refers to the shape and the number of the items. Problems can consist of few items, congruent items, many items of many different shapes and many items of relatively few different shapes.

### 2.3 Regular and Irregular Packing

The objective of a packing problem is the efficient allocation of figures in a containment region without overlap. Hence, the complexity of packing problems is strongly related to the geometric shape of the items to be packed. Concerning the geometry two types of shapes can be distinguished: regular shapes, that are described by a few parameters (e.g. rectangles, circles) and irregular shapes including asymmetries and concavities.

Regular packing problems are largely concerned with packing a set of rectangles onto a rectangular object. Packing of irregular shapes is also known as nesting e.g. in the shipbuilding industry and as marker layout problem in the textile industry (Figure 1).

Figure 1: Irregular packing problem from the textile industry

In 2D rectangular packing, the following layout types can be distinguished on the basis of the geometry of the items to be packed (Hinxman, 1980). In the case of regular items packing patterns can be orthogonal describing cuts only parallel to the sides of the stock sheet and non-orthogonal (Figure 2). Orthogonal cutting additionally distinguishes between guillotineable and non-guillotineable layouts.

Figure 2: Non-orthogonal layout

Guillotineable problems include a constraint, which prescribes that the layout has to be processed by a series of straight cuts across the full length of the remaining object (Figure 3). This type of cutting problem occurs for instance in the glass and paper industry. As non-guillotineable problems are not restricted by this rule, an item can be placed in any available position, which results in a non-overlapping layout (Figure 4).

Figure 3: Guillotineable layout

Figure 4: Non-guillotineable layout

## 2.4 Types of Packing Problems

Packing problems occur in various application areas involving different constraints and objectives. In general, strip packing and bin packing tasks can be distinguished depending on the object type.

### Strip Packing

In the paper and textile industry the raw material is available in the form of rolls. Hence the packing process aims at reducing the height of the layout. This is known as strip packing. Figure 5 illustrates a strip packing problem involving rectangles.

Figure 5: 2D strip packing problem

### Bin Packing

Bin packing refers to packing of multiple bins and can be found where the stock material is available in the form of sheets. The objective usually is to find the set of bins to accommodate all parts of the order list under minimisation of the total material used. Depending on the application, the sheets can be identical or have different size (Figure 6).

In the 1D case, bin packing is the allocation of items, whose width is identical to the ones of the bins. Hence, for the packing process only one dimension is important. Using items of different width results in 2D bin packing (Coffman et al., 1984). Regular bin packing is also frequent in three dimensions in the form of container and pallet loading.

Figure 6: 2D bin packing problem

In industrial applications bin packing appears in various forms. Depending on the objective several problem types are distinguished (Hinxman, 1980; Dowsland and Dowsland, 1992). Larger industrial problems can also appear as combinations of two or more of these basic types.

- The *trim-loss problem* concerns the allocation of the order list onto the given stock sheets. The objective is to minimise the total cost of the stock sheets needed to fulfil the order.

- The *assortment problem* involves the determination of the stock sizes necessary to fulfil the order list. The order list needs to be assigned to a supply of stock sheets such that the best selection of sheets is used.

- The *cutting stock problem* is concerned with the cutting of pieces of a given order list from a set of stock sheets. This problem can be split into two sub-problems, an assortment problem (determination of the sheets to keep in stock) and a trim-loss problem (determination of the cutting pattern to minimise waste).

- In *knapsack problems* each of the order pieces has a given value. The objective is to pack the items into fixed stock objects such that the total value of the items packed is maximised (Martello and Toth, 1990). This type of problem often occurs as a sub-problem in other areas.

- The *loading problem* describes the process of fitting a maximum number of boxes onto a pallet or into a container. The *pallet loading problem* can be regarded from the manufacturer's viewpoint, where identical boxes have to be loaded onto a pallet, as well as from the distributor's side, where the pallet has to be packed with non-identical items. *Container loading* is similar to pallet loading, though in practical applications the two variants of the loading problem can be distinguished by their constraints (Dowsland, 1985).

## 2.5 Industrial Applications

In industrial applications, a number of other factors determine the final layout apart from the objective of reducing the wastage to a minimum. Certain constraints regarding material properties, the cutting process, scheduling aspects and nesting requirements have an influence on the allocation process.

**Material Properties:**

In the sheet metal industry, the non-homogeneous properties of metal such as grain orientation limit the number of possible orientations of the items. If bending operations follow, the parts can only be rotated at a specified angle. Fabric also has certain directional properties and possibly a pattern, which restricts the orientation of the parts to 0° and 180°. It is not always possible to mirror the parts as fabric may have different properties at the other side. Natural materials such as leather consist of areas of various qualities. The quality difference can be due to defects and colour differences. The nesting process needs to match the required quality of the parts with the respective quality zones on the object.

**Cutting Process:**

The cutting technique used to obtain the parts has a great impact on the layout generation. Depending on the cutting technology (e.g. laser and plasma cutting, stamping) a minimum distance between the parts is required. This parameter is referred to as bridge width. In laser and plasma cutting the process operates with a certain width. In order not to damage the parts, a certain distance between neighbouring shapes is necessary. In stamping processes the material tends to slip at the cutting edges if the bridge width is too small. Another important parameter that determines the cutting process is the cutting length. Layouts can be optimised so that the cutting of all parts can be carried out under minimisation of the total cutting path.

**Scheduling:**

The sequence in which the parts are cut can be important for the subsequent manufacturing process. This is the case where parts need to be processed in different steps. If the layouts are large, a special allocation of the parts with respect to the sheets facilitates this. The sequence of the parts may also be important for packaging and shipping. Geometrical and weight constraints may require the parts to be packed in a certain order. Sometimes different order lists are nested in one layout to maximise material utilisation. Hence the order sequence of the parts also plays a role in despatching.

**Nesting Process:**

The parts to be nested can contain void areas, some of which may be large enough to be considered for the nesting of smaller items. This technique is referred to as in-hole-nesting and is very common in the shipbuilding industry. To reduce waste, the nesting algorithm needs to be capable of tracking and nesting into void areas of irregular shapes. Sometimes the current nesting task does not contain a sufficient number of comparatively small shapes. As the raw material is often too precious to be wasted certain filler parts can be designated and used instead. These are not part of the current order and therefore may not be required immediately, but are produced for stock.

Larger nesting tasks might involve material of different types, e.g. thickness. In the shipbuilding industry for instance sheets of different thickness can be involved in the nesting process. Whereas a number of parts require a certain sheet type, often several thicknesses are suitable for a subsection of the order list. Consequently, depending on the availability, the nesting algorithm needs to decide on the best allocation.

### 2.6 Packing and NP-Completeness

The rectangular packing problem, or rather its decision analogue, has been shown to be NP-complete (Fowler et al., 1981). As the irregular and 3D versions of this problem are more complex, they can also be regarded as NP-complete. Various constraints can be imposed on a packing problem depending on the application. Adding constraints may add to its complexity and thus the constrained versions can also be regarded as NP-complete.

According to the definition the NP-complete class has the important characteristic, that all algorithms currently known for finding optimal solutions require a number of computational steps that grows exponentially with the problem size rather than according to a polynomial function (Gary Parker, 1995). It is not worthwhile to search for an exact (optimal) algorithm, since it does not appear that any efficient optimal solution is possible. Alternative approaches that are not guaranteed to find an optimal solution are considered instead. Thus, by giving up solution quality, computational efficiency can be gained. This point of view is often adopted in cutting and packing and has led to the development of approximation algorithms, i.e. heuristics.

### 2.7 Solution Approaches

In terms of solution methods a number of approaches were proposed depending on the type and the size of the problem. For less constrained, simpler packing tasks, exact algorithms were developed along with problem-specific heuristic procedures (Hinxman, 1980, Coffman et al., 1984). For more complex packing tasks, heuristic search methods have been applied successfully for their solution (Albano and Sappupo 1980; Oliveira et al. 2000). Their success can be explained by the great flexibility in taking into account problem-specific constraints. They also offer a good trade-off between solution quality and computational effort regarding the size of the search space.

Since cutting and packing is an important issue in industrial applications, a substantial number of commercial packing software have become available recently. They are specially designed to meet industrial requirements and usually include a variety of features directed at the manufacturing process (Hopper, 2000).

## 3. Surveys and Reviews

Of a number of reviews published in the area of cutting and packing, two surveys attempt to cover the total area of cutting and packing. Dyckhoff and Finke (1992) developed a classification method, on which they based their analysis of the concrete and abstract packing problems (section 2.2). Sweeney and Paternoster (1992) chose the opposite approach and addressed the subject from the perspective of the solution approach. Their work covers more than 400 problems including books, dissertations and working papers and is the most exhaustive bibliography published in this area to date. Table 1 provides an overview of the more recent reviews and surveys in the area of concrete packing problems.

Table 1: Reviews and surveys on packing problems in the literature

This review concentrates only on a small section of cutting and packing - namely 2D regular and irregular strip packing problems. Since the geometric properties influence the complexity of the problem and the size of the search space, this paper distinguishes the various packing tasks according to their geometric features. The problems are grouped into regular or irregular packing problems.

# 4. Application of Genetic Algorithms to Packing Problems

The first researcher who implemented genetic algorithms in this domain was Smith (1985) applying them to 2D rectangular packing problems. At the same time Davis (1985) summarised the techniques applied in genetic algorithms using the example of 2D packing. During the last ten years various types of packing problems were approached ranging from regular to arbitrary shapes in two or more dimensions (Hopper, 2000). Complex problems are commonly approached by a two-stage procedure, a so-called hybrid genetic algorithm. The genetic algorithm manipulates the encoded solutions, which are then evaluated by a decoding algorithm transforming the packing sequence into the corresponding physical layout. Since domain knowledge is built into the decoding procedure the size of the search space can be reduced. The packing strategy for instance may only generate non-overlapping configurations, which restricts the search space to valid solutions only. The need for a decoding heuristic excludes certain information about the layout from the data structures the genetic algorithms operate upon. Therefore not all the information concerning the phenotype is available to the genetic operators and may therefore not be transferred to the next generation.

### 4.1 2D Regular Strip Packing Problems

Regular packing problems are largely concerned with packing a set of rectangles onto a rectangular object of unlimited height (Figure 5). To date, only one approach has been described in the literature that uses other regular items (George et al., 1995). In all cases the aim is to find the arrangement of items minimising the height of the object.

4.1.1 Non-Guillotineable Packing Problems

Several researchers approached the non-guillotineable strip packing problem with genetic algorithms. Many of these methods are hybrid algorithms combining the genetic algorithm with a placement routine. In this two-stage approach a genetic algorithm finds the sequence, in which the items are to be packed with the aid of a placement routine (Table 2 and Table 3). A second group of genetic methods incorporates more layout information into chromosomes using a tree structure (Table 4). Some research concentrated on an entirely different genetic approach, which works without encoding, but manipulates the figures in the 2D layout directly.

**Hybrid Approaches:**

Smith (1985) developed an order-based approach experimenting with two heuristic decoding routines, one of which uses backtracking. The first one ('Slide') places the rectangle in one corner from where it 'falls' to the corner furthest away under orthogonal movements zigzagging into a stable position. The second procedure ('Skyline') tries all stable positions in the partial layout. Comparisons between the two hybrid approaches show that the combination with the more sophisticated procedure generates better layouts, but is computationally more expensive. The performance of the genetic algorithms is compared with a packing method that is based on heuristics and dynamic programming. According to the author the genetic algorithms achieve the same packing densities in less time.

