Gent, september 2003

De auteur	De promotoren	
ir. E. Ducheyne	Prof. dr. ir. R. De Wulf	Prof. dr. B. De Baets

Acknowledgements

Four years ago I embarked on a new project: creating and writing a doctoral thesis. This was the most ambitious venture until that day. At the beginning of this project, I did not know what roads that I would take and whom I would meet along these roads. Looking back, I am really surprised how fortune might help even in the academic world and I count myself lucky to have been given this opportunity.

I started under the guidance of Prof. dr. ir. R. De Wulf whom I already knew well from my master's project. He suggested to me that in the domain of quantitative forestry there was an apparent lack of tools that combine optimisers and geographical information systems. He showed the many paths that I could follow and allowed me to take whichever road that I wanted to. He always encouraged me to talk to and visit other people on conferences and during visits abroad. For this I am very grateful.

Very early along the way I encountered Prof. dr. B. De Baets. He introduced me to this fairly new topic of genetic algorithms and to me genetic algorithms seemed extremely fascinating. Mimicking the struggle for life on a computer and by doing so solving forest management problems sounds even today quite extraordinary. We have spent much time together trying to figure out what the best way of tackling forest optimisation problems might be and Prof. De Baets' meticulous approach has been an example for me.

I continued working on the combination of GIS and GA but got stuck somewhere in the middle. I attended some major international conferences and on one of them I met Prof. dr. F. Lobo. He suggested me to visit him and Prof. dr. C. Fonseca at the Universidade do Algarve, whenever I wanted to. Next to Prof. Lobo, I was so lucky to meet dr. K. Matthews. He works currently at the MacCaulay Research Institute in Aberdeen, Scotland and painstakingly explained over and over how the basics of genetic algorithms work during the one week visit in Aberdeen and many times thereafter by e-mail. During my stay in Aberdeen, I also visited Prof. dr. A. Cameron. During the Socrates exchange programme some years earlier he taught me silviculture and also teaches

forest management at Aberdeen University. He keeps detailed records of all forest inventories from the master students and gladly shared the data with me. Without this data, there would have been no doctoral thesis!

The following year I found myself heading for Portugal in order to find inspiration and new ideas. Prof. dr. C. Fonseca, prof. dr. F. Lobo, dr. V. Grunert da Fonseca and A. P. Costa have exceeded their hospitality enormously and I cannot thank them enough for the wonderful time I had, both scientifically as well as personally. Muito obrigado!

Next to the people that I have met over the course of the four years, I owe also much to those that have been with me from the beginning. My colleagues and thesis students at the laboratory (in alphabetic order) Freya Danckaert, Eva De Clercq, Rudi Hoeben, Koen Mertens, Annelies Sevenant, Nancy Van Camp, Frieke Van Coillie, Lieven Verbeke, Inge Verbiese, Jan Verhoeye and Toon Westra have always been very helpful and supportive during all the stages of my work and I am very glad that even after the end of the doctoral work our roads will not part entirely! Furthermore, I must mention that the 'cake'-moments together with the laboratory of hydrology were also quite inspiring!

In the final phase of my work, the members of the examination committee, Prof. dr. ir. J. D'Haeyer, Prof. dr. C. Fonseca, Prof. dr. ir. N. Lust and dr. ir. M. Waterinckx, have given me good advise and constructive criticism. This surely improved the dissertation in many ways. I would also like to mention the enormous effort that Mrs. J. D'Hondt has put in reading my dissertation for any mistakes against Shakespeare's language. Being complete ignorant of anything that has to do with either forest management or genetic algorithms that was certainly not an easy task.

Finally, I dedicate this work to my family and friends. They were always prepared to listen to my enthusiastic tales as well as my sorrows or worries! Mama en papa, van harte bedankt voor alles! Ook voor jullie waren de laatste maanden niet altijd even gemakkelijk, zeker niet wanneer 'diepe dalen' moesten overwonnen worden. Dany, Annick, Jelle, Kaat en marraine, jullie brachten aangename verlichting in de 'bezoekjes-sperperiode'. Lulu, Jan en Bram, ook jullie stonden er altijd! Oma's, ooms en tantes, bedankt! Nathalie en Bart, Steven en Laura, Sara en Henri, Fanny, en Alain dat we elkaar nog regelmatig kunnen ontmoeten. Veerle en Gwen, bedankt voor de brieven uit Cuba, de vrijdagnamiddagen werden er alleszins mee opgefleurd. Most importantly however, I would like to say to Thomas how much he means to me. When we were writing up our dissertations together, he has been an steadfast rock to me even though he had his own worries. He always reminded me to remain calm and said that everything would work out just fine. This work shows, once more, that he is entirely right. Without him, I would not have been able to reach the end. Thomas, bedankt!

Els Ducheyne
Gent, September 2003

Contents

List of Figures

List of Tables

List of Abbreviations

BOA	Bayesian Optimisation Algorithm
CC	Combined Complexity
cGA	Compact Genetic Algorithm
COM	Component Object Model
CPC	Compressed Population Complexity
DAG	Directed Acyclic Graph
DDE	Dynamic Data Exchange
ECGA	Extended Compact Genetic Algorithm
EDA	Estimation of Distribution Algorithm
ER	Error ratio
GA	Genetic Algorithm
GD	Generational Distance
GIS	Geographical Information System
GUI	Graphical User Interface
MC	Model Complexity
MDL	Minimum Description Lenght
MOGA	Multiple Objective Genetic Algorithm
MOOP	Multiple Objective Optimisation Problem
NSGA-II	Non-dominated Sorting Genetic Algorithm-II
SDSS	Spatial Decision Support System
VBA	Visual Basic for Applications
VEGA	Vector Evaluated Genetic Algorithm

Chapter 1

Problem definition and research objectives

1.1 Problem definition

Forests should be managed as to meet multiple (often conflicting and incommensurable) objectives (Afdeling Bos en Groen, 2000). In the Flemish forest law, timber production is a very important but no longer the only management objective. In the forest management literature, however, most models focus on this purely economic function of the forest. These models were not developed to allow objectives that require spatial data such as wildlife conservation and recreation and are unable to handle spatial objectives. Linear programming, for example, uses continuous variables, and this is not suitable when spatial integrity is of concern. Integer and mixed-integer programming overcome this problem but in order to explicitly formulate the spatial requirements very large integer programs are needed. These programs cannot be solved even with today's computing power. Heuristics have been proposed as a means to handle these complicated optimisation problems, and are indeed capable of solving them. However, they suffer, as the linear or integer programs do, from the fact that they are essentially single objective optimisers. This requires that the multiple objectives have to be reformulated into a single objective function and this hampers the search for the trade-off front between these objectives. Until today, there has been no report on the use of a truly multiple objective and spatial optimiser in forest management. The integration of the optimisers together with a geographical information system has also been long advocated but in reality few (if any) real-world applications that combine GIS and optimisers online in forest management have been reported.

1.2 Objectives and research questions

Clearly, the need for an optimisation technique that can handle both multiple objectives and spatial information emerges. The main objectives in this dissertation can be stated as:

1. develop a *forest management planning tool* that generates alternative plans for *multiple objectives*;

2. this tool should be *flexible* and allow the integration of *spatial data* and *GIS functionality*;

3. this tool should be *efficient* and preferably *fast*.

These three objectives can be met if the following research questions are answered:

1. What is the current state-of-the-art of optimisation techniques in forest management and what are the shortcomings of these approaches?

2. What techniques are available in operations research or computational intelligence that optimise multiple objectives without the need for prior aggregation?

3. Which of these are flexible enough to allow integration with GIS?

4. What are the basic assumptions for using these techniques; if they are not met in the case of forest management, how can these techniques be adapted? This can be split into the following sub-questions:

    a. What is the effect of the encoding strategy on the solution quality?

    b. Are forest management problems deceiving the search and optimisation process?

5. How can these techniques be improved so that they find optimal solutions in a faster way?

1.3 Experimental setup

1.3.1 Research question 1

The first research question is answered through an extensive review of the existing techniques that have been used both in forest management planning and in related domains such as urban and land use planning.

1.3.2 Research questions 2 and 3

Genetic algorithms are proposed as optimisation tool for forest management planning because

- they do not combine multiple objectives prior to the optimisation process;
- they generate multiple alternatives in a single optimisation run due to their population-based approach;
- they allow easy integration between the optimisation module and GIS functionality.

Two forest management problems, a harvest scheduling problem and a scheduling problem involving spatial data, are first solved using a single objective genetic algorithm. Its performance is compared to that of existing methods and the validity of the alternatives is analysed in detail. This is followed by a comparative study of two widely applied multiple objective genetic algorithms in research domains such as control and civil engineering. For this study, a forest benchmark problem with known best alternatives is used so that the performance of the two optimisers in the domain of forest management can be investigated. In the next step the harvest scheduling problem and the spatial problem are solved using the best multiple objective genetic optimiser. The results from the multiple objective genetic algorithms are compared to those of the single objective optimiser and the advantages of the multiple objective approach are identified. As in genetic algorithms the function evaluations are completely separate from the optimisation process itself, it is possible to include GIS in the process. The GIS-module stores and analyses the spatial data and models and there is no need for the optimiser module to include extra decision variables for the spatial data. In the second multiple objective case study, which includes spatial information, the advantages of combining GIS and GA are shown. First a benchmark problem from literature is treated, and later this is applied to the study area.

1.3.3 Research question 4

This research question is twofold and can be split into two sub-questions.
In genetic algorithms, the decision variables have to be encoded in a chromosome. Several encoding strategies are possible and the first sub-question therefore is (research question 4a): *'How does the encoding strategy of the decision variables affect the solution quality'*. This question is dealt with separately for each case study. For the harvest scheduling case study, three encodings are possible and the effect of each of them on the solution quality is determined for both the single and multiple objective case.

Due to the two-dimensional nature of spatial information, a linear representation of the forest management decision variables might violate the basic assumptions of genetic algorithms and this could lead to suboptimal solutions.

The second sub-question can formulated as (research question 4b) : *'Are linkage learning operators necessary when genetic algorithms are used for forest management optimisation problems'*. To that end, two advanced methods that overcome this problem are applied to the spatial problem in order to ascertain whether forest management problems require specialised techniques.

1.3.4 Research question 5

Fitness inheritance is an efficiency enhancement technique that has been proposed in literature to speed up the genetic optimisation process if the function evaluations are very time-consuming. Because it has only been applied to simple problems, fitness inheritance is tested on three test functions from a benchmark test suite of functions. Their behaviour is analysed using various performance indices. After they have been tested, fitness inheritance is applied to a case study for which the assumptions for using fitness inheritance hold.

1.4 A road map to this dissertation

This dissertation consists of three main parts. The first part is a general introduction to the basics of forest management. The second part includes all theory and case studies on simple single and multiple objective genetic algorithms. The final part introduces both theory and practice of advanced genetic algorithms.

Part I consists of two chapters and answers research question 1. Chapter 2 commences with a definition of forest management and its implementation in the framework of the Flemish Community. This is followed by a literature review of models that form the basis of quantitative forest management. Chapter 2 continues with a review of the existing optimisation techniques that are commonly used in forest management and current shortcomings and pitfalls are outlined. Next to the techniques that are used in theory, those that are used in the real world are discussed. Finally, a summary and conclusion are written. The second chapter of Part I (Chapter 3) briefly describes the study area. This study area is used in the case studies throughout this dissertation.

In Part II and Part III, the design of the dissertation is as follows: first a chapter concerning the theoretical aspects is presented. This is then followed by relevant case studies. These case studies clarify the theory and show particular advantages and disadvantages of the methods that were described earlier.

Part II starts with the theoretical background on simple genetic algorithms (Chapter 4). The fundamentals of genetic algorithms are discussed, together with issues such as the correct representation of chromosomes and the use and parameter setting of genetic operators. This is followed by some guidelines for forest management problems. These guidelines are used in the two case studies

that follow this theoretical chapter. In the first case study (Chapter 5), the harvest scheduling problem, a well-studied forest management problem, is solved. The two objectives in this case are restated into a single objective formulation prior to the optimisation process. The second case study (Chapter 6) focuses on the integration of spatial information during the optimisation process. Using a simple genetic algorithm, the perimeter between old growth and clear felled areas is maximised. The results of the genetic algorithms are compared to those found in literature.

In Chapter 7 the single objective genetic algorithm is extended to the multiple objective case. The modifications needed for this extension are briefly discussed. Next to the extension, performance indices for multiple objective optimisers are reviewed. This is followed by a comparative study of two algorithms (Chapter 8) that are described in literature as good starting points. These are then compared on a forest benchmark problem. For this real-world problem the optimal values are known, and hence it is possible to test the two implemented algorithms. Both algorithms are analysed in detail and the best algorithm is retained for the rest of the dissertation. This algorithm is used for the harvest scheduling problem, which is now optimised as a multiple objective problem (Chapter 9). The objectives are simultaneously optimised without prior aggregation of the objectives. This approach is compared to the results obtained in the single objective formulation, and the validity of the harvest scheduling plans is investigated. After this case study, the focus is again on the integration of spatial data in the optimisation process (Chapter 10). Several ways of operational links between genetic algorithms and GIS are reviewed, and the multiple objective extension of the edge-dependent optimisation benchmark problem is solved. Thence, the same strategy is again applied to the study area. Overall, Part II addresses both research questions 2 and 3. Research question 4a is also answered.

In the final part of this dissertation (Part III), more advanced genetic algorithms are applied to forest management. The first chapter in this part (Chapter 11) commences with a discussion of the disadvantages of blindly applying genetic algorithms to forest management problems. Chapter 11 discusses the consequences of applying the simple genetic algorithm and tries to formulate an answer to research question 4b. As spatial data is by nature two-dimensional (or even three-dimensional), simply using linear chromosomes might violate the basic assumptions of genetic algorithm theory. Algorithms that can overcome this problem are then reviewed and two promising ones are fully described. These two algorithms are again applied to a case study in Chapter 12 to illustrate their value in the optimisation process.

Finally, a major impediment to the common use of genetic algorithms in forest management is that it can be time-consuming. Especially when complex spatial models are used to derive the objective function values, evaluating many solutions is very costly. Therefore an efficiency enhancement technique called fitness inheritance was proposed in literature. As there is little known about

where fitness inheritance can be applied, it is initially applied to a test bed of benchmark problems in Chapter 13. The effect of fitness inheritance is investigated by means of several performance indices, and after this analysis follows the framework wherein fitness inheritance can be used. This chapter provides an answer to research question 5. In the final case study (Chapter 14), fitness inheritance is applied to the harvest scheduling problem, indicating the advantages or disadvantages for a real-world application.

Part I

Basics of forest management

Chapter 2

General introduction

2.1 Defining forest management

The need to manage forests to reach multiple objectives has been present for a long time, but since the beginning of the 1970s this need has become urgent in Flanders. This is mainly due to the raise of living standards leading to more spare time and the increasing societal pressure for environmental issues (Afdeling Bos en Groen, 2000). Many countries such as the USA, Great Britain and the Netherlands explicitly recognise these objectives in their forest laws. In the Flemish forest law (Vanhaeren, 2002), timber production remains a very important forest function, but is no longer seen as the sole management objective of the forest (Janssens & de Schuyter, 1990). The forest law explicitly lists those functions that are potentially conflicting with the timber production. These functions are: *the social and educational function, the environmental protective function, the ecological function and the scientific function.*

Dealing with the complex problem of integrating timber production with other values and benefits requires a more comprehensive and spatial approach than has traditionally been applied to the management of forest ecosystems (Brown & MacLeod, 1996). Before providing a literature review of tools for the design of forest management plans, it is necessary to define what forest management is. First some definitions found in literature will be listed, and second by using their common characteristics a definition for this dissertation will be presented.

Loomis (1993) states that the management of natural resources can be defined as the organisation or coordination of natural resources and the human input of labour, capital and knowledge. Management involves not only the coordination but also the control and scheduling of used resources. Leuschner (1990) defines forest management very broadly as the application of a wide range of scientific, economic and social principles to administer and solve problems in

forested areas. These principles can include, but are not necessarily limited to, those principles developed in forest science. This is the same definition as the one given by Buongiorno and Gilles (1987):

> "The art and science of making decisions with regard to the organisation, use and conservation of forests."

In academic circles, the term forest management has a more restricted meaning (Leuschner, 1984):

> "The study and application of analytical techniques to aid in choosing those management alternatives that contribute most to the organisational objectives."

This is very similar to what Tarp and Helles (1997) mention:

> "Forest management planning involves the selection of treatments for each of the management units in the forest. These administrative units also form the basis for incorporating variations of growing conditions and weighing of primary objectives within a multi-use concept."

A more recent definition by Davis et al. (2001) has the same key components:

> (Forest management) "is the art and science of growing, harvesting, protecting and manipulating trees and related vegetation and helping land owners, affected parties and society to sort out their options and understand the trade-offs involved in achieving their contemporary mix of forestry-related goals."

According to Berck (1999), forest management is the manipulation of the forest to produce different mixes among the uses of the forest. This manipulation is undertaken to achieve a certain set of objectives. Management furthermore decides the relative importance of forest objectives. Tarp and Helles (1997), Leuschner (1984) as well as Berck (1999) therefore consider forest management planning as a technical tool to enable decision making concerning silvicultural treatments in such a way that —multiple— objectives are met. Bos (1994) finally states that forest management can be defined as the set of human activities aiming at a sustainable fulfilment of the needs of society by adjusting the forest ecosystem to those needs and adjusting society to the forest.

All these definitions are related to the arrangement or manipulation of forest management units through silvicultural treatments in order to solve problems in forests based on forestry principles and also techniques from outside the realm of forest science. Moreover, all these definitions implicitly stress the central role of decision making in forest management. The kernel of decision making in forest management is therefore to answer the following question (Lammerts van Bueren, 1983):

> "Which mix of functions and what level of fulfilment per function maximises the satisfaction of society's needs"

The answer to the question should be determined by society's needs, the physical suitability of forest land to fulfil functions, economic implications and social acceptance (Lammerts van Bueren, 1983).

Using the common characteristics found in the definitions from literature, forest management will be defined as follows:

> "A set of human actions that will produce a set of forest goods and services in such a way that the varying needs of society are met in an optimal way."

2.2 The forest management planning process

Planning in most of the forest operations is a complex task that can result in major financial losses if conducted suboptimally (Robak in Brack and Marshall (1996)). A procedure for scheduling forest operations (treatments) across space (the forest management units) and time (planning horizon) is a key component of forest management planning. According to Lammerts van Bueren (1983) the natural resource use can be evaluated by land evaluation procedures. Because society has demands and needs, some types of land use such as forest use are more suitable to fulfil these needs than others. The preferences of the user can then be defined in terms of a product that has to be realised and the methods and means to obtain this product. This product can be qualitative (e.g. nice scenery in the forest) as well as quantitative (e.g. timber volume). During the planning process the forest owner (private or public) should answer the following questions in planning (Fig. 2.1) (Loomis, 1993; Lammerts van Bueren, 1983; Speidel, 1972):

1. Where are we?

2. Where do we want to be?

3. What alternative actions will get us there - what is in our means?

4. Did we make it?

Carlsson (1999) bundles these steps into two groups:

1. Identification of objectives and alternatives;

2. Valuation of alternatives.

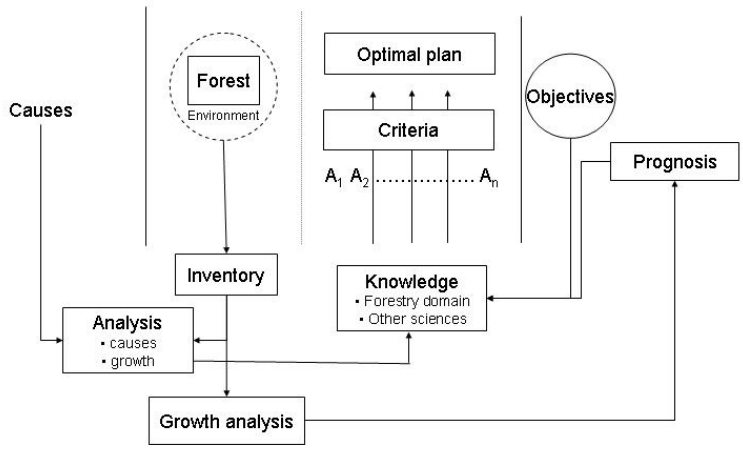


Figure 2.1: The planning process (Speidel, 1972)

The determination of the most socially desirable alternative requires agreement on the *objectives*. All the effects, both desirable and undesirable, of these objectives should be known. This implies that the first task in defining forest management is to state the objective or objectives of the management plan. This is not a trivial task and is likely to be a source of contention in itself (Mendelsohn, 1996). The purpose of analysis in this case is to ensure that all the potential effects associated with each objective have been identified, and quantified where possible. Then it is up to the agency or political decision makers to judge which objectives are most important (Loomis, 1993).

In a decision making process four possible ways exist to select between alternative strategies or solutions (Hwang & Masud, 1979):

1. No articulation of preference information is provided. This means that methods following this approach do not need any inter-objective or subjective preference information from the decision maker once the problem constraints and objectives are defined.

   - Pros: during the process of obtaining the solution the decision maker is not disturbed by the analyst
   - Cons: the analyst has to make assumptions about the decision maker's preferences

   There are no real-world forest applications of this category.

2. *A priori* articulation of preference information. In this case the preference information is given to the analyst before the optimisation procedure commences. The optimisation techniques that are traditionally being used in forest management are based on the *a priori* articulation because they require a single objective function as input.

   - Pros: a whole range of classical single objective optimisers can be applied to solve the problem.
   - Cons: requires prior preference information which can be difficult to give when the optimisation problem is very complex or when the objectives are incommensurable.

3. Progressive articulation of preference information. This class of methods relies on the progressive definition of the decision maker's preferences along with the explorations of the criterion space. During the search progress, the decision maker is repeatedly asked to give some trade-off or preference information based upon the current solution in order to determine a new solution. These methods assume that the decision maker is unable to indicate prior preferences due to the problem complexity, but can give this information at a local level to a particular solution: given a small number of solutions the decision maker can indicate the preference for the objectives.

   - Pros:
     - there is no need for prior preference information, only local preference information is needed. This means that the decision maker can articulate the preferences based on solutions produced by the analyst;
     - it is a learning process for the decision maker to understand the behaviour of the system;
     - since the decision maker is involved the obtained solution has a better prospect of being implemented.
   - Cons:
     - solutions depend on the accuracy of the local preference the decision maker can indicate;
     - there is no definitive guarantee that the preferred solution can be obtained in a finite number of iterations – that is the number of times that the analyst has to go back to the decision maker to ask for the local preference;
     - much effort from the decision maker is required.

   This has been used in group decision making for forest management problem such as in Faber et al. (1998): several stakeholders have to give their preferences over the objectives through ranking and proposal evaluation.

Combining the preferences leads to the solution which is acceptable for all parties involved.

4. *A posteriori* articulation of preference information. The methods of this class determine a subset of the complete set of solutions that make up the trade-off front between the objectives. If there is no prior preference articulation, then no solution on the trade-off front is worse than another, because an improvement in one objective dimension has the opposite effect in the other dimension. From this set the decision maker chooses the most satisfactory solution, making implicit trade-offs between objectives. In many cases the trade-off information is received from the decision maker after the method has terminated and the subset of solutions on the trade-off has been generated.

   - Pros: does not require any assumption or information concerning the decision maker's utility function.
   - Cons: these methods usually generate a large number of solutions on the trade-off front — it becomes nearly impossible for the decision maker to choose the most satisfactory one.

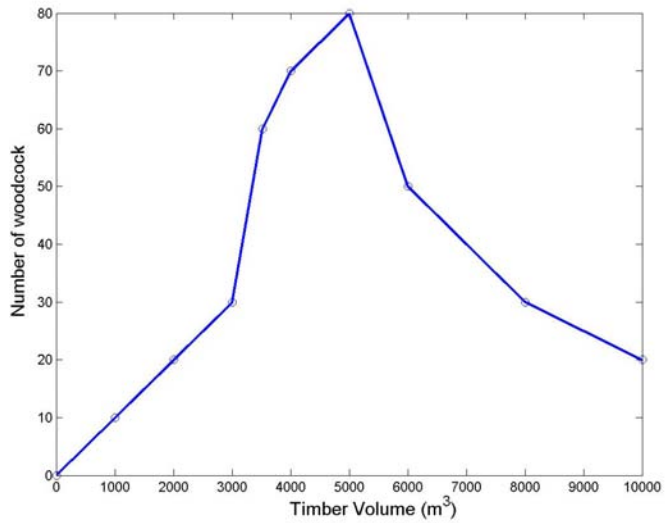
Looking for the solutions on the trade-off front and dropping inferior solutions helps to simplify a complex forest ecosystem management problem, so that it is dealt with more easily (Loomis, 1993). If an algorithm is capable of finding all these solutions, the complex problem can be reduced to choosing the best solution afterwards by the decision maker. Moreover, if the trade-offs between the objectives can be given explicitly, it also becomes easier to compare the alternatives (Carlsson, 1999).

Production possibility frontiers, or efficiency frontiers (Davis et al., 2001), have been used for this purpose and can be constructed using generating techniques (Carlsson, 1999). A generating technique is a method for solving a multiple objective problem repeatedly to obtain solutions on the trade-off front. Those solutions can then be used to define the production possibility frontier. A production possibility frontier shows the trade-offs between two products or objectives. While the production possibility frontier cannot answer the question if there is enough of one or the other objective, it is a useful starting point for debate (Davis et al., 2001). Finding the cheapest way to reach the goals is important to policy making and management planning for at least two reasons (Davis et al., 2001):

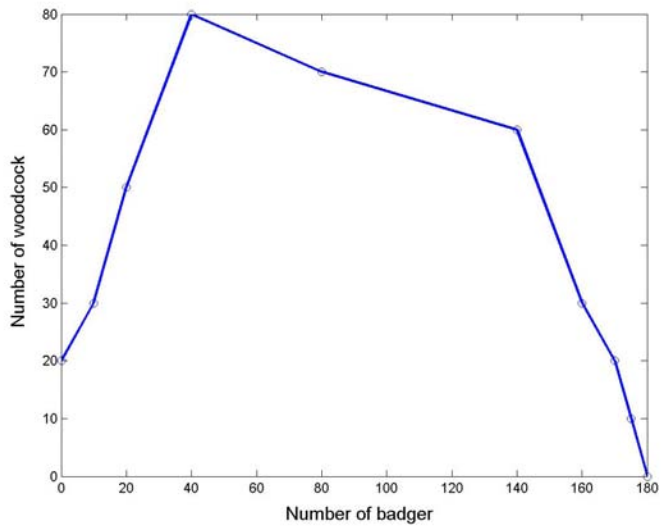
1. Inefficient solutions can result in needless conflict because the trade-offs or costs are higher than they really are.

2. The higher the costs of achieving a goal, the less likely it is that society will choose high levels of that goal.

Two examples of production possibility frontiers are given in Fig. 2.2. These trade-offs can be between market and non-market products as well as between two non-market products.



(a) Trade-off between timber volume and woodcock



(b) Trade-off between badger and woodcock

Figure 2.2: An example of a production possibility frontier between two products competing for the same resources. Fig. 2.2(a) represents the trade-off between a market product (timber volume) and a non-market product (abundance of woodcock) and in Fig. 2.2(b) the trade-off between two non-market products: abundance of badger versus abundance of woodcock is depicted

2.3 Quantitative models for planning: literature review

Many forest management problems have an almost infinite number of alternatives. A typical forest management problem recognises a multitude of stands and a variety of actions over time that could be taken in the stands to achieve the objectives of the forest management problem. As the number of choices can get very large, optimisation techniques are commonly applied. Before the optimisation techniques are described, the basic formulation of the harvest scheduling problem and other quantitative models necessary in the planning process will be elaborated in the following sections.

2.3.1 Basic formulation of the harvest scheduling problem

Classical scheduling methods were designed to regulate timber harvests. In modern forestry this approach remains essentially the same, even though social context and technological capabilities are very different (Roise et al., 2000). Much work on forest management planning emphasises economic approaches in which achievement of some goals is maximised subject to constraints of other goals. The strong economic flavour of this planning approach has some disadvantages because production efficiency is emphasised whereas spatial integrity is ignored (Davis et al., 2001).

In forest management two basic ways of formulating the optimisation problem are commonly used: Model I and Model II (Johnson & Scheurman, 1977). These two models differ in their definition of an activity. According to Johnson and Scheurman (1977) all forest optimisation problems can be reduced to one of these two formulations.

Garcia (1990) refers to the forest management from a state space view. Any system that evolves in time can be described by a state that characterises the system at some point in time, and a transition function that specifies how the state changes over time. The state of the forest at the start of period $t + 1$ is a function of the state at period t and the actions in period t. Using this concept, it is possible to represent the problem in terms of network flows (Fig. 2.3).

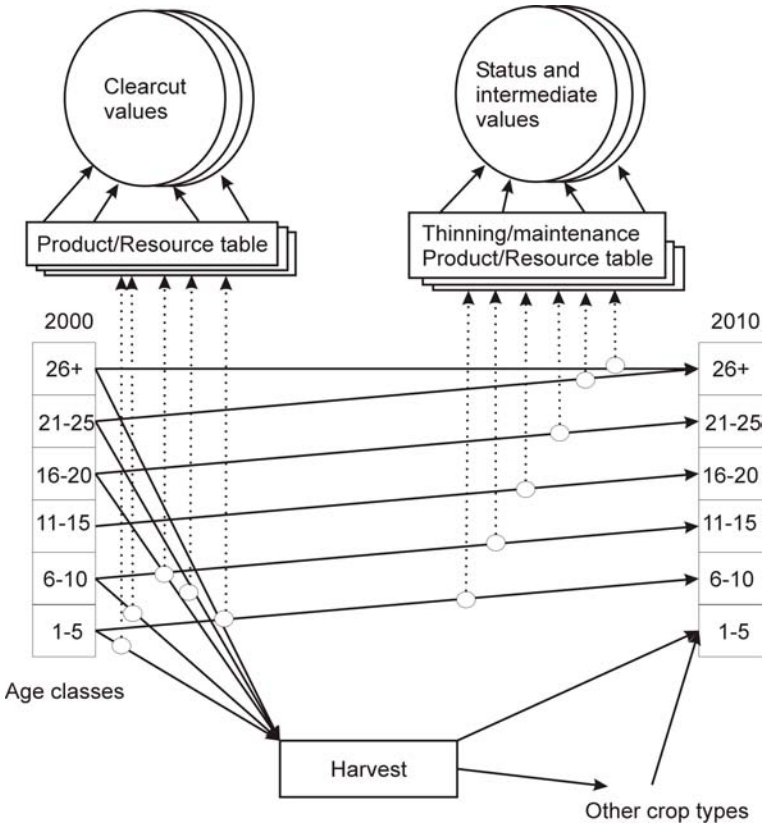


Figure 2.3: The forest management process can be represented from a state space view. The process can be seen as a system, described by its state, that evolves in time, described by its transition function (Garcia, 1990)

2.3.1.1 The Model I formulation

In Model I, an allocation option refers to a complete set of management actions that will occur on a particular land unit over the entire planning horizon. The structure of Model I is very clear: each activity represents a possible management regime per land unit with its associated inputs and outputs throughout the planning horizon. A Model I can handle both even-aged or uneven-aged management strategies (Bos, 1994). The objective function of the Model I formulation is to maximise the net present value obtained from the forest stands (Eq. 2.1) subject to area constraints for each management unit (Eq. 2.2).

