Two-dimensional Packing utilising Evolutionary Algorithms and other Meta-Heuristic Methods

A Thesis submitted to the University of Wales
for the Degree of
Doctor of Philosophy

by

Eva Hopper

University of Wales, Cardiff
School of Engineering

May 2000

DECLARATION

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any other degree.

Signed.....(candidate)

Date.....

This thesis is the result of my own investigations, except where otherwise stated.
Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed.....(candidate)

Date.....

I hereby give consent for my thesis, if accepted, to be available for photocopy and for inter-library loan, and for the title and summary to be made available to other organisations.

Signed.....(candidate)

Date.....

ACKNOWLEDGEMENTS

I would like to thank all of the people who contributed their time and support to me during my studies in Cardiff:

The staff of the Circuits and Systems Group, especially my supervisor Dr. B. Turton, for giving me the opportunity, arranging my grant and his invaluable support and guidance throughout my work, the Head of Department, Professor Tasker and the former Head of Department Dr. Horrocks for their support and guidance, and the secretarial staff Louise and Sheila for their organisational support.

The financial support from EPSRC and the WDA for providing the CASE studentship as well as Cardiff University for the A. A. Read scholarship.

J. Turton for her very useful comments on my thesis.

My fellow PhD students for their inspiration to my studies, in particular Michael, who was always willing to help out with any software and hardware problems and also maintained, updated and continuously extended the computing equipment to provide sufficient computing power for simulations and calculations.

To my family, especially my parents and my brother Peter, for their immense and never ending help during my stay abroad. A very special thanks to my boyfriend Wolfgang, who encouraged, helped and advised me so much during this work.

I could not mention everybody who helped, to all the people I have left out: Thank You.

SUMMARY

2D packing problems are a variety of combinatorial problems with a very large solution space that cannot normally be exhaustively searched. A series of hybrid approaches have been developed in this work to solve 2D rectangular and irregular strip packing and 2D bin packing problems. As the recent literature encourages the use of genetic algorithms in particular, along with other meta-heuristic methods for their solution, the main focus of this investigation has been on the application of evolutionary algorithms.

The meta-heuristic search methods have been implemented as hybrid algorithms, which use simple packing algorithms in the decoding stage for the layout generation. Apart from genetic algorithms, the meta-heuristic methods include simulated annealing, naïve evolution and stochastic optimisation. Hill climbing, random search and the downhill simplex method have also been applied to the same packing tasks.

The performance of the algorithms has been tested on a number of different sized packing problems. Some rectangular and irregular packing problems do exist in the literature, however the lack of benchmark problems has necessitated the implementation of a problem generator for rectangular packing tasks. For comparison, two commercial nesting packages were included and applied to the test problems. A heuristic packing method, which allows the user to generate rectangular layouts of very high quality, is proposed as a test method to benchmark packing algorithms.

Meta-heuristic techniques produced good results compared with published algorithms. Genetic algorithms were better suited to the solution of small to medium sized problems, at the expense of significant computation time. Simulated annealing produced very dense layouts but with an even greater increase in computation time. All of these techniques produced a limited improvement in the solution quality compared with standard packing techniques.

Table of Contents

Appendices

List of Figures

List of Tables

Nomenclature, List of Abbreviations, Glossary

1. Introduction

Packing problems are optimisation problems that are concerned with finding a *good* arrangement of multiple *items* in larger containing regions (*objects*). This type of problem is encountered in many areas of business and industry and is forms part of the combinatorial problems found in operational research. The usual objective of the allocation process is to maximise the utilisation and hence to minimise the "wasted" material.

1.1 Industrial Applications

The reduction of production cost is one of the major issues in manufacturing industries in an ever more competitive world where products are available almost instantly from anywhere on the globe. High material utilisation is of particular interest to industries with mass-production, since small improvements of the layout can result in large savings of material and considerably reduce production cost. Cutting and packing problems are encountered in many industries. The complexity of the problem and the solution approach depend on the geometry of the items to be placed and the constraints imposed. Whereas the wood, glass and paper industry are mainly concerned with the cutting of regular figures, irregular, arbitrary shaped objects are to be packed in the ship building, textile and leather industry.

The duration of the layout generation is also relevant in industrial applications. As manual generation can easily require several man-days, the packing time is an important economic factor. It is therefore understandable that industries are looking into ways to automate the packing process. Figure 1.1 summarises the steps typically involved in the industrial nesting process.

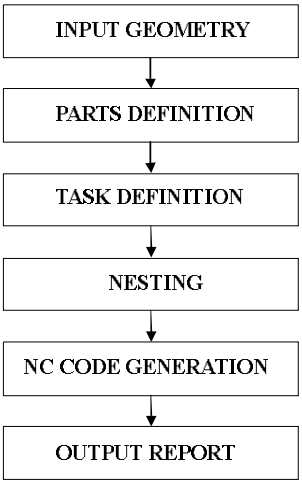

Figure 1.1: Typical steps involved in the nesting process

- input geometry from other CAD systems via DXF, DWG etc. file format or create geometry with the aid of built-in CAD module
- define parts: material, thickness, part quality, etc.
- define nesting task: number and size of object, part clearance, cutting process etc.
- automatic, manual or interactive nesting on single or multiple objects
- generate order in which parts are to be cut; output NC code for cutting process
- generate report including layout details, e.g. scrap percentage, length of strip material or sheet utilisation; cutting time and cost

The research community therefore started some time ago to investigate automatic solution approaches for cutting and packing tasks. The first scientific publications in this area appeared in the 1950s when a number of researchers highlighted the relationship between the stock cutting problem and linear programming theory, which belongs to the standard methods of operational research (Dantzig, 1951; Kantorovich and Zagaller, 1951). These early approaches concentrated only on rectangular problems.

The majority of industrial packing tasks involve irregular shapes and stock material. Certain material properties usually impose additional constraints on the packing task (section 1.1.1 to 1.1.4). Automatic packing systems were first introduced into the textile and shipbuilding industry in the 1970s and worked in interactive mode (Catastini et al., 1976; Øian et al., 1976). In the meantime a large number of fully automatic nesting systems have become available. Some are suitable for the general packing task; others are designed to meet the special requirements of certain industries (section 4.5).

Depending on the complexity of the shapes in irregular nesting tasks, the computation of layouts may require a long time. But in some applications packing problems need to be solved in real-time. As a consequence, the computation time is of paramount importance in the solution method applied. A great number of so-called on-line algorithms were proposed for this dynamic version of the packing problem (Coffman et al., 1984; Runarsson et al., 1996). The geometric algorithms in irregular nesting problems are complex and usually apply a certain degree of approximation to reduce the computation time. In cases where the solution of the nesting task is not coupled with the timing of the manufacturing process, computation time is a minor issue. When the nesting task is solved off-line, a higher degree of accuracy is possible. It is common practice in layout generation to utilise the available computational power efficiently by using overnight runs.

Recently, a class of solution methods found in the field of artificial intelligence have attracted the interest of the algorithm designers in this area (section 1.2). Inspired by nature, meta-heuristic search principles apply strategies, which work successfully in the natural process, to find good solutions. Modelled on evolution and annealing, the artificial counterparts consider a number of solutions and condense them to a few good ones (section 3.3). As the terminology suggests, the methods describe search principles rather than straightforward computation. Computation times are expected to be high which makes these methods only suitable for a limited range of nesting applications.

1.1.1 Packing in the Sheet Metal Industry

The sheet metal industry has to deal with regular as well as irregular nesting problems. Certain constraints regarding the material properties, the cutting process and scheduling aspects distinguish the packing task in this field from other industries. Apart from reducing the wastage to a minimum, there are a number of other factors that decide the final layout of the parts.

Material Properties:

The raw materials are either available in the form of sheet material with fixed dimensions or as coiled material with a fixed width. This has an impact on the objective of the nesting task. In the first case the shape of a possible remnant decides whether it can be used in a future nesting task or has to be treated as wastage. For coiled material this is usually not a constraint.

As the materials have inhomogeneous properties such as grain orientation, the number of possible orientations the parts can be nested in may be limited. This entirely depends on the application and the further processing of the shapes. If there are no bending operations to follow, the parts can be rotated in any direction; otherwise a specified angle with respect to the grain orientation needs to be applied.

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.

In certain applications, the geometry of the parts may allow special layout configurations. Especially when the order list contains many congruent shapes, rotation along a certain axis can allow two shapes to be nested efficiently in a cluster that can be repeated in the layout. Grouping is a similar technique, where two or more shapes are combined to a cluster, which can be used more than once for the layout generation. Clusters offer sub-optimal configurations of a limited number of shapes. Their repeated use increases the speed of the layout generation once a combination is computed.

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 distance.

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 or 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.

1.1.2 Packing in the Textile Industry

The textile industry usually utilises coiled material and hence is mainly concerned with the strip packing problem. The material properties limit the layout generation. As the fabric often has certain directional properties and a pattern, the orientation of the parts is usually restricted to rotation intervals of 0° and 180°. In many cases it may not be possible to mirror the parts as the fabric has different properties at the other side.

1.1.3 Packing in the Leather Industry

The nesting task in the leather industry is very complex since both the parts to be nested as well as the objects, the so-called hides, are highly irregular. As leather is a natural material, as opposed to the manufactured ones used in the textile and sheet metal industry, the hides consist of areas having various qualities. The quality difference can be due to defects and colour differences. The nesting process therefore needs to match the parts with their respective quality zones on the hide. Usually, image processing takes place prior to the nesting process to determine the shape and the quality of the hides. As this nesting task is very complex and needs to consider many specific requirements specially designed nesting packages are available to the leather industry (section 4.5).

1.1.4 Industries with Rectangular Packing Problems

The group of rectangular packing problems appears in the paper, wood and glass industry. The cutting technique, which involves shear cuts, imposes a characteristic constraint on layouts in this area. The packing patterns are required to be guillotineable, such that the parts can be obtained by straight cut through the remaining layout only (section 2.2). Packing in the glass and wood industry also needs to consider various quality ranges and defects of the raw material. Rectangular packing tasks can occur as strip and bin packing problems.

Strip Packing:
The paper industry is mainly concerned with strip packing problem, as the raw material is available in the form of rolls. Hence the packing process aims at reducing the height of the layout.

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 rather than rolls. Bin packing is not restricted to the rectangular case. In industrial applications this problem is referred to as stock cutting (section 2.3) such as in the glass, wood and metal industry. The objective usually is to find the set of sheets 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. Regular bin packing is frequent in three dimensions including problems such as container and pallet loading.

Non-Guillotineable Layouts:

Industrial applications that involve non-guillotineable layouts are less frequent. The 2D stock cutting problem occurs as a partial problem in pallet and container loading. A commonly adopted approach is to reduce the 3D problem to two dimensions. Once a set of boxes for a certain layer has been determined, the layer represents a 2D non-guillotineable packing problem. The loading patterns are then created for each layer separately (Bortfeldt, 1994). Processor allocation can also be regarded as a 2D packing problem (Hwang, 1997). In some branches of the metal industry the rectangular packing problem can occur in the non-gullotineable form when different cutting methods such as laser and plasma cutting are used to obtain the parts.

1.2 Solution Approaches

For the solution of small packing problems deterministic methods such as linear programming have been developed. Since these methods are exact, they find the optimal solution. For problems of higher complexity, the solution space is larger due to a higher number of possible combinations. In this case it is not possible to calculate the optimal solution in reasonable computing time. In order to approach these problems heuristic techniques can be used to find near-optimal solutions at reasonable computational cost. Solutions will be within a tolerable range of deviance from the optimal solution. The way the placement has to take place is described by a set of rules. Since these rules are tailored to a particular packing task, the problem-specific heuristic algorithms can only be applied successfully to that particular problem type.

More flexibility is offered by general heuristic methods, so-called meta-heuristics, which describe general search principles rather than special rules. Some of them are inspired by optimisation processes in nature, such as evolutionary algorithms and simulated annealing (section 3.3). The search through the large solution space is guided by problem-specific information. The quality of the solution depends on the implementation of this knowledge and the parameters applied to control the search process. The success of the meta-heuristics can be explained by their great flexibility in taking into account problem-specific

constraints and their good trade-off between solution quality and computational effort as opposed to a full search.

In order to decide how profitable general heuristic search methods are as a solution approach to packing problems, their performance needs to be evaluated against the cost of manually generated layouts or problems-specific heuristics, which can both offer solutions of high quality. With the increase in computing power meta-heuristic search techniques have become highly competitive in complex packing problems with large solution spaces. Genetic algorithms in particular, which allow exploration and manipulation within a large search space, have frequently been applied in the literature (section 4.3).

1.3 Outline of the Thesis

This investigation has concentrated on 2D rectangular and irregular packing problems. The rectangular packing task has been considered in the form of strip packing and bin packing. The major objective has been to develop a set of meta-heuristic solution approaches sufficiently flexible to tackle various packing tasks.

Chapter 2 introduces a classification for cutting and packing problems. Since this project focuses particularly on the research of meta-heuristic methods, the basic concept of genetic algorithms and simulated annealing problems is briefly introduced in chapter 3 along with some other heuristic search principles.

Chapter 4 provides an overview of the research activities in the area of 2D and 3D packing problems. It outlines the current state of the research in the application of problem-specific heuristics and meta-heuristics to packing problems. Particular emphasis is hereby put on genetic algorithms. The chapter finishes with an overview of commercial software packages available for industrial nesting problems and concludes with a summary of the major aspects important for the application of genetic algorithms to packing problems.

Chapter 5 introduces the solution approaches developed in this work for 2D strip and bin packing problems. Several algorithms are introduced for the generation of the rectangular test problems. The outcome of a comprehensive performance evaluation also includes a comparison with some test problems from the literature and two commercial packages and is discussed in chapters 6 to 8.

Chapter 9 summarises the major findings of this investigation and gives recommendations for possible future work in this area.

2. Cutting and Packing Problems

Packing problems arise in a large number of situations. The variety of the problems is as large as their application areas including disciplines like management science, engineering, mathematics, computer science or operational research with different industries incorporating different constraints and objectives (Figure 2.1).

Due to this diversity of problems and application areas similar packing problems appear under different names in the literature. Dyckhoff (1990) proposed a systematic classification of packing problems. Packing problems are combinatorial problems of high complexity, which are NP-complete (section 2.4). Depending on the complexity and the practical requirements different solution strategies are applied.