Table 2: Hybrid genetic algorithms for non-guillotineable 2D packing problems

Jakobs (1996) uses the bottom-left heuristic (BL) to hybridise an order-based genetic algorithm. In order to reduce computational complexity the heuristic does not necessarily search for the lowest position available in the layout, but preserves bottom-left stability in the layout. Starting at the top-right corner of the object, each rectangle is moved as far as possible to the bottom and then the left in the partial layout. The initial population is seeded with a sequence in which the rectangles are sorted by decreasing width. During the reproduction process the worst individual in the population is identified and replaced with the offspring according to steady-state replacement. The hybrid concept of this genetic algorithm was extended to polygons using a modified placement rule (section 4.3.1).

The work by Liu and Teng (1999) was aimed at improving the decoder used by Jakobs (1996). The improved bottom-left routine is based on a sliding principle and gives priority to the downward shifting of the rectangle. The authors demonstrated the better performance of the new bottom-left placement routine by using the two packing problems of Jakobs' work.

The order-based approach using a bottom-left packing routine has attracted particular attention over the recent years. Hopper and Turton (1997, 1999) applied a placement routine, which preserves the bottom-left stability in the layout (Table 3). The improved BL-algorithm can access enclosed areas in the partial layout and places the new item in the first BL-position with sufficient area. This packing routine was combined with genetic algorithms and simulated annealing. Simulated annealing generally achieved denser layouts but required longer run times. For certain problem sizes the improved BL-algorithm outperforms the layout quality achieved with genetic algorithms and simulated annealing using pre-ordered input sequences (Hopper, 2000).

Leung et al. (1999) also developed a BL-placement routine, which can access enclosed areas in the partial layout and is called 'Difference Process Algorithm'. Every insertion of a new item in the layout creates two empty rectangular spaces at its top and right side. The algorithm keeps track of the newly generated spaces selecting the one that is closest to the bottom-left corner of the object and sufficiently large for the allocation of the next rectangle. In comparison to the sliding algorithms of Jakobs (1996) and Liu and Teng (1999) the Difference Process Algorithm generates better results, because it is capable of filling enclosing empty areas in the layout.

Dagli and Poshyanonda (1997) used the genetic algorithm to generate an input sequence for the placement algorithm, which is based on a sliding method and combined with an artificial neural network (Table 3). The sliding routine places a new item next to the previously allocated one along the width of the object. If the space is not sufficient, a new row is formed. During the packing process the newly generated scrap areas are recorded and stored for subsequent allocations. Before an item is positioned onto the object, the available scrap areas are tested with an artificial neural network selecting the best match between the item and the empty areas. If no match can be found the item is allocated with the sliding routine. The matching process tries all admissible orientations of the item and is based on a matrix representation of the items and scrap areas using a grid approximation.

Lai and Chan (1997) used an evolutionary algorithm, which is combined with a heuristic routine. This algorithm does not use any cross-over operator and is only based on selection and mutation processes. The heuristic decoder is similar to the bottom-left algorithm used by Leung et al. (1999) and places the item in the position that is closest to the lower-left corner of the object. The packing task used by Lai and Chan is a stock cutting problem. Since the area of the object is limited it may not be possible to allocate all items. In addition to the classic mutation operator, a hill-climbing operation is applied during the decoding process that rearranges the rectangles of the permutation. If an item in the sequence cannot be allocated on the stock sheet the corresponding element in the permutation is shifted to the end of the sequence. Comparisons with a mathematical programming algorithm show that the evolutionary approach is computationally more efficient, but generates patterns with slightly higher trim loss.

Table 3: Hybrid genetic algorithms and evolutionary algorithm for non-guillotineable 2D packing problems

### Hybrid Algorithms Using Additional Layout Information

The data structures of the hybrid algorithms summarised in Table 2 and Table 3 may not recognise characteristic features of packing schemes in the encoding as most of them are hidden in the placement algorithm. A second category of solution approaches involving genetic algorithms is therefore directed at incorporating some of the layout information in the encoding technique (Table 4). Two approaches described in the sequel are based on binary tree structures using some additional rules to fix the position in the layout. Another approach that deals with the manufacturer's pallet problem applies a representation technique, which contains all the information about the phenotype.

The genetic algorithm by Kröger et al. (1991a, b; 1993) is based on a directed binary tree to encode the problem in which each node represents a rectangle. Two sets of edges identify those parts that are adjacent in the vertical and the horizontal direction. This representation fixes one dimension of the position of the current item in the partial layout. The second dimension is determined by the bottom-left condition. In order to generate a unique packing scheme each node is assigned a priority value, so that the rectangle with the highest priority is placed next in case of a conflict. The data structure encodes the set of rectangles and also contains information about orientation and priority. The fitness evaluation of a packing pattern considers the height and the width. The genetic operators have been adapted to the problem with the mutation operators modifying the set of edges, the orientation and the priority values. The cross-over consists of taking a sub-tree from the first parent and placing it at the root position of the offspring. The missing rectangles are then taken from the second parent while the orientations are kept and the priority values are modified such that the packing sequence is maintained. Results show that the genetic algorithm outperforms the BL-heuristic.

Herbert and Dowsland (1996) developed a 2D encoding technique for a manufacturer's pallet loading problem, which contains only identical rectangles. The layout is represented by a 2D matrix indicating available positions for vertical and horizontal placement, where the horizontal one has priority. Since this encoding contains all the information necessary to represent the geometrical layout, no decoding algorithm is required for the fitness evaluation of the layout.

The boxes as well as the pallet are considered as checkerboards of unit squares. In a 1D model the binary strings are composed of all rows in the pallet, where every bit represents a possible placement cell for the box. In order to reduce the solution space, the authors developed a reduction technique to limit the placement positions to feasible co-ordinates that are integral combinations of box lengths and widths. The geometrical meaning of this representation can be seen best in connection with cross-over, which has the effect of cutting the layout horizontally. Hence the string representation reflects proximity in the horizontal direction within the same row, but not in the vertical direction. Vertically close box positions will appear widely separated on the string.

This has been the motivation for developing a 2D matrix encoding. In order to consider the orientation of the items two rows are used in the matrix to encode each row of the pallet, with the one representing horizontal and the other one representing vertical positions. Two cross-over operators were developed cutting the layout horizontally and in a random fashion. This cross-over operation can lead to infeasible solutions, which either can be penalised in the fitness function or repaired. The authors experimented with both options investigating several fitness functions and a repair operator. After removal of overlapping boxes the optimal packing over the corresponding set of positions is calculated using a graph-theoretic model. This repair operator can be used to transform the solutions of the final population into valid layouts. An enhancement operator can also be applied throughout the search process. The enhancement operator optimally packs the removed boxes into the empty areas in the layout. Experimental results indicate it may be more profitable to remove overlap than to penalise it by the fitness function. For the small to moderately sized problems investigated 2D techniques did not have any advantages over the 1D ones. The authors concluded that their 2D approach might prove more beneficial in more complex problems.

Table 4: Comparison of the genetic algorithms for non-guillotineable 2D packing problems - approaches with encodings including layout information

**Algorithms operating on the 2D layout**

The third type solution approach operates without encoding and solves the problem in the 2D space. So far, Ratanapan and Dagli (1997a, b; 1998) developed the only evolutionary approach in this area. Starting from an initial solution, the layout is manipulated by three groups of operators performing hill-climbing, mutation and recombination operations.

Various layout modifications move one item only and are implemented in the form of hill-climbing accepting the layout change if the fitness value is better or remains the same. These operations include translation, rotation and relocation of an item. An operator rotates an item around the touch point with another item, then two operations perform translation and rotation simultaneously. The series of mutation operators aims at rearranging several items. One operator reallocates an item into a different region of the object to create room for the reorganisation of other items. If the target area is occupied the item is reallocated to the upper right corner of the partial layout. In case overlap is created in the target area, a mutation operation is performed which moves all overlapping parts out of this region. Whereas the hill-climbing and mutation operators involve one layout only, the recombination process works on two or more exchanging individual parts or a whole area. Since this can lead to invalid configurations, multiple occurrences of an item and overlap need to be eliminated.

Experiments on rectangular packing problems showed that this approach could generate layouts of up to 97% packing density. A drawback is the complexity of the various modification operators involving overlap determination and reallocation of partial layouts. Since no comparisons are made to other solution approaches in the literature it is difficult to establish the efficacy of this method.

### 4.1.2 Guillotineable Packing Problems

Guillotineable packing problems have been approached with genetic algorithms by four researchers. Three algorithms are based on tree representations applying various genetic operators. One method takes a different approach and uses a permutation and a heuristic decoder to generate guillotineable layouts (Table 5).

The slicing tree representation proposed by Kröger (1995) ensures that the packing pattern is guillotineable. The relative arrangement of the rectangles stored in the leaf nodes is described with the aid of two operators at the node above indicating either a horizontal or a vertical combination. In order to preserve the knowledge stored in the sub-trees, a special cross-over operator exchanges sub-trees. Only sub-trees with a certain packing density and at most four rectangles are transmitted to the offspring. After reducing the first parent to the sub-trees to be inherited, the sub-trees from the second parent are separately inserted into the new string together with a new cut-line. The offspring is completed by the insertion of single rectangles that are missing from the complete set. In terms of mutation five different operators are applied (Table 5). A hill-climbing strategy is implemented in the genetic algorithm aimed at improving the fitness of a recently mutated or recombined string. The solutions produced by the genetic algorithm are superior to those found by heuristic algorithms as well as random search and simulated annealing algorithm. Genetic algorithms and simulated annealing achieve significantly better results than the primitive heuristics, with the genetic algorithm being closer to the best-known solution.

Hwang et al. (1994) tackled the strip packing problem with two methods. One approach is based on a directed binary tree that can be described in the form of a string in polish notation. An operator is assigned to each tree-node indicating either the vertical and horizontal combination of two rectangles. Before the cross-over operation, the polish expression is spilt into permutation and operator parts that are manipulated separately. Four different mutation operators are applied to the chromosome (Table 5).

The second representation is order-based and applies a level-oriented packing procedure. The packing is constructed as a sequence of levels; each rectangle is placed left justified so that its bottom side rests on one of these levels. A level is defined as a horizontal line drawn through the top of the tallest rectangle on the previous level. A new level is started whenever the remaining width of any of the previous levels is too small. Two versions of this decoding algorithm were implemented placing the current rectangle into the level where it fits first (First Fit strategy, FF) or positioning it where it fits best (Best Fit strategy, BF) (Table 6).

Comparisons between the two methods show that the order-based approach achieves better packing densities. The authors conclude that the penalty term is not sufficient to deal with the width constraint.

The two genetic algorithms are compared to the First-Fit-Decreasing-Height heuristic (FFDH), which sorts the rectangles according to their height before placing them sequentially in the first available position. The two hybrid algorithms using the simple decoding routines perform equally well. Their performance is better than the one of the FFDH-heuristic.