Maximise

$$\sum_{l=1}^{U}\sum_{q=1}^{R_l} D_{lq}x_{lq} \tag{2.1}$$

subject to

$$\sum_{q=1}^{R_l} x_{lq} = A_l l = 1, \ldots, U \tag{2.2}$$

where

$x_{lq} =$ hectares of forest management unit l assigned to regeneration harvest sequence q

$A_t =$ number of hectares in forest management unit l

$U =$ number of management units

$R_t =$ number of possible regeneration harvest sequences over the planning horizon for forest management unit l

$D_{lq} =$ discounted net value per hectare of forest management unit l over the planning horizon, if assigned to regeneration harvest sequence q. D_{lq} can be decomposed into more understandable terms:

$$D_{lq} = \sum_{j=1}^{N} \frac{P_{lqj}V_{lqj} - C_{lqj}}{\gamma^j} + \frac{P'_{lqN}}{\gamma^N}$$

where

$N =$ number of periods in the planning horizon

$P_{lqj} =$ unit price of harvest in period j from forest management unit l under regeneration harvest sequence q

$V_{lqj} =$ volume per hectare harvested in period j from forest management unit l under regeneration harvest sequence q

$C_{lqj} =$ cultural treatment costs per hectare in period j for forest management unit l under regeneration harvest sequence q

$\gamma^j =$ discount rate for period j

$P'_{lqN} =$ net value per hectare placed at the end of the planning horizon, i.e. in period N on forest management unit l under harvest regeneration sequence q

2.3.1.2 The Model II formulation

In Model II, each age class containing hectares in the first period forms a management unit until these hectares are harvested. An allocation option refers to a complete set of actions that will occur on a particular land unit from the time

the land unit is regenerated until it is harvested or until it is left as ending inventory at the end of the planning horizon. This means that there are decision variables for existing stands and different ones for regenerated stands, each time a stand gets harvested it gets transferred to another decision variable. Therefore it is less easy to deal with uneven-aged management practices, because the definition is based on the existence of regeneration harvest which does not occur in uneven-age management. The objective function of the Model II formulation (Johnson & Scheurman, 1977) is written in Eq. 2.3. This objective function is subject to area constraints Eq. 2.4 and Eq. 2.5

Maximise

$$\sum_{j=1}^{N} \sum_{i=-M}^{j-Z} D_{ij} x_{ij} + \sum_{i=-M}^{N} E_{iN} w_{iN} \tag{2.3}$$

subject to

$$\sum_{j=1}^{N} x_{ij} + w_{iN} = A_i \qquad i = -M, \ldots, 0 \tag{2.4}$$

$$\sum_{k=j+Z}^{N} x_{jk} + w_{jN} = \sum_{i=-M}^{j-Z} x_{ij} \qquad j = 1, \ldots, N \tag{2.5}$$

where

$x_{ij}(x_{jk}) =$	hectares planted in period i (period j) and harvested in period j (period k)
$w_{iN}(w_{jN}) =$	hectares planted in period i (period j) and left as part of the ending inventory in period N, the hectares that are uncut at period N
$A_i =$	number of hectares present in period 1 that were planted in period i ($i = -M, \ldots, 0$), with each A_t being a constant at the beginning of the planning horizon (period 1)
$M =$	number of periods before period 0 in which the oldest age class present was planted
$Z =$	minimum number of periods between regeneration harvests
$D_{ij} =$	discounted net value per hectare from hectares planted in period i and harvested in period j. D_{ij} can be expressed as follows:

$$D_{ij} = \sum_{k=\max(i,1)}^{j} \frac{P_{ikj} V_{ikj} - C_{ikj}}{\gamma^k}$$

with

P_{ikj} = unit price of volume harvested in period k on hectares planted in period i and harvested in period j

V_{ikj} = volume per hectare harvested in period k on hectares planted in period i and harvested in period j

C_{ikj} = cultural treatment costs per hectare in period k on hectares planted in period i and harvested in period j

E_{iN} = discounted net value per hectare during the planning horizon from hectares planted in period i and left as ending inventory in period N plus discounted net value per hectare of leaving these hectares as ending inventory

$$E_{iN} = \sum_{k=\max(i,1)}^{N} \frac{P_{ikN}V_{ikN}}{\gamma^k} + \frac{P'_{iN}}{\gamma^N}$$

where

P_{ikN} = unit price of volume harvested in period k on hectares planted in period i and left as ending inventory in period N

V_{ikN} = volume per hectare harvested in period k on hectares planted in period i and left as ending inventory in period N

C_{ikN} = cultural treatment costs per hectare in period k on hectares planted in period i left as ending inventory in period N

P'_{iN} = net value per hectare of leaving hectares in period i as ending inventory in period N

Eq. 2.4 denotes the area constraints for the Model II formulation and states that the inventory of land planted at time i, this being A_i, is either cut or left standing. Eq. 2.5 denotes that the land harvested in year j from all previous standing planting is available for harvesting between year $j + z$ and year N or can be left standing (Berck & Bible, 1984).

2.3.1.3 Discussion

Both Model I and Model II have their merits for timber harvest and activity scheduling. When ecological considerations need to be incorporated, the Model I formulation has a slight advantage because a unit of land can easily be tracked from beginning to end of the planning horizon. Another advantage is that integer constraints on management units can be easily tracked from start to finish (Roise et al., 2000; Davis et al., 2001). Both models, however, are very weak in

terms of ecological forest planning and analysis because they are stratum-based models, while most ecological processes require spatial integrity. The problem with stratum-based approaches is that the links between data and their location gets destroyed, making it impossible to evaluate spatial implications of alternatives (Carlsson, 1999).

2.3.2 Growth modelling

Forest management decisions are predicated on information about both current and future resource conditions (Avery & Burkhart, 1994). Inventories taken at one instant in time provide information on current volume and related statistics. Forests, however, are dynamic systems, and it is necessary to project these changes to obtain relevant information for prudent decision making. The term *growth* of a tree is the increase over a given period of time, while *yield* can be defined as the total amount available for harvest at a given time. Hence, yield can be regarded as the accumulation of the annual increments.

A forest growth model describes the development of tree crops as they get older (Philip, 1994). The term forest growth model can be somewhat misleading because only the growth of the trees and not of the whole ecosystem is modelled. The design of a growth model depends on the resources available, on the uses to which it will be put and the structure of the tree stands, either even or uneven-aged, of a single or mixed species. The four most common uses are (Philip, 1994):

- to predict the growth of the forest so that the manager may match his harvesting and selling plans against the prediction of growth and conclude whether he is cutting more or less than, or an amount equal to, growth;
- to predict growth on a particular site to enable the land manager to make rational decisions. Often the growth model is required to provide information for conversion into economic measures to facilitate comparisons of a number of feasible investment options;
- to predict growth of crops under different management regimes and silvicultural practices in order to make comparisons and a choice;
- to predict work programmes when budgeting costs and revenues.

The growth models of forest crops can be classified as in Fig. 2.4. A *stand growth model* describes the stand and predicts growth through general parameters, such as the total basal area per hectare, mean values, and definitions of frequency distributions. A *single tree growth model* on the other hand, predicts the growth of individual trees and synthesises stand growth from the sum of a representative sample of individuals.

A *static model* predicts stand or tree volume at a stated time and infers growth by subtracting the previous from the current standing volume. A *dynamic model* predicts volume increment directly and deduces cumulative vol-

umes by summing growth. Finally, a distinction can be made between *deterministic models* and *stochastic models*. The first type predicts the expected values under a given set of conditions, whereas the second type of models includes uncertainty in the outcome through the incorporation of random variables, and thus adjusts the prediction by including the effect of stochastic elements.

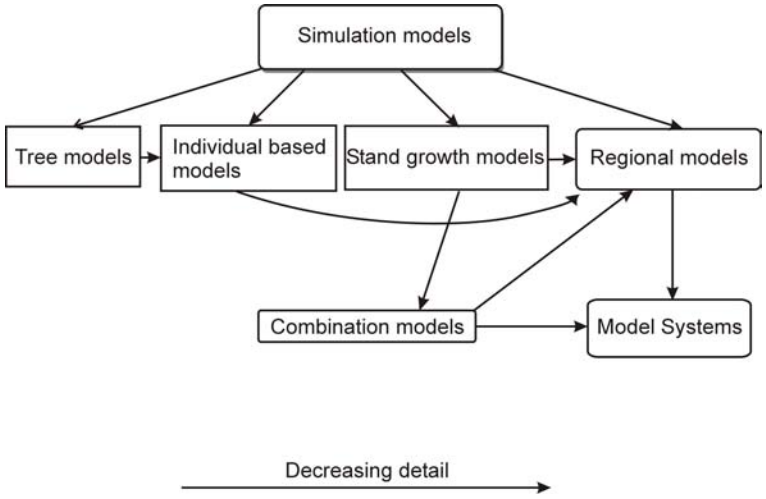


Figure 2.4: A classification of growth models (after Chertov et al. (1999))

Most authors in forest management literature for instance Carter et al. (1997), Barrett et al. (1998) and Church and Daugherty (1999), use stand growth models to predict the further growth, because of their simplicity (Philip, 1994). A limitation of stand growth models is that they consider each site in isolation. As a result, they require that the site under scrutiny is independent of all other sites (Davis & Martell, 1993). The factors that are included in these stand growth models are those factors that most closely relate to growth and yield of forest stands. These factors are (Avery & Burkhart, 1994):

1. the point in time in stand development;
2. the site quality;
3. the number of trees that occupy the site.

For even-aged stands, these factors can be expressed quantitatively through the variables of stand age, site index, and stand density. Stand age and stand density are parameters that are recorded in a standard inventory base. The site index is a measure for the site quality for the given species and is derived from the relation between the top height in a stand at a standard age. The

British Forestry Commission, for instance, designed simple yield classes for tree species commonly found in the United Kingdom. In these yield tables every curve refers to a growth model and these curves differ by intervals of 2 m^3 ha^{-1} yr^{-1} in maximum mean annual increment (Hamilton & Christie, 1971). If the yield class of the site and the age of the stand is known, it is possible to derive the stand top height. This information can be used as input in the production tables. The production tables can also be used directly with the yield class and age as input variables.

2.3.3 Forest valuation

As the revenue received from the timber production is in the future, it is necessary to transfer the future value into current value before it is possible to compare alternative plans that have different revenue at fixed points in time. A central axiom of economic forestry is that discounted net value measures the human benefit from the use of forest resources. Net present value – the discounted values of revenues and costs from the use of forest resources over time – is often used to estimate net benefit (Davis et al., 2001). The net present value can be calculated as in Eq. 2.6

$$V_0 = \frac{V_t}{(1+i)^t} \tag{2.6}$$

where V_t is the net revenue obtained at period t, and i is the discount rate.

In forestry, due to the long term planning horizon, managers favour low discount rates. As low discount rates tend to magnify the future values, they will encourage conservation and longer planning horizons. High discount rates on the other hand will lessen these future values (Duerr, 1993) and will favour alternative plans with a shorter rotation because a tree reaches financial maturity at a time long before it would be considered old growth (Davis et al., 2001).

Some forest managers propose to use a zero compound rate, because at this interest rate, a future decision, no matter how distant in the future has the same significance as a decision taken today. Any property capable of yielding a perpetual return has an incalculably high value. In general, however, the compound interest rate is set between 3% and 6% (Nalli et al., 1996; Davis et al., 2001).

One has to be cautious about compound interest. Its most dangerous implication is that the future can be clearly foreseen, but the longer the planning period and the lower the rate of interest, the greater the hazards of the compound interest are (Duerr, 1993). Some forest economists therefore argue that a compound rate can only be used for a certain period of the planning horizon (say for example the first 30 years) and then the nominal value should be used for the remainder of the planning horizon. Church and Daugherty (1999) describe alternative approaches that consider a form of equity between current and future generations as represented by the periods in the planning horizon. They combine both the nominal net revenue in each period and each period's present

net worth with values discounted relative to each period. These measures are used as criteria for welfare. Using the net present worth relative to each period will maximise economic efficiency and allows for variation in net revenue. Using net revenue focuses on a more even distribution of net revenues between the periods.

2.3.4 Ecological models

If long-term management planning should encompass multiple goals, for example economic and ecological, it is necessary to blend traditional economic forestry with ecological models (Davis et al., 2001). If ecological objective functions are included in the optimisation process, it is critical to develop the ecological models that will determine the best management activity for each forest management unit in order to maximise these ecological goals.

Ecological models model complex dynamic systems and they try to tie changes in one aspect of the model to changes in other parameters. These models should also be able to determine the impact of various management actions. These actions could include any forest management activity (Mendelsohn, 1996). A key question in forest ecosystem management is how the selected size, shape and distribution of harvests influences the spatial characteristics of natural disturbances (Seymour and Hunter, 1999 in Davis et al. (2001)). Patch size, the amount of edge, and the continuity of the patches within a forest can all affect the ability of that forest to support different species (Davis et al., 2001). The emphasis on composition, structure and processes within ecological systems directs attention to broad spatial scales and large landscapes.

In the Flemish framework, Afdeling Bos en Groen (2000) explicitly mentioned that all management plans should include management activities that promote the abundance of certain key species.

2.4 Geographical information systems in forest management

Although geographical information systems (GIS) are capable of storing, visualising, analysing and retrieving spatial information, forest management techniques rarely use the GIS during the planning or modelling phase (Taylor, Walker, & Abel, 1999; Baker & Mladenoff, 1999; Baskent & Jordan, 1991). Brief surveys by Misund et al. (1995) have shown that there exist very few systems for computer aided planning or scheduling that have capabilities to handle problems with a spatial component and that geographical information systems offer very primitive facilities for planning and scheduling.

Forest planning has traditionally assumed that spatial complexities of planning could be worked out during plan implementation or in the detailed operational planning. It is becoming clear, however, that leaving out habitat, visual, and other spatial outcomes to be worked out during the implementation can lead

to misleading results and to plans that are impossible to implement successfully (Johnson 1992 in Davis et al. (2001)). When wildlife habitat is abundant, it is easy to work out the spatial problems, but when wildlife habitat is scarce, ignoring it in the planning stage will result in projecting activity levels and outcomes that cannot be attained. Especially in the framework of the Flemish Community this is important: the size of the Flemish forests is very limited and the forests are very fragmented (Afdeling Bos en Groen, 2000, 2001) and therefore one cannot leave out the spatial outcome until the actual implementation of the plan.

GIS promises to be a major tool for management planning within administrative units and among ownerships because of the importance of the spatial scale (Franklin, 1994). Simultaneously recognising spatial scales such as patches, stands and groups of stands within watersheds or forests, together with the quantification of projection of structural characteristics, wildlife habitat and other ecosystem elements and the portrayal of spatial patterns and relationships of stands, streams and forests form three key elements of management for sustainable human-forest ecosystems (Davis et al., 2001).

Spatial decision support systems (SDSS) are tools that help the decision maker to make well-founded decisions based on the integration of spatial and non-spatial data. In general, an SDSS consists of different modules (Densham, 1991): (1) the information data base, (2) the model toolbox which includes the GIS-functionality and (3) the optimisation toolbox. A front end to the SDSS is the graphical user interface (GUI).

The use of SDSS in forest management has not been very extensively. A short review is given in the following paragraphs. Baskent and Jordan (1991) designed a spatial wood supply model based on numerical and geographical information. The geographic stand information includes harvest block configuration and stand adjacency tables. Their tool is however not designed to optimise a harvest strategy over time. It is only developed to queue harvest blocks as to not violate adjacency constraints. They conclude that spatial modelling has potential to improve management design. The reasons for this are: firstly, a spatial model determines a better assessment of wood supply as it is determined as a function of both geographic and numerical data. Secondly, the proposed strategies are much easier to implement in the field. A third reason is that harvesting as it occurs in actual practice is mimicked and finally, with the production of new spatially-oriented performance indices, a spatial wood supply model has potential to improve understanding of forest dynamics under management with both wildlife and economic objectives under consideration. Chuvieco (1993) combined linear programming and GIS for land-use modelling. He used GIS first as a tool to derive the coefficients of the objective function and after the optimisation to map the optimal solution, but GIS was not used directly during the optimisation process. Olson and Orr (1999) and Naesset (1997a) applied a similar approach: maps were generated in a GIS, and these maps were the input for linear programming. During the optimisation process there was no link to the GIS and only afterwards their optimal solutions were

visualised.

Naesset (1997b) reviews the use of GIS within decision support systems and says that GIS is a crucial technology for linking restrictions in timber management practices due to preservation of biodiversity, to practical forest management. Finally, Kurttila (2001) reviewed how the spatial structure in the forests were included in the optimisation calculations of forest planning and says that there is an increasing need to analyse the development of the spatial structure of forests and to develop the means by which spatial objectives can be explicitly included. He states that the best way to handle spatial objectives is to include them directly in the optimisation process but that this is too complicated.

In the previous paragraphs the need for an integrated approach between an optimiser and a geographical information system clearly comes forward. As there are still very little approaches that combine these two models, one of the aims of this dissertation is to develop an SDSS where the spatial data can be integrated during the optimisation process.

2.5 Multi-objective optimisation techniques in forest management

2.5.1 Basic concepts and terminology

The objective function of ecosystem management differs from previous management efforts in its complete enumeration of desired products. Ecosystem management does not preclude market outputs such as timber production from being one of the objective functions but ecosystem management does not allow it to be the only objective (Mendelsohn, 1996).

Forest managers often invoke the goal of multiple use, but, as yet, no criterion to measure the degree of multiple use has been agreed upon. It is hard to say whether one management plan contains more multiple use than another, and by how much (Davis et al., 2001). The forest management problem thus can be stated as a multiple objective optimisation problem (MOOP) where the multiple objectives have to be maximised or minimised and where a number of constraints have to be met in order to obtain feasible solutions. Korhonen et al. (1992) distinguishes between discrete and continuous multiple objective forest optimisation problems: if the forest management problem is formulated as a Model I type problem, the problem can be classified into the discrete category; a Model II formulation refers to continuous problems.

Any MOOP where it is the aim to maximise the objective functions can be written as in Eq. 2.7 (Deb, 2001):

Maximise

$$f_m(\mathbf{x}) \quad m = 1, 2, \ldots, M \tag{2.7}$$
$$\tag{2.8}$$

subject to

$$g_j(\mathbf{x}) \geq 0 \qquad j = 1, 2, \ldots, J \tag{2.9}$$
$$h_k(\mathbf{x}) = 0 \qquad k = 1, 2, \ldots, K \tag{2.10}$$
$$x_i^{(L)} \leq x_i \leq x_i^{(U)} \qquad i = 1, 2, \ldots, N \tag{2.11}$$

A solution vector to the optimisation problem, $\mathbf{x}$, is a vector of n decision variables: $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$. These decision variables are constrained by the variable bounds, restricting each decision variable x_i to take a value in between a lower $x_i^{(L)}$ and an upper $x_i^{(L)}$ bound (Eq. 2.11). These bounds determine the decision variable space $\mathcal{D}$ or in short the decision space. The above problem is characterised by K equality constraints (Eq. 2.9) and J inequality constraints (Eq. 2.10). Any solution meeting the constraints and variable bounds is called a feasible solution. Within the decision space, it is possible to determine the complete set $\mathcal{S}$ of feasible solutions for which it holds that $\mathcal{S} \subseteq \mathcal{D}$.

The main difference between single and multi-objective optimisation problems is that for multi-objective optimisation the objective functions constitute a multi-dimensional space, in addition to the usual decision variable space. This means that with multi-objective optimisation problems, an n-dimensional decision space is mapped onto an m-dimensional solution space. Fig. 2.5 illustrates the mapping process between these two spaces.

As discussed in the previous sections the indication of preferences is very difficult in the domain of forestry, even though this approach has been applied many times. Afdeling Bos en Groen (2000) mentions that many a researcher has tried to derive a total economic value but the results of this are still preliminary. If this approach is abandoned, it is necessary to define a different way to judge if one solution is better (or more preferable) than another without this prior indication of preferences. In the case of multiple objectives, it is only possible to state that one chromosome is better than another when it has at least the same objective function scores for all objectives and a better score for at least one objective.

Any two vectors u and v can be related in one of the following ways:

$$u = v \Leftrightarrow (\forall i \in \{1, 2, \ldots, k\})(u_i = v_i) \tag{2.12}$$
$$u \geq v \Leftrightarrow (\forall i \in \{1, 2, \ldots, k\})(u_i \geq v_i) \tag{2.13}$$
$$u > v \Leftrightarrow u \geq v \wedge u \neq v \tag{2.14}$$

Two decision variable vectors a and b with corresponding objective value vectors $f(a)$ and $f(b)$ can have one of the following relationships for a multiple objective

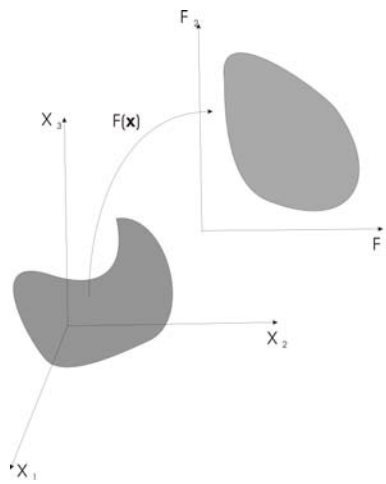


Figure 2.5: Representation of the decision space and the corresponding objectives or solutions space

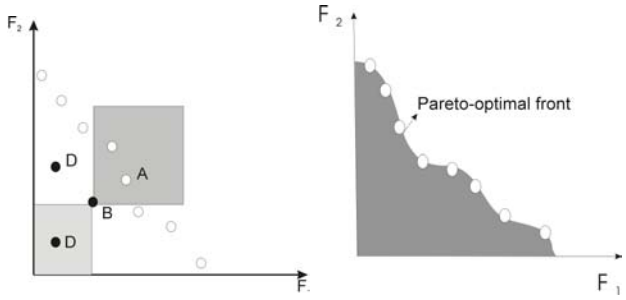
problem:

$$a \succ b \quad (a \text{ dominates } b) \quad \Leftrightarrow f(a) > f(b) \tag{2.15}$$

$$a \succeq b \quad (a \text{ weakly dominates } b) \quad \Leftrightarrow f(a) \geq f(b) \tag{2.16}$$

$$a \parallel b \quad (a \text{ incomparable to } b) \quad \Leftrightarrow f(a) \not\succ f(b) \wedge f(b) \not\succ f(a) \tag{2.17}$$

Using the concept of Pareto-dominance (Eq. 2.15), it is possible to arrange all solutions in a partial order. In Fig. 2.6(a), the light grey region indicates the solutions that are dominated by solution B and the dark grey region represents the set of solutions that dominate B.



(a) Dominance relationships (b) Pareto-optimal front

Figure 2.6: Fig. 2.6(a) For a solution B, the region indicated in light grey denotes the set of solutions that are dominated by B, the region in dark grey is the set of those solutions that are dominating B. The solutions marked as ∘ (such as A) are non-dominated solutions; the solutions marked as • are dominated solutions. In Fig. 2.6(b) the Pareto-optimal front for a maximisation problem is depicted. For any solution on this front there does not exist another solution that can improve the value of one objective without lowering the value of another objective.

A feasible decision variable vector a is said to be a non-dominated element of a reference set A if there exists no other vector that dominates a. In Fig. 2.6(a) A is a Pareto-optimal solution because there is no other solution that can improve one of the objective values without lowering another objective value. The corresponding objective value vectors of the non-dominated solutions form the Pareto-front of the reference set A (Fig. 2.6(a)). The set of all non-dominated solutions (i.e. when A is the set of all feasible solutions) is called the Pareto-optimal set. The corresponding objective value vectors are then called the Pareto-optimal front of the optimisation problem.

The aim of any optimiser is to approximate the Pareto-optimal front as well as possible and to find solutions that are evenly spaced along the Pareto-optimal front. The first goal is obligatory for any optimisation task: if the solutions obtained after the optimisation process do not converge on the optimal solution or do not approximate the Pareto-optimal front, the optimiser is not useful. The second goal is specific for multiple objective problems. If a Pareto-optimal front is to be obtained, it is necessary to find as many solutions as possible in every region of the Pareto-optimal front. Only in that way, a clear view of the explicit trade-offs between the objectives can be obtained.

2.5.2 Mathematical techniques

2.5.2.1 Linear and integer programming

Linear programming has been widely accepted as a general and good optimiser for forest management planning (Davis et al., 2001; Davis & Martell, 1993). It has been used extensively since the 1970s. FORPLAN was one of the first systems that was implemented by the United States Department of Agriculture and was entirely based on linear programming. A general single objective linear programming problem can be stated as in Eq. 2.18. In this formulation all m constraints are linear combinations of the n decision variables (Kolman & Beck, 1995) :

Maximise

$$z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$$

subject to

$$\begin{aligned} a_{11}x_1 + a_{12}x_1 + \cdots + a_{1n}x_n &\leq b_1 \\ a_{21}x_1 + a_{22}x_1 + \cdots + a_{2n}x_n &\leq b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_1 + \cdots + a_{mn}x_n &\leq b_m \\ x_j \geq 0 \qquad j = 1, 2, \ldots, n \end{aligned} \tag{2.18}$$

with x_i decision variable i, c_i the objective function coefficient associated with decision variable i, a_{ij} the coefficient of decision variable i in constraint j and b_j the upper bound for constraint j. As can be seen in Eq. 2.18, it is very easy to formulate the Model I or Model II formulation as a linear program. One limitation of linear programming is that it requires continuous variables and therefore it cannot be used for spatial objectives (Kurttila, 2001).

Integer programming or mixed-integer programming has been used to overcome the lack of spatial integrity that was present in the linear programming formulation. The problem definition remains the same as in Eq. 2.18 but extra constraints limiting the decision variables to integers are added to the formulation. The use of integer programs is therefore common when wildlife habitat maximisation is one of the objectives (Hof & Joyce, 1993; Hof, Bevers, Joyce, & Kent, 1994).

Both linear and integer programming approaches are essentially single objective optimisers. In order to deal with multiple objectives, it is necessary to recast these multiple objectives into a single objective problem. Three basic tools are employed to recast a multiple objective problem in forest management: the weighted sum method, the constraint method and a stage approach.

Weighted sum method The weighted sum method scalarises a set of objectives into a single objective one by multiplying each objective with a weight. This requires the *a priori* articulation of the preferences (Hwang & Masud, 1979). If there are M objectives, the objective function (Eq. 2.18) becomes Eq. 2.19:

Maximise

$$w_1 \times f_1(x) + w_2 \times f_2(x) + \cdots + w_M \times f_M(x)$$

subject to

$$\begin{aligned} a_{11}x_1 + a_{12}x_1 + \cdots + a_{1n}x_n &\leq b_1 \\ a_{21}x_1 + a_{22}x_1 + \cdots + a_{2n}x_n &\leq b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_1 + \cdots + a_{mn}x_n &\leq b_m \\ x_j \geq 0 \qquad j = 1, 2, \ldots, n \end{aligned} \tag{2.19}$$

In the particular case of forest management, two problems arise with this method. The main problem is the determination of the weights. They should reflect the relative importance of one objective over another. Choosing weights requires a consensus on the relative importance and this consensus can be hard to obtain. The weighting method also requires that all the objectives are commensurable. This is certainly not the case in forest management e.g. timber volume and recreational benefit are not commensurable.

Two general characteristics of this method are (1) the setting of appropriate weights is scale dependent and therefore all weights should be normalised before assignment and (2) when the Pareto-optimal front is non-convex, it is impossible to attain it using the weighting method (Mietinnen 1999 in Deb (2001)).

Constraint method The constraint method requires as much user interaction as the weighted sum method. First of all, it needs some prior preference articulation because one objective from the set of objectives has to be selected as the main objective. The other objectives are then reformulated as extra constraints. These other objectives acquire different boundary values and these target values are determined by the user. The main advantage of this method in comparison with the weighted sum method is that it can handle both convex and non-convex problems. A disadvantage is that the solution of this optimisation problem is very sensitive to the target vector stated. If these bounds are set too high (in the case of a maximisation problem) it is impossible to find a feasible solution to the problem. Especially in large problems with many constraints finding the constraints that caused infeasibility is difficult (Buongiorno & Gilles, 1987).

A stage approach Another method is the stage approach. Here the problem is solved in multiple stages by setting bounds for each objective consecutively. Strange et al. (1999) for example used this approach to evaluate management alternatives. Again this method is very sensitive to the chosen initial bounds and to the order in which the objectives are added.

2.5.2.2 Goal programming

Goal programming was first introduced by Charnes (1955 in Deb (2001)) and gained real popularity after the work by Ignizio (1976 in Deb (2001)). The main idea in goal programming is that all of the management goals are expressed as goal constraints. The objective function is then to minimise the deviation from these goal constraints and becomes (Deb, 2001):

Goal

$$f(\mathbf{x}) = t \quad \mathbf{x} \in \mathcal{S} \tag{2.20}$$

and the set of goals can be written as

$$f(\mathbf{x}) - p + n = t \tag{2.21}$$

where p is the positive deviation from the target volume and n is the negative deviation from the target volume t. The advantage of goal programming is that the optimisation problem becomes less rigid. Because the deviations are minimised, feasibility is guaranteed. The main disadvantage however is that it is possible to obtain dominated solutions if the target values are set too low. Especially in the case of forest management inferior solutions or solutions that are not on the efficiency frontier are unacceptable (Davis et al., 2001).