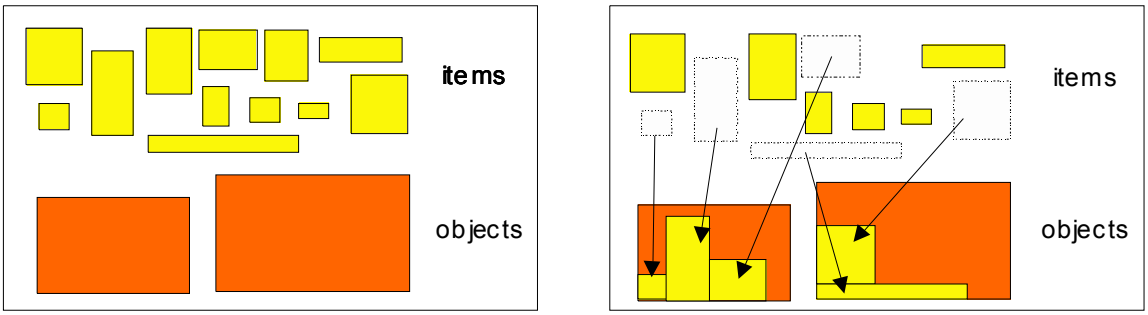

Figure 2.1: Packing task

2.1 Dyckhoff Classification

Analysing different 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 and hence research and application of solution approaches Dyckhoff (1990) identified common characteristics and properties and systematically classified packing problems.

2.1.1 Classification and Definitions

In general, packing problems belong to the field of geometric and combinatorial computing. Dyckhoff (1990) 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. The other group covers abstract “cutting and packing” problems including non-spatial dimensions such as weight, time or financial dimensions (Figure 2.2).

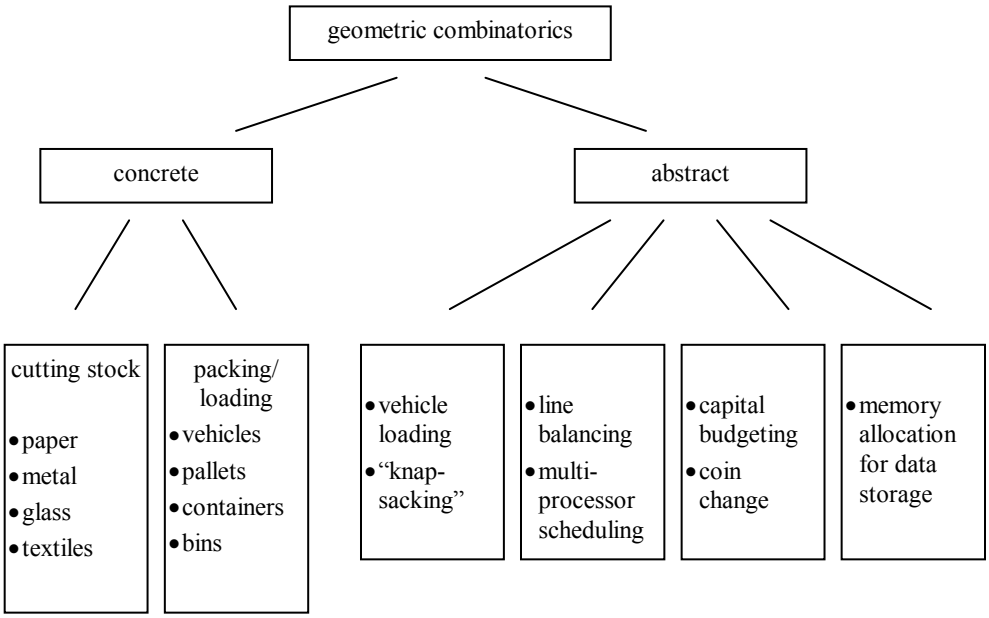

Figure 2.2: Phenomenology of cutting and packing problems (Dyckhoff, 1990)

Cutting and packing problems describe patterns consisting of geometric combinations of large objects (e.g. stock) and small items (e.g. order book). 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 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.

2.1.2 Typology

Dyckhoff's classification system describes four important characteristics of packing and cutting problems. These characteristics and the values they can take on are summarised in Table 2.1. Examples for various problem types are given in (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 has to be assigned. Four types of assignment are possible, the two most important ones are differentiated in Dyckhoff's typology:

- a selection of small items has to be combined to patterns such that a non-trivial (corresponding) pattern is assigned to *each* large object (Beladeproblem B)
- all small items have to be combined to patterns that are then assigned to a proper *selection* of large objects (Verladeproblem V).

The assortment not only considers the shape of the objects and items but also their number. The different types for large and small objects are stated in the table below. The term "figure" refers to items and objects being uniquely determined by their shape, size and orientation.

Table 2.1: Dyckhoff's typology of packing and cutting problem (Dyckhoff, 1990)

Characteristic	Symbol	Description
dimensionality	1 2 3 N	one-dimensional two-dimensional three-dimensional N-dimensional with N > 3
kind of assignment	B V	all objects and a selection of items a selection of objects and all items
assortment of large objects	O I D	one object identical figures different figures
assortment of small objects	F M R C	few items (of different figures) many items of many different figures many items of relatively few different (non-congruent) figures congruent figures

2.2 Additional Classifications

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 non-convexities.

In 2D packing, the following layout types can be distinguished on the basis of the geometry of the items to be packed. 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 2.5). Orthogonal cutting additionally distinguishes between guillotineable (Figure 2.3) and non-guillotineable (Figure 2.4) layouts. Packing of irregular shapes is known as nesting e.g. in the shipbuilding industry and as marker layout problem in the textile industry (Figure 2.6).

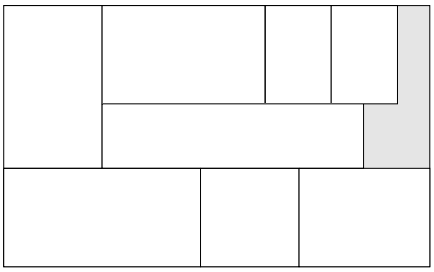

Figure 2.3: Orthogonal, guillotineable packing

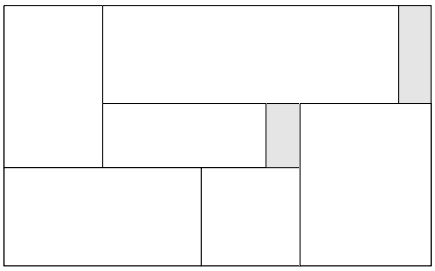

Figure 2.4: Orthogonal, non-guillotineable packing

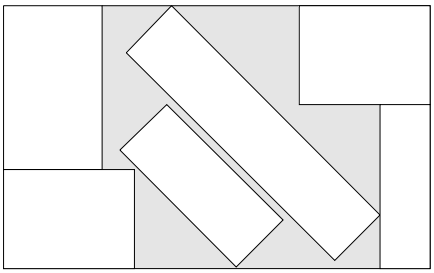

Figure 2.5: Non-orthogonal packing

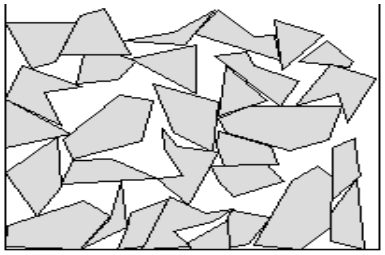

Figure 2.6: Nesting of irregular shapes

Although cutting and packing problems may have a common structure, which allows a classification like the one introduced by Dyckhoff (1990), they can differ to a great extent in the constraints imposed by various industries. Two general criteria common to all problems are that the items have to fit completely onto the objects and must not overlap. Special industrial needs are expressed as additional constraints concerning the details about possible allocation of the items. Some items are required to be placed only onto a certain area of the object (e.g. shoe manufacturing) or in certain directions (e.g. textile industry). In metal cutting for instance the items have to be placed with a minimum distance between each other. In the wood, glass or paper industry the subsequent cutting process requires the layouts to be guillotineable. Some examples of packing problems involving different geometric primitives are listed in Table 2.2.

Table 2.2: Examples of geometric shapes in 2D packing problems

geometric shape	**example for application**
circle, ellipse	loading of containers with pipes
polygons: convex or concave	packing of irregular shapes in the metal industry
free-form: parts consist of a series curved and straight line segments	marker layout problem in the textile industry
shapes with complete or partial enclosures	shipbuilding

2.3 Types of Packing Problems

Packing problems occur in various application areas involving different constraints and objectives. In the following some of the most important problems are defined briefly. Larger industrial problems can as well appear as combinations of two or more of these basic types. Further details about individual problems and examples can be found in (Dowsland and Dowsland, 1992; Dyckhoff and Finke, 1992; Hinxman, 1980).

Cutting stock problem:
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).

Trim-loss problem:
The trim-loss problem concerns the allocation of the order list onto the given stock sheets. The order list describes the set of pieces that must be allocated. The objective is to minimise the total cost of the stock sheets needed to fulfil the order.

Assortment problem:
This 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.

Knapsack problem:
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.

Bin packing:
1D 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. As the height of the bins is fixed, the packing process aims at minimising the number of bins. Using items of different width results in 2D bin packing. The concept of bin packing appears in more abstract versions in other application areas e.g. assembly line balancing (Falkenauer and Delachambre, 1992).

Loading problem:
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 (manufacturer's pallet loading problem), as well as from the distributor's side, where the pallet has to be packed with non-identical items (distributor's pallet

loading problem). Container loading is similar to pallet loading, though in practical applications the two variants of the loading problem can be distinguished by their constraints.

2.4 Packing and NP-Hardness

One of the most important conventions in complexity theory is the definition of problem classes. Combinatorial problems can be stated as decision problems where a solution corresponds to a correct yes or no response. An optimisation problem is converted by posing the question of whether of not there exists a feasible solution which has an objective function value equal or superior to a specified threshold (Garey and Johnson, 1979; Gary Parker, 1995). According to the algorithm complexity, the following problem classes can be distinguished.

Class P: Class P describes problems, which can be solved by algorithms that require at most a polynomial function of the instance size. The membership in this class means that even large problem instances can be solved with exact routines. Increases in problem size normally have a small impact on the computation time. However, the solution of large problems might become impractical using conventional computer systems. Thus, it may be still worthwhile to search for better algorithms that are of lower polynomial order.

Class NP: The definition of this class is based on the solutions to a problem rather than the algorithm. If the correct solution (i.e. a yes response) can be checked for feasibility in polynomial time, a problem is known as non-deterministic polynomial (NP). Although problems in this class are 'easy' to verify, they are not 'easy' to solve. By this definition, the class NP contains the class P.

In order to understand the class of NP-hard and NP-complete problems, first the concept of reduction is introduced. A problem P reduces to a problem P' if any algorithm that solves P' also solves problem P provided an appropriate conversion is found. Polynomial reductions are the most important ones in this context. A problem P polynomially reduces to P', if a polynomial time algorithm for P' implies the existence of a polynomial time algorithm for P.

Class NP-Hard: Problems to which all members of NP polynomially reduce are referred to as NP-hard. Thus, a polynomial time algorithm for an NP-hard problem, would produce an algorithm for every member of NP at the same time. Many optimisation problems are known to be NP-Hard.

Class NP-Complete: Problems that are in class NP-hard and also in class NP are called NP-complete. As all the members of class NP-complete are decision problems, it does not contain optimisation problems in the strict sense. Optimisation problems are NP-hard, but have analogue decision problems which are NP-complete.

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. It should be noted that adding constraints could have the opposite effect on the search space. In specific circumstances it may make the search space easier to 'navigate' and a problem which in its general form is regarded as NP-complete can then be solved with an exact algorithm.

According to the aforementioned 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. 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 (section 2.5). Thus, by giving up solution quality, computational efficiency can be gained.

This point of view is often adopted in cutting and packing and has lead to the development of approximation algorithms, i.e. heuristics. Knowing that a given packing problem is NP-complete is extremely significant in the design of an automatic packing strategy, since it helps researchers to direct their effort towards those approaches that have the highest likelihood of leading to a useful solution.

2.5 Solution Approaches

Packing problems belong to the field of geometric combinatorial computing. The complexity of the geometric properties of the items influences the magnitude of the problem and the solution approach. The search for an optimal solution for the placement of rectangular pieces is simpler involving fewer possibilities in terms of the orientation of the pieces. With the number of possible orientations and arrangements being larger for irregular shapes, the search space is much larger for this problem class and gets more complex when concave features are involved.

The solution approaches to packing problems can be divided into two groups, deterministic and heuristic methods. Deterministic methods are exact techniques that guarantee to find the optimal solution for a problem. They are usually based on linear programming or enumeration approaches such as branch-and-bound or dynamic programming. These methods are explained in more depth in (Sedgewick, 1992).

For problems of higher complexity these techniques become inefficient due to the vast number of possible solutions that have to be evaluated in enumeration based approaches. Since conventional methods fail to produce an optimal solution in reasonable amount of computing time, heuristic approaches have been developed. Unlike exact methods heuristic techniques do not guarantee to find an optimal solution to a problem. They only produce solutions that are “good enough” or near optimal, but

at a reasonable computational cost. Solutions will be within a tolerable range of deviance from the optimal solution. Heuristic techniques applied to packing problems can be problem-specific consisting of a set of placement rules. Since these rules are tailored to a particular packing task the disadvantage of problem-specific heuristic algorithms is that they can only be applied successfully to that particular problem type. On the other hand they may be extremely efficient in terms of computational cost.

Packing problems can also be seen as combinatorial optimisation problems. The application of search techniques allows finding solutions, which satisfy predefined criteria. A combinatorial optimisation problem can be formulated in the following way:

With

- the solution space S being the finite set of all solutions,
- the cost function C being a real valued function defined on members of S and the quantity to be minimised (or maximised depending on the objective)
- the problem is to find a solution or state, $i \in S$, which minimises (or maximises) C over S.

In general, search techniques operate as iterative improvement techniques attempting to find the global minimum in the solution space. Downhill techniques also operate as iterative improvement techniques. As they only accept moves that result in an immediate improvement in the objective function, they may only find a local minimum, which may be far from the global minimum. If an optimisation method is to converge to a global minimum and be insensitive to the starting point, then it must either enumerate all the local minima and then select the global minimum (multi-start method) or be able to accept moves that increase the objective function in a controlled manner.

In contrast to local descent algorithms, which may get trapped in a local minimum, other heuristic search methods can overcome this problem by accepting an increase in the value of the cost function and allowing for uphill moves. The acceptance of such moves depends on problem-specific information used to guide the search process. Heuristic search techniques, which are able to search large, constrained solution spaces due to built-in mechanisms allowing for uphill moves, are genetic algorithms, simulated annealing, tabu search, artificial neural networks, Langrangean Relaxation and others. A description of these techniques can be found in (Reeves, 1993; van Laarhoeven, 1987).