In order to reduce the complexity of the problem, Kröger (1995) introduces the concept of meta-rectangles, which describe a group of adjacent, densely packed rectangles that are combined to one large rectangle. In this way partial layouts are frozen yet the shape is still flexible enough to be grouped with other rectangles. In terms of recombination the cross-over operator has to ensure that the meta-rectangles are transmitted to the offspring. This produces a significant reduction in the run times and leads to an improvement in the average best solutions.

The solution approach applied by Rahmani and Ono (1995) is based on a binary tree, where each leaf node represents a rectangle. The node at the hierarchy level above indicates whether two rectangles perform a horizontal or vertical combination. In order to preserve the feasibility of the offspring a special cross-over operator was developed. Unlike the classical genetic algorithm where a certain amount of the population is selected for the recombination process, each individual is considered for cross-over. Once an individual is selected for cross-over using a certain node, a suitable candidate is searched and crossed. Since only sub-trees are crossed the solution only needs to be evaluated partially during the fitness calculation.

András et al. (1996) used a tree representation for this problem, where each node is either further cut into two pieces or remains uncut. In order to encode this problem a data structure has been developed with each node containing information about the dimensions of the piece, the position and orientation and the occurrence of a cut. The fitness of the individuals is related to the packing density. A combined crossover - mutation operator exchanges sub-trees between two parent strings. It may be necessary to modify the offspring after the crossover to guarantee feasible solutions, which adds a mutational component to the operation. As the quality of the solutions is not measured against another method, the general performance of the genetic algorithm cannot be evaluated.

Table 5: Comparison of the genetic algorithms for guillotineable 2D packing problems using tree representations

Corno et al. (1997) developed a hybrid genetic algorithm to solve a trim-loss problem from the glass cutting industry. The problem involves a number of constraints such as guillotineable layout, maximal distance between parallel cuts and defect areas of the glass sheet. A heuristic algorithm was developed that considers all technological constraints. The chromosome consists of a permutation of the items and contains a sequence of genes. Each gene corresponds to one item describing the geometrical characteristics, the orientation and a placement criterion flag, which links the piece with its predecessor. The mutation operators work on these genes, changing the orientation and the placement flag. The layout constraints are left to the decoder that searches through the available objects and positions to find the best. Comparisons to commercial packages show that the performance of the genetic algorithm is equal or better, especially for larger packing tasks.

Table 6: Comparison of the genetic algorithms for guillotineable 2D packing problems using order-based representation

### 4.2 Packing of Regular Shapes other than Rectangles

The only genetic algorithm designed to pack regular shapes other than rectangles was proposed by George et al. (1995). A hybrid genetic algorithm is combined with a heuristic method to pack different-sized circles into a rectangular area. During the packing process so-called position numbers are used to indicate possible locations for the remaining circles in the partial layout.

The encoding technique of the genetic algorithm makes use of the position numbers, which are defined with respect to the sides of the object and the circles already placed. Instead of evaluating every possible position of a circle in the packing pattern, only an initial position is allocated to each circle. This serves as a default position and is only modified if it causes an infeasible packing configuration. The initial positions of all circles are stored in the chromosome, with the first cell containing the position of the first circle etc. As a measure of fitness the density of the circles in the rectangle is used. The genetic operators applied are proportional selection, one-point crossover and a mutation operator that generates a random position number. The decoding procedure attempts to place a circle at a position number contained in the string. If this position is not feasible or not defined, the position number is incremented until a feasible position is found.

The genetic algorithm is compared to heuristic methods using the same decoding procedure. The comparison includes a heuristic method that generates the position number randomly. Performance comparisons for different problem types showed that genetic algorithms and random search outperformed the other heuristics, when a balance must be reached between quality and computational effort. The advantage of the data structure in George et al. (1995) is that domain information is implemented in the genetic algorithm as part of the procedure. The task of the decoder is to check the feasibility of the layout and eventually to find a new position.

### 4.3 2D Irregular Strip Packing Problems

This category of irregular problems includes the packing of polygons and arbitrary shapes on an object of fixed width and unlimited height (Figure 7). A number of researchers have approached the packing of polygons some including holes inside the shapes. Depending on the nesting algorithm, some approaches are only suitable for convex polygons. In most solution approaches the irregular items are either polygons or approximated by polygons consisting of a list of vertices. Geometric algorithms are then required to determine feasible positions in the partial layout and eventually to calculate the overlap. A second shape description technique is grid approximation where items and objects are represented by a set of equal sized squares using 2D matrices. The nesting process therefore usually involves scanning of the various matrices and matching with empty cell clusters. An outline of the algorithms is given in Table 7 to Table 10.

Figure 7: 2D irregular strip packing problem

### 4.3.1 Packing of Polygons

Fujita et al. (1993) proposed a hybrid approach combining an order-based genetic algorithm with local minimisation to solve a nesting problem involving convex polygons only. The local minimisation algorithm is used to optimise the position of an item in the layout, after initial placement in the leftmost-lowest position next to its preceding neighbour. This algorithm uses a Quasi-Newton method to manipulate the relative positions between the objects defined by a set of variables. The fitness of an individual is related to the waste and the distance of the polygons from the origin of the object and deals with the width and overlap constraint. Since the performance of the hybrid genetic algorithm was not compared to other methods, it is not possible to judge its efficiency.

Jakobs (1996) used an order-based genetic algorithm for nesting and extended the work on packing of rectangles (section 4.1.1) to polygons. The decoder only operates on the enclosing rectangles of the polygons during the evaluation stage. When the polygons are fed into the system they are first rotated into the position where the enclosing rectangle has the smallest area. The irregular aspect of the packing task is considered after the genetic algorithm has converged applying a shrinking-algorithm to the layout. This algorithm moves the polygons closer together shifting them as far as possible to the bottom and the left whilst avoiding overlap and also tests reflections of the original polygons. The shrinking-step reduces the height of the layout, and allows utilisation of the space "wasted" by the embedding process. Applying the shrinking-routine to the final layout has a major drawback. The polygons are repositioned sequentially, so empty areas in the layout may not always be reached, uniterated items can block the sliding motion of the current one. Since no comparison was made to other techniques for irregular nesting tasks, it is difficult to establish the overall performance of the method proposed.

Table 7: Hybrid genetic algorithms for 2D irregular packing problems

Dighe and Jakiela (1996) developed two genetic algorithms for the nesting task. The first approach is order-based and uses a sliding algorithm to move the irregular item into the partial layout in vertical direction. A low-level genetic algorithm is applied to find the best horizontal position at the upper side of the object from which to "drop" the item and its orientation for the sliding process.

The second genetic algorithm uses a binary tree representation technique and is also hierarchical. The tree determines the way in which two items are clustered. The nesting process of the two polygons is controlled by a low-level genetic algorithm, which searches for the configuration with the smallest enclosing rectangular area. Both approaches avoid overlap during the nesting process. The two methods were tested on jigsaw puzzles with a known optimum solution and achieve packing densities between 69% and 72%. The major drawback of these techniques is the hierarchical structure using two genetic search processes. The low-level search is extremely wasteful in terms of computation time.

Bounsaythip and Maouche (1997) applied a binary tree approach to a problem from the textile industry. Before the nesting step, the polygons are circumscribed by the bounding rectangles. The nodes in the tree contain two operators that determine the side at which the second rectangle is packed with the stationary one and its orientation. The actual nesting process is carried out by a low-level routine which finds the smallest enclosing rectangle of the cluster using a special encoding technique described in their earlier work (see below; Bounsaythip et al., 1995). A single tree in this approach does not necessarily represent the complete set of items, but rather a strip in the textile layout. The algorithm therefore has to deal with trees of different length. The cross-over and mutation operators are stated in Table 8.

In an earlier approach, Bounsaythip et al. (1995) used a different genetic algorithm to address the textile problem. Instead of dealing with a complex marker layout they focus on the generation of one strip only in the layout. The polygons are circumscribed by the bounding rectangles. The shapes are represented with a special encoding technique that describes the contour of the polygon relative to the enclosing rectangle using a set of integer values. For each of the four rectangle sides such a contour description is generated. This representation technique is very practical for nesting two shapes. Unlike many other genetic algorithms a single shape represents one individual in the population. The fitness of an individual is determined by the utilisation ratio of the bounding box. Cross-over and mutation operators are domain-specific and merge selected shapes. The performance of this genetic algorithm was enhanced through hybridisation with simulated annealing which slightly increases the packing density.

Table 8: Comparison of the genetic algorithms for 2D irregular packing problems

Petridis and Kazarlis (1994) developed a genetic algorithm, which does not require a decoding algorithm in the nesting process. Instead, the position of an item in the layout is encoded in the chromosome in form of two binary strings. The simplicity of the encoding technique, allows the traditional binary cross-over and mutation operators to be used. Furthermore, a set of mutation operations were defined to work directly on the phenotype swapping two shapes or repositioning shapes into gaps in the layout. Overlapping configurations can occur because the position of the items is determined by the encoding. These are penalised in the fitness function. The fitness function is dynamic, increasing the penalty term gradually in order to drive population away from invalid solutions towards the end of the search. Similar to some simulated annealing approaches in the literature (section 5), the rationale behind the dynamic nature is to penalise overlap less at the beginning when it is important for the shapes to pass over one another in order to reach enclosed areas. A local search technique was applied to the best solution at fixed generation intervals. Petridis and Kazarlis (1994) tested their algorithms on jigsaw problems consisting of less than 15 shapes. Comparisons showed that the optimal solution was more often found using the dynamic fitness function. The local search had a positive impact and accelerated the search process.

### 4.3.2 Packing based on Grid Approximation

Compared to the shape description based on geometric primitives such as polygons, fewer approaches use a digitised representation. Grid approximation offers the advantage that holes inside items or gaps in the partial layout can be easily described. Since the object is usually scanned for a suitable position these areas are automatically considered. One of the major advantages of this technique is that no additional routines are required to identify enclosed areas in the shapes or the partial layout. The different solution approaches are outlined in Table 8 and Table 9.

Ismail and Hon (1992) developed a genetic algorithm for the pairwise clustering of two identical polygons. After circumscribing the shape with the minimum enclosing rectangle, a grid is superimposed to convert the shape into a binary 2D matrix. When clustering two shapes, two parameters are used to describe their relative position to each other. Another four parameters are introduced to represent the mirroring of the shapes along the two axes. These parameters are combined to a binary multi-parameter string, defining a clustering solution. The fitness reflects the best orientation for maximising the material utilisation and includes a penalty for overlapping. Subsequent decoding of the string into a layout is straightforward. Comparisons to the performance of another clustering method that was developed by the authors earlier showed that the genetic algorithm produces denser packing of figures with concave features. This is mainly due to the limitations of the other method, whereas the solutions have been identical for other shape types.

Ismail and Hon (1995) extended the clustering method proposed in (Ismail and Hon, 1992) to dissimilar shapes in combination with a heuristic rule. Applying the above representation technique, the shapes and the object are first digitised and represented as a 2D grid array. Two parameters describe the relative position of a shape to the others and three parameters define mirroring and rotation. The overall genetic string is a sequence of the encodings for each individual shape. This data structure can result in infeasible solutions, which are penalised in the fitness function. The decoder uses a complex set of parameters and rules to describe the relative positions and the placement of the polygons.