2.6 Dealing with spatial data in the optimisation process

2.6.1 Integrating mathematical techniques and spatial data

It is difficult to provide spatial integrity using the linear programming method for the Model I or Model II formulation. This spatial integrity can be provided by turning a Model I formulation into an integer or a mixed-integer program with each stand receiving only one treatment (Bevers & Hof, 1999; Hof et al., 1994), but this leads to a very large integer formulation that is is usually unsolvable, even with the current computing power. Yoshimoto and Brodie (1994), for instance, noted that a spatial long-term planning problem with mixed-integer or integer programming is difficult to handle. The primary difficulty with this approach is that it requires the enumeration of all possible management regimes, a task that is combinatorial in nature (Sherali & Liu, 1990). Hof and Joyce (1992) report that the main problem for these non-linear formulations is not

the computational effort, but rather the numeric precision. As the non-linear models get larger, numeric imprecision causes the search algorithm to fail before solution time becomes a problem.

Many authors have tried to reduce the number of constraints that arise when implementing spatial features into the optimisation problems so that they could still implement integer or mixed-integer approaches. The main focus has been on the so-called adjacency constraint problem. In this problem, the forest harvesting activities have to be scheduled in such a way that two neighbouring stands are not cut within a certain temporal window. This ensures that no treated unit or collection of units will exceed a limit on the area cut usually imposed by the forest authorities (Murray, 1999) . In Flanders this is not specified in the forest law (Vanhaeren, 2002) but in the long term vision (Afdeling Bos en Groen, 2000) a maximum clear felled area of 1 ha is stated.

The simplest way to check for the adjacency constraints is by pairwise comparison. A list is made for every stand i and its neighbouring stands, the set A_i. The adjacency constraint is then defined as:

$$\sum_{j \in A_i} Y_j \leq 1 \quad \forall i \tag{2.22}$$

In Eq. 2.22, Y_j denotes that management activity j is assigned to stand i and consequently Eq. 2.22 ensures that only one stand can be cut in the group of adjacent stands of i, i.e. A_i. Equations that enforce this are referred to as Type 1 adjacency constraints (Jones et al., 1991). Type 1 constraints can only be used for groups of three or four mutually adjacent polygons (Jones et al., 1991).

If the set of mutually adjacent stands is larger, Type 2 adjacency constraints (Jones et al., 1991) can be used. These Type 2 constraints combine multiple sets of Type 1 constraints and reduce the number of adjacency constraints.

Most of the research to find methods to reduce the number of constraints have built on these two types of constraint definitions. Murray and Church (1995) investigated the effectiveness of the algorithms that reduce the number of constraints. They found that the traditional pairwise approach for imposing adjacency constraints, leads to a huge number of constraints but not necessarily to the tightest formulation. Methods that use the fewest number of constraints may produce poor structure and increase the computational difficulty of the problem.

A different approach to handle non-linearity is based on the four-colour theorem (Murray, 1999; Roise et al., 2000). This theorem says that any planar map can be coloured in such as way that adjacent polygons in the map have different colours, with just four colours. Roise et al. (2000) extended this to avoid adjacent cuttings with an exclusion period r and found that adjacency constraints can be met only for a planning horizon R if $R/r \geq 4$. This method, however, is only applicable for forests with less than 200 stands (Roise et al., 2000).

2.6.2 Heuristics

A totally different approach to integrate spatial objectives in the forest optimisation problem is based on the use of heuristics. Heuristics are based on 'rules of thumb' in order to speed up the optimisation problem. They can generate spatially and temporally feasible solutions to large problems that were previously unsolvable with mathematical techniques (Carlsson, 1999). In general heuristics start off with a randomly initialised solution and will modify this solution within a predefined neighbourhood around the former solution. They will investigate only part of the solution space and have different mechanics to escape from local maxima. A downside of heuristic algorithms often reported is that they do not guarantee optimality. The three heuristic techniques that have been used extensively in forest management are: Monte Carlo integer programming, tabu search and simulated annealing. Genetic algorithms are also heuristic procedures but have been not been used frequently.

2.6.2.1 Monte Carlo integer programming

A simple Monte Carlo search process is one where a solution is randomly generated to a mixed-integer programming problem. These solutions are tested for feasibility and for each feasible solution, the objective function is calculated. If the solution quality is better than for the previous solution, the new solution is kept as the best solution. This process is repeated for a number of times. This approach is in fact nothing but a steepest ascent (for a maximisation problem) hill climbing method. The main weakness of this approach is that it cannot escape from local optima. Clements et al. (1990) and Jamnick and Walters (1993) used Monte Carlo simulation to solve the adjacency constraint problem but did not compare it with classical techniques. Boston and Bettinger (1999) reported that Monte Carlo integer programming is limited to planning problems with a maximum of three planning periods.

2.6.2.2 Tabu search

Tabu search is a heuristic designed to escape from local optima: even if there is no better solution than the current solution x_n in its neighbourhood $V(X_n)$ then the new solution will be the best possible solution x in $V(x_n)$ or a sub-neighbourhood $V'(x_n)$ if $V(x_n)$ is too big to be explored efficiently (Pirlot, 1992). The main idea behind tabu search is simple. A memory forces the search to explore new areas of the search space. In this memory, solutions that have been examined are kept and these solutions become 'tabu' or forbidden during a number of steps (Michalewicz & Fogel, 2002). Prohibiting a given attribute is likely to be restrictive in excluding many more solutions than just the visited one. To correct this, it is possible to overrule the tabu status of a move when it leads to a solution that is good enough. When the memory should be overruled is defined in a so-called aspiration level describing what 'a good' solution is. Brumelle et al. (1998) used tabu search for finding good forest

harvest schedules satisfying the adjacency constraints. Their objective was to maximise the total volume cut, while maintaining even-flow and minimising the deviations of the adjacency constraints. If a stand i is cut at time t and the exclusion period between two successive felling activities is r then moves to include the neighbouring stands for scheduling between t and $t+r$ are put on the tabu list. If for a stand there are no non-tabu neighbours, then that solution is selected that violates the adjacency constraints the least. They compared their results to a random search and found that their solutions were superior. Boston and Bettinger (1999) found that tabu search is able to find solutions within 93.7% of the optimal values reached with integer programming. Moreover, it has a very small range in objective value between the solutions obtained in different repetitions, indicating the robustness of the method. It was also able to produce the smallest deviations from target harvest levels for even-flow objectives.

2.6.2.3 Simulated annealing

This method is based on an analogy taken from metallurgy (Michalewicz & Fogel, 2002). After metal has been heated to make it a fluid, it is cooled down again to get the proper shape, say a metal sheet. During this cooling process, the state of the molecules changes. If the temperature of the metal is lowered very quickly, the crystal formation of the molecules is suboptimal and the metal may have certain faults. If the cooling process is more gradually, the crystallisation is almost perfect but the cooling process requires more energy or time. This analogy can be used to solve optimisation problems. Starting off with a random solution, this solution can be changed from its current state into a new state. In the beginning of the 'cooling process' almost all moves from the current solution are allowed, but as the process continues, the probability to accept inferior solutions is lowered, so that the algorithm gets more and more selective. This is analogous with the Boltzmann distribution in thermodynamics (Pirlot, 1992). Simulated annealing is the heuristic that has been applied the most in forest management and it has been reported that this method obtained the best results in terms of convergence to the global optimum (Boston & Bettinger, 1999). It can be proven that simulated annealing always converges on the global optimum given enough time. Lockwood and Moore (1993) were among the first to apply simulated annealing to a harvest scheduling problem with adjacency constraints. They have tested simulated annealing on a test data set of 1648 and 27548 stands and could successfully find harvest schedules complying with the spatial constraints. These problems are far larger than could be handled by integer or mixed-integer programs. They used the basic inventory polygons as input without recombination or reclassification. As they applied simulated annealing to a real-world problem, the true optimum was unknown and likely unattainable. Therefore they argue that the use of simulated annealing is at best a tool for policy analysis. Tarp and Helles (1997) used a combination of simulated annealing and linear programming to find an optimal model. They solved the problem in four steps, from which they derived the name of the

algorithm SALP:

1. S Solve the LP model;
2. A Solve the adjacency model with SA;
3. L Compare LP and adjacency model optima;
4. P Make trade-off between incremental cost and damage cost and determine the final strategy.

They argue furthermore that the use of GIS could enhance the algorithm because then the consequences could be evaluated directly on a map. They conclude that the combination of SA and LP causes a significant improvement, partly because it contributes to reducing economic losses arising from damage caused by the adjacency problems and secondly that the non-linearity of the spatial objectives is overcome through the use of simulated annealing. Ohman and Eriksson (1998) used the concept of core area as a criterion for forming contiguous areas of old forests in landscapes and applied simulated annealing to do so. Their objective was in fact to find a maximum present value subject to core area for each planning period. Van Deusen (1999) finally used simulated annealing to generate multiple solutions by combining the different objectives into a weighted sum. He let the weights vary in order obtain the multiple solutions. He continued this work (Van Deusen, 2001) and combined the optimisation of the spatial arrangement of the stands and the harvest schedules.

2.6.2.4 Genetic algorithms

Genetic algorithms have been rarely used in forest management. Davis et al. (2001) mention only one paper by Pesonen et al. (1995). Lu and Erikkson (2000) used genetic algorithms to form harvest units, but they conclude that the results are still preliminary as they did not investigate the full options of genetic algorithms. Moore et al. (2000) used GA to find management strategies for wildlife objectives. They solved a single objective problem: maximise the bird abundance at the end of a fixed planning horizon. They found that the genetic algorithm frequently did not find the optimal value but the designs found by the GA provided objective values that were consistent with their intuition about how initial habitat stages would influence bird habitat over time. They felt that the sub-optimal results were due to insufficient investigation into the optimal settings of the genetic algorithm. Falçao and Borges (2001) also used evolutionary algorithms to solve integer forest management scheduling in Portugal. They reported that the best solution came within 0.2% of the estimated heuristic optimum. Moreover they reported a small range in the objective value between the repetitions suggesting robustness of the method. They indicated that future research should focus on the incorporation of additional spatial data such as adjacency, edge length and interior space, given that topological information is available. Booty et al. (2001) also implemented genetic algorithms in their

environmental decision support system (RAISON-DSS), reportedly because the genetic algorithm is better at avoiding local minima than linear programming. As genetic algorithms are the basis of this research they will be fully discussed in Chapter 4.

2.6.2.5 Discussion

Heuristics have been used extensively to overcome the main drawback of mathematical techniques, namely the limitations as to the problem size. Several authors have reported near-optimal solutions even though heuristics do not guarantee that optimal solutions will be attained. The main problem with the reported heuristic procedures is similar to that of mathematical techniques: they are essentially single objective optimisers. Even with modern heuristic methods it is necessary to reformulate the multiple objectives into a single objective and this reduces the potential of the above heuristics.

2.6.3 Methods applied for real-world applications

Roise et al. (2000) conducted a large informal survey in which private forest companies were asked what software they used for planning. They found that the maximisation of net present value was still the most important objective, but that some other companies used other and multiple objectives to coordinate more closely with the larger integrated company strategy. Most of the companies used linear programming in order to generate harvest plans. Very few companies applied a form of heuristic search in order to find schedules.

2.7 Summary and conclusion

In forest management literature, most models have focused on the purely economic function of the forest. It is required by many forest laws to include other functions as well, and especially the spatial objectives require a much more comprehensive approach. The mathematical techniques that were developed for the economic functions, are unable to handle these spatial objectives. Linear programming has continuous variables, and these are not suitable to ensure spatial integrity. Integer and mixed-integer programming overcome this problem but the spatial requirements lead to very large integer formulations which cannot be solved even with today's computing power. Heuristics have been proposed as a means to handle these complicated optimisation problems, and are indeed capable of solving them. They suffer, just as the mathematical techniques, of the fact that they are essentially single objective optimisers. Until now, truly multi-objective optimisers have not been used in forest management.
The integration of the optimisers together with a geographical information system has been advocated for a long time as to integrate spatial data and models in the optimisation process. This combination only exists when GIS is used for classification purposes prior to the optimisation process or afterwards for visualising the proposed alternatives. The online combination of GIS and optimisers in forest management has not been reported.

Chapter 3

Study area

In the subsequent chapters, theory will alternate with applications. All the applications and models were developed for the same study area: Kirkhill Forest in Aberdeen, Scotland. Originally the aim was to develop a tool for generating alternative forest plans for a Flemish forest. At the start of this research no Flemish data was readily available, because the field work of the First Flemish Forest Inventory (Afdeling Bos en Groen, 2001) was just finished and the data was not yet ready for use. Instead a study site in Scotland was selected. Aberdeen University uses this forest as a research forest and the students conduct an inventory every year for their master's project. This rough data set includes not only field data but also yield class (Cameron, 2000). This can be used as input to the Forestry Commission production forecast models (Hamilton & Christie, 1971). Next to inventory data, the data set included also spatial information such as contour lines. Kirkhill Forest is subject to comparable uses as in Flemish forests, i.e. timber production, recreation and wildlife conservation. It also has the same size as a the larger Flemish forests (such as Heverleebos) and therefore seemed to provide an good alternative. Finally, as I studied at Aberdeen University during a Socrates exchange programme (1998-1999), I was familiar with Kirkhill forest.

General information Kirkhill Forest is situated in Scotland, U.K. It lies between 2 °16' and 2 °14' W longitude and between 54 °11' and 58 °13' 30" latitude (Fig. 3.1). The forest is located 12 km north west of the city of Aberdeen and is dissected by the A96 Aberdeen-Inverness dual carriageway. It was previously managed by the British Forestry Commission as a productive coniferous forest but has changed ownership at the beginning of the 1990s. The recreational use of Kirkhill Forest is very high given its proximity to Aberdeen. Furthermore, it has a high landscape impact as it is predominantly surrounded by agricultural land.

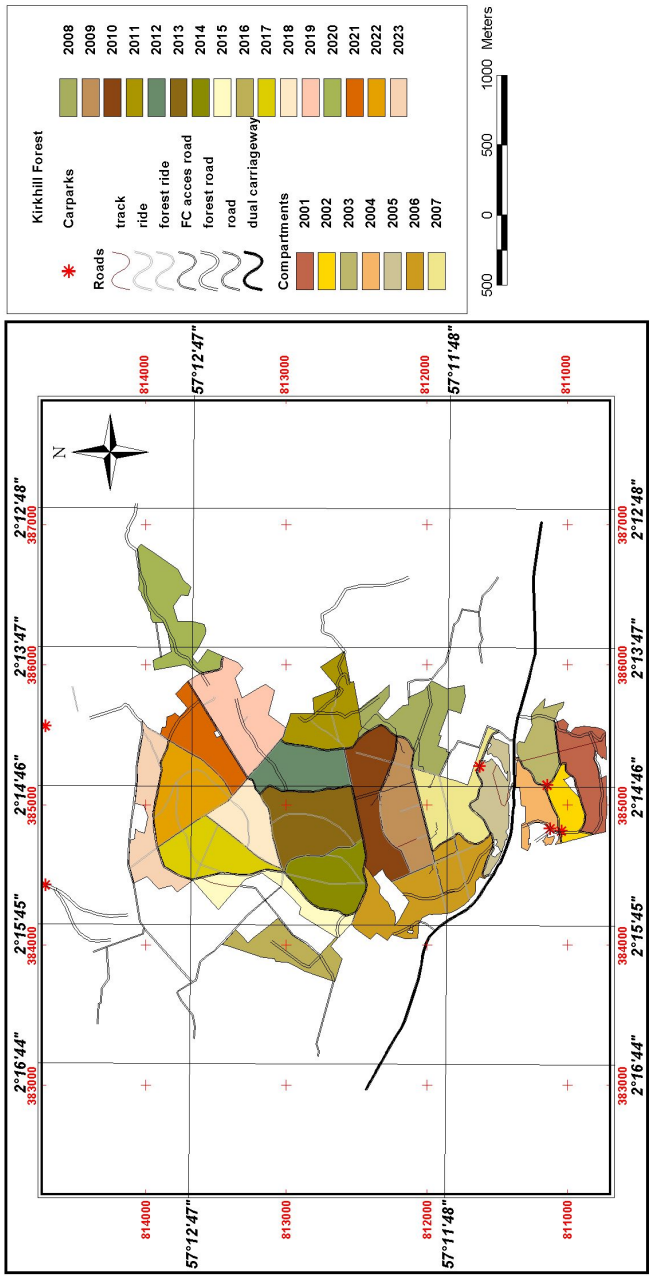


Figure 3.1: Kirkhill forest, near Aberdeen in Scotland (Cameron, 2000)

Topography and slope Kirkhill forest has a height between 80 and 250 m above sea level (Fig. 3.2). The highest peak is in the northern part of the forest. Mostly the terrain is gently undulating, with low to moderate slopes. Only near the peak of the forest, some higher slopes are present. The topography was derived from the contour lines provided by Cameron (2000).

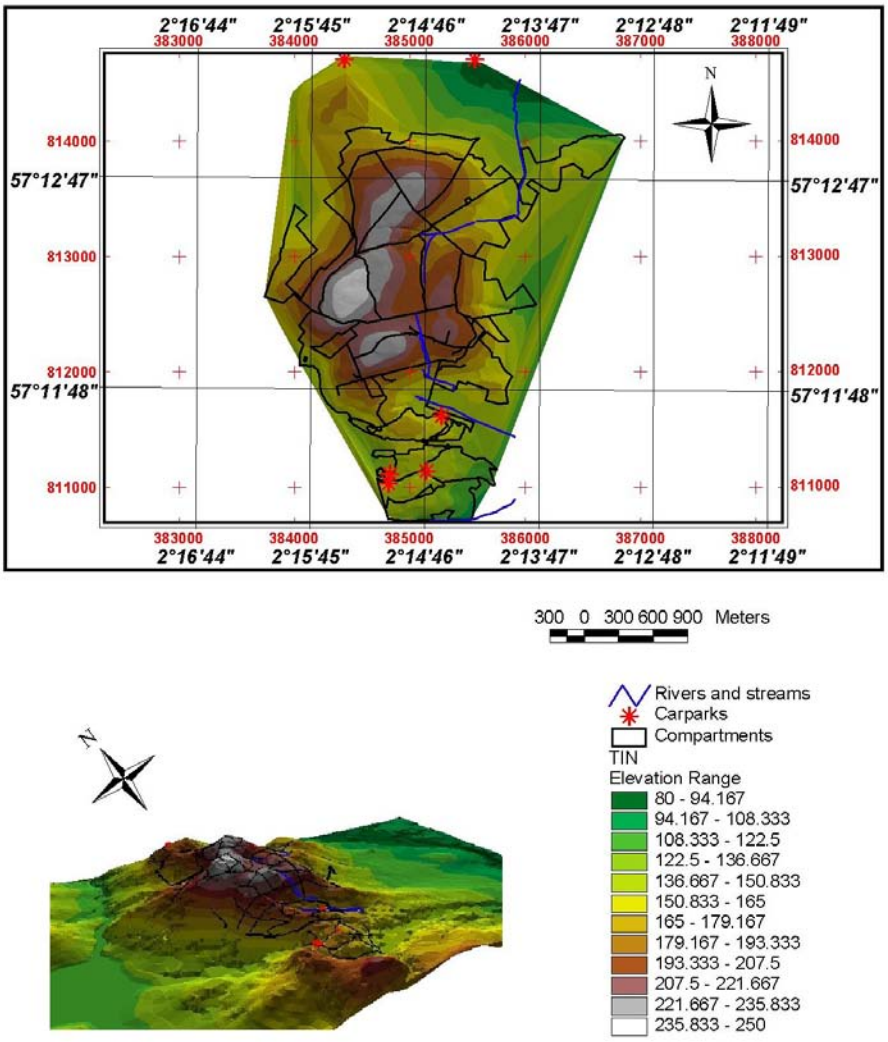


Figure 3.2: Topography of Kirkhill forest

Species distribution Kirkhill forest has an area of 454 ha and 395 stands. The predominant species in Kirkhill forest are Sitka spruce (*Picea sitchensis*), Scots pine (*Pinus sylvestris*), Larch (*Larix* spp.), Norway spruce (*Picea abies*) and Douglas fir (*Pseudotsuga menziesii*). Other conifers found in Kirkhill are Grand fir (*Abies grandis*), Noble fir (*Abies nobilis*) and Lodgepole pine (*Pinus contorta*). There is a limited presence of broadleaves such as birch (*Betula* spp.), European beech (*Fagus sylvatica*) and European oak (*Quercus* spp.). These form occasionally mixed broad-leaved stands. Currently, 50 ha of the forest is unplanted (Figs. 3.3 and 3.4).

Inventory data The area of the 299 planted stands amounts to 400 ha. The total area and mean values of basal area, volume per ha and density are summarised by species in Fig. 3.1.The site productivity varies considerably between the species and the stands. The yield classes range between low values for European larch (yield class 4 to 6) up to high to very high values for Sitka spruce (yield class 10 to 22). The age distribution of Kirkhill Forest is highly unbalanced. During the 1950s the Forestry Commission restocked many stands, and afterwards these stands were no longer managed. Therefore, there is a peak in the age distribution of the forest: up to 160 stands are in the age category between 50 and 60 years (Fig. 3.5).

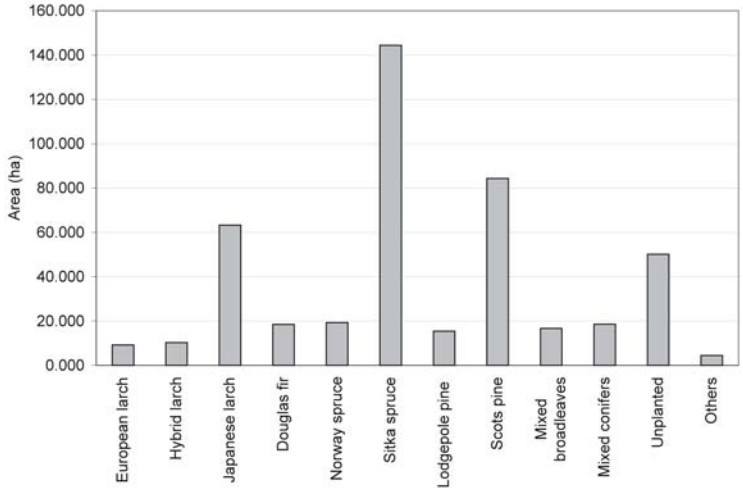


Figure 3.4: Species distribution in Kirkhill forest (Cameron, 2000)

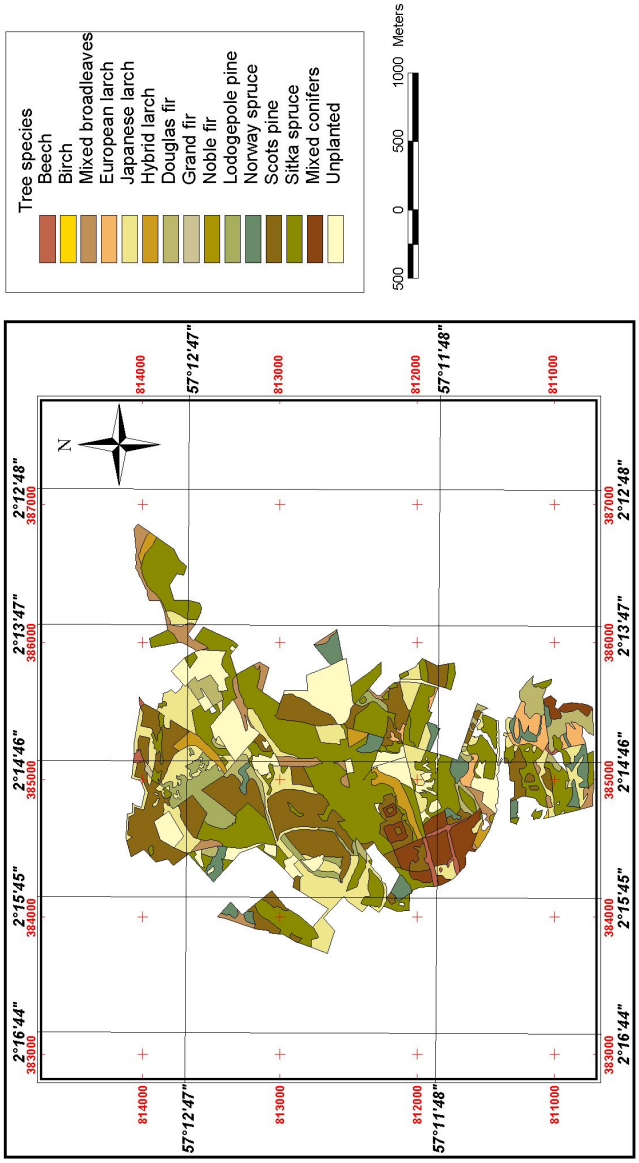


Figure 3.3: Species map of Kirkhill forest based on the inventory by Cameron (2000)

Table 3.1: Area covered by species (ha), density (trees/ha), mean basal area (m^2/ha), and mean volume (m^3/ha) in Kirkhill forest (Cameron, 2000)

Species	Area (ha)	Density (trees/ha)	Mean basal area (m^2/ha)	Mean volume (m^3/ha)
European larch	9·162	214·887	19·456	186·601
Hybrid larch	10·274	1327·000	38·167	280·457
Japanese larch	63·248	561·566	27·798	276·631
Douglas fir	18·500	745·981	53·183	574·642
Norway spruce	19·318	528·025	26·145	273·827
Sitka spruce	144·382	751·721	40·614	424·156
Lodgepole pine	15·392	958·332	21·788	139·836
Scots pine	84·343	618·108	30·117	243·654
Mixed broadleaves	16·662	–	–	–
Mixed conifers	18·596	942·875	32·083	285·376
Unplanted	50·163	–	–	–
Others	4·445	–	–	–

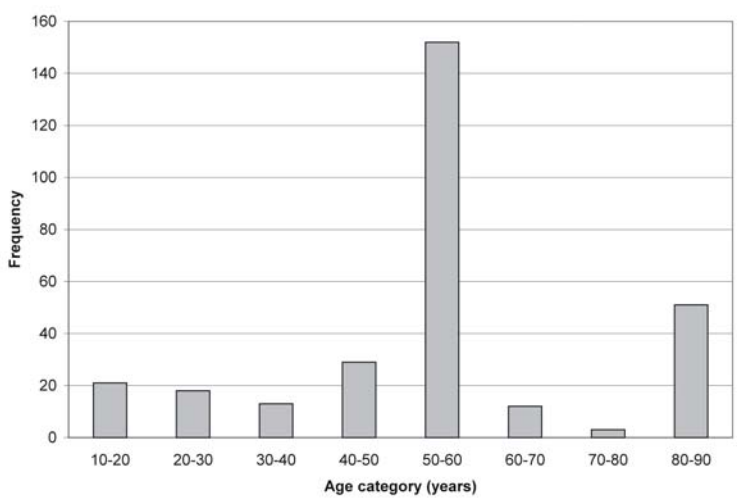


Figure 3.5: Age distribution of Kirkhill forest (Cameron, 2000). Note the high number of stands within the age category of 50–60 years

Recreation Kirkhill forest is much used for recreational purposes due to its proximity near Aberdeen. Most visitors are hikers and mountain bikers. Both hiking and cycling trails are available. The Tyrebagger sculpture trail leads hikers along several sculptures in the forest. Horse riders can also enjoy in Kirkhill forest and ride along one of the horse trails within the forest. Several car parks are provided within the forest boundaries, and the Tyrebagger trail starts at one of those points.

Wildlife Although there are few rare species of flora and fauna inside the forest, the species found are considered to constitute an integral part of the forest structure, and wildlife potential enhancement is therefore one of the objectives of the owner. Bird species include crow (*Corvus spp.*), buzzard (*Buteo buteo*) and woodcock (*Scolopax rusticola*). Red and grey squirrel (*Sciurius vulgaris* and *S. carolinensis*) are both present in Kirkhill as are roe deer (*Capreolus capreolus*) and badger (*Meles meles*). Two key species, woodcock and badger, will be used in the case study. Woodcock is a bird species typically related to old growth forest. Badger, a protected species, roams at the forest edge. They can survive in forests with a high diversity in forest structure, but as Kirkhill forest was previously used as production conifer forest, badgers have built their setts near a forest edge so that they can forage in the agricultural fields surrounding the forest while still having forest cover for protection.

In all case studies, woodcock will be considered as a species typically linked to old growth forest, whereas badgers will be considered as edge-dependent species.

Part II

Simple genetic algorithms

Chapter 4

Simple evolutionary algorithms for single objective problems

4.1 History of evolutionary algorithms

During the last thirty years there has been a growing interest in the use of the principles of heredity and evolution as a metaphor for a generic optimisation tool in areas as diverse as engineering (civil, mechanical, electrical, control, $\cdots$), industry (scheduling, planning and management) and to some lesser extent in geography, chemistry, physics, medicine, ecology and computer science and engineering (Coello et al., 2002).

The general class of optimisers based on this concept is referred to as evolutionary algorithms (Michalewicz, 1999) and encompasses distinct groups of algorithms. The three main groups are (Jacob, 1998; Michalewicz, 1999): genetic algorithms (Mitchell, 1996; Goldberg, 1989; Holland, 1975), genetic programming (Koza, 1993, 1994) and evolution strategies (Rechenberg, 1994 in Jacob (1998)). All evolutionary algorithms are search and optimisation techniques based on the principles of natural evolution and selection. The inventors of evolutionary algorithms were amazed at the seemingly very complex structures such as humans that can be created from an apparently simple coding strategy in chromosomes.