3. Optimisation with Heuristic and Meta-heuristic Search

The quality of the layout, which is constructed using heuristic placement algorithms, strongly depends on the sequence in which the rectangles are presented to the routine. Thus the full state of solutions consists of every permutation of the items with each one packed in every feasible position and orientation. Since the number of combinations is too large to be explored exhaustively in reasonable amount of time, heuristic and meta-heuristic algorithms are used as a more efficient search strategy. Efficiency of a heuristic search technique is dependent on how domain-specific knowledge is exploited and its ability to overcome combinatorial explosion.

A tree structure is a graphical representation of the numerous solutions that exist for a combinatorial problem. At each stage of the search algorithm a new intermediate solution is generated for each of the remaining elements not yet allocated. Depending on the strategy applied to search the tree several procedures can be distinguished. Depth-first, breath-first, hill-climbing, beam search and best-first belong to the basic strategies (Winston, 1984). The branch-and-bound technique, discrete dynamic programming and the A* procedure are more sophisticated search methods involving backtracking and heuristic information. Other optimisation techniques applied to simple packing problems are linear and mathematical programming.

Large combinatorial problems typically have extremely large solution spaces, so that traditional search techniques are totally impractical. Many approaches are proposed in the literature involving meta-heuristic search principles (section 4.3 and 4.4). These techniques result in a good, however, not necessarily optimal solution within reasonable computing time. This chapter describes the basic principles of the heuristic and meta-heuristic optimisation methods used in this investigation.

3.1 Hill-Climbing

Hill-climbing is a local search technique performing moves in the neighbourhood of the current state. If the quality of the new solution is better than the current one, this move is accepted and the search continues from there. If the neighbouring state does not result in an improvement, the move is rejected and the search continues from the current state. The main disadvantage of this method is that the search process might get trapped in a local minimum, which is different from the global one. A useful variation of simple hill-climbing considers a series of moves from the current state and selects the best one as the next state. This method is known as gradient search or steepest-ascent hill-climbing.

3.2 Downhill Simplex Method

The downhill simplex method was extended by Nelder and Mead (Press et al., 1995) for multi-dimensional function optimisation. One of the major advantages of this method is that it only requires function evaluations and not derivatives. This method is not very efficient in terms of number of function evaluations and therefore is mainly suitable for smaller dimensions.

The method makes use of the geometrical concept of a simplex. The geometrical figure of the simplex depends on the number of dimensions of the optimisation (N) and describes a shape with N+1 vertices. In two dimensions the simplex is a triangle and in three it is a tetrahedron.

The downhill simplex method is started with N+1 points that describe the initial simplex. It then attempts to find a way downhill through the complex N-dimensional landscape until it reaches a (local) minimum. This is done by a series of iterations which consist of moving the highest (worst) point of the simplex through the opposite face of the simplex to a lower point. These steps are called reflections. If a reflection has been successful, i.e. if a lower point than the best point so far has been found, the algorithm tries to expand the simplex in this direction (expansion). If the reflection fails and the method cannot eliminate the highest point, it will try a contraction step around the lowest (best) point. The minimisation routine ends when a certain tolerance for the decrease in the function value has been reached.

3.3 Meta-Heuristic Search Strategies

In order to overcome the main disadvantage of local search algorithms such as hill climbing, whose weakness lies in the inability to escape from local minima, more sophisticated heuristic search strategies have been designed. This can mean the temporary acceptance of a state of lower quality. A meta-heuristic is an iterative master process that guides the operations of subordinate heuristics (Voss et al., 1999).

Various meta-heuristic search principles have been developed during the last twenty years. Some of them have been inspired by nature and are modelled on processes such as evolution and annealing. Genetic algorithms, simulated annealing and tabu search are the most widely applied meta-heuristics for large combinatorial problems. The meta-heuristic search process may manipulate a single solution (e.g. simulated annealing) or a collection of solutions (e.g. genetic algorithms) at each iteration. The subordinate heuristic can be high (or low) level procedures, local search or a construction method (Voss et al., 1999).

In the area of packing it is very common to use hybrid approaches combining meta-heuristics with heuristic algorithms. The task of the meta-heuristic is to search a good ordering of the items. A placement routine is then needed to interpret the permutation and evaluate its quality.

3.3.1 Genetic Algorithms

Genetic algorithms are optimisation and search procedures that operate in a similar way to the evolutionary processes observed in nature. They are based on Darwin's theory of evolution and simulate natural selection and genetics. They are well suited for complex optimisation problems with a large search space. As in nature genetic algorithms attempt to find new and better solutions to a problem by improving present solutions. The search is guided towards improvement applying the principle known as "survival of the fittest". This is achieved by extracting the most desirable features from a generation of solutions and combining them to form the next generation.

The quality of each solution is measured by a fitness function. According to their fitness values individuals are selected for reproduction usually using a weighted probability function. Hence each individual will contribute to the next generation in proportion to its fitness. The motivation is to continue this process through a number of generations in order to reach convergence on optimal or near-optimal solutions. Mimicking the natural evolution process genetic algorithms work with probabilistic rather than deterministic rules. The probabilistic nature of genetic algorithms becomes obvious when looking at the genetic operators such as reproduction, cross-over and mutation.

The origins of genetic algorithms can be traced back to the late 1950s (Bäck et al., 1997). In the early 1970s, genetic algorithms were developed further by Holland and his colleagues at the University of Michigan (Holland, 1975). Together with "evolutionary programming" (Fogel et al., 1966), "genetic programming" (Koza, 1992) and "evolutionary strategies" (Rechenberg, 1973) they belong to the group of "evolutionary algorithms". Although genetic algorithms are an iterative improvement technique they still remain heuristic strategies and cannot guarantee to find an optimal solution. The basic concepts of genetic algorithms are described briefly in the following using simple binary encoding. They can also be applied to other encodings requiring appropriate modifications. Further theoretical and practical details can be found in (Davies, 1991; Goldberg, 1989; Mitchell, 1996; Fogel, 1998).

3.3.1.1 Problem-Specific Design Variables

The problem-specific design variables describe the nature of the problem and thus reflect its specific characteristics. These decisions involve the problem representation along with suitable genetic operators.

Representation

The representation technique, i.e. the encoding of the solution space is of paramount importance for the successful operation of genetic algorithms. First of all, the parameters of the objective function influencing the optimisation problem need to be identified and coded. The classic approach uses binary coding where the parameters are represented by strings of 0's and 1's. The parameters are then

combined resulting in a multi-parameter string - each representing a possible solution to the problem. These multi-parameter strings correspond to chromosomes in natural systems.

The classical genetic algorithm involves binary strings as genotypes that are manipulated by the genetic operators. This representation becomes less effective for more complex problems such as combinatorial problems. In this case the representation can be numerical. Additionally, order-based chromosomes are proposed (Davis, 1991; Goldberg, 1989). The encoding technique can be further adapted to the nature of the problem implementing special data structures where an individual can consist of several chromosomes rather than a single chromosome. A good representation of the problem reduces the effort at coding and decoding but also demands more complex genetic operators in the sequel.

Order-based encodings are especially suitable for combinatorial problems with a permutation representing the complete set of elements. As the standard cross-over and mutation techniques would result in invalid solutions modified genetic operators are required. Techniques appropriate for this data structure are stated along with the standard operators.

Initial Population

The initial population is the first set of strings generated before the search process is started. Usually, the strings are chosen randomly. Alternatively, the initial population can include a high-quality solution generated with heuristic rules. This technique is known as seeding and may help the genetic algorithm to find better solutions. The size of the population is usually fixed.

Fitness Function

The fitness function provides a measure for the goodness of the individuals of a population with respect to the optimisation problem. The goodness of an individual is the main criterion for the quality-driven selective decisions in genetic algorithms. Before the reproduction process the quality of each member in the population is evaluated in the context of the problem using the fitness function. The optimisation problem is described mathematically in the so-called objective function.

The simplest way to state the fitness of an optimisation problem is to use the objective function. In this case the fitness of a string is associated with the value of the objective function of this string. The main disadvantage of this method is that it cannot discriminate very well between good and bad chromosomes when the population converges to a set of similar chromosomes. Scaling or ranking procedures are superior in this case. Extending the fitness function by a so-called penalty function excludes infeasible solutions in a population from the reproduction process.

Genetic Operators

A set of genetic operators is applied to a population in order to generate a new and better generation of possible solutions. These operators are probabilistic and operate according to the “survival of the fittest”

principle. The following three operators are most frequently used. For complex problems further genetic operators were proposed which are described in connection with specific applications in section 4.3.

Cross-over

As in natural systems heredity is the motivation for cross-over. Useful features, that have been adapted in previous generations, shall be transmitted from the parents to the children. Once the members are selected for the reproduction process, the cross-over operator is applied in order to produce new offspring. During the cross-over features of two or more parents are combined to generate one or more offspring. As in nature the cross-over involves an exchange of sections of the parents' chromosomes. Therefore one or more cross-over points are randomly selected along the genetic strings. Swapping the determined parts of the parents' strings generates the new offspring. The example of the one-point cross-over is shown in Figure 3.1.

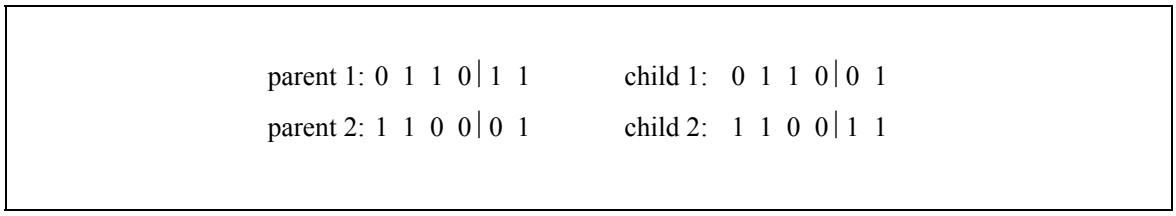

Figure 3.1: One-point cross-over

The rationale behind the cross-over operators is to produce new alternative solutions to the problem which are possibly better. As the parent chromosomes are usually the fitter individuals in the population according to the selection scheme and the creation of the new individuals is based on the combination of the their features, the idea is transfer good features to the next generation and achieve individuals which offer better solutions.

In cases where a more complex representation technique is used the cross-over operator needs to be adapted. The following operators are suitable in the recombination process when using a permutation as data structure to describe combinatorial problems (Goldberg, 1989).

- Partially Matched Cross-over: PMX crossover was developed by Goldberg (Goldberg, 1989); PMX can be implemented as two-point and one-point cross-over (Jakobs, 1996; Smith, 1985)
- Order Cross-over (OX) (Oliver et al., 1987; Goldberg, 1989)
- Order Based Cross-over (OBX) (Syswerda, 1991)
- Position Based Cross-over (PBX) (Syswerda, 1991)
- Cycle Cross-over (CX) (Goldberg et al., 1985)

Mutation

Mutation enables the recreation of genes, which are lost from the current population. This information cannot be gained if only the existing material is combined. The mutation operator is applied after the cross-over operation and randomly introduces minor modifications to the genetic strings of the

offspring. Mutation is implemented by altering the values of one unit in the genetic code with a certain probability. The principle of simple mutation is shown in Figure 3.2 using a binary coded string. The effect of this operator is to produce new unexplored areas in the search space and hence prevent the process from rapidly converging on a local optimum. Mutation therefore maintains the diversity of the population and allows a wider exploration. Since mutation also has a destructive effect as well it is only applied with a very low frequency.

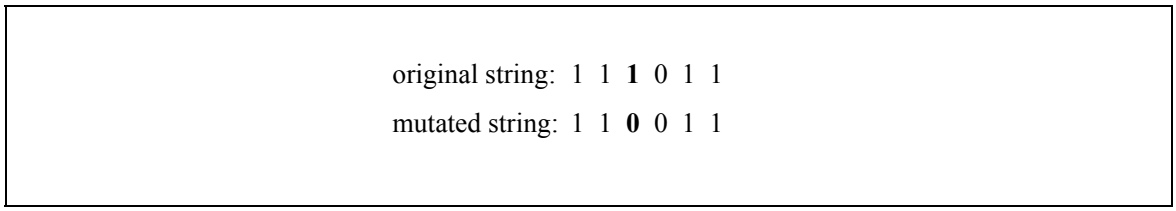

Figure 3.2: Principle of the simple mutation operator

Depending on the encoding other mutation operators can be superior to the simple one shown in Figure 3.2. In more complex encodings, usually an order relationship between adjacent loci on the strings (e.g. permutations) exists and reordering a sequence may be important. In this case the inversion operator can be applied, cutting out a section of a string and reinserting it in the reverse order (Goldberg, 1989). The following mutation operators were specifically developed for order-based encoding.

- Order Based Mutation (OBM) (Syswerda, 1991)
- Position Based Mutation (PBM) (Syswerda, 1991)

3.3.1.2 Generic Design Variables

The generic design variables concern the probability of the genetic operators and the reproduction scheme. The selection scheme and replacement strategy can be implemented in different ways.

Reproduction Scheme

During the reproduction process individuals are selected from the population and copied into a mating pool for the production of the next generation. The generation of the mating pool is a probabilistic process where the fitter individuals are more likely to receive more than one copy and the unfit ones are more likely to receive no copies. In the next step pairs of individuals are taken out of the pool and mated at random. The new generation can either replace the previous generation entirely (generational replacement) or partially (steady-state model).

With the selection of the strings taking place randomly according to their fitness, individuals with a higher value have a higher probability of contributing one or more offspring. The probability function that determines the selection process can reflect the actual fitness value, a scaled value or the rank. The reproduction of fit individuals provides the pressure for improvement. At the beginning of the search the values for each gene of an individual are randomly distributed. This produces a wide spread of

fitnesses. During the progress of the search certain values for each gene start to dominate. With the population converging, the range of fitnesses of a population reduces.

Proportional Selection

The proportionate selection technique chooses an individual for the reproduction process on basis of its objective function value. The probability of selection of an individual is proportional to its fitness. This technique can be implemented in different ways. The simplest one is known as roulette wheel selection, because it simulates the spin of a roulette wheel. Each string of a population is allocated a slot sized in proportion to its fitness. Spinning the weighted wheel corresponds to generating a random number and selecting a parent. As a fitter individual has a higher chance of being selected it thus contribute more copies to the mating pool.