Poshyanonda and Dagli (1993) extended the order-based genetic algorithm developed for rectangles to the nesting of irregular shapes (section 4.1.1). The decoder consists of an artificial neural network that matches an incoming shape with the available empty areas in the partial layout. For this purpose the items and the object are presented as binary 2D matrices. The algorithm selects the best match or triggers a sliding algorithm if no match is found.

Table 9: Hybrid genetic algorithms for 2D irregular packing problems

Gwee and Lim (1996) studied a special type of irregular packing problem originating from the world of jigsaw puzzles. The items consist of rectilinear blocks, so-called polyominoes, which are placed onto a rectangular board. Since these puzzles have a known optimal solution, the performance of the genetic algorithm can be easily measured. The objective function considers three aspects, which are important in the search for the optimal configuration (Table 9). The set of polyominoes is represented as a permutation. The decoding stage uses a circular placement technique, which places the shapes in circular fashion starting at one corner of the board, and continues in anti-clockwise direction towards the centre of the board. Several orientations are tried selecting the one that yields to the highest number of contact edges. The idea of this technique is to build up good groupings of polyominoes starting from the corners. Comparisons with two hill-climbing techniques show that the genetic algorithm finds the optimal solution quicker, in particular when the problems consist of a higher number of pieces.

Jain and Gea (1998) designed a special encoding method, which describes the complete layout as a 2D matrix. Before the encoding step, the items are digitised and consist of a cluster of unit squares. In the 2D matrix the corresponding cells are marked with the item number. In that way the phenotype is completely contained in the genotype making a decoding algorithm redundant. A set of problem-specific cross-over and mutation operators were developed to work on this representation scheme and are stated in (Table 10). Since these operations can easily result in overlapping configurations, repositioning of items is frequently required. In order to increase the density of the layout, subsequent compaction steps shift the items left and down in order to fill vacant positions.

The method developed by Ratanapan and Dagli (1997b, 1998) is different from the other approaches described so far, since it does not make use of a data structure to represent the problem. The irregular items are represented using a grid approximation. After the initialisation process, which places all items into non-overlapping positions on the object, a series of genetic operators is applied consisting of hill-climbing, mutation and recombination processes. These operations are described in connection with their earlier work on rectangle packing in section 4.1.1.

Table 10: Genetic algorithms operating on the phenotype for 2D irregular packing problems

Genetic algorithms are not the only meta-heuristic techniques that have been applied successfully to packing problems. A number of researchers experimented with simulated annealing, tabu search and neural networks.

## 5. Application of Simulated Annealing

Simulated annealing is a meta-heuristic search method whose design was inspired by the metallurgical process of annealing. Eglese (1990) investigated the application of simulated annealing as a tool in operational research. Simulated annealing was applied to rectangular and irregular packing tasks, a selection of which is described below.

### 5.1 Regular Packing Problems:

Only few researchers have applied simulated annealing to 2D rectangular packing problems (Table 11). One of the first researchers working on simulated annealing and packing problems was Kämpke (1988). He applied simulated annealing to 1D bin packing comparing different cooling strategies.

Dowsland (1993) experimented with simulated annealing on pallet loading problems involving identical as well as non-identical boxes. In the identical case, the number of feasible positions is reduced to the co-ordinates, which are multiples of the item length. The neighbourhood is defined as the set of solutions, which is obtained, when each item is moved to any other position with some restrictions. Since these movements lead to overlapping patterns, this constraint has been dealt with in the objective function. In the extension to non-identical boxes, the condition for the feasible position is that it needs to be at a valid combination of lengths and widths of the other item types starting from the container edge. The results indicate that simulated annealing is only capable of producing near optimal solutions, which could be improved by other optimisation routines.

Faina (1999) developed a hybrid simulated annealing algorithm for guillotineable and non-guillotineable stock cutting problems. The set of items is represented as a permutation indicating the order of packing. Two heuristic decoders are used to pack the objects whilst taking into consideration the guillotine constraint. The algorithm for the non-guillotineable layout places the current item either at the top-left or the bottom-right corner of the previously positioned rectangle. The choice between the two insertion points is random. In the guillotineable case, the algorithm keeps track of the remaining empty areas in the layout. After placing a rectangle, two empty areas are created at the top and the right side, which are stored and treated as objects in the subsequent packing processes. Although the placement algorithms are formulated for a stock cutting problem, the performance evaluation only involves one object of unlimited height (i.e. strip packing). Comparisons show that the algorithm for non-guillotineable layouts achieves much higher packing densities than the algorithm developed for non-guillotineable problems due to the better nesting technique.

Leung et al. (1999) also applied the order-based approach developed for use with genetic algorithms to simulated annealing (section 4.1.1). Their results indicate that genetic algorithms outperform simulated annealing.

Table 11: Comparison of the simulated annealing approaches for 2D rectangular packing problems

### 5.2 Irregular Packing Problems

Most simulated annealing approaches for irregular packing tasks do not make use of any encoding technique. The packing problem is represented as an allocation of 2D items, which must be compacted. Usually, overlap is permitted during the search process and penalised in the evaluation function. The representation technique for this approach differs from the ones involving genetic algorithms. With one exception (Ratanapan and Dagli, 1997b, 1998) the genetic methods from the literature operate on encoding. Overlap is usually avoided through the application of placement rules and only permitted in few approaches. A number of 2D nesting tasks have been approached with simulated annealing (Table 12).

Jain et al. (1988) addressed a blank nesting problem from the metal cutting industry, where two congruent items are nested for continuous strip stamping applications. The blanks have arbitrary shape and are approximated by polygons. In order to accommodate interlocking shapes it is necessary to allow the shapes to move over one another producing intermediate overlap. Overlap is penalised in the fitness function consisting of two terms: the wasted area and a penalty for the total overlap.

Marques et al. (1991) developed a simulated annealing algorithm for the packing of polygons and applied it to a problem from the textile industry. A neighbourhood move is achieved by translation, rotation or reflection of an item whilst only accepting valid configurations. The quality of the layout is described by the sum of three components: the area of smallest enclosing rectangle and parameters indicating the distance of each item from the centre of the object and proximity of the items to each other. In order to reduce the processing time for the verification of the layout legality, only the overlap between corresponding enveloping circumferences of the items is tested initially.

The efficiency of the search process conducted by simulated annealing largely depends on careful construction of the cooling schedule. Theodoracatos and Grimsley (1995) experimented with polynomial-time cooling schedules and showed their impact on computational efficiency. The authors applied simulated annealing to the packing of circles and polygons. In addition to that, an adaptive penalty function was proposed to penalise overlapping configurations to a minor extent at the beginning when items need to slide over each other in order to find feasible positions in the partial layout.

The simulated annealing algorithm in Han and Na's work (1996) is used to improve an already existing layout created by an artificial neural network (ANN). The irregular items are first approximated by polygons and circumscribed by a minimum enclosing rectangle. The polygon is then described as a composition of basic geometric shapes i.e. rectangles and circles that are placed in the void areas within the enclosing rectangle as well as within the item itself. A move to a neighbouring solution consists of translation, rotation or a swap of two items. Since these operations can result in invalid configurations the fitness function considers the overlap constraint in the form of a penalty. In order to achieve dense layouts a second parameter describes a force driving an item leftwards and downwards. The neighbourhood move achieved through translation is implemented in two ways. The large perturbation within the entire object area is directed at the global optimisation of a layout whereas the small perturbation in the lower leftward direction is used to optimise the layout locally. Low starting temperatures were used in this approach, since the starting solution obtained from the ANN has already generated a reasonably good quality. It is difficult to judge the merits of these hybrid approaches involving two intelligent search processes due to the lack of comparisons with other methods.

Burke and Kendall's work (1999) is different from the approaches described above, since the neighbourhood moves are not performed directly in the layout. Instead the problem is represented as a permutation. A neighbouring configuration is reached through one of the re-ordering techniques stated in Table 12. As a consequence a placement routine is required to transform the list of items into the layout. The authors developed an algorithm, which nests two polygons in turn using the No Fit Polygon (NFP, Adamowicz and Albano, 1976) and local search to determine the best position. Before a new polygon is placed, all positions along the vertices of the NFP are tried and the cluster with the smallest convex hull is used. In case the configuration exceeds the bin width a new 'row' is started. Results show that the simulated annealing technique produces better results than hill-climbing and there is a difference between the various neighbourhood operators.

Table 12: Comparison of the simulated annealing approaches for 2D irregular packing problems

## 6. Application of Other Meta-Heuristic Search Methods

### 6.1 Tabu Search

Tabu search is a search technique that is guided by the use of adaptive or flexible memory structures. It is different to heuristic methods such as simulated annealing and genetic algorithms, tabu search contains some in-built memory mechanisms that prevent the search algorithm from returning to recently executed moves for a number of iterations. A tabu list is maintained which contains all moves, which are not allowed in the current iteration step. The search is guided by an objective function in order to find the best admissible move in a neighbourhood (Reeves, 1993; Glover and Laguna, 1993). With respect to packing problems, fewer solution approaches with tabu search have been proposed than with genetic algorithms and simulated annealing. The first work in this area was perfomed by Blazewicz and his co-researchers in the early 90's.

The only investigation into the application of tabu search to rectangular problems was presented by Lodi et al. (1999) who focused on 2D bin packing. The two constraints that are imposed on the packing process concern the fixed orientation of the items and the layout, which has to be guillotineable. The initial layout is generated by a simple heuristic algorithm, which is then improved by tabu search. The tabu search algorithm is based on two possible neighbourhood moves. The first one attempts to remove an item from the worst bin redistributing it among the other used bins. In the latter move the algorithm tries to accommodate the item by recombining the items of two other bins. The bin layout is generated with a heuristic level-oriented algorithm. The performance of the tabu search is better than any one of the two bin packing heuristics and comparable to a branch-and-bound algorithm.

Blazewicz et al. (1993) were the first to apply tabu search to irregular packing problems. Starting with a feasible layout solution, which is produced by a simple placement procedure, a tabu search process is used to further improve the existing layout. After selecting a single item, several new positions are tried and the best one is kept. The move describes a change of the allocation of one item from one position to the other, prohibiting overlapping configurations. Items that have changed their position during recent iterations are members of the tabu list. The best admissible move is determined by the objective function aiming at placing the rightmost elements into the void areas of the layout. In comparison with Albano and Sapuppo’s (1980) heuristic search algorithm, the tabu search achieved better results.

### 6.2 Artificial Neural Networks

In some approaches to rectangular and irregular packing problems, neural networks have been used. They were also applied in combination with other meta-heuristic methods where they either served to generate the initial layout or to perform the nesting process. Two examples are briefly described below.

Dagli and Poshyanonda (1997) used a neural network in combination with a genetic algorithm for a rectangular packing problem (section 4.1.1). The genetic algorithm is used to generate an input sequence, which is decoded into the layout by the neural network. Every time a new item is placed into the partial layout all new scrap areas are recorded and stored for subsequent nesting processes. Before an item is allocated the neural network searches through all empty areas and returns the best match. If no match is found the item is allocated next to the partial layout using a sliding algorithm. The matching process is based on a grid description of the items and scrap areas.