The evolution of simple towards more complex systems has occurred due to selective pressure from the environment. Species that are more successful at avoiding death have the opportunity to produce more offspring than those that die young. This offspring inherits some of the beneficial traits from the parents, allowing it to survive even better under the environmental conditions. On average, the survival fitness of a generation increases due to this selective pressure. Furthermore, because of the recombination of the genetic material of the parents, the offspring develops new traits that were not present in one of

the parents. Mutation once in a while throws in a wild card. This wild card is sometimes bad causing early death, but occasionally it creates a somewhat different trait that allows an individual to be even more successful than would have been the case after simple recombination of the parents' genetic material. In fact mutation increases the diversity in a population. As the environment (the predators for example) is dynamic, there is a constant struggle of all species to stay at the edge of the current 'genetic' technology. If that species does not invent something new, it will get extinct some time or another.

Imitating this strategy, evolutionary algorithms were designed to solve optimisation problems. It was hypothesised that a general optimiser could be developed based on the principles of selection, recombination and mutation. A number of alternatives is proposed (either generated randomly or seeded with previously known solutions) and these alternatives are evaluated for their performance for the problem at hand. The solutions that are performing better are selected as 'promising' and used to make new alternatives. Through the recombination and mutation of bits and pieces of the promising alternatives, it is expected that the best alternative is reached. The main steps in the evolutionary algorithm are: initialisation, evaluation, selection and variation. These steps are repeated until a stopping criterion is reached (Alg. 1).

Algorithm 1: The overall pseudocode for evolutionary algorithms

```
t ← 0;
initialise population P(t);
initial evaluation P(t);
while stopping criteria not met do
    t ← t + 1;
    select P(t) from P(t − 1);
    change P(t);
    evaluate P(t);
end
```

The three variants of evolutionary algorithms differ in (1) the data structure that they apply, and (2) the values on which several parameters of the algorithm are fixed on during the optimisation process. Evolution strategies are based on a variable-length vector of real values, genetic algorithms use a fixed-length vector of a discrete alphabet with low cardinality and genetic programs employ tree structures with a variable length.

For land-use planning (Matthews et al., 1999, 2000), the automatic formation of harvest units based on tree attributes (Lu & Erikkson, 2000) and urban planning (Feng & Lin, 1999) genetic algorithms were applied, because in those cases the number of management units is known beforehand and the number of treatments per unit is discrete. A fixed-length string is a logical data structure for this type of optimisation problems.

In Section 2.3.1.3 was already mentioned that a Model I approach is the best model for forest management problems that include spatial data. When this Model I approach is used, the forest stands have a one decision variable for the entire duration of the planning horizon. This ensures a fixed number of management units and thus a fixed number of decision variables. As this number is known beforehand and it is fixed, a fixed-length string is used in this dissertation and therefore genetic algorithms will be the focal point

4.2 Fundamentals of genetic algorithms

4.2.1 Terminology

The vocabulary of genetic algorithms is heavily based upon biological terminology. The basic data structure of a genetic algorithm is a fixed-length binary encoded string and is referred to as a *chromosome*. This binary encoded string, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ with n variables or *genes*, is mapped from its *genotype* into its *phenotype* through the mapping function f. Each gene can have several values or *alleles*. In the case of a single objective problem, the *expression* of the genotype will result in one *trait* or objective value. This is mathematically written as:

$$\mathbb{R}^n \rightarrow \mathbb{R} : \mathbf{x} \rightarrow g(\mathbf{x}) \tag{4.1}$$

For a multiple objective problem this results in a vector of objective values.

A set of individuals is called a *population*. After the creation of the initial population, a binary *crossover* operator and a unary *mutation* operator are applied with a certain probability after a biased selection of individuals that are seemingly better than others. These selected individuals are put into the *mating pool* before *recombination*. This process repeats itself until a predefined stopping criterion is met. Each cycle is referred to as a *generation*. When the best individuals of the previous generation are used in the next one, *elitism* is applied.

4.2.2 Theoretical foundation

A genetic algorithm processes short promising pieces of different chromosomes. They are promising because they increase the fitness of an individual more than other pieces do. These short pieces can be represented by templates, so-called *schemata*, over the chromosomes in the population. The *order* of a schema S is the number of fixed genes in that schema. The *defining length* of a schema S is the difference in position between the two utmost fixed positions in the string. The promising schemata receive more trials of being in the next generation than the schemata that are less promising. The assignment of the number of trials is exponential and this is under a noisy fitness environment the best strategy (Goldberg, 1989).

A genetic algorithm combines instances of these templates through recombination and the final sum of all pieces adds up to the global optimum. Since the global optimal can be build up using these smaller parts, these pieces are referred to as *building blocks*.

The fundamental theorem of genetic algorithms can be stated as the following theorem (Theorem 1) (Holland, 1975; Goldberg, 1989; Michalewicz, 1999):

Theorem 1 (Fundamental Theorem of GA) *Short, low-order, above-average schemata receive exponentially increasingly trials in subsequent generations of a genetic algorithm.*

and the theoretical foundation of genetic algorithms is mostly based on the following basic hypothesis of genetic algorithm (Hypothesis 1) (Holland, 1975; Goldberg, 1989; Michalewicz, 1999).

Hypothesis 1 (Building block hypothesis) *A genetic algorithm seeks near-optimal performance through the juxtaposition of short, low-order, high-performance schemata called the building blocks.*

A lot of genetic algorithm theorists have tried to prove this building block hypothesis, but for problems other than laboratory problems, most evidence of this hypothesis has been purely empirical (Michalewicz, 1999).

The building block hypothesis states that only short and low order schemata are processed and this has a direct consequence on the proper representation of an optimisation problem: the encoding strategy that is used has to comply with the basic assumption of genetic algorithm. Indeed, the encoding strategy might even be critical for the success or failure of the optimisation process.

4.3 Setting up a simple genetic algorithm

As discussed before, recombination and mutation will explore the search space, whereas selection will exploit the knowledge learned from the individuals. The balance between exploration and exploitation is critical in order to achieve reasonable behaviour for the genetic algorithms (Bäck, 1994; Grefenstette, 1997a; Michalewicz, 1999). In the following sections a brief review of the possible operators that are available for a genetic algorithm is presented.

4.3.1 Representation issues

Before a genetic algorithm can be run, the proper data structure has to be chosen. Davis (1991) and Michalewicz (1999) stress that a good representation is critical for the success or failure of the genetic algorithm. The success of a data structure can be evaluated by the best objective value obtained after running the genetic algorithm with the different data structures.

Matthews et al. (2000) investigated the effect of two different representations for a land use planning problem. At first, they applied a fixed-length string

referred to as the land block representation: each gene represents a land block and the allele value is the land use type that will be allocated to that land block. The second representation encodes target land use as percentages. The priority for allocating these land uses is determined by the order in which they appear in the genotype. This representation is called the percentage and priority representation. When this last operator is translated into an actual allocation, the land blocks are allocated 'greedily' starting from those that have the best performance per unit. This allocation continues until either the target land use percentage is exceeded or no land blocks remain to be allocated.

They found that there was no significant difference in solution quality for the two representations. Hence for forest management problems the data structure is simple. As the management units are fixed for the entire duration of the planning period using a Model I approach, a fixed-length chromosome is suitable.

Next the question of the most suitable encoding must be addressed. Depending on which optimisation problem is solved, there is a choice between binary encoding, gray encoding or integer encoding. Binary encoding is the most commonly known representation of integers on a base 2. This representation has the disadvantage however that two consecutive integers might differ in more than one bit place. The number of different bit positions between two consecutive integers in the binary representation is referred to as the Hamming distance. As the Hamming distance is larger than one on some occasions, discrepancies in the search space may be present because two genotypes that differ in only one bit place might be decoded into two phenotypes with two integer values that are not consecutive (Goldberg, 1989). Gray codes are an adapted form of binary encoding and ensure that the Hamming distance between two consecutive integers is always one. According to Goldberg (1989) the use of gray coding results in a smoother optimisation process. Michalewicz (1999) (p. 98) presents some conversion tools from binary to gray encodings. Davis (1991) and Michalewicz (1999) advocate however that the representation should be as close as possible to the problem domain and therefore integer encodings are a logical encoding scheme as well. As an illustration the different encoding strategies for integers in the interval $[0, 7]$ are given in Table 4.1 .

4.3.2 The genetic operators

4.3.2.1 Selection operators

As seen in the basic evolutionary algorithm (Alg. 1) the first step in a genetic algorithm is the selection procedure. Three main selection schemes are (1) proportional selection, (2) selection strategies based on ranking procedures and (3) tournament selection.

Proportional selection Fitness proportionate selection was originally proposed by Holland (1975) and has been widely used in evolutionary computation (Grefenstette, 1997a). Before the selection phase, a probability distribution is

Table 4.1: Different encoding strategies: binary, gray and integer encoding for integer values in the interval $[0, 7]$. Note that for the gray encoding the Hamming distance is always one

binary	gray	integer
000	000	0
001	001	1
010	011	2
011	010	3
100	110	4
101	111	5
110	101	6
111	100	7

created such that the probability of selecting an individual for reproduction is proportional to the individual's fitness. The sampling procedure is referred to as the *roulette wheel sampling algorithm*, because one can think of the probability distribution as defining a roulette wheel on which each slice has a width corresponding to the individual's selection probability, and the sampling can be imagined as spinning the roulette wheel and testing which slice ends up top (Fig. 4.1).

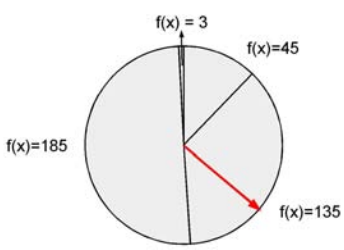


Figure 4.1: Roulette wheel sampling

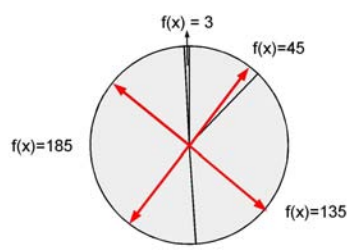


Figure 4.2: Stochastic universal sampling

The advantage of the fitness proportionate selection is that it is easy to understand. Given the building block hypothesis and the schema theorem, fitness proportionate selection is a natural selection strategy. A first drawback of the roulette wheel sampling scheme is that the probability distribution is sampled N times because the complete population is replaced. If the roulette wheel is called N independent times, this may result in a high variance in

the number of offspring assigned to each individual. Baker (1987) developed an algorithm called *stochastic universal sampling* to reduce the variance. Stochastic universal sampling also divides the roulette wheel into slices, but N equal spaced pointers are used to determine which individuals will get selected (Fig. 4.2). The roulette wheel is then spun to determine the selected individuals. Stochastic universal sampling is more efficient than roulette wheel sampling because it takes only a single pass to assign all the individuals.

The main problem associated with proportional selection is that it is scale sensitive. In the beginning of the genetic algorithm, some individuals will have a fitness that is a lot larger than the fitness of the second best individual. This individual will take up a very large part of the roulette wheel, and will get selected many times. The other individuals do not really stand a chance to be represented in the mating pool. This can cause premature convergence towards a local suboptimum. In the last generations, all individuals are more or less equal in fitness. Each individual will receive an equal portion of the pie, and the probability to become selected will be uniform. The search capacity of the genetic algorithm will reduce and the search behaviour will become a random walk. This can be overcome to some extent by using fitness scaling (Goldberg, 1989). Fitness scaling rescales the fitness values between the maximum and minimum value, and modifies the probability distribution function. The disadvantage of this is that the probability distribution function is sensitive to the scaling parameters and that choosing different scaling parameters influences the selective behaviour of the operator even though the fitness of the solutions is the same.

Rank-based selection As discussed in the previous section, the proportional selection schemes are easily influenced by so-called *super-individuals* at the beginning of the search process and are characterised by a weak search capacity at the end of the optimisation process unless fitness scaling is applied. In order to prevent these side-effects, rank-based selection procedures have been proposed. Ranking simplifies the mapping from the objective function to the fitness function (Grefenstette, 1997b) and also eliminates the fitness scaling procedures, since selection pressure is maintained even if the objective function values within the population converge on a very narrow range. Grefenstette (1997b) states that ranking may be a natural choice for problems in which it is difficult to state an objective function, for example if the objective function involves some subjective preference for alternative solutions. In that case, the exact value of the objective function is not so important. Additionally, Michalewicz (1999) states that rank-based selection operators control the selective pressure better than proportional schemes, and focus the search process better. On the other hand, these approaches have some apparent drawbacks (Michalewicz, 1999). Firstly, it puts the responsibility to the user when to use these mechanisms. Secondly, rank-based procedures ignore the information about the relative evaluations of different chromosomes and thirdly, all cases are treated equally regardless of the magnitude of the problem. Finally, selection procedures based on ranking

violate the Schema Theorem (Michalewicz, 1999).

Tournament selection In tournament selection a group of q individuals is chosen randomly from the population either with or without replacement. This group of individuals takes part in a tournament, this means that the fitness values are compared between the q individuals and the best individual is selected. This individual wins the tournament. Often tournaments are held between two individuals, but this can be generalised to any arbitrary group size. The process is repeated N times and this will lead to a high variance as was the case with proportional selection. In comparison with the previous techniques, tournament selection can be implemented efficiently and has a time complexity of $\mathcal{O}(\mathcal{N})$ because no sorting of the population is required (Blickle, 1997). Furthermore, tournament selection is invariant under rescaling.

4.3.2.2 Crossover operators

During the recombination phase, crossover is performed with a certain (fixed or variable) crossover probability. For many genetic algorithm theorists and practitioners crossover is the most important operator. These theorists and practitioners believe that if crossover is deleted from the algorithm the result is no longer a genetic algorithm because crossover distinguishes the optimisation process of genetic algorithms from classical optimisers (Davis, 1991). The main problem associated with the recombination operators is that the purely syntactic operation on the chosen alphabet must produce semantically valid fitness values (Booker et al., 1997; Falkenauer, 1998). If these semantically correct results cannot be found with classical operators then it is up to the user to design specialised operators. All these crossover operators should be consistent with the Mendelian inheritance: the requirement that each gene carried by an offspring is a copy of a gene inherited from one of its parents (Booker et al., 1997). In the following paragraphs a brief overview of classical crossover operators is given. This is by no means a detailed analysis of their performance. Further details are provided by Booker et al. (1997).

One-point crossover One-point crossover was devised by Holland (1975). Goldberg (1989) also favours this operator and the schema theorem is based on this simple operator. It has three main steps. Firstly two parents are randomly chosen from the mating pool. Secondly, a random point along the string is chosen as a breakpoint. This delineates the parts of information to be exchanged. Finally, the two parent chromosomes swap this information to produce two children. One-point crossover has a very high positional bias: longer schemata will become disrupted more easily than the shorter ones. On the other hand it does not have a distribution bias, i.e. the cross site is chosen uniformly along the string.

Two-point and multi-point crossover The one-point crossover operator has been extended by De Jong (in Spears (1998)) into the n-point or multi-point crossover operator. In n-point crossover, n positions are randomly chosen along the chromosome and the information between two consecutive points are exchanged. According to De Jong two-point recombination is less likely to disrupt long schemata than one-point crossover because it is a cyclic operator. Therefore the longer schemata are preserved. There is no consensus about the advantages and disadvantages of multi-point crossover for $n \geq 3$.

Uniform crossover Syswerda (1989) introduced uniform or shuffle crossover. Unlike the other operators it does not use split points but it takes a decision of which parent to copy from based on a (fair or biased) flipped coin. He compared the performance of this operator with one-point and two-point crossover and found that while uniform crossover is more disruptive of schemata than one-point or two-point crossover, it does not have a length bias. This means that the defining length does not influence the probability of disruption. Moreover, he showed that the more disruptive nature of uniform crossover is likely to construct instances of higher order schemata. Spears (1998) extended this work by providing a common framework upon which to compare all n-point crossover operators and uniform recombination operators. Radcliffe ((1991) in Booker et al. (1997)) points out that uniform crossover is the only crossover operator that can generate all possible instances of offspring. Uniform crossover is the most commonly applied recombination operator in applications (Booker et al., 1997).

Discussion Spears (1998) analysed the effect of these operators on the recombination of building blocks. His conclusion is that the disruptive nature of the n-point crossover is affected by both the defining length and the order of the schemata, whereas uniform recombination is only affected by the order. All forms of recombination are more disruptive if there are higher order schemata and become less disruptive when the population converges.

4.3.2.3 Mutation

Although crossover is a very potent means to explore the search space, it does have a negative side-effect. As it only swaps existing knowledge, it cannot escape from suboptimal solutions after convergence. The only way to keep the exploration phase going is by inserting diversity in the population. This can be done by a mutation operator. Falkenauer (1998) calls the effect of the mutation operator the smallest possible random modification of an individual. Many authors use mutation with a low probability (Holland, 1975; Goldberg, 1989; Falkenauer, 1998) but empirical and theoretical investigations demonstrated the benefits of the role of mutation as a search operator (Bäck et al., 1997). Spears (1998) found that high levels of mutation are the most disruptive and

also achieve the lowest levels of construction. This means that by using high levels of mutation the chance that new building blocks are found decreases.

4.3.2.4 Setting the parameters

Before the genetic algorithm can be run the user has to determine how to set the various parameters. In general there are four variables that need to be set before good results can be obtained: the population size, the crossover probability, the mutation probability and the stopping criterion. The setting of these parameters is more of 'an art than a science' (Michalewicz, 1999) and for real-world applications they are mostly sought through trial and error. There are however some rules of thumb and theoretical models that have been worked out for small problems such as the bit-counting or One Max problem. For this problem the goal is to maximise the number of ones along the chromosome. Because of its simplicity it has been used widely for testing purposes. A fifth but implicit parameter that can largely influence the behaviour of a genetic algorithm is the initial seed for the random population (Osyczka, 2002). Running any genetic algorithm with a different random starting seed might produce very different results and this should be kept in mind when making comparisons between algorithms. For real-world applications this means that repetitions of experiments are needed in order to remove the random effect.

Population size The population size has to be considered carefully. If the population size is too small, the population will soon suffer from premature convergence because the diversity in the population is too low. On the other hand, if the size is too large the convergence towards the global optimum is slow and the memory requirements to run the genetic algorithm increase a lot. Lobo (2000) did some experiments with the One Max problem and a real-world network expansion problem. For his real-world application, the aim was to expand an existing electricity network to newly built houses. In order to do so, the company had the choice between building transformers and putting new cables between the houses and either the new transformers or existing substations. It was also possible to connect several transformers before connecting them to one of the existing substations. The total cost of the network expansion is the sum of all the costs needed to build the transformers and the costs of putting new cables. The overall goal of the electricity company is of course to provide electricity to all houses with a minimum cost. For One Max and the real-world application, he found that indeed an undersized or oversized population influenced respectively the solution quality and the time to convergence a lot.

Several authors (Harik et al., 1999; Pelikan et al., 2000a) have spent much effort on trying to design a population sizing model and although their models have been checked against some real-world problems, they are not really applicable for the practitioner (Lobo, 2000) because they require a lot statistical data on the fitness values such as the order of the building blocks or the variance

and means of the schemata. These parameters are not readily available for real-world problems. Lobo suggested to use a parameter-less GA that determines the population size through a rat race between several parallel populations with a different population size but in reality most researchers still determine the population size through trial and error.

Crossover and mutation probability Recombination and mutation is performed with a certain fixed or variable probability. Again the setting of these parameters is the subject of debate. Whereas the more classical genetic algorithm theorists fervently advocate the use of a high crossover probability (in the range $[0.8, 1]$ (Goldberg, 1989; Holland, 1975)) and a low mutation probability (in the range $[0, 0.01]$ (Goldberg, 1989; Holland, 1975)), recent work has shown that the issue of the mutation probability has been underestimated. Bäck et al. (1997) cites three important results on the mutation operator but the main (practical) result is that the lower bound for the mutation probability is $p_m = 1/l$ with l the length of the chromosome (Mühlenbein 1992 in Bäck et al. (1997)). Fonseca (1995) provides a more mathematical background for the mutation parameter. The probability that a chromosome with length l is not modified by mutation is:

$$P_s = (1 - p_m)^l \tag{4.2}$$

where p_m represents the bit mutation probability. If there is no crossover operator the probability of survival P_s should be no less than the inverse of the expected number of offspring, μ, of the best individual: $P_s \leq \frac{1}{\mu}$. It follows that

$$p_m \leq 1 - \mu^{-1/l} \tag{4.3}$$

In the presence of crossover, the actual mutation probability should be somewhere below this limit. If the expected number of offspring is 2, $p_m = 0.7/l$ was found to be a good parameter setting. Other authors suggest a variable mutation probability, either over the generations (Fogarty 1989 in Bäck et al. (1997)) or along the chromosome (Michalewicz, 1999). Lobo (2000) on the other hand states that the performance of the genetic algorithm is not so much influenced by these operators than by the population size. According to him the performance of a genetic algorithm is not very influenced by these setting. He suggests that the combined effect of recombination and selection (ignoring the effect of mutation) should result in a net growth factor of building blocks of 2. This ensures that the schemata can mix given an adequate population size and that the population will not converge prematurely.

Stopping criterion There are three main ways to stop the generational loop:

- allele convergence
- a predefined number of generations

- when the optimum is reached

It is evident that for real-world problems only the second option is open. Waiting for a full allele convergence can cost too much time, and it is very likely that the optimal solution has already been in the population for a long time. The third possibility is only possible for laboratory problems such as One Max where the optimum is known beforehand. The problem with the predefined number of generations is that again it needs user interaction. The combination of population size and number of generations provide the total number of function evaluations. The population size has the largest effect: running a genetic algorithm when the population has converged already does not make any sense. In reality the number of generations is usually set according to the time one is prepared to wait before a solution is achieved.

4.4 Implications for forest management problems

Representation For all forest management problems, the data structure that will be applied is a fixed-length chromosome given its performance in similar domains such as land use and urban planning. The encoding strategy should be tested, since there are no comparative studies as yet. Therefore, for the relevant problems integer, gray and binary encodings will be implemented and compared.

Selection operator In general for all applications tournament selection will be used as selection operator because (1) it is a fast method, (2) it is fitness scale insensitive and (3) it does not require any scaling or ranking.

Recombination operators and their parameters One-point crossover and uniform mutation were selected because of their ease of implementation and their theoretical foundation. Uniform mutation will be applied and the mutation probability is fixed for the entire duration of the search process. On all occasions the crossover probability was in the range $[0.8, 1]$ and mutation probability was above the lower bound of $0.7/l$ provided in literature.

Analysing the outcome All the experiments were repeated at least ten times (Osyczka, 2002) and mean and median values were used to compare the effect of the parameters or algorithms so that the effect of the initial seed could be eliminated from the analysis.

Chapter 5

Case study : solving a harvest scheduling problem with single objective genetic algorithms

5.1 Problem definition

The issue of harvesting scheduling was studied extensively in the past (Hoganson & Rose, 1984; Hof & Joyce, 1992, 1993; Lockwood & Moore, 1993; Murray & Church, 1995; Snyder & Revelle, 1997; Tarp & Helles, 1997; Borges & Hoganson, 1999; Snyder et al., 1999; Falçao & Borges, 2001). Forest managers need to schedule management treatments over a planning horizon. The two objectives that are mostly used in the harvest scheduling problems in literature are (1) to maximise net present worth, and (2) to minimise the deviations between the different cutting periods. Using a Model I harvest scheduling formulation this can be written as (Johnson & Scheurman, 1977):

$$\text{Maximise} \quad f = \sum_{i=1}^{N} \sum_{j=1}^{M} c_{ij} x_{ij} \tag{5.1}$$

$$\text{Minimise} \quad g = \sum_{j=1}^{M} \left(V_j - \overline{V} \right) \tag{5.2}$$

where N is the number of stands, M is the number of time periods, c_{ij} is the present value obtained when applying the management treatment to stand x_i in period j. V_j is the total volume summed over all stands (m^3) cut in period j and $\overline{V}$ is the average volume over all cutting periods. Eq. 5.1 expresses the management objective of maximising the net present value, while Eq. 5.2

expresses the objective of minimising deviations in timber volume between the different cutting periods.

5.2 Research rationale

For N cutting periods and k management activities, the harvest scheduling problem has N^k possible combinations. Constraining the problem to an integer program, requires $N * k$ decision variables. This increases the number of decision variables to such an extent that it can only be solved using heuristics. Therefore genetic algorithms will be used.
Because the problem can only be solved using heuristics the global optimum for the bi-objective problem is unknown. The global optimum for this problem under no even-flow assumptions on the other hand can be derived. In that case, all felling activities are postponed to the end of the planning horizon and the maximum present value that can be obtained under that scenario amounts to € 914232. This value can be used as a benchmark to compare the solutions found with the genetic algorithm.
Because the encoding strategy might influence the results from the genetic algorithm, the effect of three encoding strategies will also be investigated.

5.3 Material and methods

5.3.1 Input data

For each of the stands the yield class is known. This was used as input for the production tables from the Forestry Commission (Hamilton & Christie, 1971), and from these forecast tables the cumulative volume from thinning and felling activities can be derived. To simplify the problem, it was assumed that all timber, both from thinning and felling, was sold at the felling date even though this is not very realistic. Prices were real prices per diameter class published in 2000 (Anonymous, 2000) (Appendix A). The diameter class at each period was derived from Hamilton and Christie (1971). The discount rate was 3% and the present value was calculated as in Eq. 2.6 (p. 23).

The difficult task of assigning values to weights can be simplified by working with relative weights. If one is indifferent to either of the objectives, then the objectives should be rescaled between 0 and 1 in order to remove differences of scale magnitude (Buongiorno & Gilles, 1987). However, this might lead to numeric imprecision and therefore the weights are usually multiplied by a fixed factor. As the present value is in magnitude 100 times larger than the harvest volumes, the present value was divided by 100. The objective function can thus be written using the notations from Eq. 5.1 and Eq. 5.2 as in Eq. 5.3:

Maximise

$$f = \frac{\sum_{i=1}^{N} \sum_{j=1}^{M} c_{ij}}{100} + w \cdot \sum_{j=1}^{M} \left(V_j - \overline{V}\right) \tag{5.3}$$

with w the weight for the second objective.

5.3.2 Implementation

A genetic algorithm with binary tournament selection, one-point crossover with a probability of 0.8 and uniform mutation with a probability of 0.01 ($> 1/l$) was implemented in Java (Appendix B). The population size was 100 and the number of generations was set to 50. No fitness scaling was implemented as binary tournament selection is insensitive to the fitness differences. Because genetic algorithms can lose good solutions during the optimisation process, elitism was applied. In this particular case of elitism, the parent population and the child population were merged and sorted according to their fitness values. The best N individuals were used to continue the search process. Binary, gray and integer encodings were initially tested with equal weights. For the binary and gray codes, 3 bits for each harvesting period were used as there are 8 periods in total over the complete planning horizon. This was repeated 10 times (Osyczka, 2002).

After selection of the representation that led to the best solution, the weights were varied. The weight w was initially linearly distributed on the half-open interval $]0, 1]$ in steps of 0.2. If the weight 0 is included in the interval then the optimisation problem is unbounded and all felling activities will be planned at the end of the planning horizon (period 8). Two additional weights $(0.01, 0.05)$ were evaluated in a later phase to get more information on the Pareto-front between the two objectives. For each weight the genetic algorithm was repeated 10 times.

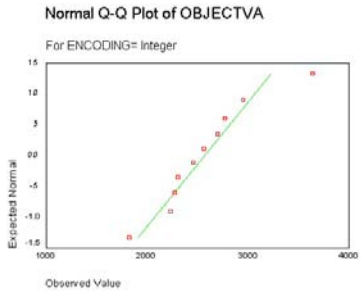
5.4 Results and discussion

Encoding strategy Initially, the influence of the encoding strategy on the solution quality was tested. In Table 5.1 the mean objective function, mean values for present value and mean sum of deviation in volume for binary, gray and integer coding are presented.

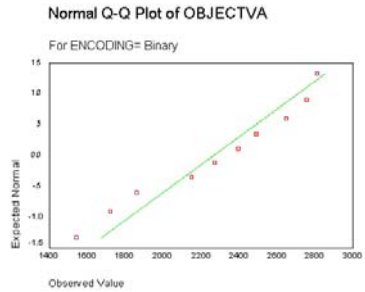
Table 5.1: Influence of binary, gray and integer coding on the performance of the genetic algorithm. The experiment was repeated 10 times for each encoding. OV is the combined objective value, PV is the present value, V_i the total volume summed over all stands cut in period i and $\overline{V}$ is the average volume over all cutting periods

Encoding type	mean OV	mean PV (*€ 100)	$\sum_{i=1}^{n} V_i - \overline{V}$ (m^3)
binary	2265·50	3308	1043·10
gray	2684·50	3534	850
integer	2573·50	3326·68	753·58

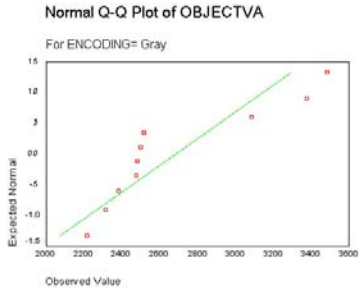
The significance of the differences between the mean objective values is tested using a One Way ANOVA, which has as null hypothesis H_0: *the means of k samples are equal*. One Way ANOVA assumes that the samples are independent, are drawn from a normally distributed population and that the variances between the samples are equal. The assumption of normality was tested using the Shapiro-Wilk test which has proven to be a better test than the classical Kolmogorov-Smirnov test (Zar, 1999). The null hypothesis for the Shapiro-Wilk test is H_0: *the sample is drawn from a normally distributed population.* QQ-plots, depicting the deviation of the sample distribution from a normal distribution, were inspected visually. Equal variances were analysed graphically with boxplots as well as statistically using Levene's test for homogeneity of variances (H_0: *variances are equal*).



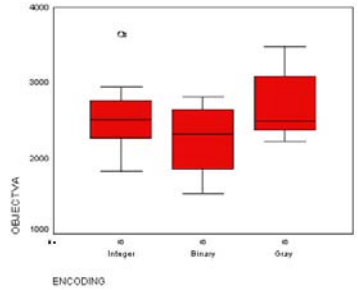
(a) The QQ-plot for integer encoding



(b) The QQ-plot for binary encoding



(c) The QQ-plot for gray



(d) Boxplot for the three encodings

Figure 5.1: QQ-plots and boxplot for the integer, binary and gray encoding strategies for a harvest scheduling problem. The effect of the encodings on the weighted objective value is tested. There are 10 repetitions

From Figs. 5.1(a) to 5.1(c), it can be concluded that the data is close to normal distribution. This is only confirmed by the Shapiro-Wilk test statistic for the integer and binary codes and not for the gray codes. The test statistics for the samples are:

- $p_{int} = 0.535 > 0.05 \Leftrightarrow H_0$ is accepted;
- $p_{bin} = 0.602 > 0.05 \Leftrightarrow H_0$ is accepted;
- $p_{gray} = 0.026 < 0.05 \Leftrightarrow H_0$ is not accepted.