In order to prevent premature convergence the reproductive trials should not be allocated in direct proportion to the raw fitness. Alternative methods re-map the raw fitness onto a new scale and include fitness scaling, fitness windowing and fitness ranking. Some of the most important selection techniques are briefly described in the following.

Rank Selection

If the fitness of some strings becomes very large, the fittest individuals can dominate the recombination process. This may lead to a loss of the genetic material. In order to prevent this reduction of genetic diversity ranking selection can be applied. After sorting the individuals according to their raw fitness value reproductive fitness values are assigned according to rank: linear (Baker, 1985) or exponentially (Davis, 1989). A parent string is selected with a probability proportional to its rank rather than the absolute fitness value. Consequently, the ratio between maximum and average fitness is normalised to a particular value. Hence the effect of one or two extreme individuals will be negligible. According to experiments carried out by Baker (1985) and Whitley (1989) ranking is superior to proportionate selection.

Scaling Mechanisms

Scaling mechanisms are applied to maintain a certain level of competition throughout the search process. Since there is a tendency that a few “super-individuals” may dominate the search process in the first generations, a take-over can be avoided by scaling down their fitness values. At the later stages when the population converges scaling up helps to differentiate between individuals with otherwise similar fitness values. Several scaling techniques exist including linear scaling and sigma truncation (Goldberg, 1989).

Termination criterion

Genetic algorithms are an iterative search technique. Being heuristic a convergence on the optimal solution cannot be guaranteed. Hence criteria for the termination of the process need to be stated. This

can be done by fixing the number of generations for the search process. An alternative method is to terminate when the process converges on an optimum, i.e. when no more improvements have been achieved for a specified number of generations.

3.3.1.3 A Brief Description of the Algorithm

Figure 3.3 shows the general description of a simple genetic algorithm. After the creation of the initial population the processes of reproduction, cross-over and mutation are repeated until the termination criteria is reached. The actual population at this stage represents the solutions to the problem. The characteristics of the genetic algorithm method are briefly summarised below.

Characteristics of genetic algorithms:

1. random choice as tool to guide, but not random search
2. searches from population of points, not a single point
3. uses objective function, not auxiliary knowledge
4. probabilistic, not deterministic transition rules

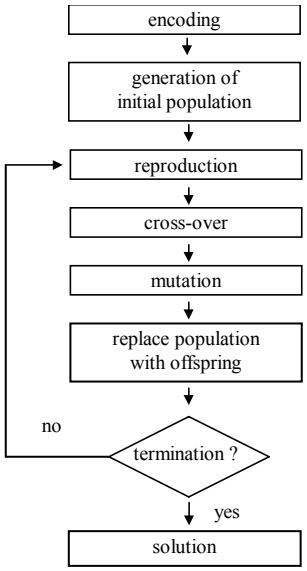

Figure 3.3: Flow diagram of genetic algorithm

3.3.2 Naïve Evolution

The rationale for naïve evolution is the same as for genetic algorithms. The algorithm operates according to the same principle, but does not apply cross-over to manipulate the search space. Only the mutation operator is used for the generation of the next population. A naïve evolution algorithm can be used to test the efficiency of the crossover operator in a genetic algorithm. Falkenauer (1998) applied this technique in experiments on 1D bin packing problems. Naïve evolution has been used throughout this work to establish the performance of the cross-over operators implemented in various approaches to strip and bin packing problems.

3.3.3 Simulated Annealing

Eglese (1990) investigated the application of simulated annealing as a tool for Operations Research. A number of researchers applied it to packing problems (section 4.4.1). Simulated annealing was introduced as an optimisation tool in the 1980's when the concept of physical annealing was first applied in combinatorial optimisation.

In a thermodynamic system low energy configurations of a solid are achieved by initially melting the substance and subsequently decreasing the temperature slowly. If the cooling process takes place too fast, the configuration does not achieve its lowest energy state and will be frozen into a locally optimal structure. Transferring this model to combinatorial problems the energy states correspond to the various feasible solutions and the energy of the system to the cost function to be minimised.

Simulated annealing can be seen as a variant of hill-climbing. Instead of only accepting neighbouring solutions that result in an improvement, also inferior solutions may be accepted with a certain probability. This probability depends on the increase in cost and a control parameter, i.e. temperature in physical annealing. The temperature refers to the analogy with physical annealing. The smaller the increase in the cost and the higher the temperature the more likely uphill moves will be accepted.

During the annealing process the temperature is gradually lowered according to the cooling schedule. This means the algorithm becomes more and more selective in accepting new solutions. At the end of the process only moves, which result in an improvement are accepted in practice. The search process terminates when it arrives at a lower bound for temperature or cost.

3.3.3.1 Problem-specific Design Variables

The problem-specific design variables describe the characteristics of the problem and consist of:

- representation
- fitness
- initial solution
- neighbourhood: i.e. set of states that can be reached from the current state

3.3.3.2 Generic Design Variables

The generic choices for the implementation of a simulated annealing algorithm are summarised in the annealing or cooling schedule and include:

- initial temperature
- length of Markov chain: number of moves the temperature is held constant based on the number of successful and unsuccessful moves
- decrement rule: mathematical expression describing how the temperature is lowered
- termination criterion: number of iterations, time, convergence, unsuccessful moves

4. Automated Packing – State of current Research

Over the last 30 years interest in packing and cutting problems has been very large. This is reflected by the great number of publications in this area. Cutting and packing problems arise in various industrial applications and are not restricted to the manufacturing industry. Packing problems for instance are encountered in operational research and the financial sector in a more abstract form. 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. For more complex packing tasks, i.e. irregular problems heuristic search methods have been applied successfully for their solution. 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 nesting packages have become available recently. They are specially designed to meet industrial requirements and usually include a variety of features directed at the manufacturing process.

The following review considers only work on concrete packing problems in the area of 2D rectangular and irregular strip packing as well as bin packing. Particular emphasis is hereby put on solution approaches involving genetic algorithms. A brief introduction into techniques for 1D and 3D packing problems is also given. With regard to the solution methods, the research activities on other meta-heuristic and heuristic search techniques are also included. In addition, an overview of commercial software packages in the area of 2D and 3D cutting and packing is given.

4.1 Surveys and Reviews

Over the years a considerable amount of literature on cutting and packing problems has been presented. A number of reviews tried to keep track of the recent developments in this area - a task that is virtually impossible due to the size of the subject. Despite the vast extent of the literature, two surveys attempt to cover the total area of cutting and packing. Dyckhoff and Finke (1992) developed a classification method, on which they base their analysis of the concrete and abstract packing problems. Sweeney and Paternoster (1992) chose the opposite approach and addressed the subject from the perspective of the solution approach. The problems encountered in the literature are grouped into three classes according to their solution methodologies consisting of sequential assignment heuristics[1], single-pattern

[1] set of rules determining the order and the orientation of the items

generating procedures[2] and multi-pattern generating procedures[3]. 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 4.1 provides an overview of the more recent reviews and surveys in the area of concrete packing problems.

Table 4.1: 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

4.2 Application of Heuristic Algorithms to Packing Problems

A number of exact and heuristic algorithms were proposed for regular packing problems. Some researchers developed exact methods, which solve 1D and 2D problems of moderate size to optimality such as the linear and dynamic programming models used by Gilmore and Gomory (1961, 1963 and

[2] dynamic-programming based algorithms, which attempt to reapply a single 'optimal' pattern configuration

[3] linear programming based algorithms, which consider interactions between pattern; the solutions are approximated, hence heuristic

1965). The majority of packing problems are constrained and hence more complex. With respect to practical packing tasks arising in industrial applications, the solution quality is not always the major issue. It is traded against ease of implementation and computational cost. Near-optimal layouts achieved in reasonable time are usually sufficient for industrial needs. Heuristic methods, which provide fast solution approaches, were therefore studied intensively in the literature.

4.2.1 Regular Packing Problems

A significant amount of research is devoted to regular packing problems. Most frequently rectangular packing has been investigated. Dowsland's (1985) survey provides an excellent overview of exact and heuristic algorithms for the 2D and 3D rectangular packing tasks. This review was extended by Dowsland and Dowsland (1992) to address a number of bin packing problems.

4.2.1.1 1D Strip and Bin Packing Problems

In 1D packing problems the width of the items is equal to the object width. Hence only a single dimension is significant for the packing process. Several variants of this problem are encountered in the literature covering bin packing (Coffman et al. 1984), knapsack problems (Rönnquist, 1995) and strip packing problems (Dyckhoff, 1981; Gilmore and Gomory, 1961, 1963). Typical industrial applications concern the cutting of paper, pipes and wood. More abstract problems in this area deal with process scheduling, timetabling, processor allocation and file distribution.

Since 1D problems have a small solution space compared to higher dimensional ones they can be solved to optimality at reasonable computational cost. Exact methods, which were successfully applied, are linear and dynamic programming and branch-and-bound techniques. A number of heuristic algorithms for larger strip packing as well as bin packing problems were used such as the First Fit (FF), Next Fit (NF) and Best Fit (BF) (Rayward-Smith and Shing, 1983; Coffman et al., 1984). As the name suggests the FF-routine places an item in the first bin with sufficient space. Whereas the FF-rule searches the list of bins from the beginning, the NF-algorithm starts at the previously used bin and continues with a new one if the item does not fit. The BF-algorithm searches through all available bins and chooses the one that results in the highest performance defined by an evaluation function. In particular, pre-ordering the items according to decreasing height in combination with the FF- and NF-algorithms improves the average performance of the simple placement routines. These routines are referred to as FFD and NFD respectively. Figure 4.1 and Figure 4.2 illustrate the operation of the FFD- and NFD-algorithms placing a sequence of items, numbered from A to G. After sorting them by decreasing height, the sequences are allocated with the FF- and the NF-routine to the bins.

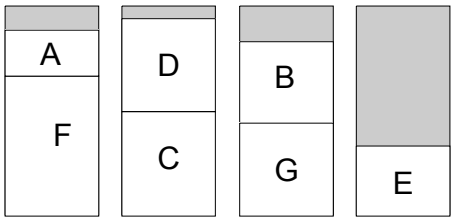

Figure 4.1: FFD-rule (First Fit Decreasing)

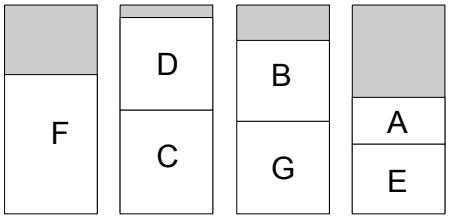

Figure 4.2: NFD-rule (Next Fit Decreasing)

4.2.1.2 2D Strip and Bin Packing Problems

The majority of the literature in this area concentrates on 2D rectangular packing problems. 2D packing tasks are either concerned with packing items into an object of fixed width minimising its height (Bengtsson, 1982) or into objects of fixed size minimising of the number of bins (Bengtsson, 1982; Berkey and Wang, 1987; Frenk and Galambos, 1987).

Many heuristic algorithms have been developed over the years to solve this problem. A fairly simple approach is the Bottom-Left-rule (BL), which places an item as near to the bottom of the strip as it will fit and then as far to the left as it can be placed at that bottom-most level. A number of placement rules which belong to the class of algorithms preserving bottom-left stability in the layout are described in section 5.5.4.1 to 5.5.4.4. A different approach is taken by the so-called level-algorithm, where the rectangles are placed in different rows of the strip in a left-justified manner. The packing is constructed as sequence of levels, each rectangle being placed so that its bottom side rests on one of these levels. The height is determined by the tallest rectangle at that level. The level technique is then combined with 1D rules like FF, NF or BF to determine at which level the next item is positioned. Again, pre-ordering according to height or width (e.g. FFDH or FFDW) can improve the average performance (Coffman et al., 1984). Figure 4.3 and Figure 4.4 highlight the operation of the FFDH and NFDH-algorithms, which apply a height-sorted sequence in combination with the FF-rule and the NF-rule respectively.

Level oriented methods are also used in two-stage algorithms for the bin packing problem such as the FFDH * FFD, where the FFDH is first used to solve a strip packing task which serves as basis for the subsequent bin packing process with the FFD. This method is described in greater detail in section 4.3.2.3 where it was used in connection with a genetic algorithm.

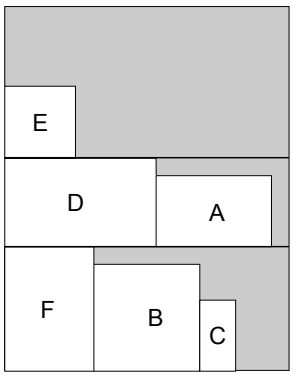

Figure 4.3: FFDH (First Fit Decreasing Height)

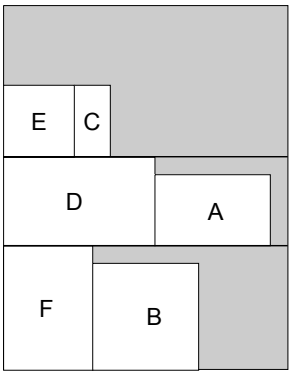

Figure 4.4: NFDH (Next Fit Decreasing Height)

Regarding industrial applications, the concept of bin packing is found in rectangular stock cutting problems, where an order list has to be assigned to a supply of stock sheets. These problems arise frequently in manufacturing processes, which are concerned with the cutting of large and flat rectangular objects as in the glass and wood industry. A number of publications introduced solution approaches for the two variants of the stock packing problem, the trim loss problem (Adamowicz and Albano, 1976b; Albano and Orsini, 1980; Wang, 1983; Rode and Rosenberg, 1987; Riehme et al., 1996) and the assortment problem (Beasley, 1985; Yanasse, 1991). Industrial packing techniques need to accommodate more constraints than the general 2D packing task. Layouts usually have to be guillotineable, which is the case in all examples quoted above, and may contain defective areas, like in the glass industry, which have to remain unpacked (Hahn, 1968).

Pallet loading can be regarded as bin packing, where a number of boxes have to be packed onto a pallet (Bischoff et al., 1995). The classical problem, the so-called manufacture's pallet loading problem only deals with identical boxes (Smith and De Cani, 1980; Dowsland, 1984; Scheithauer and Terno, 1996), for which Dowsland (1987) proposed an exact algorithm.

Some special packing problems like the classical pallet loading problem or moderate guillotineable packing problems were solved with exact algorithms using linear programming techniques (Gilmore and Gromory, 1963, 1965) and tree search algorithms (Hadjiconstantinou and Christofides, 1995; Christofides and Hadjiconstantinou, 1995). Most of the 2D packing tasks, however, were approached by heuristic techniques.