Han and Na (1996) used a neural network to produce an initial solution for a 2D irregular problem. After a 'good' initial solution is obtained the non-overlapping layout is further improved by simulated annealing (section 5). The learning algorithm of the neural network is based on a Kohonen network. At the beginning of the nesting process all shapes are allocated around the centre of the object by assigning small random values to their position vectors describing the distance to the centre. The position vectors, which indicate the direction of the motion for the items, are modified by the neural network. The finite position is determined such that the overlap of the items is minimal using leftmost-lowest placement. The cost function is a combination between a penalty term for the overlap and the moments of area driving items to the left and to the bottom side of the object.

### 6.3 Other Heuristic Search Techniques

One of the main characteristics of meta-heuristic search processes as opposed to local search is that they contain a means of escaping local minima. Whereas optimisation with hill-climbing terminates when a locally optimum solution is found, meta-heuristics can escape this situation by temporarily accepting solutions of lower quality. Some researchers used the concept of these uphill moves and implemented new meta-heuristic search principles in addition to the standard methods like genetic algorithms and simulated annealing.

Healy and Moll (1996) proposed a minimisation algorithm for a 2D rectangular packing problem. The algorithm is a variant of a hill-climbing technique and designed such that it allows moves in the other direction in order to escape local minima.

Pargas and Jain (1993) developed a stochastic optimisation algorithm, which borrows some principles from hill-climbing and genetic algorithms. The method was applied to a 2D irregular packing problem. Stochastic optimisation operates on a population of solutions manipulating them with the aid of a mutation operator. Unlike in genetic algorithms, only one solution is modified at a time. A new state in the search process can be obtained in two ways. The first one selects a solution from the population using ranking and generates a certain number of neighbouring states as in steepest-ascent hill-climbing. The best solution of the neighbourhood compared with the current solution is taken. If its fitness is better, it replaces the current one in the population. With a probability of around 10% the second method generates a new state randomly in order to maintain diversity in the population. The termination criteria are based on convergence or a maximum number of iterations.

In the implementation for an irregular packing problem the items are represented as a permutation. A grid approximation technique is applied. The allocation routine scans the object for the first leftmost-uppermost cell, which allows a valid configuration. If overlap occurs the item is rotated by 90°. Unfortunately, the authors did not compare this approach to other meta-heuristic techniques. Therefore relative performance in terms of solution quality and speed are not known.

## 7. Summary

Meta-heuristic search methods have been implemented for the solution of a large variety of 2D packing problems. The solution space of combinatorial problems is enormous and increases rapidly with the complexity of the problem, in particular with the geometry of the objects to be allocated. With most of the packing problems being NP-complete, heuristic search procedures are used, since exact algorithms cannot solve the problem efficiently; their time function is described by a polynomial. During recent years, researchers have proposed an increasing number of meta-heuristic approaches for the solution of rectangular and irregular packing tasks that offer the ability to search large and complex solution spaces in a systematic and efficient way.

## 7.1 Solution Approaches

The major features of the existing solution approaches with respect to encoding technique, shape representation and algorithm design are briefly summarised in the following section, which highlights their major advantages as well as disadvantages.

### 7.1.1 Encoding

The strength of genetic algorithms lies in the ability to search large and complex solution spaces in a systematic and efficient way. Not being dependent on a particular problem structure allows the user to utilise different methods for the encoding of the genotype. The performance of a search process is strongly related to the representation of the packing problem. It is important that the encoding technique, which describes possible packing patterns, utilises characteristic features in the packing schemes. It may be advantageous to design the data structure such that sub-structures of layouts are accessible and can easily be manipulated. For packing problems, order-based chromosomes can be used to represent packing sequences. An appropriate modification of the data structure may maintain certain efficient sub-structures of the layout. At the same time the genetic operators need to be adapted to the encoding technique, so that they support the inheritance of important layout features, which are meaningful and effective for the packing objective.

### 7.1.2 Type of Approach

With respect to the packing problems described three types of solution approaches involving genetic algorithms can be distinguished (see below). The common feature of genetic algorithms developed for packing problems is their two-stage approach. The genetic algorithm is used to explore and manipulate the solution space, and a second procedure is used to evaluate the solutions. The phenotype needs to be constructed in order to check quality and feasibility of packing scheme.

In the first group the genetic algorithm is only used to determine the sequence of packing. Therefore a placement routine is then needed to find the allocation of the items on the object. A heuristic decoder can limit the genetic algorithm. It may not support the inheritance of certain features by the offspring since the domain knowledge is hidden in the placement routine. In order to avoid the dependency of the performance of the genetic algorithm on the decoding method, it seems beneficial to develop a data structure that calculates the fitness from the genotype rather than the decoded phenotype. A second category of solution approaches attempts to incorporate more layout information into the data structure of the genetic algorithm. Some additional rules are still needed to fix the position in the layout. The third group of genetic solution methods resolved this matter by transferring the genetic search process into the 2D layout domain. Since the genetic operations are performed directly on the 2D shapes this method does not require an encoding technique.

The concept of performing a search process entirely in the layout domain has long been applied in simulated annealing and tabu search (Marques et al., 1991; Blazewicz, 1993) and is common to most approaches in this area. Applying an indirect optimisation process via the use of an encoding is a very recent idea (Faina, 1999; Burke and Kendall, 1999).

The benefits of an operation in the 2D space are evident, since it enables a meaningful implementation of the abstract meta-heuristic principles and operators describing concepts such as neighbourhood and neighbourhood moves as well as features of the phenotype, cross-over and mutation. The operation on the layout rather than an encoded data structure raises a number of other issues, such as overlap. Overlapping configurations are invalid solutions and need to be resolved either by rejecting, correcting or temporarily accepting them. Rejection wastes precious computation time and may result in less dense layouts for highly irregular shapes, since the slightest change in position or rotation could lead to invalid configurations, which will no longer contribute to the search process. Correcting invalid configurations seems a better option, since often only minor re-positioning is necessary to obtain a valid solution. This contributes to the computation time, especially, if the re-positioning task turns out to be more complex.

Accepting an invalid configuration temporarily offers a balance between these two measures. Often a series of moves will result in a valid solution. This is beneficial when shapes pass over each other in order to reach enclosed areas in the layout or other shapes. The acceptance of an invalid layout requires a penalty term in the evaluation function. The penalty expression needs to be carefully designed balancing between layout compaction and overlap generation. According to Davis (1991) penalties are a less efficient guide to the search than a decoding algorithm that avoids producing constrained results.

When the search process operates on an encoding the packing rules applied by the decoding algorithm guarantee that all solutions considered in the search process are valid. There has been much speculation on whether this is beneficial with respect to the transmission of specific layout to the next generation and the next state in the neighbourhood respectively. The literature is reluctant so far to give a satisfactory answer to this problem. The different solution approaches have not been compared with each other. Since much of their performance strongly depends on the packing task with respect to the formulation of the objective and the shapes involved it is not sufficient to judge their performance purely on the basis of the packing densities achieved. This emphasises the need for commonly accepted benchmark tests and problems (section 7.3).

#### 7.1.3 Computation Time

The decoding method has a great influence on the computational effort of the hybrid algorithm. The importance of computation time in a certain nesting task depends on the respective application. Meta-heuristics are computationally very expensive due to the high number of function evaluations. This results in long run times especially in irregular problems, where geometric computations required for the nesting process are time intensive. Type and implementation of geometric algorithms contribute to the computation time, especially when a high accuracy for the shape approximation and description is used.

#### 7.1.4 Shape Representation

The representation of the shapes to be placed is strongly related to the strategy chosen to tackle the nesting task. Two main methods can be distinguished in the irregular examples in the literature. In approaches where the allocation in the object is found on the basis of a scanning process, shapes are represented as matrices. The second option is the description in the form of geometric primitives such as polygons and circles and implies that geometric routines are used to compute the relation between items in the layout. The approximation of arbitrary items as they occur in the textile and metal industry by concave polygons or the convex hull raises the issue of accuracy. For instance, a popular method in the nesting process is the clustering of two polygons using the convex hull or some outer boundary of the configuration in the subsequent nesting steps (Burke and Kendall, 1999). The convex hull is not an accurate description of the partial layout, but might be sufficient for the generation of a layout of acceptable quality. The basic question to resolve in this context is how much accuracy is needed in terms of layout quality and how much is affordable in terms of computation time.

The issue of shape representation reflects on the encoding technique. In a hybrid algorithm the domain knowledge is stored outside the meta-heuristic part, since an additional procedure is used for decoding into the phenotype. In approaches, that do not involve a decoding algorithm, the geometry of the figures necessarily needs to be considered in the data structure (e.g. Jain and Gea, 1998).

### 7.2 Meta-heuristics

Despite some comparisons with problem-specific search processes and local optimisation methods such as hill-climbing, only a few attempts have been made so far to compare the performance of various meta-heuristics in the area of packing. Burke and Kendall (1999) and Leung et al. (1999) carried out some research in this area. The first work indicated that tabu search and simulated annealing outperform genetic algorithms. Leung et al. (1999) implemented a genetic algorithm which was better than their approach with simulated annealing.

Most researchers in the area of genetic algorithms seem to take a successful search process of their particular implementation for granted. The operation of the genetic operators with respect to the outcome of the search process is hardly verified. This is of paramount importance where novel encoding structures and problem-specific operators are proposed. A verification step would normally be quite straightforward and easy to implement. As most genetic algorithms make use of both genetic operators, omitting the cross-over operation reveals its impact on the final outcome and on the course of the search process. Despite the simplicity, this technique is not a part of the 'standard test tools' researchers use in this area. So far, it was only applied by Falkenauer (1996) and is referred to as naïve evolution.

A second, at least as powerful tool for the performance evaluation of genetic algorithms, is random search. Executed over the same number of iterations as the meta-heuristic algorithm, it allows the quality of the search to be established. Since the genetic operators as well as the neighbourhood moves are intended to guide the search process to good solution areas in the extremely large solution space, the outcome of a search, which conducts a pure random exploration reveals how well this objective has been met.

### 7.3 Benchmarks

The discussion of performance comparison reveals the lack of comparisons with known benchmarks. Despite some effort by two on-line libraries (Hopper, 2000), there is no test suite available which could enable comparisons between algorithms intended for packing problems. Although some researchers acknowledge and regret this fact in their work, no further work has been done in this area. Performance evaluation mainly continues to only consider 'self-made' test problems, which are not publicly available in most cases. A commonly agreed test suite benefits the development of algorithms as well as the industrial user, who has to select the most appropriate packing method considering various criteria. Solution quality and computation time, are only two out of many criteria to be considered.

A number of standard packing methods is also of advantage for performance comparison, especially in the area of rectangular packing, a large number of simple heuristics exist which could be applied as such a standard method for this purpose. Simple heuristics are easy to implement and achieve very dense layouts under certain conditions. Meta-heuristics are expected to perform comparatively better in terms of solution quality. Therefore it may seem to be a waste of time. However, even a relative comparison to a standard method is a useful and valid measure for comparisons between more complex algorithms. Although the task of determining a benchmark method may be more difficult in irregular packing, some of the heuristic search techniques (Albano and Sappupo, 1980; Oliveira et al., 2000) have proved to be flexible and extremely powerful on a variety of test cases. Therefore even a benchmark method for irregular packing could be established.

In order to keep test problems flexible regarding parameters such as problem size, aspect ratio of items or availability of known optimum solution, a further task is the design and implementation of problem generators (Hopper, 2000). Whereas this is certainly simpler for the rectangular strip and bin packing problems, a careful consideration of parameters to determine the irregular packing task in certain applications is necessary for the irregular case.