The boxplot (Fig. 5.1(d)) indicates that the variances of the groups are equal. The test statistic of Levene's test backs this up: $p = 0.133 > 0.05 \Leftrightarrow H_0$ is accepted. As the data is not normally distributed, a One Way ANOVA procedure cannot be applied. The most common alternative to the One Way ANOVA statistics, is the Kruskal-Wallis test. This procedure is based on ranking the results from all samples in ascending order and then calculating the mean rank for each of the factors. The power of this test is lower than that from an ANOVA test, because the type II error, where the H_0 hypothesis is not rejected when in fact it is false, increases (Zar, 1999). It is thus possible that the null hypothesis is accepted, while there are significant differences between the two factors under consideration. On the other hand, if the null hypothesis is rejected then it is most certain that there are differences due to the factors present.

Table 5.1 shows that gray codes results in a higher objective value than integer and binary codes. The Kruskal-Wallis test statistic ($p = 0.263 > 0.05$) shows that there is no significant difference between the three groups.

> As there is no difference, the integer codes will be used in the following case studies because they are the most natural representation for the problem and this is recommended by Davis (1991).

Changing the weight In Table 5.2 and Fig. 5.2 the mean values per weight combination for integer encoding are presented. A first observation is that linearly distributing the weights on a small interval does not result to evenly spaced solutions along the Pareto-front. This has some implications: if a forest manager decides to investigate only weights in the interval $]0, 1]$ in steps of 0.2, a considerable amount of information on the shape of the Pareto-front will be lost. Changing the weight from $w = 0.2$ to $w = 0.05$ and then to $w = 0.01$, changes the slope of the Pareto-front substantially. Beyond $w = 0.2$ a small increase in present value results in a large increase in the sum of all deviations and a large increase of the present value. The effect of the even-flow objective is small even when the present value is deemed 5 times more important than even-flow, but this drastically changes once the present value is considered 100 times more important than even-flow. The present value obtained with a weight of 0.01 amounts to € 666811, which is 72.9 % of the maximum value that can be obtained if there are no even-flow assumptions.

Table 5.2: Results for the different weight combinations. The experiment was repeated 10 times. PV is the present value, V_i the volume cut in period i and $\overline{V}$ is the average volume over all cutting periods

	mean OV	mean PV (*€ 100)	$\sum_{i=1}^{n} V_i - \overline{V}$ (m^3)
1	2573.0	3326.58	753.58
0.8	2744.6	3396.14	814.43
0.6	3151.3	3659.14	846.40
0.4	3435.8	3752.19	790.98
0.2	4042.7	4211.20	1084.93
0.05	5482.1	5508.46	2636.38
0.01	6469.8	6668.11	19830.60

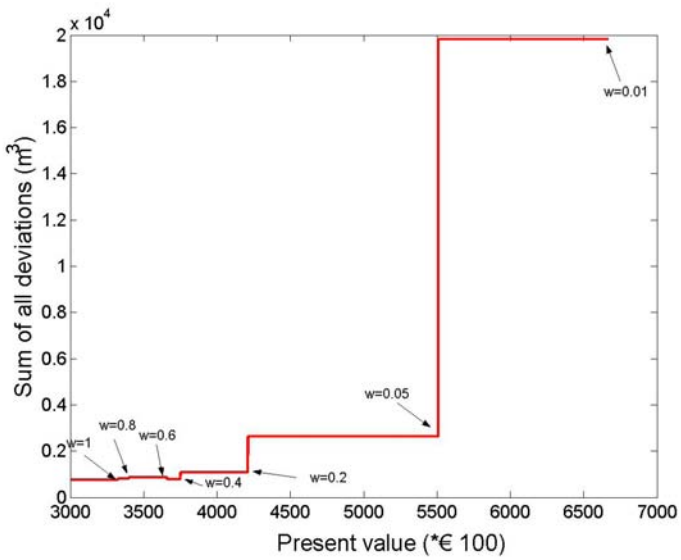
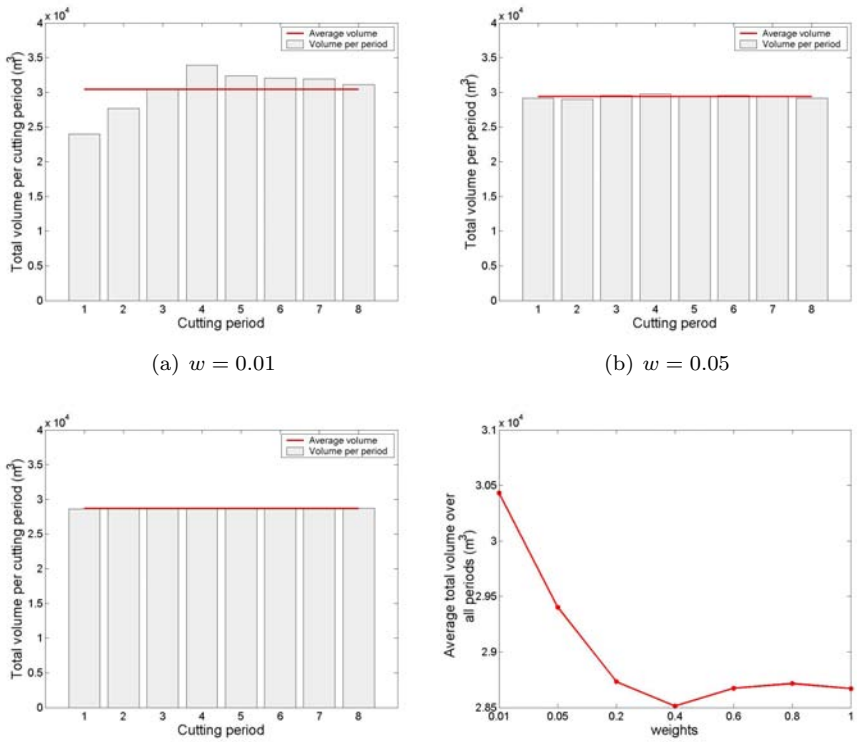


Figure 5.2: Results for the different weight combinations. The experiment was repeated 10 times. On the x-axis the present value (*€ 100), and on the y-axis $\sum_{i=1}^{n}(V_i - \overline{V})$

Validity In Fig. 5.3 the volumes per cutting period are presented for all the weights. From Fig. 5.3(a) to Fig. 5.3(c) the even-flow constraint is strengthened. If this constraint is strengthened then the average volume obtained over all periods declines when increasing w from 0.01 to 0.2. For a weight $w = 0.01$ the volume harvested per year amounts to 6.70 m^3/ha/yr. For equal weights this is only 6.30 m^3/ha/yr. To illustrate that Kirkhill Forest is similar for production as the Flemish forest: there the average over all forests per year is 4 m^3/ha/yr. The other weights produce similar average volumes. Even-flow constraints do not only have an effect on the present value but also have a negative side-effect on the average volume over all periods.



(a) $w = 0.01$

(b) $w = 0.05$

(c) $w = 1$

(d) Effect of w on $\overline{V}$

Figure 5.3: The influence of weight w on the variation in volume between the different cutting periods. From 5.3(a) to 5.3(c), the even-flow constraint is strengthened. In (d) the effect of the weights on the average volume $\overline{V}$ is shown

The felling age of the forest stands (Fig. 5.4) after equal weights indicates that the rotation age should be increased in order to obtain a normal age dis-

tribution. Up to 1/3 of the stands have a felling age over 80 years (Fig. 5.4). From Fig. 5.4, it also follows that some stands are cut very young in order to obtain an even-flow of timber volume.

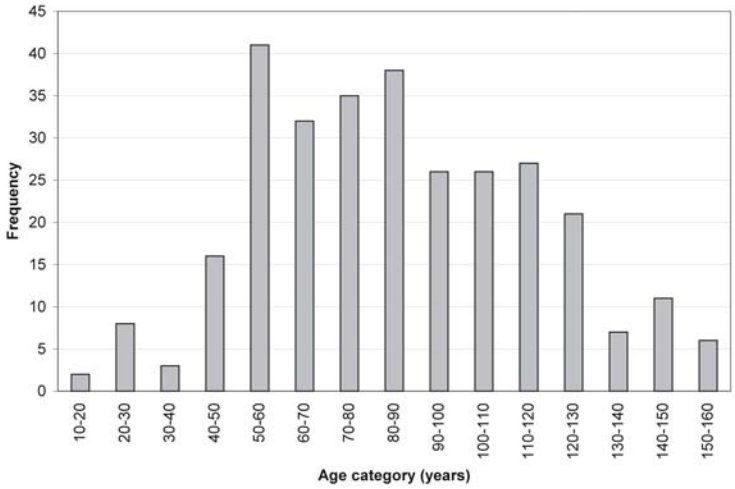


Figure 5.4: Felling age of the forest stands

The age distribution of the forest is affected by the harvesting plan. Looking at the effect of the plan where the two objectives were equally important, the age distribution of the forest almost resembles an age distribution of a normal forest (Fig. 5.5). This is caused implicitly by the even-flow objective. This is based on a volume control and the age distribution is implicitly adjusted to a normal state. This is confirmed in Fig. 5.6: if the even-flow objective is relaxed, the state of the forest reduces to a normal forest but to a lesser extent. Running the genetic algorithm for another planning horizon stabilises the age distribution, if the objectives are equally important, even more (Fig. 5.7). Starting from the age distribution with equal weights in the first planning horizon and running it again for a second planning horizon with a weight of 0.01 affects the age distribution a little: it becomes less stable (Fig. 5.8).

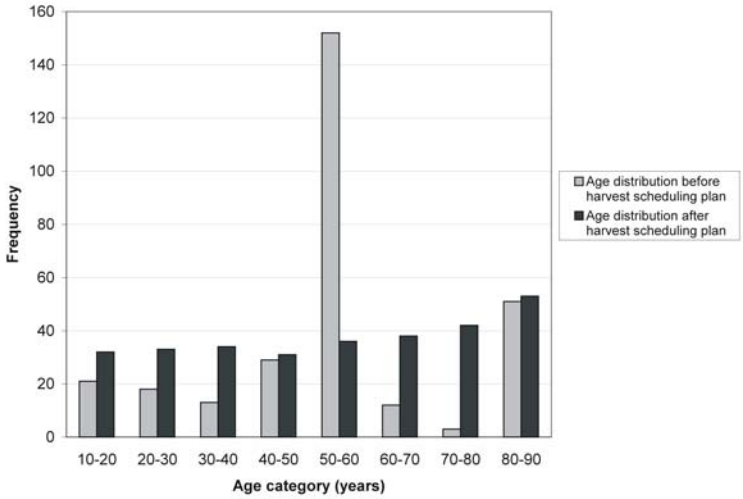


Figure 5.5: Age distribution of the forest before and after the harvest scheduling plan with two equally important objectives

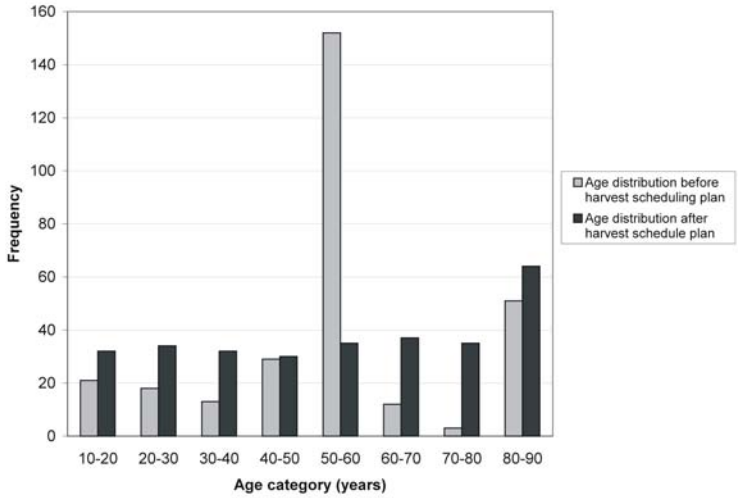


Figure 5.6: Age distribution of the forest before and after the harvest scheduling plan with the present value objective 100 times more important than the even-flow objective

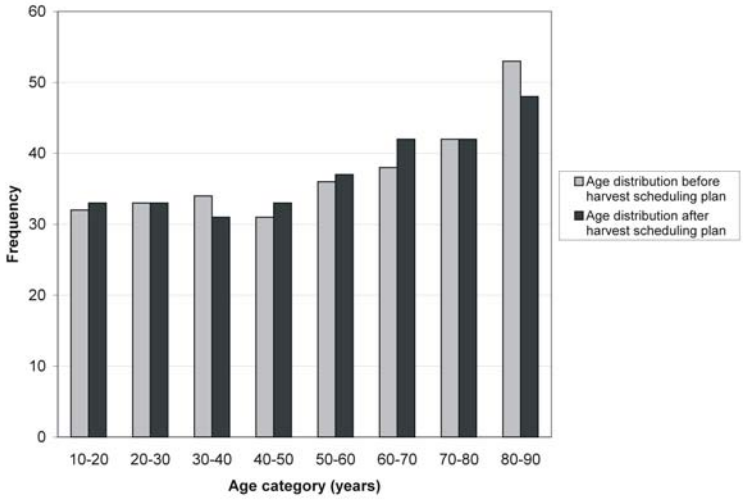


Figure 5.7: Age distribution of the forest before and after the harvest scheduling plan after a second planning horizon with two equally important objectives

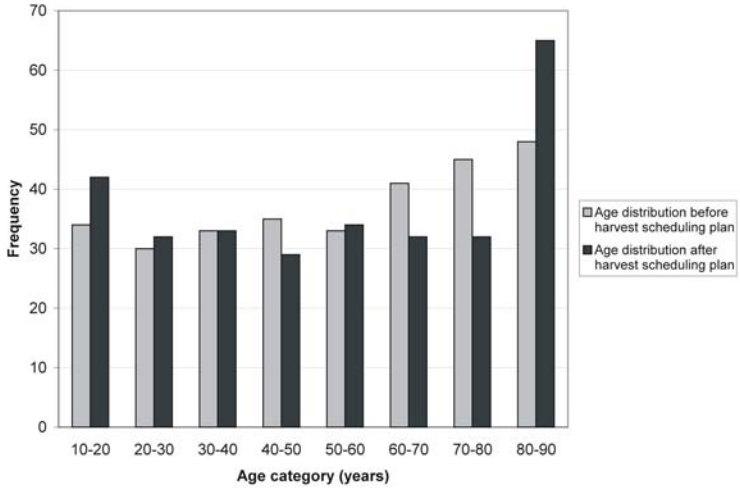


Figure 5.8: Age distribution of the forest before and after the harvest scheduling plan after a second planning horizon with the present value 100 times more important than the even-flow objective

5.5 Conclusion

Genetic algorithms are capable of solving a harvest scheduling problem. The encoding strategy did not affect the quality of the solutions; there was no significant difference between the different codes. As there is no difference, integer codes were used for the other experiments, since this is the most natural representation for the decision variables and this is generally recommended.

In order to find the Pareto-front, the weights were initially linearly distributed on the interval $]0, 1]$. It was found that this did not lead to evenly spaced solutions along the Pareto-front. In order to gain more information, two additional weights were chosen: $w = 0.01$ and $w = 0.05$. When the present value was 100 times more important the slope of the Pareto-front changed a lot. This implies that a user without prior knowledge on the problem, investigating the effect of the weights on the two objectives might loose a lot of information on the Pareto-front if these weights are linearly distributed on a small interval.

Both the age distribution and the average volume are affected by the even-flow objective. Running the genetic algorithm in order to maximise the present value and minimise the deviations between the periods, produces harvesting plans enforcing a balanced age distribution. Relaxing the even-flow objective has an effect on the age distribution, but then some variations in frequency between the different age classes are still present. The present value obtained with a relaxed even-flow constraints amounts to 72.9 % of the total maximum attainable present value. The even-flow objective also influences the harvested volume: this declines as the even-flow objective becomes more important.

A practical drawback using weights is that it is very cumbersome. Rerunning the genetic algorithm or any single objective optimiser for several weight combinations is a tedious job and requires large amounts of computing time.

In Chapter 9, a multiple objective genetic algorithm will be applied in order to optimise the two objectives simultaneously. If a multiple objective genetic algorithm is applied, no weights have to be set as the combination of these weights are evaluated implicitly. Consequently, a single run of the multiple objective genetic algorithm produces the Pareto-front.

Chapter 6

Case study: maximising the abundance of badgers using GAs

6.1 Problem definition

In Kirkhill forest, badger (*Meles meles*) is present. Badger is protected in Great Britain and is considered as edge-dependent species (Neal & Cheeseman, 1996). This means that the abundance is both influenced by the forest cover and the presence of nearby agricultural fields and by open areas in the forest where it can forage. Imagine in the simplest case that the abundance of badger is linearly related to the perimeter between the forest and the open areas and a simulated forest consisting of nine forest management units as in Hof and Joyce (1992). Each block has a size of 3-by-3 km. The goal is to maximise the abundance of edge-dependent species. As the relationship between the perimeter and the edge-dependent species is assumed to be linear, this is equivalent to maximising the perimeter between the clear felled and old growth blocks. Hof and Joyce (1992) formulated the optimisation problem as follows :

Maximise E subject to

$$
\begin{aligned}
N &= \sum_i \sum_j C_{ij} \\
D &= (N \cdot 12) - 3[C_{11} \cdot (C_{12} + C_{21}) \\
&\quad + C_{12} \cdot (C_{11} + C_{13} + C_{22}) \\
&\quad + C_{13} \cdot (C_{12} + C_{23}) \\
&\quad + C_{21} \cdot (C_{11} + C_{22} + C_{31}) \\
&\quad + C_{22} \cdot (C_{12} + C_{23} + C_{32} + C_{21}) \\
&\quad + C_{23} \cdot (C_{13} + C_{22} + C_{22} + C_{33}) \\
&\quad + C_{31} \cdot (C_{21} + C_{32}) \\
&\quad + C_{32} \cdot (C_{31} + C_{22} + C_{33}) \\
&\quad + C_{33} \cdot (C_{32} + C_{23})] \\
E &= 1.261573 \cdot D
\end{aligned}
$$

where N is the number of cells left in old growth, $C_i j$ is the management activity assigned to the stand on row i and column j and D is the amount of edges between old growth and harvested areas.

The amount of edge D is calculated as follows: the edge from each C_{ij} that equals one, i.e. old growth, is 12 if all adjacent C_{ij} are zero, i.e. clear felled. For each adjacent cell C_{ij} that is not zero the edge is overestimated twice by 3 km. The constraint thus starts with the maximum individual edge $N \cdot 12$ and deducts the overestimates for the cells. The optimal objective function value is 60 and occurs when a checker board pattern of clear felled and old growth blocks is present (Fig. 6.1).

1	0	1
0	1	0
1	0	1

Figure 6.1: The optimal solution is a checkerboard pattern. The blocks that remain standing have value 1, the cut blocks have value 0.

6.2 Research rationale

In this case study, models for spatial allocation in geographic information systems are not yet integrated with the genetic algorithm. Because in literature no benchmark problems exist where GIS and GA or any other optimiser are combined during the optimisation process, a more basic spatial problem is solved with genetic algorithms but without GIS as to have a common bed of comparison with problems described in literature. In later case studies, GIS functionality will however be combined with the flexibility of GA optimisation.

6.3 Material and methods

In order to test the effectiveness of the genetic algorithm a series of experiments was performed. In each of the experiments a fixed-length binary string was used with length $l = 9$, each block being represented as a bit. As only two management activities were allowed (clear fell and do-nothing) a fixed-length data structure with binary encoding was sufficient. A simple genetic algorithm with a crossover probability of 0.9 and mutation probability of $0.1 \approx 1/l$ was implemented. As the search space for the problem has only 512 solutions, for a population size of 30 all possible combinations can be generated after 18 generations. Therefore the maximum number of generations was fixed on 20. The optimisation process was repeated until the best solution was found or until there was a time out, which occurred when the maximum number of generations was reached. This experiment was repeated 1000 times for every population size. The population size was varied. Initially a large population was used, and gradually this was lowered in the experiments until the percentage of repetitions that reached the optimal solution was insufficient ($< 80\%$). This could indicate the minimal population size needed to solve the problem.

6.4 Results

For a population size of 30, the optimal value is only obtained in 90.5% of the cases before the maximum number of generations is reached (Table 6.1). The mean number of fitness evaluations is 245, which is after approximately 8 generations. The population size could be lowered to a size of 24 and still the optimum is found in 84.6 % of the cases. Lowering the population size even more does not yield satisfactory results. Even though the genetic algorithm does not find the global optimum for such a seemingly simple problem always, the results were not considered too bad as no attention was paid to the setting of the parameters of crossover probability or mutation probability at this stage of the research.

Table 6.1: Percentage of repetitions where the optimal value was obtained before maximum number of generations was reached for forest with a 3-by-3 grid layout for an edge-dependent maximisation formulation

P	Optimum (%)
30	90.5
24	84.6
18	75.7

Reformulating the problem as an adjacency problem eases the optimisation process. If for the simulated forest the objective function is to prevent adjacent cuts, then the optimal solution is reached in 99.5% of the cases after on average 155 function evaluations (Table 6.2). This means that the global optimum is found within the fifth generation. It is possible to reduce the population size to 12 and still obtain the global solution in 82.5% of the cases before time out. If the population size is decreased even more, the global solution is reached in less than 80%.

Table 6.2: Percentage of repetitions where a global optimum is obtained before maximum number of generations was reached for forest with a 3-by-3 grid layout for an adjacency formulation

P	Optimum (%)
30	99.5
24	97.4
18	92.3
12	82.5

The adjacency formulation was applied to a larger grid of 9-by-9 blocks, leading to a search space of $2^{81} \approx 2 \cdot 10^{24}$. The crossover probability was set to 0.9 and the mutation probability 0.01. The experiment was repeated 20 times for a population size of 80 and for a population size of 200. As the initial seed influences the result (Osyczka, 2002) the mean evolution over the repetitions is used for comparison.

In Fig. 6.2 this mean evolution over the 50 generations is depicted. The number of non-adjacent stands that is reached on average for a population size of 80 is 74. Increasing the population to 200 slightly improves the average number of correct stands up to 76 (Fig. 6.3). This means that over the complete solution 2 adjacent stands do not have a correct management activity assigned to them.

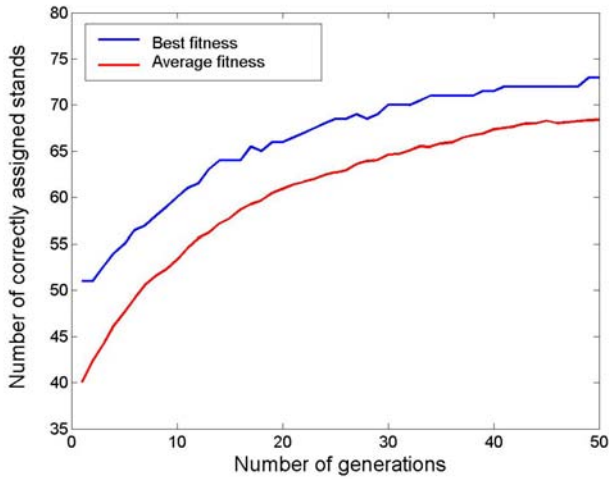


Figure 6.2: Evolution of the fitness value for a forest with a 9-by-9 grid layout for an adjacency formulation. The population size is 80 and number of generations is 50

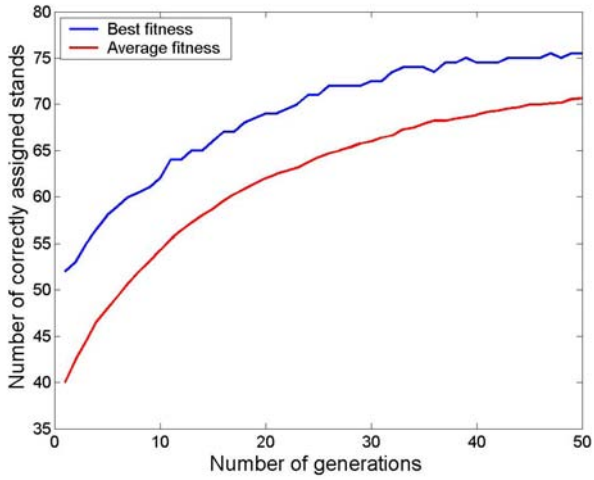


Figure 6.3: Evolution of the fitness value for a forest with a 9-by-9 grid layout for an adjacency formulation. The population size is 200 and number of generations is 50

As the population had not converged, the number of generations was increased up to 150. This yielded the evolution as in Fig. 6.4. The same best value was obtained and once more no convergence occured. The best and mean fitness values do level off on a plateau and do not change anymore beyond the 55th generation.

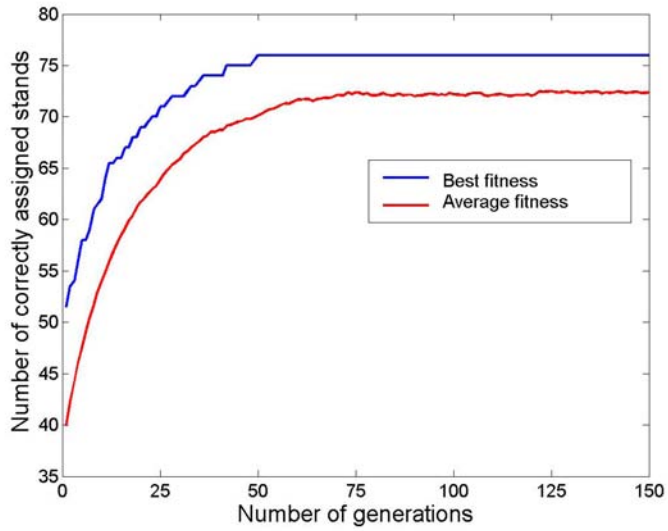


Figure 6.4: Evolution of the fitness value for a forest with a 9-by-9 grid layout for an adjacency formulation. The population size is 200 and number of generations is 150

An explanation as to why the adjacency problem is more readily solved than the perimeter problem is the following. In the adjacency problem the following cutting patterns are equivalent to each other:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

For the edge formulation however, the first structure is a local optimal solution (perimeter= 48). This local optimal pattern is the solution that is found after time out. If the population converges towards this local optimum, it is very difficult to step from that solution to the global optimum as it requires all the bits to be switched. This problem was also reported by Hof and Joyce (1993). They mention that the fitness surface is not convex and that local optima are present. The complete fitness surface is presented in Fig. 6.5.

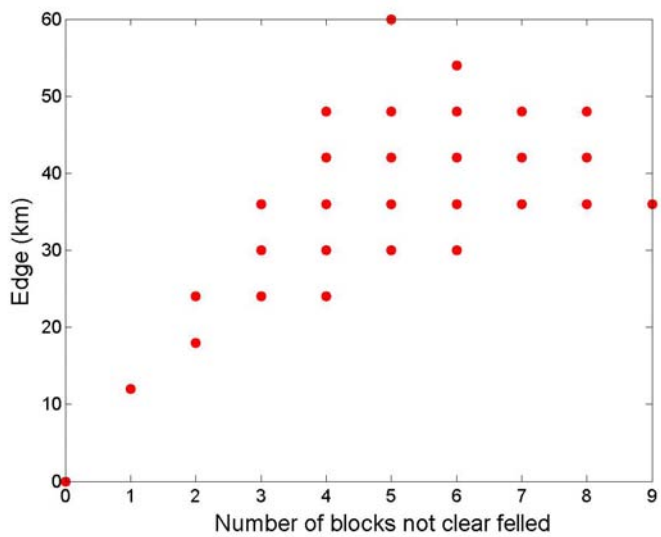


Figure 6.5: Fitness surface for the different combinations, on the x-axis the number of blocks that are not clear felled, on the y-axis the edge between old growth and cut blocks

A possible solution to overcome this problem is to ensure diversity in the population with diversity-stimulating measures. These will be explained in Chapter 7. Another possibility is through the use of techniques that try to process schemata as a whole which will be handled in Chapters 11 and 12.

The technique described by Hof and Joyce (1992) guarantees optimality but is not extendable towards larger problems. In the case of a genetic algorithm however, the relationship between the cutting blocks and the perimeter can be determined through the use of a geographic information system and therefore it is expected that even larger problems can be solved using GA. An application based on this approach will be presented in Chapter 10.

6.5 Conclusion

A simple genetic algorithm is not capable of finding the global optimum for an edge-maximisation problem in all cases: only in 90.5% of the repetitions for a population size of 30 the global optimum is attained before 600 solutions are evaluated. Reformulating the problem as an adjacency problem leads to better results: for a population size of 30 the optimum is reached in 99.5% of the cases. The difference in performance of the genetic algorithm is due to the existence of local optima for the edge maximisation problem. A possible method to overcome this problem is through the use of genetic algorithms that search for good schemata and process these as a whole. This subject will be treated in Chapter 11.
The genetic algorithms have an advantage over the classical non-linear technique. The equations for the edge formulation cannot be extended to larger problems. The genetic algorithm offers the advantage that it can be integrated with a GIS because the genetic algorithm requires only the fitness value. Inside the GIS, the edge length can be determined and the result can be fed back to the genetic algorithm. An example of such an application will be given in the Chapter 10 on multiple objective genetic algorithms.

Chapter 7

Extending the simple genetic algorithm to multiple objectives

One of the objectives in this dissertation is to implement an optimiser that facilitates simultaneous optimisation of multiple objectives without having to combine the objectives into a single objective function.

In order to use genetic algorithms for multiple objectives two major modifications are necessary. A first problem arising from multiple objectives is the assignment of fitness values and the selection of individuals. As multiple objectives are defined, the selection criterion can no longer be based on the raw fitness values of the objectives. New selection criteria will be discussed in Section 7.1.

Another feature of single objective genetic algorithms is that the population usually converges on a single solution. For multiple objectives this is unwanted because the aim is to to find a set of evenly spaced Pareto-optimal solutions. To counter this convergence, techniques have been proposed to enforce population diversity and they will briefly described in Section 7.2.