4.2.1.3 3D Strip and Bin Packing Problems

3D packing problems occur less frequently in the literature. Container and pallet loading problems are mainly concerned with the packing of rectangular boxes, which can be either identical (George, 1992) or non-identical (Bischoff and Marriott, 1990). In the general case, certain parallels can be noticed between pallet loading and container loading problems. Once practical constraints are involved such as stability (Carpenter and Dowsland, 1985), pallet and container loading have to be treated as two different

problems. Instead of volume utilisation, the objective may be the maximisation of the value of the boxes, in which case the problem can be formulated as a knapsack problem (Mohanty et al., 1994).

In terms of solution approaches many algorithms transfer the task into a 2D problem applying layer-by-layer strategies. Due to the enormous size of the solutions space, only heuristic approaches are applied to the 3D packing problem in the literature. The methods range from simple heuristics such as the 3D First Fit and the 3D Next Fit (Corcoran and Wainrwright, 1992) to meta-heuristic algorithms, some of which are described in section 4.3.4.

4.2.2 Irregular Packing Problems

Irregular packing tasks occur in the manufacturing processes of the leather, shipbuilding and sheet metal industry dealing with different types of irregular shapes. Whereas in the shipbuilding industry many items have rectangular features, packing problems in the leather industry are concerned with highly irregular shapes. Irregular items and objects increase the complexity of the packing process. Since this additional complexity is highly dependent on the nature of the shapes, irregular packing tasks can be distinguished according to the types of irregular items involved. Whereas most packing techniques use rectangular or polygonal approximations of the items (Adamowicz and Albano, 1976a; Nee et al., 1986), others like commercial nesting packages are capable of true-shape nesting (section 4.5).

Despite the irregular features of the items, the complexity of the packing task is determined by the degree to which the irregularity of the shapes is considered in the basic packing strategy. This refers to the representation of the problem. Irregular problems can be easily transformed into regular ones simplifying the packing process to a great extent. Approximating irregular items by simpler geometric shapes like rectangles reduces the complexity of the packing task. The complexity of the overlap computation for instance strongly depends on the representation method. The shape description is therefore one of the key elements in irregular packing problems.

The techniques used for shape description depend very much on the demands of a given application and its underlying objectives. The question of shape representation has to be seen together with the placement strategy as it influences the type of placement rules to be applied and vice versa. Due to this mutual influence the irregular packing problems reviewed in the following section are categorised by the shape description used rather than the type of the original shape and the main placement strategy. In the majority of problems the items are approximated by rectangles and polygons representing the irregular shape by a sequence of segments. Using a higher number of segments can increase the accuracy of the approximation. Other techniques are grid approximation and quad-tree representation. The major goals of these geometric representations are to reduce the time for overlap and intersection computation and the complexity of the nesting algorithm. With respect to the application in packing, they need to be regarded as a trade-off between accuracy and computational effort.

4.2.2.1 Packing of Rectangular Modules

Many irregular packing problems are transformed into rectangular ones by enclosing arbitrary shaped items by rectangles, because simple, existing algorithms can then be used in the packing process. Some researchers studied the sub-problem of finding the minimum enclosing rectangle. Freeman and Shapira (1975) developed an algorithm that computes the minimal rectangle for an arbitrary closed curve. Martin and Stephenson (1988) proposed a number of algorithms, which pack 2D items such as arbitrary polygons and curved objects into rectangles.

Haims and Freeman (1970) applied the minimum enclosing rectangle concept to a cluster of irregular items. Before the packing stage, up to eight irregular shapes are clustered and circumscribed with the minimum enclosing rectangle. These modules are then optimally packed into the object by means of dynamic programming.

Adamowicz and Albano (1976a) addressed the problem of nesting irregular items into rectangular modules. They introduce a technique that allows the efficient clustering of two polygons by calculating the No Fit Polygon (NFP). The NFP was originally presented by Art (1966) and describes all possible locations that an orbiting polygon (P2) can take with respect to a fixed one (P1) so that the polygons touch but do not overlap (Figure 4.5). This technique can be used to find the minimum enclosing rectangle of two polygons. The authors applied this method to cluster two and more polygons that are then packed on basis of the minimum enclosing rectangle approach. The ideas in that work were later developed further by Albano (1977) in the form of an interactive packing system that allowed the operator to manipulate the proposed layout. The rectangular modules are first grouped then allocated on the rectangular object using dynamic programming.

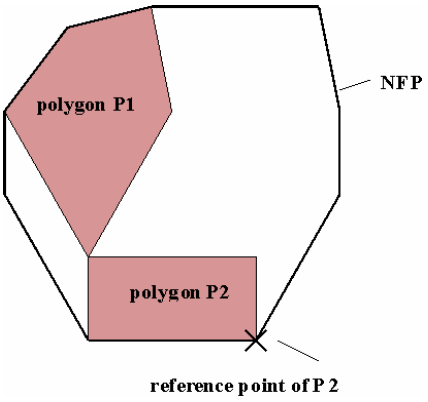

Figure 4.5: No Fit Polygon for two polygons

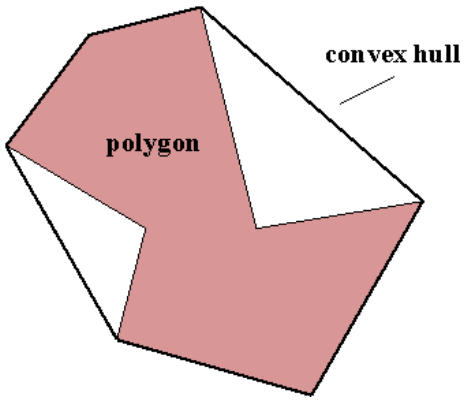

Figure 4.6: Convex hull of a concave polygon

Dagli and Tatoglu (1987) solved an irregular packing problem involving multiple objects. Using a two-stage approach the irregular part of the packing task is simplified to a problem only dealing with single objects. In the first step, the irregular items are enclosed by rectangles and allocated to objects applying mathematical programming techniques. Based on this initial allocation they are then packed by a heuristic procedure on the respective objects. After selecting an item according to a set of priority rules it is moved towards the cluster of previously placed items. By repetitively matching the sides of the two figures and rotating the new item, the algorithm searches for the configuration, which has the smallest enclosing rectangle. Although this procedure may achieve good layouts, it is computationally inefficient for irregular figures with a large number of sides.

The ideas developed by Nee (1984a, 1984b) regarding the pairwise clustering of blanks, were used by Nee et al. (1986) for the nesting of irregular items. Arbitrary items are approximated by polygons and pairwise clustered, if they are repetitive in small batches, before they are circumscribed by the minimum enclosing rectangle. Since pairwise clustered modules may not always be advantageous for the subsequent packing process, the clustering procedure is only carried out if a certain utilisation ratio is achieved. A subsequent rectangular packing procedure places the modules in both orientations onto the object trying all valid Pivot points, which are created by neighbouring boundaries of the already placed rectangles or the object. The non-intersecting configuration, which produces the smallest enclosing rectangle with its neighbour, is finally selected. The list of modules is sorted by decreasing area.

Cheok and Nee (1991) chose a rectangular packing approach for the automatic nesting of ship/ offshore plates, which consists of three steps. The first one is concerned with the shape representation, which approximates the items by a series of line segments ignoring minor features such as drain holes and fillets. In the next step the items are grouped into a number of classes according to their geometric properties as well as their relative sizes and void areas. The authors have developed two separate local optimisation processes. The first one is directed at the clustering of congruent items according to predefined arrangements, since in particular ship/ offshore plates are symmetrical about the centre line and come in pairs. After the clustering process the new figure is enclosed by a rectangle. The second local optimisation process attempts to place smaller items in the void areas of the enclosing rectangle of the larger ones. In the last stage of the packing system, the rectangular modules are allocated onto the object according to a rectangular placement procedure applying various sorting criteria to the sequence of rectangles. The classification of the shapes and the size in this approach helps to select the most appropriate local optimisation process and therefore contributes significantly to the efficiency of the search process.

Since in the shipbuilding industry a high percentage of the items have rectangular features, rectangular packing strategies may be applied in order to reduce the computational effort. Qu and Sanders (1987) use composites of rectangles to describe the irregular items decomposing them into maximal five non-overlapping rectangles. The system orientates each part such that its length is larger than its height and its largest complimentary void area is at the upper right corner. After sorting the list according to the height

of the parts, the items are placed onto the object according to a bottom-left procedure, composing the layout from the lower-left corner to the top-right one. Since the height of the next item cannot exceed the one of its left neighbour, the series of placed items can be enclosed by a stair-shaped line. At the same time the algorithm attempts to fill the blank areas below this line testing these positions first. When no more parts can be fitted within the stairs, the layer determined by the height of the first item is filled, before a new layer is started.

One of the main advantages of the shape representation of this approach is that the complexity of the arbitrary shapes is reduced to rectangles. Since their geometric properties can be stored efficiently, the computation of the layouts is fast. At the same time this description is very flexible hence if necessary a higher accuracy can be achieved by increasing the number of rectangles in the shape approximation. This approach offers many advantages for packing problems that involve a high percentage of rectangular items. For problems that consist mainly of highly irregular shapes the approximation technique and the orthogonal manner in which the layout is built have disadvantages.

4.2.2.2 Packing of Polygons

Some approaches to irregular nesting problems attempt to find efficient clusters of two or more identical or different shapes, which are combined to form larger polygons. These configurations can then be packed onto the object by a placement procedure. With respect to that, Grinde and Cavalier (1995) developed an algorithm that finds the convex enclosure with the smallest area for two concave polygons. The majority of solution approaches to irregular nesting use heuristic techniques which model the problem as a search tree and apply a placement strategy to allocate an item in the partial layout. The tree structure is a graphical representation of the numerous solutions consisting of permutations. The objective is to find the best path from the initial state to the goal state. At each node, a set of heuristic rules is applied to decide which branch to expand next taking into account the layout information. By expanding only the best node at each level the selection process eliminates the majority of permutations. A different field of research in the area of nesting focuses on layout compaction of existing layouts as opposed to layout generation.

In particular in the metal cutting industry, many cutting tasks are concerned with the nesting of congruent items, which are small compared to the object size. This problem is referred to as blank nesting. A possible approach is to find an appropriate description of the item such that it can be placed repeatedly on the infinite object without leaving any gaps or overlaps. This technique is called paving or tiling. Dori and Ben-Bassat (1984) developed a two-stage technique for the efficient nesting of congruent convex figures, which circumscribes an item by a convex polygon with a sufficient number of sides. In the second step, the convex polygon is enclosed by another polygon that is suitable to tile the plane. Koroupi and Loftus (1991) later extended this approach to concave polygons. The objective is to find a paving polygon, which adds as little area as possible to the original shape.

A different approach towards the nesting of congruent figures is taken by Nee (1984a, 1984b). Instead of constructing paver polygons the items are pairwise clustered before they are placed onto the object. A number of configurations are tried rotating the polygons at various angles and placing them side by side. The solution with the highest utilisation ratio is adopted taking into consideration a number of constraints such as bridge width. Various aspects of metal cutting to be considered for the layout generation are highlighted. Prasad and Somasundaram (1991) and Prasad et al. (1995) addressed the blank nesting problem. Their nesting algorithm is based on a sliding technique and finds all feasible arrangements of two items. This technique can also be applied to the nesting of three shapes. After clustering two selected blanks their boundary is traced and then clustered with a third item. The paper by Prasad and Somasundaram (1991) also contains a comprehensive list of practical constraints to be considered in packing systems for the sheet metal industry.

Albano and Sappupo (1980) developed a tree-search approach to address a problem from the textile industry. The items are placed by a leftward placement strategy allocating them to the right of the current profile through the construction of the No Fit Polygon. This single pass algorithm is improved by backtracking when the current solution path produces less desired results than a previous node. Since it is impossible to search the whole solution space, two bounds are used to reduce the increasing number of expanded nodes. The first one describes the quality of the partial layout and measures the true waste in the area left of the profile. For the second one, the potential waste, which will be generated by the remaining items, is estimated using a fixed proportion of their area. The branch that minimises both bounds is chosen next. The search tree is pruned restricting the number of options at each node by a set of rules. Each remaining part generates only one branch at each node. Only a fixed number of branches are explored and backtracking is only allowed to nodes that are significantly further up the tree.

Amaral et al. (1991) developed a sliding algorithm for a problem from the textile industry. This algorithm moves a new item towards the polygonal profile of the partial layout until a collision occurs. Several iterative sliding steps follow depending on the type of collision and the collision distances, until an appropriate position is found. The layout is built up from the left to the right. The intersection tests in this algorithm are based on the concept of D-functions that reduce them to a number of arithmetical operations proportional to the product of the number of sides of the polygons tested (Dowsland et al., 1998). For the selection of the next shape an interactive as well as a heuristic procedure is introduced. In the heuristic technique, the choice of the next shape is related to the analysis of the current profile. An appropriate item is then chosen ensuring that the larger items are placed first in the leftmost placement area. The layout quality could be improved through pairwise clustering of identical shapes.

Unlike the steel industry, which mainly involve rectangular stock sheets, the leather industry deals with objects (hides) and items that are highly irregular and can have defects and various quality levels. The fast solution method developed by Heistermann and Lengauer (1993) approximates the irregular figures by polygons, which are partitioned into various polygonal quality zones. Consequently, each point of an item must be placed on a point in the hide of the same or higher quality. In contrast to placement policies that

fill an object from a certain direction (e.g. leftward-strategy) the authors place a new item along the contour of the unfilled hide. The location for the next placement is called focus and is selected on basis of several parameters such as smoothness of the contour, curvature and distance to defects. The profile of the items is coded by a set of parameters and grouped into several topology classes. A number of items are chosen for the next placement according to their suitability for placement on the current focus. The item that best fits contributing the least to the added waste is finally positioned on the object. The authors claim that their method outperforms other comparable software in this area.

Lamousin et al. (1996) adopted Albano and Sappupo's (1980) heuristic search approach for the nesting of irregular items. In order to deal with the complexity of industrial shapes they applied an efficient representation technique and developed a new placement strategy that allows packing into void areas of larger items. In terms of shape representation the authors used several modification steps in order to obtain a simplified profile of the arbitrary shape. The convex hull serves as first approximation (Figure 4.6). Since this approximation introduces considerable wastage to certain types of shapes (e.g. L-shapes), concave regions of the shape are incorporated in the simplified profile if they are large compared to the area of the part. The placement strategy constructs the No Fit Polygon (NFP) to find a feasible allocation in the partial layout using leftmost, lowest strategy. For the nesting of voids the authors develop a technique called the internal NFP, which determines all feasible locations for a polygon inside void regions.