## 8. Conclusions

Evolutionary algorithms are the most widely investigated meta-heuristics in the area of cutting and packing. The work done to date almost exclusively uses order-based and tree-structure representations. Comparisons as far as they have been possible given the limited number of benchmark problems in the area have shown, that order-based genetic algorithms achieve layouts of similar density as the approaches that include layout information into the encoding structure.

Simulated annealing approaches concentrate on irregular packing problems. A considerable quantity of work remains to be done in the area of tabu search.

Due to the lack of benchmarking, it is difficult to decide which method is better suited to approach packing problems. To-date only a few attempts have been made to compare meta-heuristic techniques.

At present meta-heuristic techniques are usually not benchmarked against efficient heuristic methods. Where this has been done, the indications are that the heuristic techniques perform very well. Consequently, such comparisons should be regarded as necessary for further research in this field.

An alternative method for a provisional assessment of a meta-heuristic would be to use one of the commercial nesting packages. However, commercial organisations do not make the algorithms available and therefore they are not suitable for publication purposes in the academic literature.

A simple, but effective method to check to successful performance of meta-heuristic search processes is to compare them against random search. The performance gain achieved over random sampling of the search space could be used as an indicator for the effectiveness of the intelligent search algorithm. At the same time it should become standard practice to test novel cross-over operators through the application of naïve evolution.

Rotation is rarely considered for any of the irregular packing problems other than 90° steps. This is mainly due to the constraints imposed by the industrial application, e.g. textile and metal industry. However, a study into the impact of the rotation interval on the layout quality could be useful in a theoretical context.

## 9. Acknowledgements

We thank J. Turton and Dr. C. Valenzuela for their helpful comments on this paper.

## 10. References

Adamowicz M. & Albano A. (1976*). Nesting two-dimensional shapes in rectangular modules*. Computer Aided Design **8:** 27-33.

Albano A. & Sappupo G. (1980). *Optimal Allocation of two-dimensional irregular shapes using heuristic search methods.* IEEE Transactions on Systems, Man and Cybernetics, **SMC-10**: 242-248.

András P., András, A. & Zsuzsa, S. (1996). A genetic solution for the cutting stock problem. In Proceedings of the *First On-line Workshop on Soft Computing,* 87-92. Nagoya University.

Blazewicz J., Hawryluk P. & Walkowiak R. (1993). *Using a tabu search approach for solving the two-dimensional irregular cutting problem.* Annals of Operations Research **41**: 313-327.

Bounsaythip C., Maouche S. & Neus M. (1995). Evolutionary search techniques application in automated lay-planning optimization problem. In Proceedings of the *IEEE Conference on SMC*, 4497-4502.

Bounsaythip C. & 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, 3425-3430.

Burke E. & Kendall G. (1999). Applying Simulated Annealing and the No Fit Polygon to the Nesting Problem. In Proceedings of *the World Manufacturing Congress*, 27-30. Durham, UK.

Coffman, E. G., Garey, M. R. & Johnson, D. S. (1984). Approximation algorithms for bin-packing - an updated survey. In Ausiello, G., Lucertini, M. and Serafini, P. (eds.), *Algorithms Design for Computer Systems Design*, 49-106. Springer, Vienna.

Coffman E. G. & Shor P. W. (1990). *Average-case analysis of cutting and packing in two dimensions.* European Journal of Operational Research **44**: 134-144.

Corno F., Prinetto P., Rebaudengo M. & Sonza Reorda M. (1997). Optimising area loss in flat glass cutting. In Proceedings of *Second International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, *GALESIA '97*, 450-455. Glasgow : University of Strathclyde, UK.

Dagli C. H. & Poshyanonda P. (1997). *New approaches to nesting rectangular patterns.* Journal of Intelligent Manufacturing **8**: 177-190.

Davis L. (1985). Applying adaptive search algorithms to epistatic domains. In Proceedings of the *9th International Joint Conference on Artificial Intelligence*, 162-164. Los Angeles.

Davis L. (1991). *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York.

Dighe R. & Jakiela M. J. (1996). *Solving Pattern Nesting Problems with Genetic Algorithms Employing Task Decomposition and Contact Detection.* Evolutionary Computation **3**: 239-266.

Dowsland, K.A. (1993). *Some experiments with simulated annealing techniques for packing problems.* European Journal of Operational Research **68**: 389-399.

Dowsland K. A. & Dowsland W. B. (1992). *Packing problems*. European Journal of Operational Research **56**: 2-14.

Dowsland K. A. & Dowsland W. B. (1995). *Solution approaches to irregular nesting problems.* European Journal of Operational Research **84**: 506-521.

Dowsland W. B. (1985). *Two and three dimensional packing problems and solution methods.* New Zealand Journal of Operational Research **13**: 1-18.

Dowsland W. B. (1991). *Three-dimensional packing-solution approaches and heuristic development.* International Journal of Production Research **29**: 1673-1685.

Dyckhoff H. (1990). *Typology of cutting and packing problems*. European Journal of Operational Research 44, 145-159.

Dyckhoff H. & Finke U. (1992). *Cutting and Packing in Production and Distribution.* Springer Verlag, Berlin.

Eglese R. W. (1990). *Simulated annealing. A tool for operational research.* European Journal of Operational Research **46**: 271-281.

Faina L. (1999). *Application of simulated annealing to the cutting stock problem.* European Journal of Operational Research **114**: 542-556.

Fowler, R. J., Paterson, M. S. & Tatimoto, S. L. (1981). *Optimal packing and covering in the plane are NP-complete*. Information Processing Letters **12**: 133-137.

Fujita K., Akagji, S. & 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*, Part 1, 477-484. Albuquerque, NM, USA.

Gary Parker R. (1995). *Deterministic Scheduling Theory*. Chapman Hall.

George J. A., George, J. M. & Lamar, B. W. (1995). *Packing different-sized circles into a rectangular container.* European Journal of Operational Research **84**: 693-712.

Glover F. & Laguna M. (1993). Tabu search. In Reeves, C. R. (ed.), *Modern Heuristics for Computational Problems.* Basil Blackwell: Oxford.

Golden, B. (1976). *Approaches to the cutting stock problem*. AIIE Transactions **8**: 265-274.

Gwee B. H. & Lim M. H. (1996). *Polyominoes tiling by a genetic algorithm.* Computational Optimisation and Applications **6**: 273-291.

Falkenauer E. (1996). *Hybrid grouping genetic algorithm for bin packing.* Journal of Heuristics **2**: 5-30.

Han G. C. & Na S. J. (1996). *Two-stage approach for nesting in two-dimensional cutting problems using neural network and simulated annealing.* Proceedings of the Institute of Mechanical Engineers, Part B, Journal of Engineering Manufacture **210** B6: 509-519.

Healy P. & Moll R. (1996). *A local optimisation-based solution to the rectangle layout problem.* Journal of the Operational Research Society **47**: 523-537.

Herbert E. A. & Dowsland K. A. (1996). *A family of genetic algorithms for the pallet loading problem.* Annals of Operations Research **63**: 415-436.

Hinxman A. I. (1980). *The trim loss and assortment problems.* European Journal of Operational Research **5**: 8-18.

Hopper, E. (2000). *Two-dimensional packing utilising evolutionary algorithms and other meta-heuristic methods.* Ph.D. diss., Cardiff University.

Hopper E. & Turton B. C. H. (1997). Application of Genetic Algorithms to Packing Problems - A Review. In Chawdry, P. K., Roy, R. and Kant, R. K. (eds.), In Proceedings of the *2nd On-line World Conference on Soft Computing in Engineering Design and Manufacturing*, 279-288. Springer Verlag, London.

Hopper E. & Turton B. C. H. (1999). *A genetic algorithms for a 2D industrial packing problem.* Computers in Engineering 37, 375-378.

Hopper E. & Turton B. C. H. (2000). *An Empirical Investigation of Meta-heuristic and Heuristic Algorithms for a 2D Packing Problem.* European Journal of Operational Research **128/1**: 34-57.

Hwang S. M., Cheng Y. K. & Horng J. T. (1994). On solving rectangle bin packing problems using genetic algorithms. In Proceedings of the *1994 IEEE International Conference on Systems, Man and Cybernetics*. Part 2, 1583-1590. San Antonio, TX, USA.

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

Ismail H. S., Hon K. K. B. (1995). *Nesting of two-dimensional shapes using genetic algorithms.* Proceedings of the Institution of Mechanical Engineers **209** Part B: 115-124.

Jain P., Fenyes P. & Richter R. (1988). *Optimal blank nesting using simulated annealing.* Journal of Mechanical Design - Transactions of the ASME **114**: 160-165.

Jain S. & Chang Gea H. (1998). *Two dimensional packing problems using genetic algorithms.* Engineering with Computers **14**: 206-213.

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

Kämpke T. (1988). *Simulated annealing: use of a new tool in bin-packing.* Annals of Operations Research **16**: 327-332.

Kröger B. (1995). *Guillontineable bin-packing: A genetic approach.* European Journal of Operational Research **84**: 645-661.

Kröger B., Schwenderling P., Vornberger O. (1991a). Parallel genetic packing of rectangles. In *Parallel Problem Solving from Nature 1st Workshop*, 160-164. Springer Verlag: Berlin.

Kröger B., Schwenderling P. & Vornberger O. (1991b). Genetic packing of rectangles on transputers. In Welch, P. (ed.), *Transputing 2*, 593-608, IOS Press: Amsterdam.

Kröger B., Schwenderling P. & Vornberger O. (1993). *Parallel genetic packing on transputers.* In: J. Stender (Ed.), Parallel Genetic Algorithms: Theory and Applications, IOS Press, Amsterdam, pp. 151-185.

Lai K. K. & Chan W. M. (1997). *An evolutionary algorithm for the rectangular cutting stock problem.* International Journal of Industrial Engineering **4**: 130-139.

Leung T. W., Yung C. H. & Chan C .K. (1999). Applications of genetic algorithm and simulated annealing to the 2-dimensional non-guillotine cutting stock problem*. IFORS '99*. Beijing China.

Liu D. & Teng H. (1999). *An improved BL-algorithm for genetic algorithm of the orthogonal packing of rectangles.* European Journal of Operational Research **112**: 413-419.

Lodi A., Martello S. & Vigo D. (1999). *Approximation algorithms for the oriented two-dimensional bin packing problem.* European Journal of Operational Research **112**: 158-166.

Marques V. M. M., Bispo C. F .G. & Sentieiro J. J. S. (1991). A system for the compaction of two-dimensional irregular shapes based on simulated annealing. In Proceedings of the *1991 International Conference on Industrial Electronics, Control and Instrumentation - IECON '91*, 1911-1916. Kobe, Japan.

Martello S. & Toth P. (1990). *Knapsack problems: algorithms and computer implementations.* John Wiley & Sons Ltd., Chichester.

Oliveira J. F., Gomes A. M. & Ferreira S. (2000). *A new constructive algorithm for nesting problems.* OR Spektrum **22:** 263-284.

Petridis V. & Kazarlis S. (1994). Varying quality function in genetic algorithms and the cutting problem. In Proceedings of the *IEEE Conference on Evolutionary Computation*, 166- 169.