7.1 Fitness assignment in a multiple objective environment

In a multiple objective environment, one individual will have two or more objective values, one for each objective function. Because there is an objective value vector, the criterion for the selection operator must be redefined. Even with multiple objective evolutionary algorithms, the selection criterion has to be based on a single fitness value. This value must be derived from the objective values and any monotonic transformation is sufficient as long as one ensures that the individuals are at least as fit as the solutions they dominate (Fonseca & Fleming, 1997). There are many ways to define this monotonic transformation

and these can be categorised into three main categories (Fonseca & Fleming, 1995):

- plain aggregation procedures;
- non-Pareto-based approaches;
- Pareto-based approaches.

The first category will not be discussed in detail because this is the approach that has been used commonly for all classical multiple objective optimisers. The other groups are briefly described in the following sections.

7.1.1 Non-Pareto-based approaches

In the first generation of multiple objective evolutionary algorithms non-Pareto-based approaches were used. The first real multiple objective approach was made by Schaffer (1985) (in Fonseca and Fleming (1995)). His Vector Evaluated Genetic Algorithm (VEGA) splits the population into several subpopulations and each subpopulation was used to evaluate one of the objective functions. The selection phase took place on the subpopulations but the reproduction phase was performed on the complete population. One of the main drawbacks was that this approach biases the search towards more extreme solutions, because the selection was biased towards the individuals performing well in one or the other objective direction but tends to ignore the compromise solutions (Fonseca & Fleming, 1995; Goldberg, 1989). This is referred to as *speciation*.

7.1.2 Pareto-based approaches

Pareto-based fitness assignment was initially proposed by Goldberg (1989) and has been very popular in the domain of multiple objective evolutionary algorithms (Van Veldhuizen & Lamont, 2000): in the category of *a posteriori* techniques twice as many Pareto-based selection approaches exist than the sum of non-Pareto-based and aggregating approaches. The Pareto-based approaches are founded on the notion that solutions on the same level of domination should receive equal fitness values. According to a survey by Van Veldhuizen and Lamont (2000), the two mainstream Pareto-based approaches are a scheme proposed by Goldberg (1989) and by Fonseca and Fleming (1993).

Goldberg (1989) proposed iterative fitness assignment: initially all the non-dominated solutions are identified in the population and these solutions are assigned rank 1. Then this set of solutions is removed from the population and the next set of non-dominated solutions is assigned rank 2. This continues until all solutions have been assigned their rank. This procedure was used in Srinivas and Deb (1994) and Deb et al. (2000) in the design of the Non-Dominated Sorting Algorithm (Srinivas & Deb, 1994) and Non-Dominated Sorting Algorithm II (Deb et al., 2000).

Fonseca and Fleming (1993) use a different approach: each individual receives a Pareto-rank corresponding to the number of individuals that dominated this individual. This ensures that all non-dominated solutions are assigned the same ranking. Dominated solutions are penalised according to the density of solutions along the Pareto-front.

7.1.3 Discussion

Van Veldhuizen and Lamont (2000) list some recommendations for the design of multiple objective evolutionary algorithms. In general they recommend algorithms that incorporate Pareto-based approaches because they seek the Pareto-optimal front explicitly. Given these recommendations the Non-Dominated Sorting ranking procedure is applied. Essentially there should no difference between the two ranking procedures other than implementation details (Fonseca & Fleming, 1997), because in both approaches individuals at the same level of domination receive the same ranking and the order between the solutions is respected between the two approaches.

7.2 Niche formation methods

7.3 Introduction

Both the previous Pareto-ranking procedures ensure that all individuals at the same level of dominance receive the same fitness value, but this does not ensure that the solutions will be evenly distributed along the Pareto-front. Due to finite population size, genetic algorithms tend to drift towards a single (sub)-optimal solution even when the problem is multi-modal. This phenomenon is referred to as genetic drift. As it is the aim of multiple objective evolutionary algorithms to approximate the Pareto-optimal front, some sort of diversity-stimulating measures have to be applied to counter this drift.

Most of the proposed techniques are a metaphor from nature. Species have evolved over time in such a way that they avoid overlap for resources as much as possible. Each species has its own function and resource in nature and this is called the species' niche. Every species has adapted in such a way that it fulfills its jobs within the environment in the best possible way. If the abundance of that species increases due to a favourable environment, the resources available per capita decline. This increases the normal level of selective pressure on the population and consequently the number of individuals that produce offspring declines. This self-regulating effect reduces the population size until there is enough resource available once more for each individual.

Many techniques based on this metaphor of niching have been proposed to ensure diversity for multiple objective genetic algorithms: restricted mating (Fonseca & Fleming, 1993), clustering (Zitzler, 1999), adaptive grid methods (Knowles & Corne, 2000) amongst others. Because niche inducing methods were

originally designed for multi-modal functions, Purshouse and Fleming (2002) investigated whether they perform well in the case of multiple objective problems. They concluded that the incorporation of a niche inducing method enhances the spread along the Pareto-front well, and that the performance of multiple objective genetic algorithms is significantly better due to the diversity stimulating operators.

As it is by no means the aim in this dissertation to compare all niching techniques, two techniques were selected: the fitness sharing strategy with σ_{share} calculated as by Fonseca and Fleming (1993) because this technique has proven its merit in land use planning (Matthews et al., 2000) and the crowding distance assignment as proposed by Deb et al. (2000) and Deb (2001) because it has a reduced computational complexity.

7.3.1 Fitness sharing

Fitness sharing was proposed by Goldberg and Richardson (1987 in Goldberg (1989)) and decreases the fitness value of an individual in relation to the number of similar individuals. The similarity is expressed either in the genotype space: the number of genes where the individuals differ, or in the phenotype space, the distance in objective values of two individuals. For a population of size N the shared fitness value of an individual is calculated in three steps. Initially a sharing function is determined as follows

$$Sh(d) = \begin{cases} 1 - \left(\frac{d}{\sigma_{share}}\right)^{\alpha} & \text{if } d \leq \sigma_{share} \\ 0 & \text{otherwise} \end{cases} \tag{7.1}$$

All individuals within a distance of σ_{share} of each other will decrease each other's fitness (Fig. 7.1). Dissimilar individuals have no impact on each other's fitness value.

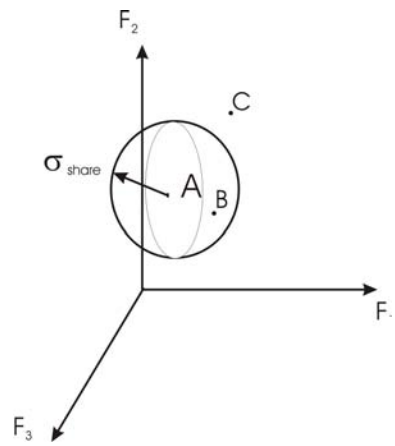


Figure 7.1: All the individuals within a distance σ_{share} of individual A (such as B) will decrease the fitness value of A. Sharing can be performed in either the decision variable or objective function space

A second step is to determine the *niche count* $NC(i)$ of each solution i. For a solution i this is the sum of all distances obtained by applying the sharing function to the complete population and is thus determined as

$$NC(i) = \sum_{j=1}^{N} Sh(d_{ij}) \tag{7.2}$$

with d_{ij} the distance between individual i and individual j. Finally the fitness value of individual i f_i' is

$$f_i' = \frac{f_i}{NC(i)} \tag{7.3}$$

with f_i the original fitness value. The problem with this technique is the determination of σ_{share}. Fonseca and Fleming (1993) provide a mathematical background to calculate this parameter. Since for multiple objectives evenly spaced solutions along the Pareto-optimal front are desirable, then ideally the distance between successive solutions along the Pareto-front is L/N where L is the perimeter length of the front and N is the population size.

In reality this perimeter is unknown and therefore they have suggested a procedure to dynamically update this parameter at every generation. If the search space is divided into different hypervolumes that are a distance σ_{share} apart from each other then all ideally solutions lie in a different hypervolume. If the distance Δ_i between current maximum objective value $\max_i$ and minimum the objective value $\min_i$, for each of the k objectives, is a surrogate for the perimeter length of the Pareto-front, then σ_{share} is calculated as

$$N\left(\sigma_{share}\right)^{(k-1)} = \frac{\prod_{i=1}^{k}(\Delta_i + \sigma_{share}) - \prod_{i=1}^{k}\Delta_i}{\sigma_{share}} \tag{7.4}$$

where k is the number of objectives and

$$\Delta_i = \max_i - \min_i \tag{7.5}$$

Before fitness sharing is used, all fitness values should be normalised into $[0, 1]$ to avoid that scale differences of the objectives influence the distance measures. Fonseca and Fleming (1993) suggests that normalising the objectives by the best estimate of Δ_i available at each generation yields good results as they showed in Fonseca and Fleming (1998).

7.3.2 Crowding distance operator

Another approach, the crowding distance operator, was proposed by Deb (1999). In fitness sharing, the niche count has to be determined and this requires the comparison of each individual with each other individual in the population. As this is computationally inefficient, they proposed a method where each individual is only compared with its neighbours instead of comparing it with the entire population. The two closest individuals are those that have the nearest objective function value in every objective dimension. For a bi-objective problem, the crowding distance of an individual is calculated as the sum of the sides of a rectangle between the two closest individuals (Fig. 7.2). The distance between these neighbours is then a measure for the density around the solution: if the closest neighbours are far away this implies a good spread along the Pareto-front. If the neighbours are very close to the solution, allowing new solutions in this area should be discouraged.

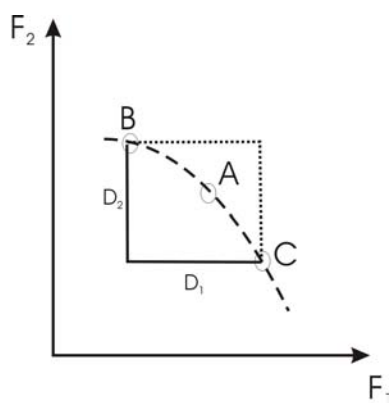


Figure 7.2: Crowding distance measure: the distance between two closest solutions of point A (B and C) is determined. The distances for each objective dimension are added together (D_1 and D_2) and yield the crowding distance. If this distance is low, the crowding around A is high and the probability of A winning a tournament is lowered

Individuals are then selected according to a modified tournament selection operator. In Deb's approach binary tournaments are held and the diversity measure is based on the crowding distance instead of a simple count of other individuals within a distance σ_{share} of the individual under investigation. The tournament relationship can then be defined as (Deb, 2001):

> An individual i wins in a tournament with another solution j if any of the following conditions are true:
>
> 1. solution i has a better rank than individual j: $r_i < r_j$
> 2. both solutions have the same rank but individual i has a better crowding distance than individual

7.4 Comparing multiple objective evolutionary algorithms

Given the fact that the outcome of multiple objective evolutionary algorithms is a set of solutions, it becomes very difficult to determine whether a set of parameters results in better performance than another set or whether one algorithm gives better results than another. In the following sections the mainstream performance indices are reviewed, together with their shortcomings and advantages.

7.4.1 Performance indices

Various unary performance indices have been proposed in the literature. The question arises which indices might adequately measure the multiple objective genetic algorithm results or allow meaningful comparisons of the implementations (Coello et al., 2002). Appropriate measures must be selected upon which to base multiple objective genetic algorithm performance claims. There is a difference between the performance indices that have been typically used in operations research versus those commonly applied in the domain of multiple objective genetic algorithms. In general the indices from operation research focus on the genotype space and they try to measure the distance of a point to the Pareto-optimal set, whereas those from the genetic algorithm domain most commonly measure the distance from a solution to the Pareto-optimal front in the phenotype space. Because two algorithms might have different behaviour in the genotype and phenotype space, it is certainly not an easy task to determine which of the algorithms is better given the indices.

The performance indices that have been formulated are designed to evaluate the two characteristics of a multiple objective genetic algorithm:

- closeness to the Pareto-optimal front

- spread along the Pareto-optimal front

Testing closeness Van Veldhuizen and Lamont (1999) proposed two measures: the *error ratio* and the *generational distance*. The error ratio is simply calculated as the ratio of the number of solutions that are not on the Pareto-optimal front to the population size.

$$\frac{\sum_{i=1}^{n} e_i}{n} \tag{7.6}$$

where $e_i = 0$ if individual i is on the Pareto-optimal front P^* and $e_i = 1$ if it is not. This measure is bounded between 0 (all solutions are on the Pareto-optimal front) and 1 (none of the solutions are on the Pareto-optimal front). One of the main drawbacks of this measure is that it can only indicate if there are any of the solutions on the Pareto-optimal front. If for two algorithms none of the solutions are on this Pareto-optimal front the error ratio cannot distinguish between these two algorithms (Deb, 2001). Therefore Deb (2001) redefines the error conditions as follows: if the minimum distance between an individual i and the Pareto-optimal front P^* is larger than a threshold δ then the individual is counted as an error. If a suitable threshold is used, then the δ-error ratio can give an indication about the proportion of individuals within a distance δ of the Pareto-optimal front (Deb, 2001).

Another measure proposed by Van Veldhuizen and Lamont (1999) is the generational distance (GD). The generational distance is a value representing how far the current front is from the optimal front. The value is defined as

$$G = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \tag{7.7}$$

where n is the number of individuals in the current front and d_i is the distance between each of the individuals and the nearest individual on the Pareto-optimal front. This measure can also be used as a measure of fluctuation between several repetitions. Deb (2001) proposes to use the variance in the distance values to test the robustness of an algorithm. Both measures defined by Van Veldhuizen and Lamont (1999) are scaling dependent.

Testing spread among the non-dominated points Schott (1995) introduced a spacing measure based on a variance-like measure. For the spacing measure the variance between the distances of each solution on the Pareto-front and the mean distance along the Pareto-front is calculated.

Deb and Jain (2000) extended this measure because the spacing measure does not take into account the maximum spread between the most extreme solutions.

$$\Delta = \frac{\sum_{m=1}^{M} d_m^e + \sum_{i=1}^{|Q|} (d_i - \overline{d})}{\sum_{m=1}^{M} d_m^e |Q| \overline{d}} \tag{7.8}$$

where $|Q|$ denotes the number of solutions in the Pareto-front, d_i the Euclidean distance of solution i to the closest solution j in the Pareto-front, $\overline{d}$ is the average distance over all solutions and d_m^e is the distance between the extreme solutions of Pareto-front and Pareto-optimal front according to objective dimension m.

If the spacing and spread indices are averaged over a number of repetitions, then a low mean value indicates evenly spaced solutions. If this space measure is high the solutions are not very well distributed along the front. Knowles and Corne (2000) indicate that these measures can only be used in conjunction with other indices. In that case it provides information about the vector distributions. An advantage of these measures is their low computational cost.

Combining spread and closeness Another measure defined by Zitzler (1999) is the hypervolume or $\mathcal{S}$-measure. It calculates the hypervolume enclosed by a solution set A and a reference point. It computes the area of the search space dominated by this solution set. In many aspects this measure is a very good one (Knowles & Corne, 2002; Zitzler et al., 2002) but it has one caveat: the size of the dominated space is easily influenced by the reference point, and care has to be taken before any decisive conclusions based on this measure can be made. A disadvantage is the larger computational overhead.

Comparing the indices Several approaches can be used to compare the indices. These approaches are guided by the underlying distribution assumptions. If the independent samples are normally distributed and have equal variances a usual One Way ANOVA test can be used to determine whether the differences between the indices are significant. If the above stated assumptions are not met, then a non-parametric Kruskal-Wallis test provides an alternative.

A different strategy is proposed by Purshouse and Fleming (2002). As the results from several runs are influenced by the initial seed, some of the differences may be the result of this initial seed. In order to exclude the random effect, they use a non-parametric test based on bootstrapping (Hollander & Wolfe, 1999). The bootstrapping test procedure in a one-sample non-parametric framework can be described as follows (Hollander & Wolfe, 1999): suppose a parameter $\theta = \theta(F)$ has to be estimated and there are n samples $X_1, X_2, \ldots, X_n$ randomly drawn from an unknown distribution F. The bootstrapping procedure is then

1. Make n random draws with replacement from the sample $X_1, \ldots, X_n$
2. Perform this step a large number of times (e.g. 1000) (B). For each draw calculate the estimate of the parameter θ, this is $\widehat{\theta}$. Denote the B values of $\widehat{\theta}$ as $\widehat{\theta}^{*1}, \ldots, \widehat{\theta}^{*B}$, these are the bootstrapping replications of $\widehat{\theta}$
3. Sort the bootstrapping values in ascending order

Using the B sorted bootstrapping estimators the confidence interval can be derived without any assumptions about the underlying distribution. For a confidence level $100(1-\alpha)\%$ the lower bound θ'_L and the upper bound θ'_U can be

derived in the following manner:

$$\theta'_L = \widehat{\theta}^{*(k)} \ , \ \theta'_U = \widehat{\theta}^{*(B-k+1)} \tag{7.9}$$

where k is the largest integer less than or equal to $(B+1)(\alpha/2)$. This confidence interval is referred to as the percentile interval. Hollander and Wolfe (1999) indicate that the number of bootstrapping values B should be > 2000.

When comparing the performance of two multiple objective genetic algorithm, the difference in means from the two algorithms has to be estimated. Purshouse and Fleming (2002) therefore extend the original bootstrapping idea into the following procedure:

1. For each of the two algorithms, each with a sample size of n, calculate the mean of e.g. the hypervolume measure
2. Combine the results of the two algorithms in one set S
3. Make $2n$ random draws with replacement from the combined set S
4. Split the set up into two new samples and assign the first half to algorithm 1 and the other half to algorithm 2
5. Perform steps 3 and 4 a large number of times B. For each draw calculate the estimate for the mean of each sample and then calculate the estimate of the difference in means between the samples
6. Sort the estimated difference in means values in ascending order
7. Calculate the confidence interval as in Eq. 7.9

Purshouse and Fleming (2002) set the number of bootstrapping iterations to $B = 5000$. All the mean differences can then be put into a histogram, and the original mean difference can be used as a test measure. If this test value lies within the 95% (or 99%) confidence interval, the mean difference can be seen as one originating from randomly assigning the output results and therefore it cannot be concluded that one method is better/worse than another. If the test value is outside the boundaries, the difference in means is very unlikely caused by randomness and consequently one algorithm is better than the other.

Some care needs to be taken when using this approach: it is possible to derive different estimates of bootstrapping confidence intervals when the same data is bootstrapped (Hollander & Wolfe, 1999). Especially in cases where the test measure is very close to the confidence interval boundaries different conclusions might be drawn when repeating the bootstrapping method several times. This is in contrast to the basic assumption of statistics stating that given the same data the conclusion should always remain the same.

The procedures to calculate the above indices are further elaborated in Appendix C.

Number of indices to use Zitzler et al. (2002) proved mathematically that it is impossible to state that one algorithm is better than another given a finite set of performance indices. Deb and Jain (2000) however indicate that even though the use of performance indices is mathematically incorrect, they do provide a lot of information about the performance of the two algorithms as long as one does not revert to strong statements such as: '*Algorithm A is significantly better than B*'.

7.4.2 Statistical approaches

A totally different approach for performance comparison avoiding the issues of the number of indices to use, is based on statistics and was proposed by Fonseca and Fleming (1996). They use the so-called attainment function as a measure of performance.

If a multiple genetic algorithm is run for several times, the search space can be divided into three areas :

- part of the search space that is always dominated by all of the runs;
- part of the search space that is dominated by some of the runs;
- part of the search space that is never dominated by any of the runs.

The part of the search space that is always dominated is bounded by the set of the 'tightest goals' and this corresponds to attaining the goals in 100% of the repetitions. Using the repetitions, it is also possible to draw the set of goals that are attained in 50% of the cases and this is the median attainment surface over all runs. Next to the median, 25% and 75% percentiles can also be drawn. Using these attainment surfaces, a Mann-Whitney test can be applied to establish the differences in performance between the several algorithms. To this end, Knowles and Corne (2000) grid the median and for each of the grid points they determine whether a solution on median A is dominating the corresponding solution on median B point or vice versa. Using this information, they give test statistics in a matrix form, showing the area of the search space where algorithm A beats algorithm B and vice versa. A second approach by Fonseca and Fleming (1996) uses a Kolmogorov-Smirnov like test, to determine whether the two distributions along the attainment functions from the two samples originate from the same distribution or not.

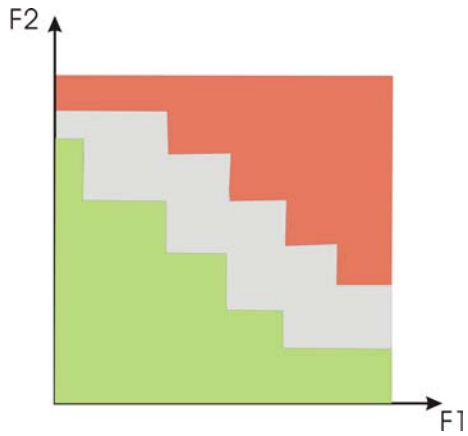


Figure 7.3: When the multiple objective genetic algorithm is run for several times, the search space can be divided in three areas. For a maximisation problem, the green region marks the area that is dominated by the Pareto-fronts of all runs and the red area bounds the region that is never dominated by any of the runs. The grey region is reached by some Pareto-fronts but not by all of them. The upper boundary of the green region is the 100% attainment surface, the lower boundary of the red region is the 0% attainment surface

7.5 Conclusion

Because there is no consensus on the use of performance indices, and because they can only highlight certain qualities of multiple objective genetic algorithms, it was decided to use all of them combined. In this way, the general performance of an algorithm can be gauged. It should be clear however that this general performance cannot give strong statements such as algorithm A is better than B and the statistical analysis is only performed per indicator separately. Many of the measures require that the Pareto-optimal front is known and cannot be used for the real-world case studies. The complete set of indices will however be used for the initial comparative study.

Chapter 8

Multiple objective genetic algorithms for forest management: a comparative study

8.1 Research rationale

Multiple objective genetic algorithms have not been used in forest management yet. Therefore there is no information available on which algorithms perform well for this type of problem. In order to get this information, a comparative study was conducted on a forest management problem with a known Pareto-optimal front. Two multiple objective algorithms were tested: the Multiple Objective Genetic Algorithm (MOGA) implemented by Fonseca and Fleming (1993) because this has proven its merit in a land use planning problem (Matthews et al., 1999, 2000) and the Non-Dominated Sorting Algorithm II (NSGA-II) (Deb et al., 2000, 2002) because of its reported efficiency. These two algorithms are also recommended by Van Veldhuizen and Lamont (2000) as starting points. The outcome of these were compared with a random search strategy to determine whether the genetic operators were efficient.

8.2 Introduction

Because the Pareto-optimal front is unknown for the harvest scheduling problem, a forest management problem defined by Gong (1992) was chosen as forest benchmark problem (Ducheyne et al., 2001).

For this benchmark problem, the first objective is to maximise the harvest volume V. Since harvest volume is negatively correlated to the standing volume left in the forest, it is possible to write the first objective as in Eq. 8.2:

Maximise

$$f_1 = \sum_{i=1}^{i=N} v(i,a) \tag{8.1}$$

$$\sim \frac{1}{v_{stand}} \tag{8.2}$$

where

$v(i,a)$ = harvest assoc. with stand i and decision a
v_{stand} = volume left standing
N = number of forest stands

The second objective is to maximise the benefit that people obtain from the standing forest, measured by a utility function U. According to the law of diminishing returns, this function can be modelled using the square root of the standing volume (Gong, 1992). The more standing volume left in the forest, the more trees present for people to enjoy. However, the increase in benefit derived from the trees will decrease when the forested area is larger as the marginal gain of leaving an extra tree declines. Therefore the second objective can be written as in Eq. 8.3:

Maximise

$$f_2 = \sum_{i=1}^{i=N} u_i \tag{8.3}$$

$$\sim \sqrt{v_{stand}} \tag{8.4}$$

where

u_i = utility assoc. with stand i
v_{stand} = volume left standing
N = number of forest stands

As follows from Eqs. 8.2 and 8.3, the two objectives are conflicting. Moreover, the Pareto-front between the two objectives is non-convex (Fig. 8.1). This prohibits the use of weighted sum formulations and Gong (1992) reverted to a complex dynamic programming formulation in order to solve it.

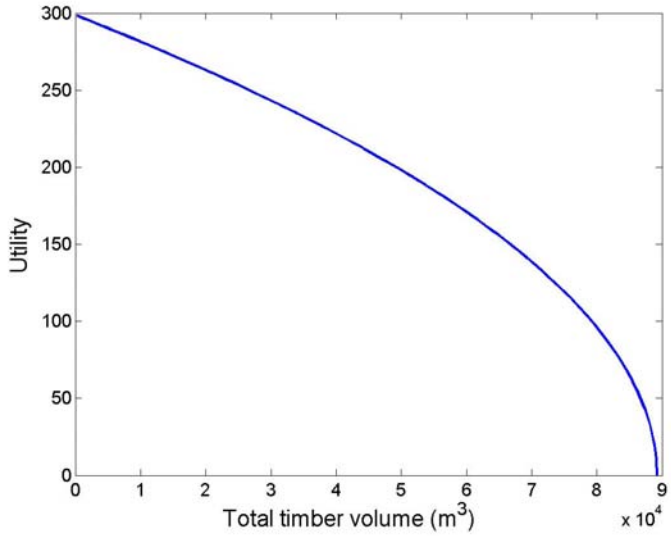


Figure 8.1: The Pareto-optimal front: the front is non-convex and cannot be found using the weighted sum approach

8.3 Methodology

There are four different functions for this benchmark problem that can be assigned to the forest: an economic function which involves clear felling and habitat conservation, passive and active recreation which involves leaving the stands. Each stand can receive only one set of management activities over the complete planning horizon. The functions are mapped using two bits per forest stand. 295 stands from the 399 stands were retained, the other stands were excluded because they were either unplanted or not yet productive during the initial period. A simple land block assignment procedure as in Matthews et al. (2000) was applied. The different algorithms (MOGA, NSGA-II and the random search strategy) were all programmed in Java. The Java-documentation files are included in Appendix B.

For the two genetic algorithms, the population size P, the number of generations T, crossover rate for uniform crossover p_c and mutation rate p_m were fixed: $P = 100$, $T = 50$, $p_c = 0.8$ and $p_m = 0.01$. All algorithms were repeated 30 times and for each run the error ratio, generational distance, spacing, spread and hypervolume measure were determined (Section 7.4.1 p. 89). These measures were implemented in MATLAB 6.5 (Appendix C) and were analysed using either the One Way ANOVA or the Kruskal-Wallis statistical tests (both in SPSS) and the bootstrapping method, which was implemented in MATLAB

(Appendix C). The significance level for all statistical tests was set to 95%. The 50% attainment surface was derived in MATLAB (Appendix C) for visual comparison and this is also used as input for the Mann-Whitney test statistics provided by Knowles and Corne (2000).

8.4 Results and discussion

8.4.1 Visual interpretation

The following median attainment surfaces were obtained for MOGA and NSGA-II; for the random simulation the median attainment surfaces is not presented because this is too small, therefore the solutions found by the random simulations over the 30 runs are presented (Fig. 8.2).

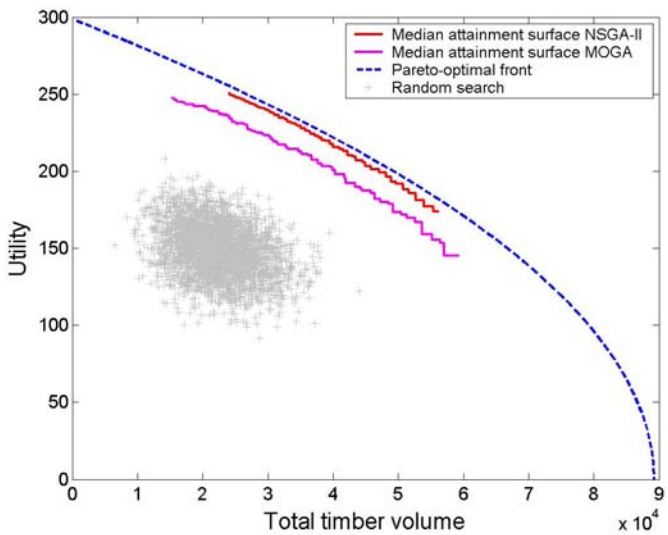


Figure 8.2: Comparison of the median attainment surface non-dominated front between random search, MOGA and NSGA-II. Both algorithms outperform the random strategy. NSGA-II approached the Pareto-optimal set more closely than MOGA, but MOGA is better at maintaining spread along the Pareto-front

Fig. 8.2 shows that both MOGA and NSGA-II perform much better than the random search strategy. NSGA-II approached the Pareto-optimal front better than MOGA. MOGA on the other hand is capable of finding more extreme solutions and this results in better spread along the Pareto-front. None of the algorithms are capable of finding the extreme solutions. Because the apparent

lack in spread might be caused by implementation errors in the crowding distance operator (NSGA-II) or sharing function (MOGA), the algorithms were tested on a non-convex test function provided by Zitzler (1999), Zitzler et al. (2000). Test function 2 is a three objective non-convex function:

$$f_1(x_1) = x_1$$

$$g_1(x_2, \ldots, x_n) = 1 + \frac{9 \cdot (\sum_{i=2}^{n} x_i)}{n-1} \tag{8.5}$$

$$h_2(f_1, g_1) = 1 - \sqrt{\frac{f_1}{g_1}} \tag{8.6}$$

where $n = 30$ and $x_i \in [0, 1]$. The Pareto-optimal front is formed when g_1 equals 1. In Zitzler (1999), Zitzler et al. (2000) a comparison between different algorithms on this test function is made and this can be used to compare the results between the adapted implementations used in this work with the performance of the other implementations.