Oliveira et al. (2000) developed a heuristic search method using the No Fit Polygon to determine a set of the feasible positions in the layout. The best allocation is selected according to three nesting strategies minimising area, length or overlap between the rectangular enclosures of the two items to be nested. In each iteration several items in several orientations are considered for placement. The process of adding a new item to the layout is controlled in two different ways. The pieces are either ordered according to a sorting criterion and tried in all orientations or are tried in all orientations according to the chosen nesting strategy.

Unlike the heuristic search strategies described above, Dowsland et al. (1998) presented a "jostle" algorithm, which was inspired by granular products. When stored in a container any unevenness in the surface can be removed by shaking the container up and down. Simple pass algorithms, which use a set of pre-determined rules to pack a sequence of items towards one end of the stock sheet with a fixed width, often generate layouts with a jagged profile at the open end of the bin. Less area will be required for a layout that has a flatter profile. The jostle method models the shaking process by a series of repetitions of the leftmost placement strategy. After the items have been placed according to a leftmost placement policy they are re-ordered in decreasing order of their rightmost points and packed according to a rightmost placement strategy. In the next iteration the selection is done by the leftmost points of the parts and the placement policy is again the leftmost one. This process is continued for a fixed number of times. The authors used a modified bottom-left placement strategy, which fills holes behind the current profile.

Results showed that the performance of the method depends on the characteristics of the items, but is very effective for a variety of different problems.

Li and Milenkovic's (1995) work concentrates on a different aspect of nesting problems in the cloth industry, i.e. compaction of existing layouts. The efficiency of manually generated layouts can be increased by shortening the length of the layout, removing the overlaps in an overlapping layout and moving the items such, that larger void areas are created. Two types of motion can be distinguished, the compaction which will only lead to feasible (non-overlapping) configurations and separation, which transfers an overlapping configuration into a non-overlapping one. The authors discuss two optimisation models for the compaction and separation tasks.

4.2.2.3 Packing Based on Grid Approximation and Quad-Tree Representation

The common feature of the packing techniques discussed is their approach to the shape representation. Prior to the packing process, the item is approximated by geometric figures like polygons or rectangles. The shape is then described by a set of geometric entities such as vertices and angles. The number of vertices determines the resolution. In contrast to this method are shape representation techniques, which store the digitised image of an item as a set of pixels. Although this type of representation can be computationally expensive, it is advantageous for the description of highly irregular figures. In some cases it can be very difficult to elaborate a suitable set of geometric parameters that sufficiently describe the geometric properties of a shape for further processing. It allows the description of the geometric features of a shape with less loss of information as when the enclosing rectangle or the convex hull is applied to concave polygons. The resolution of the approximation is determined by the grid size.

Batchelor (1991) described a technique, which digitises the object in order to determine a suitable position for the next item. The arbitrary items are circumscribed with their minimum area bounding rectangle prior to packing. During the packing process the object is scanned until a position is found which is sufficiently large to place the rectangle without causing overlap. Due to the scanning procedure this method is able to pack smaller shapes between larger ones. This technique achieves denser patterns than the conventional rectangular placement strategy, which positions the rectangles rather than the original shapes.

Whelan and Batchelor (1991, 1992) and Whelan (1996) extended the ideas developed in Batchelor's (1991) approach to solve a nesting problem from the leather industry. The digitising technique is used to describe highly irregular objects as well as items. Since the search for a feasible location based on the enclosing rectangle contains some limitations (Batchelor, 1991), the authors have developed a packing method which makes use of computer vision techniques namely mathematical morphology. The morphological transformations allow to determine the feasible placement region and to describe the new partial layout again as a digitised image. The placement policy works on a sorted list of rectangles. Since this technique allows items to interlock it achieves higher packing densities than the one developed earlier

(Batchlor, 1991). Prior to placement, a heuristic component is implemented that determines the orientation and the order in which the items are positioned. The heuristic algorithm distinguishes between polygon and "blob" shapes. The ordering and orientation rules for the blob shapes mainly concern the circularity and size of concavities, whereas the polygons are ordered according to their size and are aligned prior to positioning such that their main direction corresponds with the main direction of the unpacked region of the object.

Han et al. (1997) developed a special part decomposition procedure to reduce the overlap computation time at the nesting stage. After calculating the minimum enclosing rectangle of the item, the scrap areas outside as well as holes inside the polygon are encoded using a quad-tree technique. The nesting procedure consists of two main heuristic algorithms using a SOAL (self-organisation assisted layout) to generate an initial layout (section 4.4.3) and simulated annealing for compaction (section 4.4.1).

4.3 Application of Genetic Algorithms to Packing Problems

A considerable amount of research has been carried out to develop algorithms for the solution of packing problems. Deterministic methods such as linear programming techniques have been developed to obtain exact solutions. Since exact algorithms only work efficiently up to a certain degree of complexity heuristic techniques are applied to more complex combinatorial problems. Apart from a vast number of problem-specific heuristics, that were proposed for certain packing problems, there are general heuristic methods like genetic algorithms, simulated annealing and tabu search. These algorithms are suitable for packing tasks as well as other combinatorial problems.

Although genetic algorithms were developed in the early 70s, it was not until the mid 80s that they were applied to packing problems. The first researcher who implemented genetic algorithms in this domain was Smith (1985) applying them to a 2D rectangular packing problem. At the same time Davis (1985) summarised the techniques for the application of genetic algorithms to epistatic domains 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.

Complex epistatic 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 of 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. In the following review, the solution approaches using genetic algorithms are distinguished according to the classification described in the next section.

4.3.1 Classification of Packing Problems Approached with Genetic Algorithms

To date, a variety of packing problems have been approached by genetic algorithms. Most of the problems encountered in the literature are 2D strip packing tasks dealing with regular, mainly rectangular shapes. The complexity and the implementation of a genetic algorithm depend on the type of problem. In order to compare the quality of the solution approaches suggested in the literature the following review distinguishes between several problem types.

Since the geometric properties influence the complexity of the problem and the size of the search space, the various packing tasks are distinguished according to their geometric features. The spatial dimensions of the problem are also very important criteria. Both characteristics have been used to classify the solution approaches in the literature (Figure 4.7). The problems are grouped into regular or irregular packing problems. Whereas regular figures such as rectangles and circles can be determined by a few parameters (Dyckhoff, 1990), the term irregular applies to convex and concave polygons as well as highly irregular, arbitrary shapes.

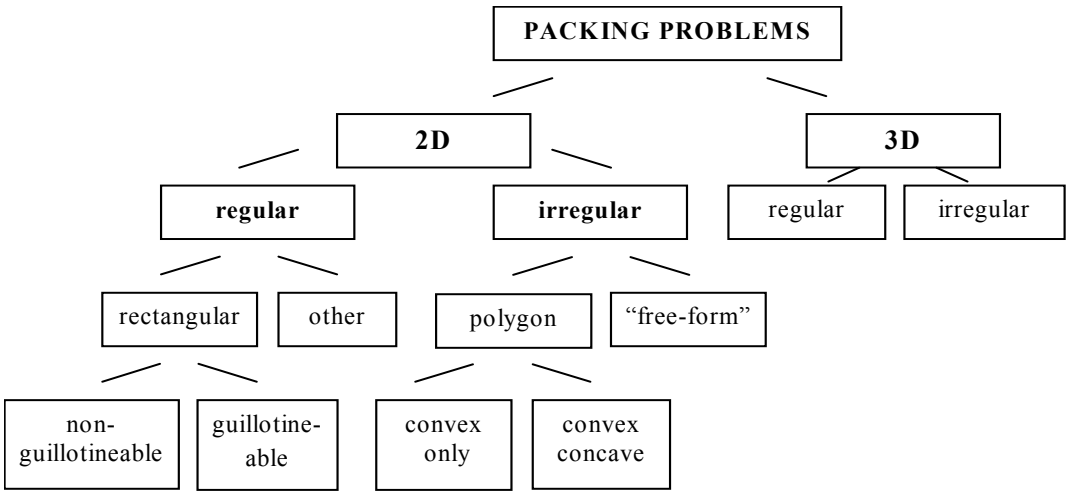

Figure 4.7 Classification of packing problems approached with genetic algorithms

The representation of the problem is the key to the efficient application of genetic algorithms to packing problems. The following review therefore puts particular emphasis on the description of the encoding technique and the genetic operators and summarises most of the problem-specific design variables in form of tables. Where available, the performance of the genetic algorithms in comparison to other solution approaches is briefly stated.

4.3.2 2D Regular Strip Packing Problems

The vast majority of the literature that concentrates on regular packing problems is concerned with the rectangular strip packing problem where a set of rectangles has to be allocated onto a rectangular object of unlimited height. In general, the packing of rectangular items distinguishes between guillotineable and non-guillotineable patterns. A modification of this problem is 2D bin packing that consists of multiple objects of fixed size. To date, only one approach has been described in the literature that uses other regular items. In all cases the aim is to find the arrangement of items producing the least waste either minimising the height of the object or reducing the number of bins.

4.3.2.1 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 4.2 to Table 4.5). A second group of genetic methods incorporates more layout information into chromosomes using a tree structure (Table 4.6). Some research concentrated on an entirely different genetic approach, which works without encoding, but manipulates the figures in the 2D layout directly.

Hybrid Approaches:

One of the first researchers who implemented genetic algorithms in the domain of packing is Smith (1985). He experimented with two heuristic decoding routines, one of which implements backtracking. The first one (Slide algorithm) 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.

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 4.3.3.1).

Table 4.2: 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[4]	improved bottom-left algorithm[5]

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 demonstrate the better performance of the new bottom-left placement routine using the two packing problems of Jakobs' work. The BL-algorithm as well as the improved version have been used for comparison purposes in this investigation and are described in detail in section 5.5.4.1 and 5.5.4.2.

The order-based approach using a bottom-left packing routine has attracted particular attention over the recent years. Leung et al. (1999) developed a placement routine, which preserves the bottom-left stability in the layout (Table 4.3). The improved BL-algorithm 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. This packing routine was combined with genetic algorithms and simulated annealing. In comparison to the sliding algorithms the Difference Process Algorithm generated 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 4.3). 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

[4] BL = Bottom Left heuristic; based on sliding principle (section **Error! Reference source not found.**)

[5] based on sliding principle; referred to as BLLT-routine throughout this work (section **Error! Reference source not found.**)

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

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

Hybrid Algorithms Using Additional Layout Information

The data structures of the hybrid algorithms summarised in Table 4.3 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 4.4). 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. The evaluation against the BL-heuristic allows a relative comparison with the hybrid algorithms developed in this work (section 6.6).

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, whereby 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 4.4: 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
representation	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 + bottom-left condition	none	none

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 of from an initial solution, the layout is manipulated by three groups of operators including hill-climbing, mutation and recombination. The evolutionary algorithm is summarised in Table 4.5.

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 at least remains the same. These operations include translation, rotation and relocation of an item. An additional operator rotates an item around the touch point with another item. Further to that, 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. One of its major drawbacks 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.

Table 4.5: Evolutionary algorithm for non-guillotineable 2D packing problems - solution in 2D space

	Ratanapan and Dagli (1997a,b; 1998)
problem	strip packing; 90° rotation
objective	minimise height of bin
representation	2D geometric objects
fitness	packing density
cross-over	none
mutation	series of hill-climbing, mutation and recombination operations
decoder	none

4.3.2.2 Guillotineable Packing Problems

Guillotineable packing problems have been approached with genetic algorithms by four researchers. Most algorithms are based on tree representations applying various genetic operators (Table 4.6). Two methods take a different approach and use a permutation and a heuristic decoder to generate guillotineable layouts (Table 4.7).

Hwang et al. (1994) tackled three rectangular packing problems involving strip packing, packing under minimisation of the area and bin packing (section 4.3.2.3). The strip packing problem is approached with two different encodings. The first one uses 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 4.6).

The second representation that has been implemented by the authors is order-based using a level-oriented packing procedure (Table 4.7). The packing is constructed as a sequence of levels, each rectangle being placed left justified so that its bottom rests on one of these levels. Each level is defined by 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).

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

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 4.6). 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. Genetic algorithms and simulated annealing achieve significantly better results than the primitive heuristics with the genetic algorithm performing best.

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 4.6: 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 90° rotation	strip packing; 90° rotation	packing of a single closed bin; no rotation	packing of a single closed bin; 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

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 previous one. The mutation operators work on these genes, changing the orientation and the placement flag. The layout constraints are entirely 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 4.7: 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 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[6] and BF[7]	heuristic algorithm that considers all technological constraints

4.3.2.3 Bin Packing

The genetic algorithm approaches to bin packing concentrate mainly on the 1D version. The objective of bin packing is to allocate a set of items into a minimum number of bins. Consequently the fitness function has to take into account the utilisation of the bins. Two of the following algorithms are concerned with bin packing in the classical sense and one method is aimed at dynamic bin packing (Table 4.8). Less work has been done on 2D bin packing.

The hybrid algorithm by Hussain and Sastry (1997) is order-based and uses the Next Fit algorithm (NF) to generate the bin layout. Comparisons with the First Fit Decreasing (FFD) and the Best Fit Decreasing (BFD) algorithm which sort the items according to decreasing height show that the genetic approach outperformed the simple heuristics for many test cases. The heuristic routines are described in section 4.2.1.2.

The representation scheme applied by Falkenauer and Delchambre (1992) and Falkenauer (1996, 1998) in their so-called grouping genetic algorithm focuses on the bins rather than the items. According to the authors an encoding technique that uses one gene per item to represent the bin, into which it is packed, does not perform well in combination with the classic crossover and mutation operators. Therefore an alternative data structure was proposed using a two-part chromosome, which consists of an item part describing the permutation of items and a group part. The group part contains the bins used in the solution. The order of the bins in the chromosome is irrelevant to the genetic algorithm. The idea behind this encoding scheme is that groups, which are the meaningful building blocks in grouping problems, are represented by the genes of the chromosome.

[6] FF = First Fit heuristic (section 4.2.1.2)

[7] BF = Best Fit heuristic (section 4.2.1.2)

The genetic operators only work on the group part and have to handle strings of variable length. They make use of two heuristic procedures, the First Fit (FF) and the First Fit Descending heuristic (FFD). The FF places an item into the first bin with sufficient space and starts a new bin in case there is not enough space. The FFD first sorts the items according to decreasing height before applying the FF-rule (section 4.2.1.2).