Pargas R. P. & Jain R. (1993). A parallel stochastic optimisation algorithm for solving 2D bin packing problems. In Proceedings of the *9th Conference on Artificial Intelligence for Applications*, 18-25.

Poshyanonda P. & Dagli C. H. (1993). Genetic neuro-nester for irregular patterns. In Proceedings for *Artificial Neural Networks in Engineering Conference (ANNIE '93)*, vol. 3, 825-830. ASME Press, New York.

Rahmani A. T. & Ono N. (1995). An evolutionary approach to two-dimensional guillotine cutting problem. In Proceedings of the *IEEE Conference on Evolutionary Computation*, 148-151.

Ratanapan K. & Dagli C. H. (1997a). An object-based evolutionary algorithm for solving rectangular piece nesting problems. In Proceedings of the *IEEE Conference on Evolutionary Computation*, *ICEC*, 989-994. IEEE, Piscataway, NJ, USA.

Ratanapan K. & Dagli C. H. (1997b). *An object-based evolutionary algorithm for solving irregular nesting problems.* In Proceedings for *Artificial Neural Networks in Engineering Conference (ANNIE '97),* vol. 7, 383-388. ASME Press, New York.

Ratanapan K. & Dagli C. H. (1998). An object-based evolutionary algorithm: the nesting solution. In Proceedings of the *International Conference on Evolutionary Computation 1998, ICEC '98*, 581-586. IEEE, Piscataway, NJ, USA.

Rayward-Smith V. J. & Shing M. T. (1983). *Bin packing*. Bulletin of the Institute of Mathematics and its Applications **19**: 142-146.

Reeves C. (1993). *Modern Heuristics for Computational Problems.* Basil Blackwell: Oxford.

Sarin S. C. (1983). *Two-dimensional stock cutting problems and solution methodologies.* ASME Transactions, Journal of Engineering for Industry **104**: 155-160.

Smith D. (1985). Bin-packing with adaptive search. In Grefenstette (Ed.), *Proceedings of an International Conference on Genetic Algorithms and their Applications*, 202-206. Lawrence Erlbaum.

Sweeney P. E. & Paternoster E. (1992). *Cutting and packing problems: a categorised, application-orientated research bibliography.* Journal of the Operational Research Society **43**: 691-706.

Theodoracatos V. E. & Grimsley J. L. (1995). *Optimal packing of arbitrarily-shaped polygons using simulated annealing and polynomial-time cooling schedules.* Computer Methods in Applied Mechanics and Engineering **125**: 53-70.

Whelan P. F. & Batchelor B. G. (1993). *Automated packing systems: Review of industrial implementations*. SPIE, Machine Vision Architectures, Integration and Applications **2064**: 358-369.

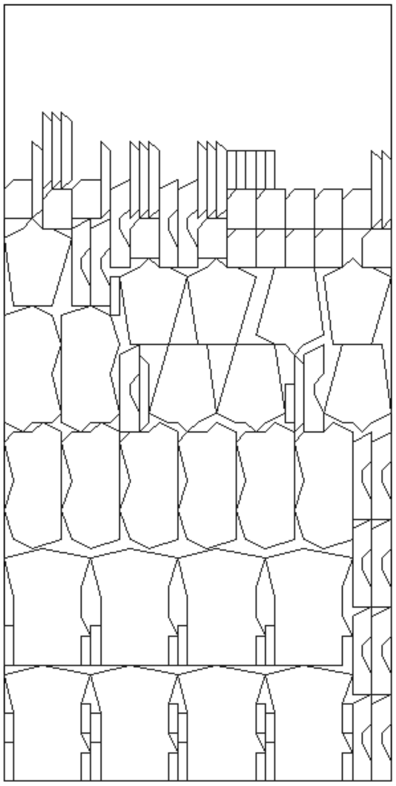

Figure 1: Irregular packing problem from the textile industry

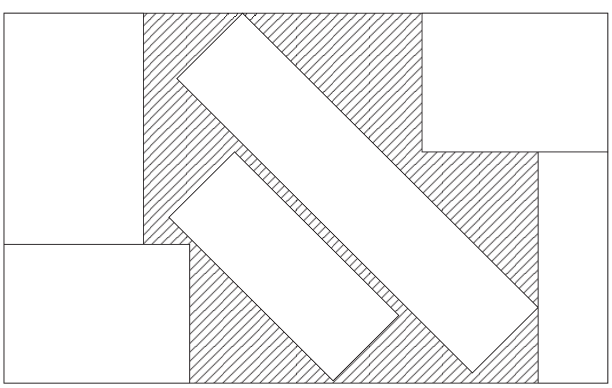

Figure 2: Non-orthogonal layout

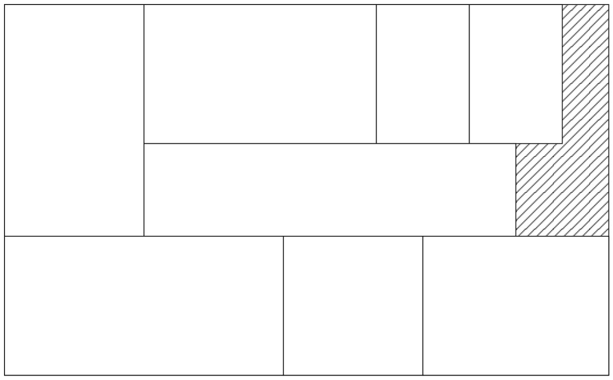

Figure 3: Guillotineable layout

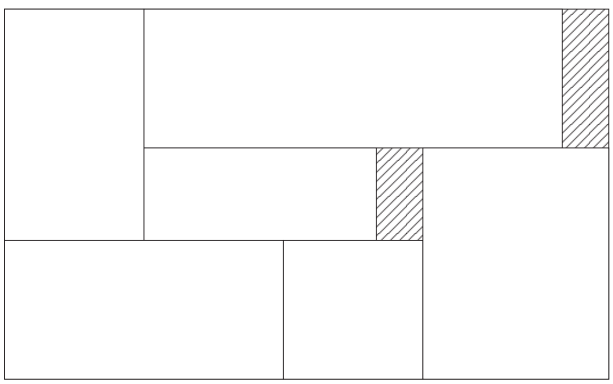

Figure 4: Non-guillotineable layout

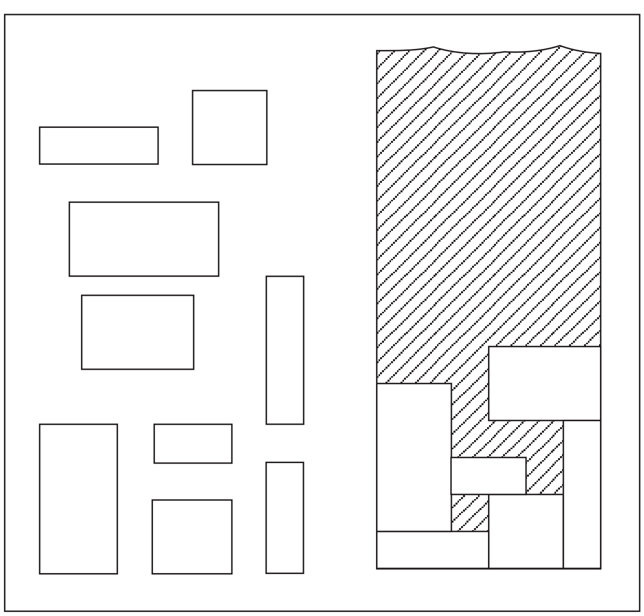

Figure 5: 2D strip packing problem

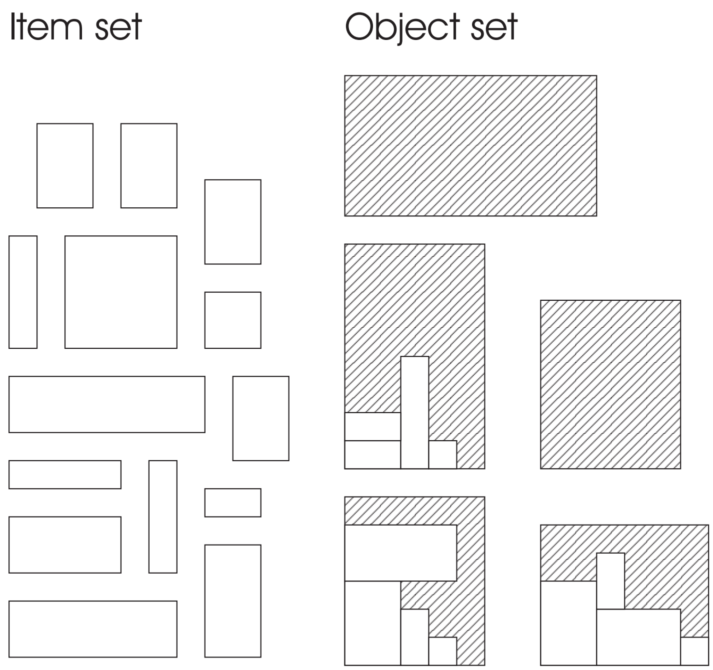


Figure 6: 2D bin packing problem

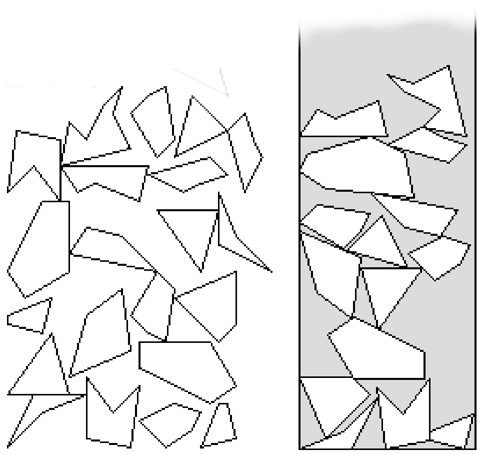

Figure 7: 2D irregular strip packing problem

Table 1: Reviews and surveys on packing problems in the literature

| **authors** | **topic** | **classification of packing problems** |
|---|---|---|
| Dyckhoff and Finke (1992) | analysis of large variety of problems | Dyckhoff classification |
| Sweeney and Paternoster (1992) | more than 400 problems including books, dissertations and working papers | dimension, solution methodology, special topics |
| Golden (1976) | 2D cutting stock problems | solution methodology |
| Hinxman (1980) | 2D trim-loss and assortment problems | dimension, solution methodology |
| Rayward-Smith and Shing (1983) | 1D and 2D bin packing | dimension |
| Sarin (1983) | 2D cutting stock problems | solution methodology |
| Coffman et al. (1984) | bin packing | type of bin packing, dimension |
| Dowsland (1985) | 2D and 3D rectangular problems | problem type, dimension |
| Coffman and Shor (1990) | 2D regular packing problems | on-line, off-line; probabilistic analysis |
| Haessler and Sweeney (1991) | 1D and 2D cutting stock problems | dimension, solution methodology |
| Dowsland (1991) | 3D problems | solution methodology |
| Dowsland and Dowsland (1992) | 2D and 3D packing problems, mainly regular | problem type, dimension |
| Whelan and Batchelor (1993) | industrial implementations of automated packing systems for 2D irregular packing problems | application, focus on leather industry |
| Dowsland and Dowsland (1995) | 2D, irregular packing problems | methods for clustering, packing, computational geometry |
| Hopper and Turton (1997) | 2D and 3D, regular and irregular packing problems and genetic algorithms | geometric characteristics of items, dimension |