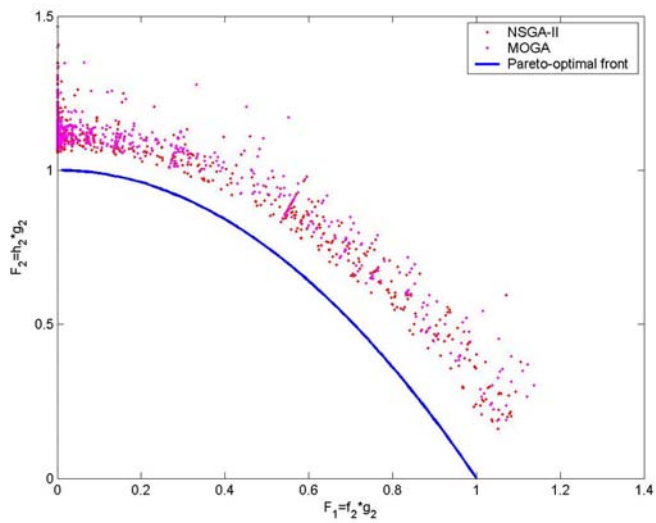


Figure 8.3: Performance of MOGA and NSGA-II for a non-convex test function. In this case all extreme solutions are found, indicating good implementation of the two algorithms

From Fig. 8.3 follows that both algorithms have a good spread and they approximate the Pareto-optimal front well for the non-convex test function. The performance of both algorithms on the test function is similar. The lack of spread in the forest management problem is therefore not caused by implementation errors, but is most likely caused by the discreteness of the problem.

MOGA can handle this discreteness better than NSGA-II in terms of spread but NSGA-II is capable of approximating the Pareto-optimal front quicker.

8.4.2 Performance indices

8.4.2.1 Testing closeness to the Pareto-optimal front

The generational distance (GD) and the δ-error ratio, with $\delta = 0.05$, were calculated for both genetic algorithms. The output results were normalised using the respective minimum and maximum values in each objective dimension in the Pareto-optimal front because both error ratio and generational distance are scaling dependent: the difference in magnitude between the objectives disregards the effect of the objective with the lowest magnitude. The means of error ratio and generational distance as well as their standard deviations are listed in Table 8.1 . In Fig. 8.4 the result of the bootstrapping method is shown.

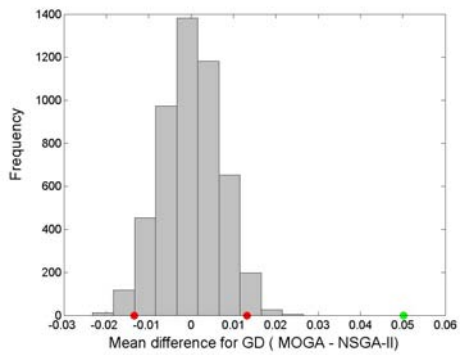
Table 8.1: The mean error ratio with $\delta = 0.05$, generational distance and the standard deviation over 30 runs for MOGA and NSGA-II

	MOGA	NSGA-II
GD	0.066 ± 0.009	0.016 ± 0.002
Error ratio	0.8921 ± 0.01	0.003 ± 0.011

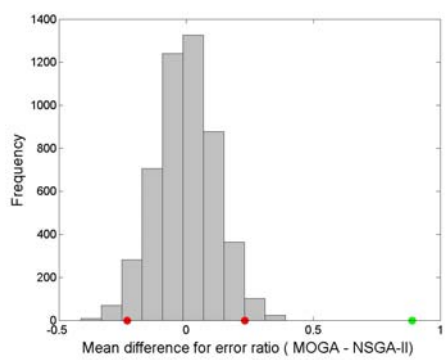
The data is not normally distributed for the error ratio ($p = 0.01 < 0.05$) and for both generational distance and error ratio, the assumption of equal variances is also not fulfilled (generational distance: $p < 0.05$ and error ratio: $p = 0 < 0.05$). Therefore a non-parametric Kruskal-Wallis test in combination with the bootstrapping method is applied for the statistical analysis. From both the Kruskal-Wallis test (Table 8.2) ($p = 0.0 < 0.05$) as well as the bootstrapping method (Fig. 8.4) clearly follows that NSGA-II performs better than MOGA at a confidence level of 95%. In Fig. 8.4 the test measure is positive, indicating that the mean generational distance as well as mean error ratio is higher for MOGA than for NSGA-II. As for these two test indices lower values are better, it follows that NSGA-II performs better in terms of closeness to the Pareto-optimal front. The standard deviation of the generational distance is smaller for the NSGA-II than for MOGA (Table 8.1), and this can be interpreted as a more robust behaviour of the NSGA-II algorithm.

Table 8.2: Kruskal-Wallis statistics for generational distance and error ratio over 30 repetitions for MOGA and NSGA-II

	MOGA	NSGA-II
GD	45.50	15.50
Error ratio	45.50	15.50



(a) Generational distance



(b) Error ratio

Figure 8.4: Bootstrapping results for the error ratio (Fig. 8.4(b)) and the generational distance (Fig. 8.4(a)). The confidence interval boundaries are marked in red ($\alpha = 95\%$), the test measure is marked in green. Both test indices are outside the boundaries.

8.4.2.2 Testing spread

The spread was measured by the spacing measure and by the spread measure. The spacing from NSGA-II is lower than the spacing from MOGA indicating that the crowding distance function spreads the solutions better than the sharing function (Table 8.3).

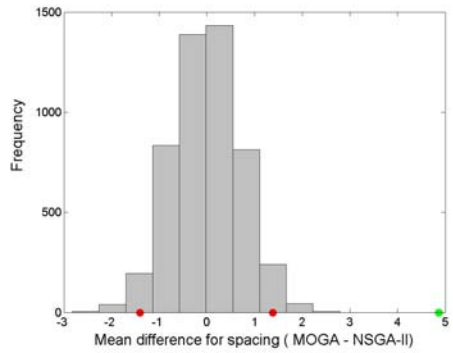
Table 8.3: Mean spacing and spread for MOGA and NSGA-II over 30 runs. A low value of spacing and spread indicates evenly spaced solutions

	MOGA	NSGA-II
spacing	18.37 ± 1.61	13.50 ± 0.297
spread	0.502 ± 0.053	0.525 ± 0.04

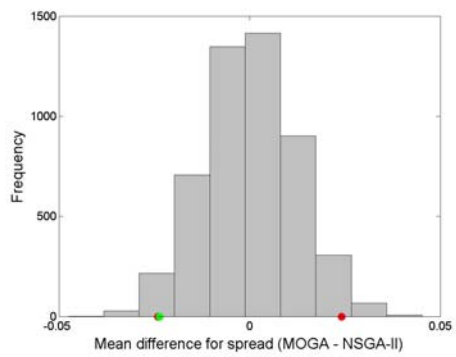
For spacing and spread the variances were not equal ($p = 0.001 < 0.05$), therefore the Kruskal-Wallis-test procedure was used (Table 8.4). The means for spacing are significantly different at a 95% level ($p = 0.009 < 0.05$) and this is confirmed using the bootstrapping procedure (Fig. 8.5). The test indices for the spacing are in the 5% tails of the histogram. Once more lower values are better, therefore NSGA-II has more evenly spaced solutions than MOGA. If the distance to the most extreme solutions is included as in the spread measure by Deb et al. (2000), however, MOGA has a better performance because it can reach the extremes better. From the bootstrapping results (Fig. 8.5), it follows that there is no difference between the two algorithms, but as noted before, the test value is just inside the boundaries and the boundaries differ between two different bootstrapping procedures. The Kruskal-Wallis test (Table 8.4) cannot detect any significant difference at a level of 95% ($p = 0.095 > 0.05$).

Table 8.4: Kruskal-Wallis statistics for spacing and spread over 30 repetitions for MOGA and NSGA-II

	mean rank MOGA	mean rank NSGA-II
spacing	15.50	45.50
spread	34.27	26.73



(a) Spacing



(b) Spread

Figure 8.5: Bootstrapping results for spacing (Fig. 8.5(a)) and spread (Fig. 8.5(b)). The confidence interval boundaries are marked in red ($\alpha = 95\%$), the test measure is marked in green. The test measure for spacing is outside the confidence interval boundaries, the test measure for spread is just within these boundaries

8.4.2.3 Combining spread and closeness

The hypervolume measure calculates the size of the dominated space and is a combined measure for both spread and closeness. For this measure the data was not normalised as it is scaling independent. The mean hypervolume indices are listed in Table 8.5

Table 8.5: Mean hypervolume for MOGA, NSGA-II and random search over 30 runs

Algorithm	$\mathcal{S}$
MOGA	13491904·43
NSGA-II	13998229·19
random search	7339239·14

Both normality ($p > 0.05$ for all groups) and homogeneity of variances ($p = 0.426 > 0.05$) assumptions are fulfilled and from statistical analysis (One Way ANOVA) follows that NSGA-II is significantly better than MOGA at a 95% level, and that both genetic algorithms are significantly better than the random strategy. From the bootstrapping method the same conclusion can be drawn when comparing MOGA and NSGA-II (at a 95% level).

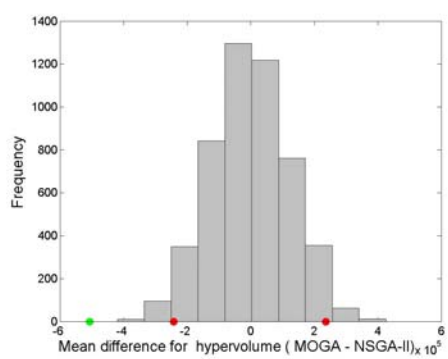


Figure 8.6: Results from the bootstrapping method. In the x-axis, the mean differences are represented, on the y-axis the frequency counts. In red, the confidence interval of 95% is indicated, in green the test measure is marked.

8.4.3 Statistical approaches

The median attainment surfaces for MOGA and NSGA-II have already been represented in Fig. 8.2. These surfaces can be used as input for statistical comparison. Knowles and Corne (2000) provided a test measure based on these attainment surfaces showing where algorithm A outperforms B and vice versa.

Table 8.6: Test statistics based on the comparison of the median attainment surface from MOGA and NSGA-II. The test statistics show the part of the search space where the NSGA-II and MOGA are not dominated by any algorithm and the part where they are dominated by the another algorithm

	MOGA	NSGA-II
non-dominated	20.6	83.6
dominates	16.4	79.4

From this measure (Table 8.6), it follows that NSGA-II dominates MOGA in 83.6% of the covered search space and that MOGA dominates NSGA-II in 20.6% of the cases. These statistics can be explained because MOGA reaches the extreme solutions better than NSGA-II does and therefore MOGA beats NSGA-II in part of the search space. In the largest part of the search space the solutions from MOGA are dominated by NSGA-II because NSGA-II is closer to the Pareto-optimal front.

8.5 Conclusion for the forest management problem

Both MOGA and NSGA-II have shown a better performance than a random search strategy. They both approximate the Pareto-optimal front well, but suffer from a lack of spread. Especially the NSGA-II is not capable of finding the more extreme solutions. This lack of spread, however, is not caused by any implementation errors: both algorithms have a very good spread over the complete Pareto-front for a non-convex test function commonly used as reference function in GA literature.
NSGA-II is capable of approximating the Pareto-optimal front faster than MOGA and has more evenly spaced solutions. If the distance from the extreme solutions in the Pareto-front to the extremes of the Pareto-optimal front are included in the spread measure, MOGA scores better than NSGA-II. However, this is not significant. The variance between the several runs both in generational distance is smaller for NSGA-II than for MOGA, highlighting that NSGA-II is more robust than MOGA in terms of approximation of the Pareto-optimal front.
When the algorithms are compared in terms of both spread and closeness, the hypervolume measures indicates that the NSGA-II dominates a higher portion of the solution space than MOGA does.
Using the attainment surfaces similar conclusions can be drawn: the Mann-Whitney test procedure shows that NSGA-II beats MOGA in the larger portion of the search space.
Overall, the NSGA-II algorithm shows a better performance for the forest management problem and therefore this algorithm will be used in the subsequent case studies.

Chapter 9

Case study: solving a harvest scheduling problem as a bi-objective problem

9.1 Introduction

The harvest scheduling problem as defined in the single objective case will now be solved using the multiple objective genetic algorithm NSGA-II. The original objective functions (Eqs. 5.1 and 5.2) are the direct input for the genetic algorithm and do not have to be combined in any way beforehand. The same data and production forecast models as in the single objective case are used to allow for comparison of both approaches.

9.2 Methodology

As for the single objective case, the effect of encoding was investigated. Next to looking the approximation of the Pareto-optimal front, the spacing of the solutions along the Pareto-front was closely examined. Again binary, gray and integer encodings were used to represent the eight different cutting periods per management unit. The effect of the encoding strategies was inspected visually as well as using the hypervolume measure and the statistical analysis via the attainment surfaces. Other indices for closeness could not be applied because the Pareto-optimal front is unknown. The spacing measure was also used. Later on, the population size was increased from 500 up to 1000 individuals in steps of 250. For each of these population sizes, the effect on convergence and spread was determined. For each encoding type and population size, the experiment was repeated 10 times.

The multiple objective genetic algorithm with the best encoding strategy was then also run for 50 generations and a population size of 100, using the same parameters as for the single objective problem. The outcome of the single objective and multiple objective genetic algorithms was compared. For all experiments binary tournament selection with the non-dominance selection criterion was used, together with one-point crossover with a crossover probability of 0.8 and uniform mutation with a probability of 0.01. Once more the elitist strategy was applied.

9.3 Results and discussion

9.3.1 Effect of encoding on spread and Pareto-optimality

Visual interpretation Integer encoding proves to be the best encoding strategy in terms of approximating the Pareto-front (Fig. 9.1), but again gray encoding is a very close second. The three encodings show a similar spread.

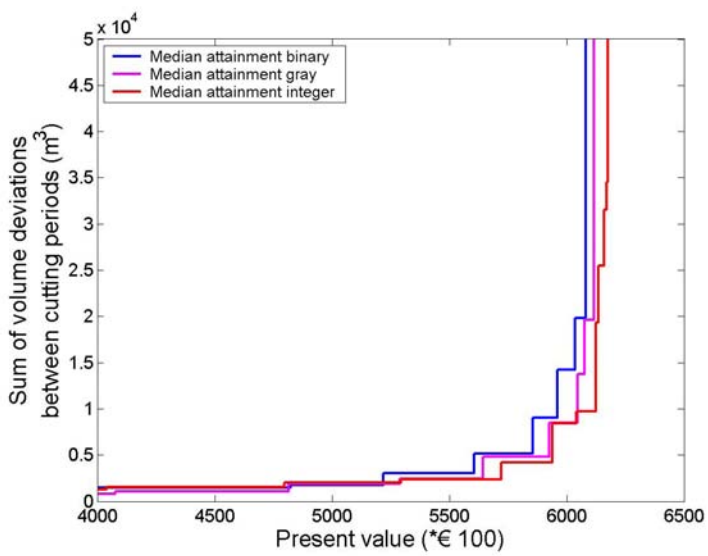


Figure 9.1: Median attainment surfaces for binary, gray and integer encoding

Performance indices The performance of the integer encoding is confirmed by the spacing measure and by the hypervolume measure (Table 9.1). The data for the spacing measure is not normally distributed ($p_b = 0.010 < 0.05$, $p_g = 0.185 > 0.05$ and $p_i = 0.01 < 0.05$). Therefore the Kruskal-Wallis test was used as test procedure next to the randomised testing procedure. For the

Table 9.1: Mean hypervolume and spacing measure for binary, gray, and integer encodings. The results are averaged over 10 runs.

	binary	gray	integer
hypervolume	6.1e10	6.1e10	6.2e10
spacing	1603.80	431.41	485.42

hypervolume measure the data is normally distributed ($p_b = 0.576 > 0.05$, $p_g = 0.728 > 0.05$ and $p_i = 0.823 > 0.05$) and the variances are equal ($p = 0.784 > 0.05$) and for the hypervolume measure a One Way ANOVA test was applied together with the randomised testing procedure.

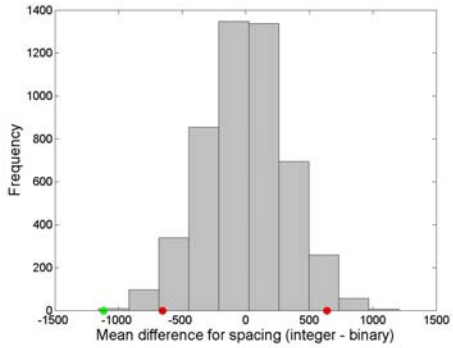
There is a significant difference between the groups for the spacing measure (Table 9.2). Again gray and integer codes score best and both are significantly better than binary codes according to a non-parametric post-hoc test. The bootstrapping test procedure confirms this, in both Fig. 9.2(a) and Fig. 9.2(c) the test value is outside the confidence intervals indicating a significant difference between integer and binary codes and between gray and binary codes. In Fig. 9.2(b) the test value is within the boundaries of the interval showing that there is no difference between gray and integer codes.

The One Way ANOVA test statistic for the hypervolume measure (Table 9.2) indicates that there are no significant differences ($p = 0.656 > 0.05$) and this is also confirmed by the bootstrapping results (Figs. 9.3(a) to 9.3(c)).

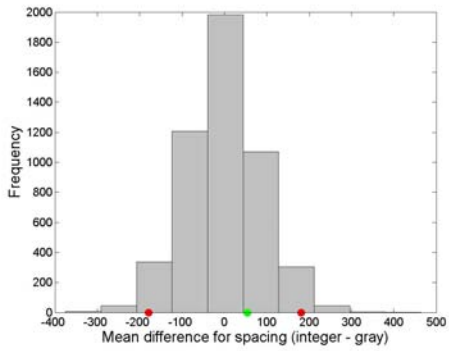
Integer codes will be used to solve the harvest scheduling problem because they are the most natural representation for the problem.

Table 9.2: Kruskal-Wallis ranks for spacing measure over 10 repetitions for binary, gray and integer encoding

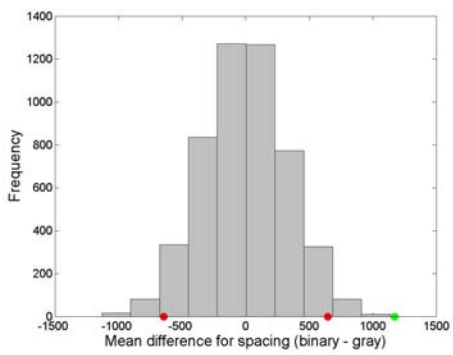
	binary	gray	integer
spacing	24.50	12.20	9.80



(a) integer - binary

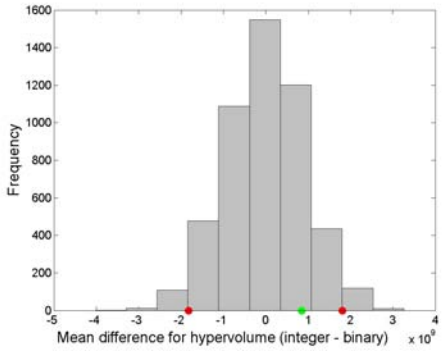


(b) integer - gray

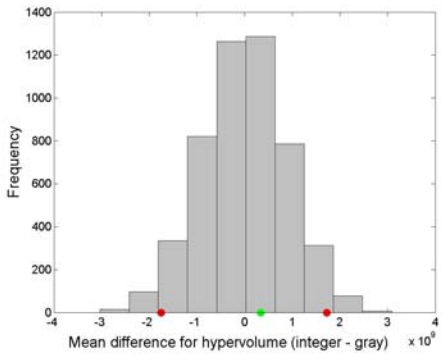


(c) binary - gray

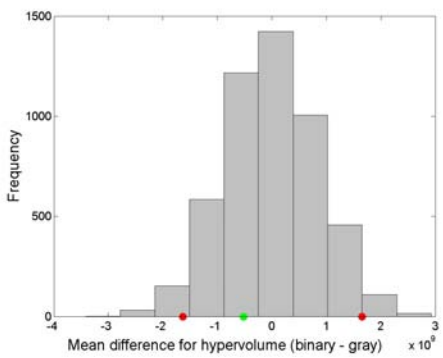
Figure 9.2: Bootstrapping results for the difference in mean spacing for integer, binary and gray encodings. In Fig. 9.2(a) the difference between integer and binary encoding, Fig. 9.2(b) between integer and gray encoding and Fig. 9.2(c) between binary and gray encoding



(a) integer - binary



(b) integer - gray



(c) binary - gray

Figure 9.3: Bootstrapping results for the difference in mean hypervolume for integer, binary and gray encodings. In Fig. 9.3(a) the difference between integer and binary encoding, Fig. 9.3(b) between integer and gray encoding and Fig. 9.3(c) between binary and gray encoding

9.3.2 Effect of population size on solution quality

Visual interpretation In a last phase the effect of the population size on solution quality as well as spread was investigated. The population size was increased from 500 to 1000 in steps of 250. This results in the following median attainment surfaces (Fig. 9.4).

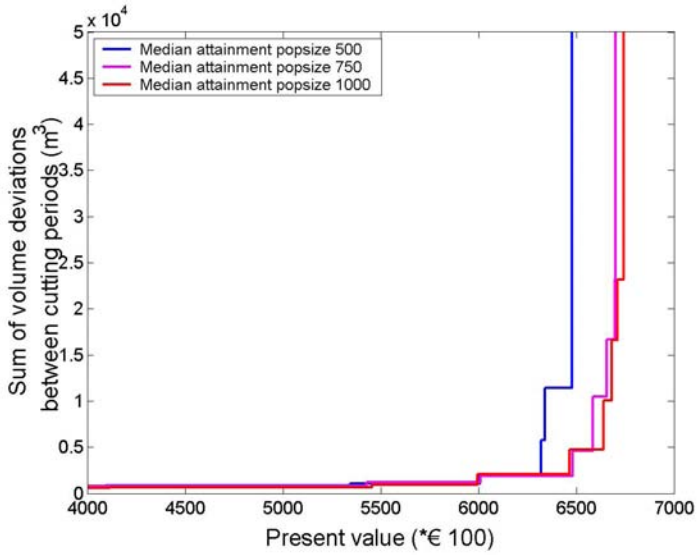


Figure 9.4: Median attainment surfaces for population sizes 500, 750, and 1000 over 10 runs

The median attainment surfaces for the three population sizes are very similar. They approximate the Pareto-optimal front the same and the spread of the solutions along the attainment surface is even along the Paret-front. The fact that they approximate the same front indicates that they are very close to the (unknown) Pareto-optimal front.

Performance indices The spacing and hypervolume measure are determined for the different population sizes. The mean values are listed in Table 9.3. The mean value for the hypervolume measure is almost the same for the three population sizes, the spacing along the Pareto-front is also very similar across the different population sizes.

Table 9.3: Mean spacing and hypervolume measure for population sizes 500, 750 and 1000. The results are averaged over 10 runs.

	500	750	1000
spacing	2853.04	1865.74	3041.27
hypervolume	6.5e10	6.7e10	6.7e10

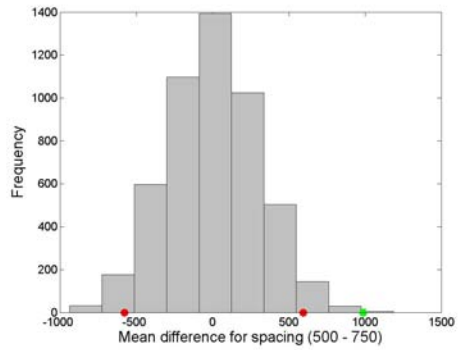
For the spacing measure the data is normally distributed ($p_{500} = 0.237$, $p_{750} = 0.625$ and $p_{1000} = 0.394$). The data is not homoscedastic and therefore the Kruskal-Wallis procedure is applied. From the test value ($p = 0.0 > 0.05$), it follows that there are significant differences. These differences are found, according to a non-parametric posthoc-test, between the population size of 750 and the population sizes of 500 and 1000.

A One Way ANOVA test can be used to test whether the means of the hypervolume measure are equal or not for the three population sizes, because the assumptions of normality ($p > 0.05$ for all groups) as well as homogeneity of variance ($p = 0.085 > 0.05$) are fulfilled. The statistical analysis shows that there is significant difference between the different population sizes for the hypervolume measure ($p = 0.001 > 0.05$) and according to Tukey's posthoc-test this is between the population size of 500 on the one hand and the population sizes of 750 and 1000 on the other hand.

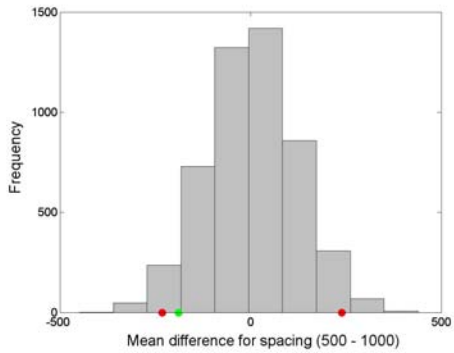
A population size of 750 is sufficiently large enough to solve the harvest scheduling problem.

Table 9.4: Mean Kruskal-Wallis ranks for spacing over 10 repetitions for population sizes 500, 750 and 1000

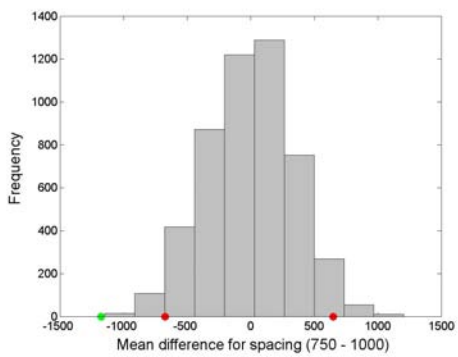
	500	750	1000
spacing	18.30	5.90	22.30



(a) population sizes 500 - 750

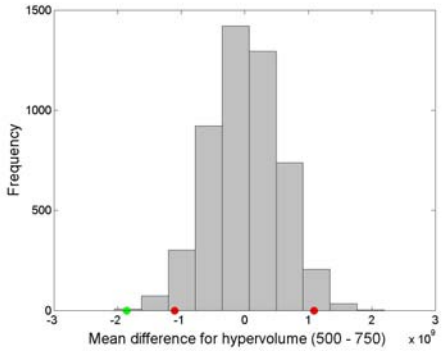


(b) population sizes 500 - 1000

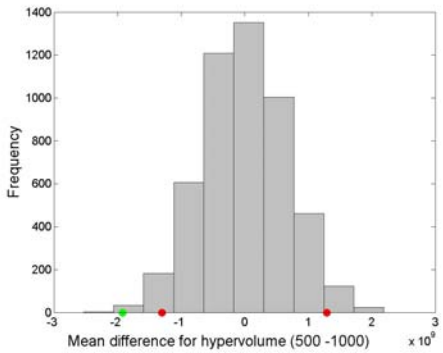


(c) population sizes 750 - 1000

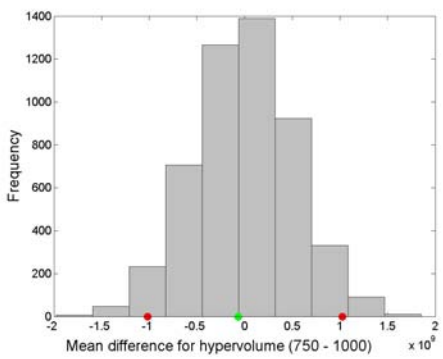
Figure 9.5: Bootstrapping results for the difference in mean spacing for population sizes 500, 750 and 1000. In Fig. 9.5(a) the difference between population sizes 500 and 750, Fig. 9.5(b) between population sizes 500 and 1000 and Fig. 9.5(c) between population sizes 750 and 1000



(a) population sizes 500 - 750



(b) population sizes 500 - 1000



(c) population sizes 750 - 1000

Figure 9.6: Bootstrapping results for the difference in mean hypervolume for population sizes 500, 750 and 1000. In Fig. 9.6(a) the difference between population sizes 500 and 750, Fig. 9.6(b) between population sizes 500 and 1000 and Fig. 9.6(c) between population sizes 750 and 1000

9.3.3 Comparing the single and multiple objective genetic algorithm

Running NSGA-II with integer encoding and a population size of 50 enables comparising the results of the single objective and the multiple objective optimiser. The multiple objective genetic algorithm was also run for the same number of generations. Overlay of the median attainment surface from the multiple objective optimisation runs with the median attainment from the single objective optimisation is depicted in Fig. 9.7.

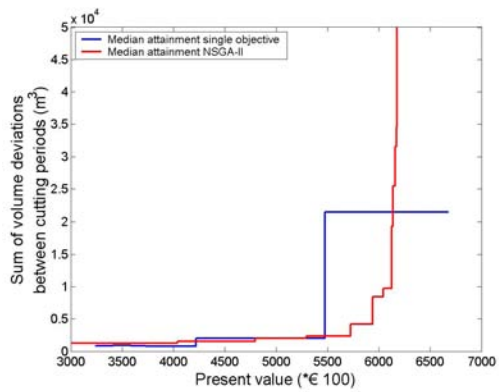


Figure 9.7: Overlay of the median attainment surface found with the single objective optimiser and the best solutions obtained with a multiple objective genetic algorithm with a population size of 50. On the x-axis the present value (*€ 100) and on the y-axis the sum of deviations in volume (m^3)

The two median attainment surfaces are very similar. Only the most extreme solution is missing from the Pareto-front found by NSGA-II. Running the multiple objective genetic algorithm has particular benefits in terms of computer efficiency. For both algorithms the same population size and number of generations was chosen. The product of population size and number of generations yields the number of function evaluations. In the case of the single objective optimiser, this total number should be multiplied by five as the genetic algorithm has to be run five times to get points along the Pareto-front. This results in $50 * 100 * 5$ function evaluations. For the multiple objective genetic algorithm, with the same population size and number of generations, the number of function evaluations is only one fifth of the total number of function evaluations needed for the single objective optimiser.

Running NSGA-II with a population size of 750, which proved to be a good population size in the previous section, and for 50 generations yields the following Pareto-front (Fig. 9.8). The maximum present value that is attained in

50% of the repetitions amounts to € 670300 (73,3% of the maximum attainable present value) and has a total sum of volume deviations of 398991 m^3. For a weight of 0.01 this was € 667435 (73% of the maximum attainable present value) and a total sum of volume deviations of 21535.5 m^3. The median values are similar for the single and multiple objective optimiser.