The cross-over procedure ensures that important information concerning the items stored in certain bins is transmitted. After random selection of two cross-over points in the group part, the bins between the cross-over section are inserted into the first offspring. In order to avoid the multiple appearance of items in the solution, the relevant bins originating from the first parent are deleted. Since the deleted bins may contain items that are not present in the bins coming from the second parent, these missing items are reinserted with the FFD-procedure creating additional bins if necessary. For the mutation process a few bins are selected randomly reinserting their items in random order with the FF-algorithm. An inversion operator is implemented changing the order of the bins in the group part of the chromosome. This can improve the transmission of certain genes to the offspring. Comparison with the FFD-algorithm showed that the performance of the genetic algorithms is superior.

The genetic grouping algorithm was extended implementing the so-called dominance criterion (Falkenauer, 1996) which was used earlier in an approximation algorithm for the bin packing problem. This criterion is based on the observation that under a certain geometric condition the arrangement of the items in a bin is better than another one. This is exploited in the form of local optimisation in the genetic algorithm. Experiments showed that this technique works better than the original grouping algorithm.

Table 4.8: Comparison of the genetic algorithms for 1D bin packing problems

	Falkenauer et al. (1992) Falkenauer (1996, 1998)	**Hussain and Stastry (1997)**	**Runarsson et al. (1996)**
problem	bin packing	bin packing	dynamic bin packing
objective	minimise number of bins	minimise number of bins	minimum weight for each bin, maximise utilisation
representation	two-part chromosome	permutation	binary strings representing bin allocation
fitness	bin capacity	unfilled proportion of the bin	set of fuzzy objective functions
cross-over	problem specific operator	special cross-over operator from TSP	binary crossover (2 point)
mutation	deletion of bins, reinsertion of objects with FFD-rule	swap of 2 elements	binary
decoding	First Fit-algorithm	Next Fit-algorithm	

Runarsson et al. (1996) applied genetic algorithms to a 1D dynamic bin packing problem. A continuous supply of items has to be packed into a limited number of bins assuring minimum weight of each bin. Once a bin is filled, an empty one replaces it. Similar to on-line algorithms, the items must be packed sequentially, though the algorithm sees a number of items simultaneously. The algorithm uses the binary

coding technique for the chromosomes, which assign the series of 'visible' items to a limited number of bins. The authors developed a set of fuzzy objective functions to guide the search.

The only work, which has so far approached the 2D bin packing problem with genetic algorithms, was conducted by Hwang et al. (1994). The items are represented by a permutation and packed into the bins by a two-stage heuristic. In the first stage the level-oriented First Fit heuristic procedure as described to the 2D strip packing approach in section 4.3.2.2 places them onto a strip of unlimited height constructing the layout as a sequence of levels. Each level forms a rectangular block containing one or more items. In the next step the packed strip is decomposed at each level. Each block can be viewed as a rectangle of fixed width and the height of the level. The blocks are then packed with the FFD or BFD-rule into fixed size bins reducing the problem to a 1D bin packing task. The authors implemented two hybrid genetic algorithms using the FFD and the BFD-routines in the heuristic decoding algorithm. Comparisons are made to the Hybrid First Fit-algorithm (HFF), which is a combination between the FFDH and FFD-routines (Coffman et al., 1984). The genetic algorithms consistently outperformed the heuristic one (HFF), whereby the one using the BFD-heuristic performed best (section 4.2.1.2).

4.3.2.4 Packing of Regular Shapes other than Rectangles:

The only genetic algorithm designed for packing 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 cross-over 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.

4.3.3 2D Irregular Strip Packing Problems

This category of irregular problems includes the packing of polygons and arbitrary shapes. 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 4.9 to Table 4.12.

4.3.3.1 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 4.3.2.1) 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. To improve the shrinking-step, reflections of the original polygons are also tested. The shrinking-step reduces the height of the layout, since it allows utilisation of the space "wasted" by the embedding process. Applying the shrinking-routine to the final layout has a major drawback. Since the polygons are repositioned sequentially, empty areas in the layout may not always be reached, since items, which are not yet iterated, can block the sliding motion of the current one.

Hybrid algorithms which operate on a sliding principle and pack the polygon rather than the rectangle in the partial layout have been found to achieve better packing densities (section 7.5), although they are

computationally more expensive. 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 4.9: 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

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. An additional (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 in which way 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 4.10.

In an earlier approach, Bounsaythip et al. (1995) used a different genetic algorithm to address the textile nesting 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 4.10: 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

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 from of two binary strings. Due to the simplicity of the encoding technique, the binary cross-over and mutation operators can 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. Since the position of the items is determined by the encoding, overlapping configurations can occur and are penalised in the fitness function. The fitness function is dynamic, increasing the penalty term gradually in order to drive the population away from invalid solutions towards the end of the search. Similar to some simulated

annealing approaches in the literature (section 4.4.1), 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. In addition to that, 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.

4.3.3.2 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 4.11 and Table 4.12.

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 into 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. Additional to the two parameters, which describe the relative position of a shape to the others, 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 4.3.2.1). 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 4.11: 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

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 4.11). 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 4.12. 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 4.3.2.1.

Table 4.12: 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

4.3.4 Overview of 3D Packing Problems

Unlike 2D packing problems, the 3D versions deal mainly with regular and in most cases cuboid objects, which have to be loaded onto pallets or into containers. A number of problems were approached with genetic algorithms in the literature. An overview of the algorithms for regular 3D problems is given in Table 4.13. The complexity increases when irregular objects are to be placed. So far, only one group of researchers have applied genetic algorithms to the packing of arbitrary 3D items.

4.3.4.1 Regular Packing Problems

Prosser (1988) developed two genetic algorithms for a highly constrained pallet loading problem. The problem consists of loading stacks of plates onto a minimum number of pallets without violating geometrical and weight constraints. The main drawback of the first genetic algorithm is the evaluation function calculating the number of pallets used. Since it only produces a few distinct values for densely packed pallets, there is a lack of guidance for the search process.

To overcome these disadvantages, a hybrid genetic algorithm was developed in order to find one maximally loaded pallet. Loading one pallet after the other up to the weight limit reduces the number of possible loading patterns and results in a faster convergence. Apart from the weight, the number of items on one pallet is also limited. The pallet is loaded up to the first violation of the constraints. The genetic algorithm is then applied iteratively to the reduced problem until the allocation of the stacks is concluded.

For example if a maximum of six elements of a string can be loaded onto a pallet the inversion operator is modified such that elements are randomly swapped between the first six positions of a string and beyond. The computational effort of the evaluation is lower than in the first genetic algorithm. The comparison with a branch-and-bound method showed that the second genetic algorithm produces better solutions in less time.

Corcoran and Wainwright (1992) proposed a hybrid genetic algorithm for a 3D loading problem using a single, open bin. The algorithm is based on the so-called level technique, which places items level by level. In extending this strategy to three dimensions the bin is divided into "slices" along the height and "levels" along the length. In order to evaluate the fitness of the packing pattern two heuristic algorithms are used. The Next Fit algorithm (NF) places an item into the next available position on the level without exceeding the width of the bin. If that fails a new level is started. The First Fit procedure (FF) searches from the beginning until a suitable position is found. The performance comparison for various sets of boxes shows that the genetic algorithms produce better packing patterns than the two heuristics, which have equal packing efficiencies.

House and Dagli (1993) approach the container loading problem with a permutation and use a heuristic decoder to pack the boxes. Lists keep track of the empty spaces that are generated after allocating a box.

A different approach to a container loading problem was taken by Lin et al. (1993). Their algorithm takes into account additional constraints such as the weight of the boxes. A heuristic method enhanced by a simulated annealing algorithm is combined with a genetic algorithm. In order to consider the weight constraint, i.e. placement of heavier boxes below lighter ones, the loading layout of the container is encoded in the form of a multi-chromosome string with each partial chromosome representing the loading pattern of one layer. Cross-over and mutation only operate on the corresponding relative chromosomes without inter-chromosome exchanges. A set of heuristic rules is used in the decoding stage. In terms of solution quality the genetic algorithm outperforms the heuristic method enhanced with the simulated annealing algorithm.

Bortfeldt (1994) developed a similar method to the level technique by Corcoran and Wainwright (1992) for a container loading problem, where a set of different boxes is packed maximising space utilisation. The genetic algorithm does not require a decoding stage since all information concerning the phenotype is stored in the encoding. Instead a heuristic algorithm is used during the cross-over and mutation operation based on a layer technique. The depth of a layer is determined by the first box (layer defining box = LDB) in the layer, width and height are the same as the container dimension. A further procedure is implemented to fill the layer, where each allocated box generates three spare spaces inside the layer: beside, in front and above. The fill algorithm searches for the best fitting pair of boxes and places them into the layer.

In order to keep the computational effort low, a problem-specific data structure is applied. The chromosome includes the number of layers and for every layer the LDB, its rotation and the set of boxes

allocated in that layer. The main advantage of this encoding technique is that the genotype of the individuals is sufficient for evaluation of the fitness. It guarantees the feasibility of the individuals generated by the genetic operators without the need for decoding. The rationale behind the cross-over technique is the transfer of the best layers of both parents. When no more layers can be taken and the offspring is still incomplete, new layers are generated by the heuristic algorithms described above. During the mutation stage the boxes with the lowest frequency of appearance in the new population are determined. New packing plans are then generated by the heuristic algorithm, which uses these boxes as the LDB of the first layer. The individuals generated in this way are exchanged for the ones with the lowest fitness in the population. A performance comparison shows that the genetic algorithm produces packing patterns with a high space utilisation.

Table 4.13: Comparison of the genetic algorithms for 3D regular packing problems

	Prosser (1988)	**Corcoran and Wainwright (1992)**	**House and Dagli (1993)**	**Lin et al. (1993)**	**Bortfeldt (1994)**
problem	pallet loading with weight constraints	loading of single open container	loading of single container with weight	loading of single container with weight constraint	loading of single container
objective	minimise number of pallets	minimise container size	maximise number of items	maximise number of items	maximise number of items
representation	permutation	permutation	permutation	multiple-chromosome	problem-specific
fitness	weight of pallets	height	container utilisation		utilisation and deviation of utilisation of single layers
cross-over	OBX, adapted to loading constraint	PMX, Cycle, Order2; Rand1	not stated	PMX	problem-specific with FF rule
mutation	exchange of elements from first 6 positions with rest	elements swapped and rotated	not stated	inversion	problem-specific
decoder	set of constraints	First Fit rule Next Fit rule	heuristic that caters for constraints	set of heuristic rules	none

4.3.4.2 Irregular Packing Problems

Ikonen et al. (1996) and Ikonen and Kumar (1997) proposed a framework of genetic algorithms for packing irregular items into a cylindrical container. The items need to be packed utilising cavities of larger ones with no orientation restrictions. In order to represent the order and the orientation, in which

the parts are packed a multi-chromosome representation is used. The first chromosome consists of a permutation, which encodes the packing order. The second one contains the orientation of each element. A swapping operator in the mutation stage changes the position of two parts in the chromosome and a mutation operator is applied to the second chromosome changing orientation values randomly. The geometry of the parts is included in the decoding procedure that translates the strings into their packing layout and evaluates their fitness. The fitness of an individual is related to the distance of each part from the origin. In order to reduce the probability of infeasible solutions an adaptive penalty function is incorporated in the fitness function. This penalty function is especially advantageous when it is not known in advance how difficult it is to find feasible solutions and how effectively the objective function differentiates between solutions.

4.4 Application of Meta-heuristic Methods to 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. Since simulated annealing is one of the meta-heuristic methods, which are used to compare the performance of the hybrid genetic algorithms developed in this work (chapters 6 to 8), the major solution approaches in this area will be briefly described.

4.4.1 Simulated Annealing

Simulated annealing is a meta-heuristic search method whose design was inspired by the metallurgical process of annealing (section 3.3.3). Simulated annealing was applied to rectangular and irregular packing tasks, a selection of which is described below.

4.4.1.1 Regular Packing Problems:

A number of approaches in the literature have applied simulated annealing to 2D rectangular packing problems (Table 4.14). One of the first works on simulated annealing and packing problems was carried out by 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 non-guillotineable one 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 4.3.2.1). Their results indicate that genetic algorithms outperform simulated annealing.

Table 4.14: 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' (section 4.3.2.1)

4.4.1.2 Irregular Packing Problems

Most simulated annealing approaches for irregular packing tasks do not make use of any encoding technique like genetic algorithms. 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. With one exception (Ratanapan and Dagli, 1997b, 1998) the genetic methods from

the literature operate on an 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 4.15), whereas less work has been done in the area of 3D irregular problems (Szykman and Cagan, 1995).

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 accepting only 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 the 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 (1999b) 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 4.15. 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, Figure 4.5) 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 4.15: 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 (1999b)**
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

4.4.2 Tabu Search

Tabu search is a search technique that is guided by the use of adaptive or flexible memory structures. In contrast to heuristic search methods such as genetic algorithms and simulated annealing 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 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, 1996; Glover and Laguna, 1993, 1997; Pirlot, 1996). 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 conducted by Blazewicz and his co-researchers, which dates back to the early 90's.

The only investigation into the application of tabu search to rectangular problems was presented by Lodi et al. (1999a, b) 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 (section 4.2.1.2). 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 ones 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.

4.4.3 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 4.3.2.1). 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, 1997) 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 4.4.1). 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 only indicate the direction for the motion of the items, are then 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.

4.4.4 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. The stochastic optimisation method proposed by Pargas and Jain (1993) has been used during this investigation to compare the performance of the hybrid genetic algorithms with other meta-heuristic methods (section 6.4.2 and 8.4.2).

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.

4.5 Commercial Packing Software

Cost reduction via efficient production is one of the major issues in the manufacturing industries where products are mass-produced. In particular, the efficient use of raw material can result in significant savings. Therefore the industry started some time ago to automate packing and cutting processes. With the increased use of computers in the manufacturing environment, fully automatic and interactive nesting systems have become available offering a large variety of tools for nesting.

Most of these products are not only directed to improve material utilisation, but the manufacturing process as a whole containing modules on scheduling, cost estimation and inventory. Some of them have been developed for special industries, e.g. the leather industry. In general, they are multi-purpose nesting systems for a wide range of industrial applications. Table 4.16 and Table 4.17 give an overview of commercial packing software for 2D rectangular and irregular packing tasks (see Appendix C for Internet addresses). Most irregular nesting systems include or provide optional modules with algorithms for rectangle packing.