Table 2: Hybrid genetic algorithms for non-guillotineable 2D packing problems

| | **Smith (1985)** | **Jakobs (1996)** | **Liu and Teng (1999)** |
|---|---|---|---|
| **problem** | packing of single closed bin; 90° rotation | strip packing 90° rotation | strip packing 90° rotation |
| **objective** | maximise number of items in the bin | minimise height | minimise height |
| **representation** | permutation | permutation | permutation |
| **fitness** | ratio of packed to unpacked area | remaining area and height | remaining area and height |
| **cross-over** | OX (1point) | OX (1point) | OX (2point) |
| **mutation** | random reordering of string; rotation | inversion, swap of 2 elements, rotation | inversion, swap of 2 elements, rotation |
| **decoder** | Slide algorithm Skyline algorithm | BL-algorithm[1] | improved bottom-left algorithm[2] |

[1] BL = Bottom Left heuristic; based on sliding principle

[2] based on sliding principle; referred to as BLLT-routine throughout this

Table 3: Hybrid genetic algorithms and evolutionary algorithm for non-guillotineable 2D packing problems

| | **Hopper and Turton (1999, 2000)** | **Leung et al. (1999)** | **Dagli and Poshyanonda (1997)** | **Lai and Chan (1997)** |
|---|---|---|---|---|
| **algorithm** | **GA** | **GA** | **GA** | **EA** |
| **problem** | strip packing; rotation | strip packing; no rotation | strip packing; 90° rotation | packing of a single closed bin; no rotation |
| **objective** | minimise height | minimise trim loss | minimise height | minimise trim loss |
| **representation** | permutation | permutation | permutation | permutation |
| **fitness** | trim loss, height | trim loss | height, width | trim loss |
| **cross-over** | PMX (2 point) | PMX, CX, OBX, OX (1, 2 point) | OX | none |
| **mutation** | swap of 2 elements | swap of 2 elements | inversion | swap of 2 elements; hill-climbing during allocation process |
| **decoder** | 'Bottom-Left-Fill Algorithm' | 'Difference Process Algorithm' | sliding algorithm and ANN to match free areas with item | placement closest to the bottom-left corner |

Table 4: Comparison of the genetic algorithms for non-guillotineable 2D packing problems - approaches with encodings including layout information

| | **Kröger et al. (1991a, b; 1993)** | **Herbert and Dowsland (1996)** | **Herbert and Dowsland (1996)** |
|---|---|---|---|
| **problem** | strip packing 90° rotation | pallet loading; 90° rotation | pallet loading; 90° rotation |
| **objective** | minimise height | maximise number of boxes placed | maximise number of boxes placed |
| **re-presentation** | directed binary tree | 1D binary string | 2D binary matrix |
| **fitness** | height, width | number of boxes placed; penalty for overlap | number of boxes placed; penalty for overlap |
| **cross-over** | problem-specific | uniform (1 and 2 point) | problem-specific |
| **mutation** | variation of set of edges, orientation, priority | bit change | bit change |
| **decoder** | encoding structure + BL-condition | none | none |

Table 5: Comparison of the genetic algorithms for guillotineable 2D packing problems using tree representations

| | **Hwang et al. (1994)** | **Kröger (1995)** | **Rahmani and Ono (1995)** | **András et al. (1996)** |
|---|---|---|---|---|
| **problem** | strip packing<br>90° rotation | strip packing;<br>90° rotation | packing of a single closed bin;<br>no rotation | packing of a single closed bin;<br>no rotation |
| **objective** | minimise height | minimise height | minimise waste | minimise area |
| **re-presentation** | directed binary tree | string representing tree structure | tree representation | tree representation |
| **fitness** | bounding rectangle to be close to square; excess width penalised | height | utilisation ratio | packing density |
| **cross-over** | PMX and uniform | exchange of sub-trees under certain conditions | exchange of sub-tree under certain conditions | exchange of sub-trees |
| **mutation** | rotation, swap of two items; move of operator, complement of operator | swapping of sub-trees, inversion of cut-line or rectangle orientation, rotation of rectangle | inversion of cut-line; shifting of a cutting position | combined with cross-over: repair of infeasible configurations |
| **decoder** | combination of 2 items: position in containing larger rectangle is bottom-left justified | none | none | none |

Table 6: Comparison of the genetic algorithms for guillotineable 2D packing problems using order-based representation

| | **Hwang et al. (1994)** | **Corno et al. (1997)** |
|---|---|---|
| **problem** | strip packing<br>90° rotation | packing of a single closed bin; 90° rotation; constraints: defects, distances, etc. |
| **objective** | minimise height | maximise utilisation |
| **representation** | permutation | permutation with flags for orientation, placement, geometry |
| **fitness** | height | utilisation ratio |
| **cross-over** | PMX | OBX |
| **mutation** | rotation, swap of 2 elements | swap of 2 elements, flip rotation, flip placement criterion flag |
| **decoder** | level-oriented FF[3] and BF[4] | heuristic algorithm that considers all technological constraints |

[3] FF = First Fit heuristic

[4] BF = Best Fit heuristic

Table 7: Hybrid genetic algorithms for 2D irregular packing problems

| | **Fujita et al. (1993)** | **Jakobs (1996)** | **Dighe and Jakiela (1996)** | **Dighe and Jakiela (1996)** |
|---|---|---|---|---|
| **problem** | convex polygons only; free rotation | polygons; 90° rotation | polygons; free rotation | polygons; free rotation |
| **objective** | minimise waste | minimise height | maximise density | minimise height |
| **re-presentation** | permutation | permutation | binary tree | permutation |
| **fitness** | waste, distance to origin, width; penalty for overlap | height, remaining area | packing density | height |
| **cross-over** | OX (1point) | OX (1point) | exchange of sub-trees | OX (1 point) |
| **mutation** | random removal and reinsertion of one element | inversion; exchange of 2 elements; rotation | none | random removal and reinsertion of 1 element |
| **decoding** | placement in leftmost-lowest position; local minimisation algorithm | placement of enclosing rectangles in BL-position; then shift algorithm; overlap omitted | determined by low-level GA: pairwise clustering of nodes items; overlap omitted | determined by low-level GA: vertical sliding from top of object into partial layout; overlap omitted |

Table 8: Comparison of the genetic algorithms for 2D irregular packing problems

| | **Petridis and Kazarlis (1994)** | **Bounsaythip and Maouche (1995)** | **Bounsaythip and Maouche (1997)** |
|---|---|---|---|
| **problem** | polygons; no rotation | irregular items from textile industry; considering one strip only; 90° rotation | irregular items from textile industry; 90° rotation |
| **objective** | minimise height | minimise length of strip | minimise waste |
| **representation** | binary string encoding position in layout | string consisting of 4 sub-strings; represents a single shape or cluster | binary tree |
| **fitness** | dynamic; overlap, used area; x-position of shapes | packing density | density of the strip layout formed by each tree in relation to the overall layout |
| **cross-over** | multi-point, binary | interchange of sub-strings | exchange of sub-trees |
| **mutation** | binary | swap of sub-string within one individual | change of operator; swap of 2 elements; deletion of 1 element |
| **decoding** | none | none | best relative position of 2 clusters determined by low-level algorithm together with operator info in tree; overlap omitted |

Table 9: Hybrid genetic algorithms for 2D irregular packing problems

| | **Poshyanonda and Dagli (1993)** | **Ismail and Hon (1995)** | **Gwee and Lim (1996)** |
|---|---|---|---|
| **problem** | irregular items; 90° rotation | rectilinear shapes; 90° rotation | polyominoes; 90° rotation |
| **objective** | minimise height | minimise area used | optimal solution |
| **representation** | permutation | multi-parameter string including relative position and rotation of both items; binary | permutation |
| **fitness** | height | density of packing, penalty for overlap | number of boundary edges; number of void and overlapping cells; number of items without overlap |
| **cross-over** | OX | binary (1point) | PMX |
| **mutation** | inversion | bit change | |
| **decoding** | ANN to match scrap areas with item + sliding algorithm; overlap omitted | set of heuristic rules | circular placement starting from the centre of the object |

Table 10: Genetic algorithms operating on the phenotype for 2D irregular packing problems

| | **Jain and Gea (1998)** | **Ratanapan and Dagli (1997b, 1998)** |
|---|---|---|
| **problem** | irregular items 90° rotation | strip packing; free rotation |
| **objective** | minimise layout area | minimise height |
| **representation** | 2D matrix | 2D geometric objects |
| **fitness** | used area; total moment of inertia | packing density |
| **cross-over** | exchange of items in sub-area of matrix | none |
| **mutation** | rotation; swap of 2 items; random new position for 1 item | series of hill-climbing, mutation and recombination operations |
| **decoding** | none | none |

Table 11: Comparison of the simulated annealing approaches for 2D rectangular packing problems

| | **Dowsland (1993)** | **Faina (1999)** | **Leung et al. (1999)** |
|---|---|---|---|
| **problem** | pallet loading with identical and non identical boxes; 90° rotation | strip packing; guillotineable and non-guillotineable; no rotation | strip packing; no rotation |
| **objective** | finding a feasible arrangement of a fixed number of boxes | minimise area used | minimise trim loss |
| **representation** | position in layout; overlap allowed | permutation | permutation |
| **fitness** | minimise number of overlapping boxes | packing density | height |
| **neighbourhood move** | set of position composed of width and length of boxes | swap position of two elements | swap position of two elements |
| **cooling schedule** | geometric | geometric | geometric |
| **decoder** | none | left-justified routines considering guillotine constraint | 'Difference Process Algorithm' |

Table 12: Comparison of the simulated annealing approaches for 2D irregular packing problems

| | **Jain et al. (1988)** | **Marques et al. (1991)** | **Theodoracatos and Grimsley (1995)** | **Han and Na (1996)** | **Burke and Kendall (1999)** |
|---|---|---|---|---|---|
| **problem** | polygons; clustering of 2 and 3 identical shapes; free rotation | polygons; textile industry; 90° rotation | 1. circles; 2. polygons; free rotation | circles, polygons with enclosures; improvement of existing layout 90° rotation | polygons; strip packing |
| **objective** | minimise waste | minimise area | maximise number of circles | minimise height | minimise height |
| **repre-sentation** | 2D layout overlap permitted | 2D layout overlap omitted | 2D layout overlap permitted | 2D layout overlap permitted | permutation overlap omitted |
| **fitness** | waste; penalty for overlap | area of enclosing rectangle; sum of distances from centre; proximity to neighbours | waste; penalty for overlap | overlap area; moment of area in the bottom-left direction | area used by each 'row' in layout |
| **neigh-bour-hood move** | translation, rotation | translation, rotation: large and small perturbation; reflection | 1. translation 2. translation and rotation | translation, rotation; swap of 2 elements | swap 2 adjacent items; swap 2 random items; re-order polygons according to type |
| **cooling schedule** | geometric | geometric | polynomial-time | geometric | linear; geometric |
| **decoder** | none | none | none | none | routine using NFP and local search |