Table 4.16: Commercial packages for rectangular packing problems

product	**company**	**description**
Cut Planner	Remarkable Software Ltd.	cutting optimisation for rectangular parts for multiple sheets
DEC++	Advanced Technology Institute	packing software for rectangular parts for multiple sheets
Itemizer	MID-OHIO SYSTEMS	PC-based packing system for rectangular items
Ritmo4	ALMA Manufacturing Software	rectangular and linear cutting optimisation software package for woodwork, mirror manufacturing, sheet metal and cutting plastics or composites
Shear/Miser	Fab/Trol Systems, Inc.	packing system for rectangular parts for multiple sheets with remnant tracking

Table 4.17: Commercial packages for rectangular and irregular packing problems

product	company	description
UG/Sheet Metal Nesting	Unigraphics Solutions Inc.	fully automatic and interactive nesting for the sheet metal industry
PowerNest	ALMA Manufacturing Software	fully automatic nesting of system mainly for the mechanical industry with; includes a specific function for the nesting of rectangular parts; available as library containing the nesting algorithms
Nestix2	Nestix Oy	automatic nesting module part of 3D CAD package for the shipbuilding industry
HUMANCAD Optitex	Humantec Industriesysteme	template construction and nesting for the apparel, the upholstery and the automotive industry; interactive and automatic nesting is supported
Materials Manager 2.7	MID-OHIO SYSTEMS	general purpose true-shape nesting and optimisation system; runs inside of AutoCAD; automatic and interactive nesting
OptiNest	Remarkable Software Ltd.	general purpose true-shape nesting program; automatic and interactive mode
SigmaNEST	SigmaTEK Corporation	true-shape automatic nesting system; automatic and interactive mode; for the metal and wood cutting industry
SS-NEST	Striker Systems	automatic nesting software for sheet metal
SS-STRIP	Striker Systems	automatic strip layout; also interactive
AccuFABTM	DynaSys, Inc.	multi-purpose fully automatic nesting system; also interactive
Nestlib	Geometric Software Solutions Co. Ltd.	fully automatic true-shape automatic nesting; also available as DLL containing the nesting algorithms
SMP/IS	Merry Mechanization, Inc.	fully automatic and interactive true-shape nesting system sheet metal fabrication; with modules for just-in-time requirements, special cutting processes etc.
PINS XL 2000	Generative N/C Technology	automatic nesting system
Helix Manufacturing	MICROCADAM, Inc.	automatic true-shape nesting system for the sheet metal industry
AutoNest	Virtek	fully automatic dynamic nesting software specifically designed to optimise yields on leather hides
Friendly 2D	DNT	automatic nesting software for the leather cutting industry; fully automatic and interactive mode
Lasernest	Humantec Industriesysteme	fully automatic nesting software for the leather industry

Since material utilisation is one of the major industrial issues, most nesting algorithms work with true-shape representation rather than approximations. Other desired features of nesting packages are a high speed of operation and the capability to perform fully automatic nesting. Most of the commercial products on the market at present offer the following major features with respect to the packing process:

- flexibility aimed at industrial problems from a wide area of applications
- support for a range of file formats, e.g. DXF
- fully automatic nesting
- possibility for interactive modifications

- true-shape nesting
- suitability for regular and irregular problems involving rectangular or non-rectangular sheets
- facility to nest multiple sheets of different sizes in a single run
- in-hole-nesting, i.e. facility to optionally nest parts within the holes of larger parts
- support for incremental nesting, i.e. nesting on a pre-nested sheet(s)
- part rotation (user defined step angle for each part)
- suitability for several cutting techniques
- cutting sequence generation

The features listed above are not a full description of the commercial packages but focus mainly on the packing aspect. Commercial nesting software typically offers more features especially with respect to the cutting process, providing support for many cutting techniques such as laser and plasma cutting and punching. Additional modules are available for special industrial needs such as part scheduling, cost estimation and inventory control. The price for the commercial products starts at $400 for rectangular packing systems and can be as much as $20,000 for irregular nesting products, depending on the complexity of the software and the modules acquired. Two of the nesting packages listed in Table 4.17 have been used for the evaluation of the rectangular and irregular packing algorithms developed in this work (section 5.5 to 5.7).

A number of products are available for 3D packing tasks specialised on container and pallet loading (Table 4.18). Although a first step has been made towards packing of irregular items and spaces in two of the packages in Table 4.18, there is still no commercial product available, that supports complex 3D shapes as they occur in the field of rapid prototyping. Research activities in this area have already started (Ikonen et al., 1996). With the availability of computational power continuously widening, it should not be long until commercial products for this task will be launched.

Table 4.18: Commercial packages for 3D container/ vehicle and pallet loading problems

product	**company**	**description**
PowerPack	Remarkable Software Ltd.	flexible 3D packing tool for packing boxes on containers, pallets etc.
CubeIQ	Remarkable Software Ltd.	loading optimiser for rectangular and irregular shaped containers, i.e. air craft
CARGOMANAGER	Gower Optimal Algorithms Ltd	3D system for packing boxes into containers, pallets etc.
PALLETMANAGER	Gower Optimal Algorithms Ltd	powerful package to pack pallets with rectangular boxes and cylindrical boxes

4.6 Conclusions

This review of solution approaches to 2D and 3D packing problems reveals that heuristic and meta-heuristic search methods have been implemented for the solution of a large variety of 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 in polynomial time. 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.

4.6.1 Meta-heuristics

Genetic algorithms and to a smaller extent simulated annealing have been applied to 2D packing tasks. 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 (1999a) 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, whereas genetic algorithms were better in the second investigation.

In addition to that, most researchers in the area of genetic algorithms seem to take a successful search process of their particular implementation for granted. Furthermore, 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 as a consequence 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 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 can be used to assess the quality of the search space operators. 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.

4.6.2 Benchmarks

The discussion of performance comparison leads on to another topic which has not entirely been neglected, but certainly has not been resolved to satisfy the needs of a coherent research community. A key weakness behind much of the work is the lack of comparisons with known benchmarks in the widest sense by pointing at standard test methods as well as standard test problems. Despite some effort by two on-line libraries (section 5.3), there is no test suite available which could enable at present 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.

Apart from the range of benchmark problems 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 (section 4.2.1). As meta-heuristics are expected to perform comparatively better in terms of solution quality this may seem to be a waste of time. However, even a relative comparison to a standard method is a useful and a 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. They also were already used by a few other researchers. 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. 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.

4.6.3 Solution Approaches

The literature review shows that a large variety of genetic algorithms and simulated annealing approaches were developed for strip and bin packing problems. 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.

4.6.3.1 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.

4.6.3.2 Type of Approach

With respect to the packing problems described in the review, three types of solution approaches involving genetic algorithms can be distinguished. The common feature of most genetic algorithms developed for packing problems is their two-stage approach. The concept of the genetic algorithm is used to explore and manipulate the solution space, whereas a second procedure is needed to evaluate the solutions generated. Therefore the phenotype needs to be constructed in order to check quality and feasibility of packing scheme. As the genetic algorithm is used to determine the sequence of packing, 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; Marques et al., 1998) 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, 1999b).

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 to be considered, 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 additionally to the computation time, especially, if the re-positioning task turns out to be more complex involving several steps.

Accepting an invalid configuration temporarily offers a balance between these two extreme measures, since often a series of moves automatically 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 less efficient to guide the search than using a decoding algorithm that avoids producing constrained results.

The opposite approach is taken by hybrid algorithms, where 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 4.6.2).

4.6.3.3 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.

4.6.3.4 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, 1999b). 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).

4.6.4 New Solution Approach to Rectangular and Irregular Packing Problems

The major objective of this investigation is to develop a general meta-heuristic solution approach to 2D rectangular and irregular packing problems involving strip as well as bin packing. As the review of the literature shows genetic algorithms are most favoured for this task and obtained promising results in a large range of applications. They are therefore primarily considered in this work. At the same time a comparison with other meta-heuristic methods is regarded as highly important.

4.6.4.1 Problem Representation and Outline of the Algorithm

The issue of representation, which refers to a description either in the 2D domain or in form of encoding technique, has not been resolved satisfactorily in the literature considering all its advantages and disadvantages. Therefore it has been decided to use an encoding technique. After all the natural example genetic algorithms are derived from is based on this principle. Before any assumptions concerning the quality of a certain encoding can be made, it is important to first investigate the traditional, though very basic, order-based technique, which by now can be seen as the standard encoding technique for combinatorial problems. After comparison with more sophisticated principles it may eventually be developed further.

Rectangular Strip Packing:

The chosen approach is a two-stage order-based technique, where the meta-heuristic method acts as tuner for a packing routine. Approaches which used this principle (Kröger et al., 1991a, b and 1993; Hwang 1994; Jakobs, 1996; Liu and Teng 1999) outperformed heuristic solutions with respect to rectangular strip and bin packing problems. The heuristic decoders used by Jakobs, Liu and Teng as well as Hwang et al. show a certain weakness since they do not necessarily place the rectangle into the best position from the point of view of a human operator. One of the design tasks of the hybrid algorithm is to improve the packing routine with respect to the packing density allowing access to enclosed areas in the partial layout.

At the time this work was started the publication by Leung et al. (1999), which presents a similar idea, did not exist, but was conducted independently. In order to establish the performance of the improved heuristic decoder comparisons are made to other hybrid techniques in this area as well as the heuristic algorithms themselves. The latter one will help to identify the advantages of the hybridisation concept for the layout generation.

In packing problems, the simulated annealing approaches have always operated directly on the 2D layout. The order-based encoding technique had not been incorporated into this search method at the time this investigation was outlined and seemed worth exploring. In the meantime, a number of simulated annealing approaches were based on this principle (Faina, 1999; Burke and Kendall, 1999b; Leung et al., 1999). However, the rectangular ones do not consider the orientation of the items as a neighbourhood move.

Rectangular Bin Packing:

In addition to strip packing, 2D rectangular bin packing is considered as a second application of the hybrid meta-heuristic algorithms. Apart from one investigation by Hwang et al. (1994), most genetic algorithm approaches focus on 1D bin packing. Although Hwang's method uses a heuristic decoder, it does not necessarily allocate an item in the best way to generate a dense layout. The improved decoder used in the strip packing case should also achieve better bin layouts. Unlike in the literature (Hwang, 1994; Faina, 1999), different-sized bins are used incorporating the sequence of the objects into the encoding such that it can be manipulated by the search process.

Irregular Strip Packing:

The concept of the hybrid algorithms developed for rectangular packing tasks is also applied to irregular packing extending the bottom-left placement routines to polygons. Although hybrid principles have been applied in the literature in this area they have some drawbacks. The irregular placement algorithm used by Jakobs (1996) considers the irregularity of the packing task only after the search process. This can be improved by treating the shapes as polygons during the decoding stage rather than as rectangles. Many redundant calculations in Dighe and Jakiela's (1996) method, that applies a low-level search process in connection with a sliding routine to position an item, can be avoided using a simple decoder. In this work

a number of decoding methods are investigated which are based on a sliding principle, placing the polygons in a bottom-left manner similar to the rectangular placement routines.

4.6.4.2 Performance Testing

When testing algorithms, it is important to consider their application. Apart from the solution quality, computation time is also an important issue in industrial applications and strongly depends on the problem size. To judge the suitability of the algorithms for a certain packing task, the impact of the problem size on the performance of the hybrid algorithms is investigated and compared with simple heuristic methods.

Problem Generators:
Due to the lack of benchmark problems in this area (section 5.3.2) new problem generators are developed for the rectangular case to create test problems automatically under consideration of certain parameters. So far, no algorithms for the automatic test case generation for 2D strip and bin packing problems have been published.

Benchmark Problems:
One of the major objectives in this work is to compare the performance of the hybrid algorithms with other meta-heuristic solutions. Since there is no standard test method, comparison is only possible where authors compare their work with other algorithms in the area or where the same test problems can be used. Unfortunately, the first option is not standard praxis, apart from a few exceptions (Blazewicz et al., 1993; Oliveira et al., 2000). Therefore a list of test cases will be compiled that were applied in earlier solution approaches to rectangular as well as irregular packing tasks. In that way, at least comparisons to a few other methods are possible.

Benchmark Tests:
At the same time an attempt is made to identify a possible benchmark method for rectangular packing tasks, which is easy to implement and performs reasonably well with respect to packing density. The heuristic nature of the method will help to reveal the merits of meta-heuristic search techniques over heuristic ones.

Commercial Nesting Software:
Since the meta-heuristic methods in this work have been developed for looking at possible industrial applications, standard industrial practice should be taken into consideration when judging the suitability of algorithms for certain tasks. Currently, the manufacturing industry has the choice between a number of commercial nesting packages. Apart from one work (Corno et al., 1997), researchers in this area do not seem to regard them as a possible and inevitable way to test whether their algorithms match industrial practice, if intended to do so. A number of software packages, which are suitable for rectangular and

irregular strip and bin packing tasks, have been identified in section 4.5. Two of them are used to measure the performance of the hybrid algorithms developed in this work.

Testing of Meta-Heuristics:

The performance of the hybrid algorithms is measured by applying a comprehensive test procedure. A comparison is made with the simple heuristic packing routines, which are used to hybridise the meta-heuristic methods. Since the meta-heuristics function as a tuner for the simple algorithm, they are expected to perform better in terms of layout quality.

In order to see whether a possible performance gain is purely due to a higher number of iterations, random search is applied testing the suitability of the genetic and neighbourhood operators to guide the search process into good areas of the solution space. The operation of the cross-over operator is verified studying its impact on the final solution quality with naïve evolution. Possible merits of seeding are analysed.

Apart from genetic algorithms, other meta-heuristic and heuristic methods were applied to packing tasks in the literature. Since little evidence exists whether some are more favourable than others, a selection has been included in this work involving hill-climbing, simulated annealing and stochastic optimisation, which has been previously used in the literature (Pargas and Jain, 1993). A comparison with some of the heuristic search techniques in the literature is possible through the application of the same test problems.

4.6.4.3 Implementation

One of the key issues in algorithm development is efficiency and reliability. Especially, the nesting algorithms used in irregular packing require a large number of complex and time intensive geometric computations. The implementation of geometric algorithms does not only consume precious time during the algorithm development, which could be better spent at different stages of the project, but also seems like "re-inventing the wheel" considering the amount of professional software available in this area. A software library has been used in the project, which offers reliable data types and routines for geometric primitives. An overview of software packages for computational geometry as well as description of the ones used is given in the following chapter.