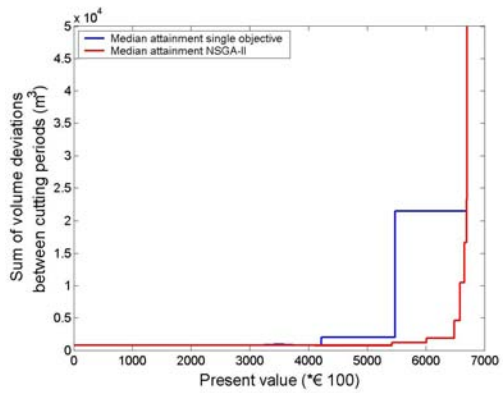


Figure 9.8: Overlay of the best solutions found with the single objective optimiser and the solution front obtained with a multiple objective genetic algorithm with a population size of 750. On the x-axis the present value (*€ 100) and on the y-axis the sum of deviations in volume (m^3)

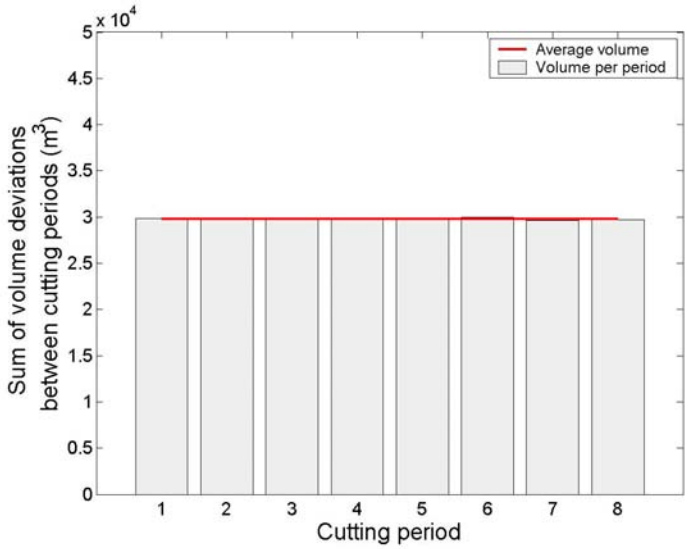
9.3.4 Validity of the plans

Two plans will be investigated more closely as to their validity: the harvest schedule plan with the most strict even-flow objective and the plan with the best present value. The objective values for the two plans are listed in (Table 9.5). The volume per period and the harvest pattern in the forest illustrated in Fig. 9.11(a) and Fig. 9.11(b).

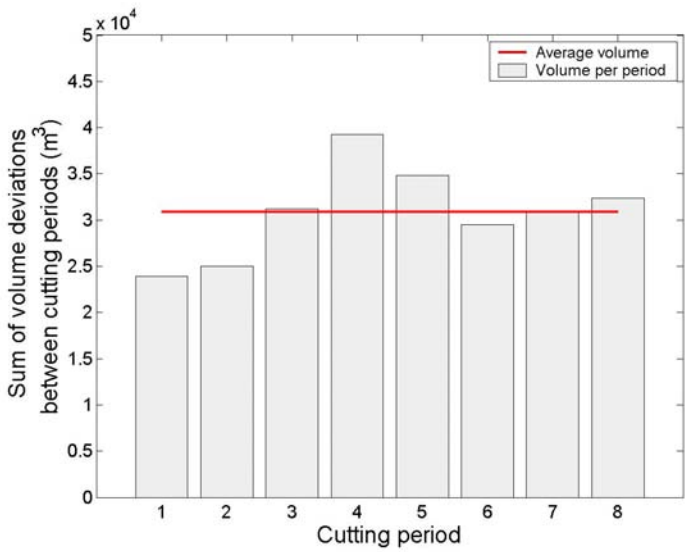
Table 9.5: The present value PV and the average volume $\overline{V}$ over all cutting periods for the best even-flow harvest plan, the compromise plan and the best present value plan

PV (*€ 100)	$\sum_{i=1}^{n} V_i - \overline{V}$ (m^3)
5878	600
6851	28514

From Figs. 9.11(a) and 9.11(b) a conclusion similar to that of the single objective case follows: the age distribution is forced by the proposed harvest plans towards a normal age distribution. If the even-flow objective becomes more important this effect is stronger than when the present value objective is more important. Again the average volume that is attained with the relaxed even-flow objective (6.80 m^3/ha/yr) is higher than when the objective of even-flow becomes more important (6.56 m^3/ha/yr). From the harvest pattern, it follows that in order to get a better present value, more stands are scheduled for cutting in the later planning periods than when the even-flow objective is important. From the detailed Pareto-front follows that there is a very narrow range where low deviations from the average volume can be obtained. This shows that forest managers need to design their plans very carefully so as to avoid too large deviations, and that the range is limited.

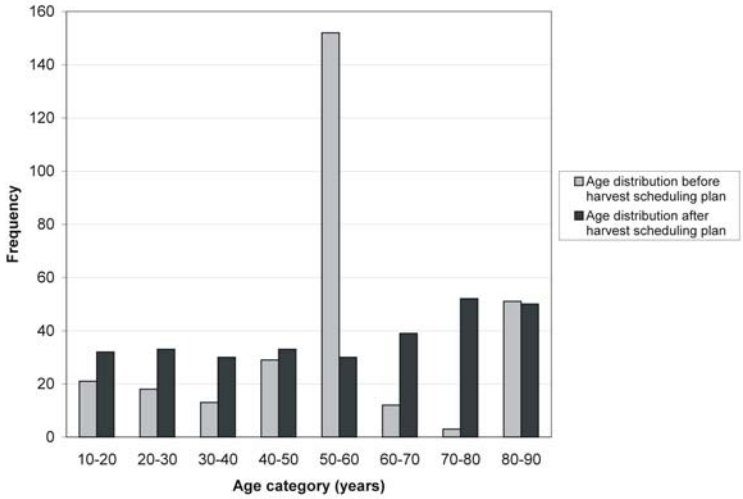


(a) Even flow

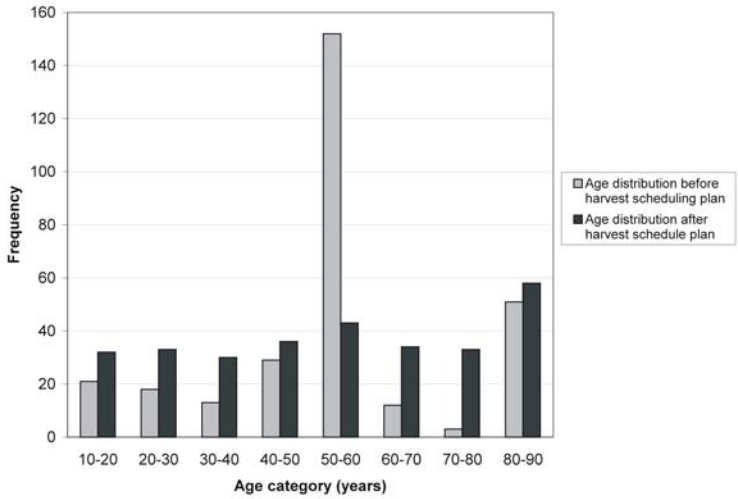


(b) No even-flow

Figure 9.9: The variation in total deviation in volume (m^3) between the different cutting periods. From Fig. 9.9(a) to Fig. 9.9(b), the even-flow constraint is strengthened.

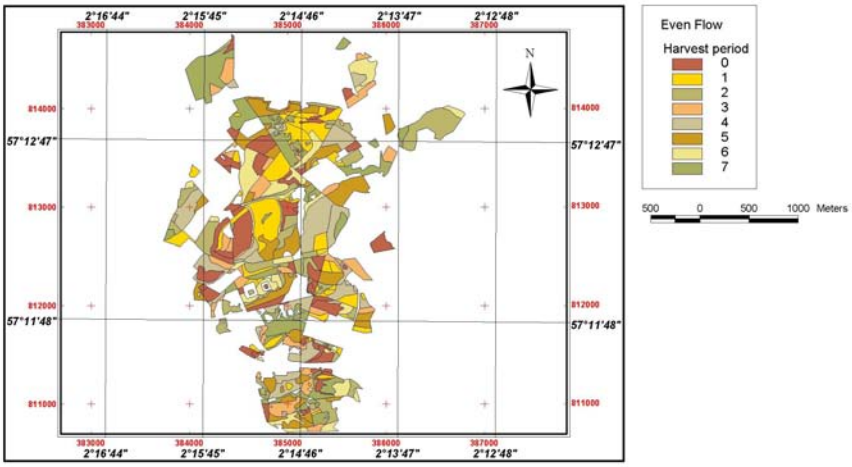


(a) Even flow

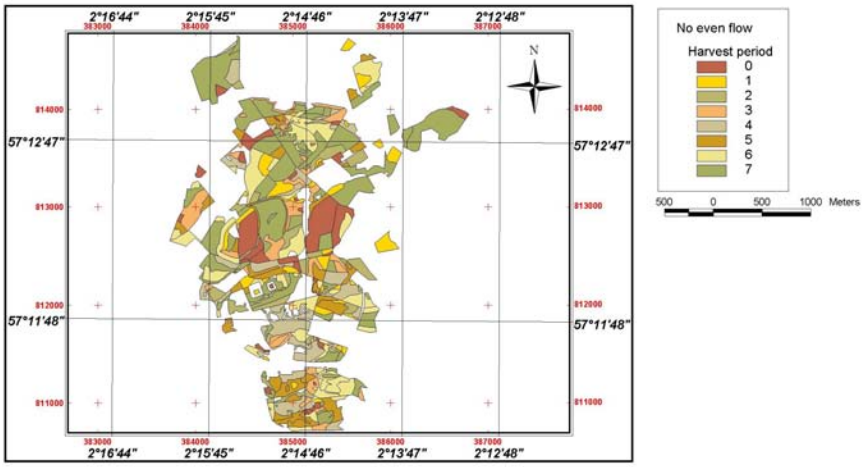


(b) No even-flow

Figure 9.10: The effect of the even-flow objective on the age structure. From 9.10(a) to 9.10(b), the even-flow constraint is strengthened.



(a) Even flow



(b) No even-flow

Figure 9.11: The effect of the even-flow objective on the harvest pattern. From 9.11(a) to 9.11(b), the even-flow constraint is strengthened.

9.3.5 Conclusion

The encoding strategy is important in terms of approximation of the Pareto-front. The best encoding strategies are gray and integer encoding. As is suggested in literature, binary encoding does not perform very well. There is an effect if the population size is increased from 500 to 750, the Pareto-optimal front is approximated more closely. This effect is no longer present when the size increases even more to 1000.
Using multiple objective genetic algorithms instead of single objective genetic algorithms to solve the harvest scheduling problem speeds up the optimisation process: in order to find solutions linearly distributed along the Pareto-front a single run suffices. For both optimisers the effect of the plans on the age structure of the forest is the same: if the even-flow objective becomes more and more important, the age structure resembles that of a normal forest due to the volume control, even though this is not explicitly mentioned in the objective functions. If the even-flow objective is relaxed, the stands are scheduled in later planning periods than when the even-flow objective is very important. Finally, the Pareto-front is very steep, indicating that forest managers have to design their plans carefully to meet their objectives.

Chapter 10

Case study: solving a multiple objective problem using GAs and GIS

10.1 Research rationale

In the single objective case, the abundance of edge-dependent species was maximised using explicitly formulated spatial constraints. These equations were created manually and the spatial modelling was performed without a geographical information system. This is hardly feasible for large problems and creating the non-linear constraints becomes impossible if the forest stands are in an irregular pattern. As Hof and Joyce (1993) state, the shortcomings of their approach is that (1) global optima cannot be assured and (2) only relatively small problems can be solved.

In the following case study, the multiple objective problem as defined by Hof and Joyce (1992, 1993) is solved using a multiple objective genetic algorithm in combination with a GIS. This addresses the second shortcoming of the former approach but does not guarantee global optima. The combination of the optimiser and the GIS fulfils the requirements for a Spatial Decision Support System (SDSS): the multiple objective genetic algorithm generates alternatives for the decision maker using spatial information and spatial modelling. These solutions help the decision maker to make well-founded decisions.

10.2 Linking genetic algorithms and GIS

From SDSS literature can be learned that an SDSS consists of several distinct modules (Armstrong & Densham, 1990; Seffino et al., 1999) such as the analysis and optimiser toolbox and the model base, usually the GIS. The integration of GA and GIS fits well within the framework of SDSS. Because the functionality of the genetic algorithm is clearly distinguished from the fitness evaluation, any model requiring spatial data can be evaluated directly inside a geographical information system. In order to do this the entities should be able to communicate with each other. The inter-operability is the ability of several components to communicate with each other through the exchange of data and services with one another (Twery et al., 2000). The coupling between the different entities can take place in the following ways (Goodchild, 1992; Ungerer & Goodchild, 2002):

- stand alone, which is not very efficient because in that case the functionality of the two systems cannot be exploited efficiently;
- loose coupling: the data exchange operates either through ASCII- or binary files. This has two strengths: (1) each task is tackled by the package that is best suited to solve the task and (2) there is no need to build new software. Its disadvantage is the dependence on file formats in the case of ASCII-files and the fact that it requires the loading of both the analysis/optimiser toolbox and the model base at the same time. This might cause hardware-related problems;
- close or deep coupling: The analysis/optimiser toolbox is called directly from the GIS. This has much potential but requires that the underlying data structures can be accessed and there is the extra need for code creation;
- full integration: this is the best way to work but requires complete openness of the GIS software and this is as yet not the case for most available commercial software.

Twery et al. (2000) mention that the existing forest ecosystems management decision support systems are largely monolithic structures without open general purpose communication and control standards that provide interoperability. Only very recently some GIS have allowed better communication through the Component Object Model protocol (COM-compliance) such as Idrisi32 and ArcGis8.1 and can be useful in the creation of a close coupled system (Ungerer & Goodchild, 2002).

In this dissertation, a loose coupling strategy (Fig. 10.1) is proposed because of its ease of implementation and because of the optimal task distribution. Moreover, this approach is platform-independent. When the software is developed using the COM-protocol, it has to use Visual Basic for Applications (VBA), and this inhibits the use of the developed software on operating systems such

as Linux, Unix or Macintosh. The solutions generated by the genetic algorithm were written as ASCII-files. The GIS was prompted to read the input files and the proposed management activity for each stand was linked through the relational key to the attribute table in the GIS. After the files had been linked, the GIS performed several scripts. While the GIS was running, the genetic algorithm waited for the response, repeatedly asking the GIS whether it was done or not. The geographic information system returned the different objective function values in an ASCII-file.

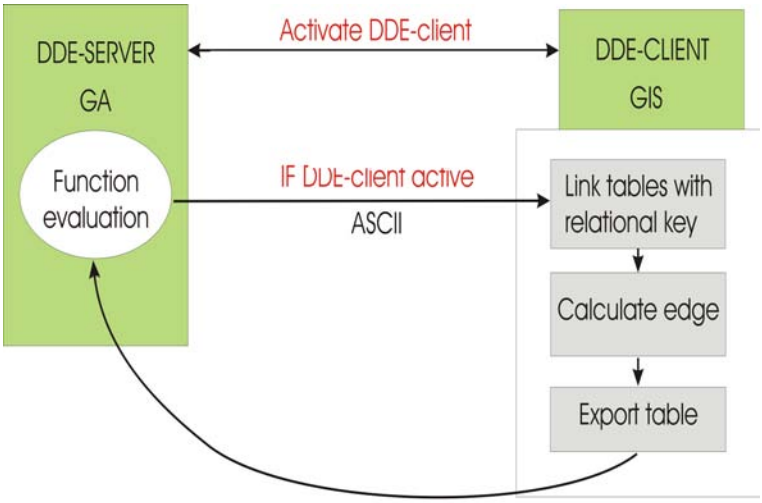


Figure 10.1: Loose coupling between the analysis/optimiser toolbox and the GIS. The data exchange is through ASCII-files

Originally the aim was to combine the genetic algorithms with ArcView. The reason for this was twofold:

- ArcView3.1/ArcGIS is the best known and most widely applied GIS in public agencies because of its user-friendliness and its possibility to customise the graphical user interface (Zhang & Griffith, 1997; Ducheyne, 1999)
- ArcView3.1 allows communication between two programs through the Dynamic Data Exchange protocol (DDE). This enables prompting ArcView to execute a script from another program

When ArcView was used numerous errors and broken connections between the modules occurred and in the initial stages of this research the genetic algorithm had to be restarted many times. This was also noted by Westra (2002). Therefore, it was opted in a later stage to use open source GRASS GIS (4.3;5.0)

(Neteler & Mitasova, 2002) as an alternative GIS. This free GIS runs on a Linux platform and is as customisable as ArcView through the use of tcltTk for GRASS. Because all the software was developed in Java and there was loose coupling, the genetic algorithm could be transferred to the Linux platform without any modification. Any GIS script could be called and executed immediately from the GA, as all source code and executables were directly available in GRASS GIS. A disadvantage for operational applications is the lower level of user-friendliness and that it is less well known.

10.3 A multiple objective spatial problem

10.3.1 The objective functions

As in Hof and Joyce (1992, 1993), timber volume, the abundance of edge-dependent species as well as the abundance of old growth species are maximised for a single time period. For simplification reasons, Hof and Joyce (1992, 1993) assumed in their application that the timber volume V per m^2 is $1m^3$, and this assumption is also made here for comparison purposes. The abundance of edge-dependent species E is linearly related to the length of the edge between clear felled stands and old growth stands (internal edge) and between agricultural fields and old growth stands (external edge). The abundance of the old growth species O is linearly related to the area of old growth forest. Two management activities are possible, clear felling stand i ($C_i = 0$) and leaving stand i in old growth ($C_i = 1$). The following objective functions are formulated for M stands with a size of l-by-l km (in a grid layout as in Chapter 6), with δ_{ij} a matrix where $\delta_{ij} = 1$ if j is a neighbour of i and $\delta_{ij} = 0$ if j is not a neighbour of stand i, and C_j is the management activity assigned to stand j:

$$\text{Maximise} \quad V = \sum_{i=1}^{M} 1 - C_i) \tag{10.1}$$

$$\text{Maximise} \quad E = \sum_{i=1}^{M} C_i \times (4 \times l) - l \times \left[\sum_{i=1}^{M} \left(\sum_{j=1}^{M} \delta_{ij} C_j \right) \right] \tag{10.2}$$

$$\text{Maximise} \quad O = \sum_{i=1}^{M} C_i \tag{10.3}$$

10.3.2 Evaluating fitness values in GIS

The area of each stand is fixed, and therefore it is much faster to use a lookup table to calculate the total area of clear felled and old growth stands instead of using time-consuming spatial analysis tools. The edge objective on the other hand is dynamic and has to be evaluated continuously in the GIS. The evaluation of the length of the edge was initiated from the genetic algorithm, which served

as DDE server. In DDE communication, there is a DDE server and a DDE client. The DDE server calls the client to check whether the client is running or not, and then sends a command towards it to activate the client. While the DDE client is working the DDE server checks for output. In this application, the genetic algorithm fired a main script in ArcView GIS, the DDE client, and this main script controlled the internal ArcView scripts (Fig. 10.1). The first step of the internal scripts linked the ASCII-file with proposed management activity for each stand, to the relational database in ArcView. This was followed by dissolving the boundaries between the stands that have the same management activity attribute and by the calculation of the perimeter. In a last phase, the result was exported as an ASCII-file and all objects that were created during the script were removed. These scripts were written in the macro language Avenue (ESRI, 1996) provided by ArcView, and were based on either the standard commands from ArcView itself or on freely available scripts from the ESRI forum (ESRI, 2003). When ArcView was activated as DDE client, it tended to time out before the complete command was sent through, and this was the main reason to use GRASS GIS for the spatial modelling.

In GRASS GIS a similar approach was followed: a main script controlled all steps that had to be performed by the GIS and was called from the GA. This initiated both the linkage of the tables and the perimeter calculation. In GRASS GIS, a set of landscape ecology measures were provided by Baker and Cai (1992) and these could be used to determine the perimeter between the clear felled and old growth stands.

10.3.3 Solving the benchmark problem

10.3.3.1 Methodology

NSGA-II was applied to solve the benchmark problem. Because the search space of the initial problem is small for a three-by-three grid (size of the search space = $2^9 = 512$), the population size was kept low ($P = 30$) and the maximum number of generations was set at 10. This means that 300 alternatives were evaluated and this is approximately half of the search space. A binary encoding strategy is sufficient to represent the problem using the land block representation. Later on, the size of the benchmark problem was increased from a three-by-three grid to a nine-by-nine grid making the problem harder to solve (size of the search space = 2^{81}). For the nine-by-nine grid the population size was increased up to $P = 100$ and the NSGA-II was run for 50 generations. For both problems the crossover probability was kept at 0.08 and the mutation rate at 0.01.

10.3.3.2 Results and discussion

The 3-by-3 grid Solving the problem on a three-by-three grid produces the following Pareto-front (Fig. 10.2). In Fig. 10.2, the solutions from the initial population as well as the solutions from the final population are presented.

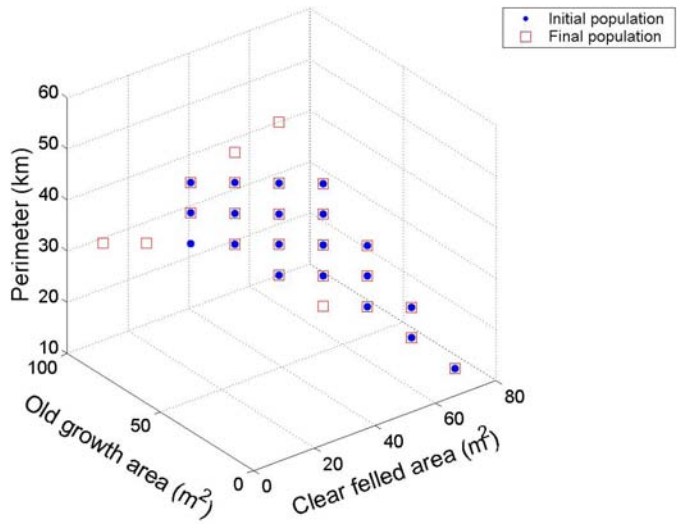


Figure 10.2: Pareto-front over 10 runs for the 3-by-3 grid. The initial population consists of almost all optimal solutions, and there is no effect of the genetic algorithm

Because the search space is so small, the population size was kept low but even at a low population size, many of the optimal solutions are already present in the initial population (Fig. 10.2). This means that the problem is too easy to solve with a genetic algorithm. Some optimal solutions that are missing from the initial population are found in later generations by the GA, but in general the problem is solved without exploiting the search capacity of the genetic algorithm.

The 9-by-9 grid Solving the nine-by-nine grid problem shows the advantages of the GA. In the initial population a smaller portion of the optimal solutions is present (Fig. 10.3) and because the search space is much larger, the search capacity of the GA has an effect on finding (sub-)optimal solutions quickly.

From Fig. 10.3 follows that the genetic algorithm optimised the three objectives. The solutions produced by the initial population are dominated by the final Pareto-front and the Pareto-front has the shape of the Pareto-optimal front. It also follows that, in contrast to the harvest scheduling problem, the genetic algorithm is now capable of finding the extreme solutions more easily than the compromise solutions. This is mainly due to the local suboptima located in the centre of the fitness surface. These local suboptima are not found at the boundaries and the chance of finding an optimal solution is much larger at the boundaries than at the centre of the fitness surface.

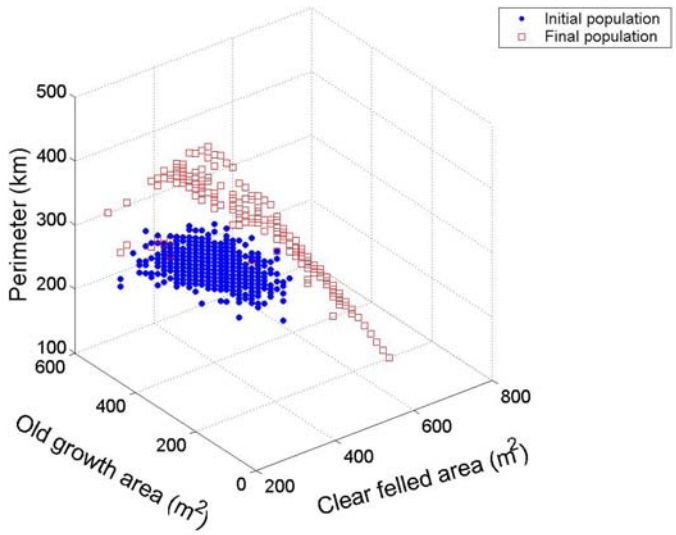


Figure 10.3: Pareto-front over 10 runs for the 9-by-9 grid. There is a difference between the initial population and the population found at generation 50

10.3.4 Application to a Kirkhill forest

10.3.4.1 Methodology

Kirkhill forest consists of 399 different stands, and problems of this size can no longer be solved using the procedure proposed by Hof and Joyce (1992, 1993). In this case, the chromosome length is 399, one bit for each stand. The population size was set to 100 and the total number of generations to 50. Crossover probability was set to 0.08 and uniform mutation occurred in 1% of the cases. The initial population was randomly initialised.

10.3.4.2 Results and discussion

The total running time was 4h on a Intel Pentium III processer (651 MHz and 384 MB of RAM). This included the entire communication between GA and GIS. As the global optima are not known beforehand, it is impossible to know whether the resulting Pareto-front is optimal. The Pareto-front between the different objectives for a single run is presented in Fig. 10.4.

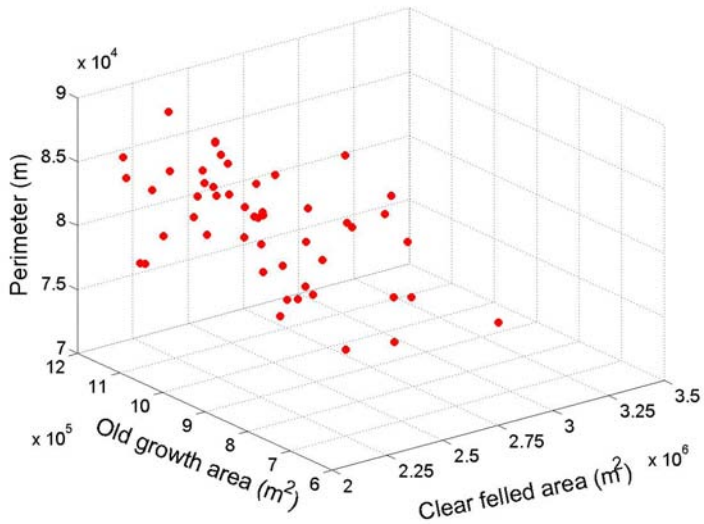


Figure 10.4: The Pareto-front between (1) clear felled area (m^2), (2) old growth area (m^2) and (3) perimeter (m)

From Fig. 10.4, it follows that different compromise solutions are found. Again, maintaining the spread for the real-world application proves to be a difficult task because all solutions are in the centre, no extreme solutions for the timber or old growth objectives are found (e.g. clear felling all stands and leaving all stands). The three most extreme solutions are presented in Table 10.1 and Figs. 10.5 to 10.7.

Table 10.1: Objective function values under the scenario of maximum timber production, maximum abundance of old growth species and maximum abundance of edge-dependent species

	timber	old growth	edge
Total timber production (m^3)	288·60	184·14	220·01
Abundance of old growth species	88·97	133·30	118·11
Abundance of edge-dependent species	78·818	83·460	85·821

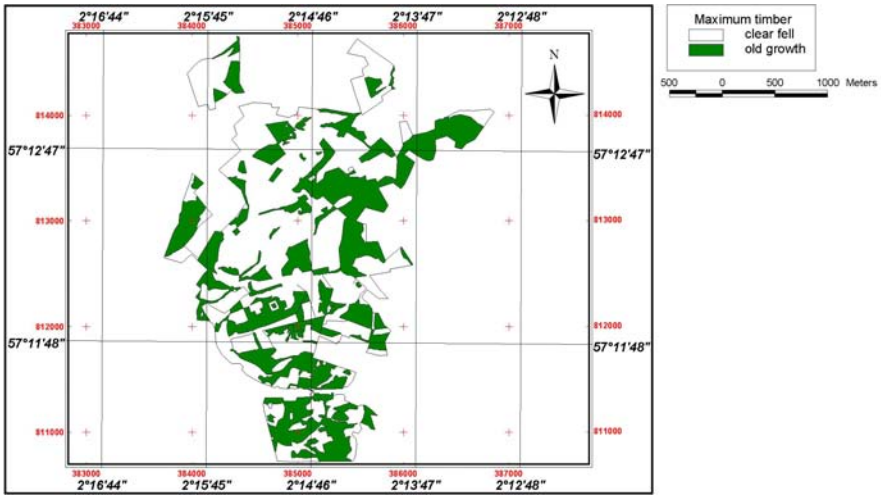


Figure 10.5: Harvesting pattern for a maximum timber production

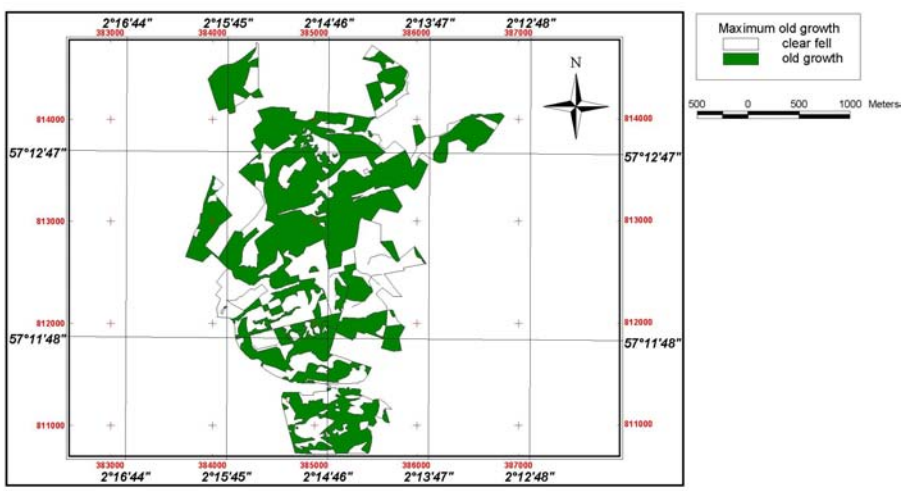


Figure 10.6: Harvesting pattern for a maximum abundance of old growth species

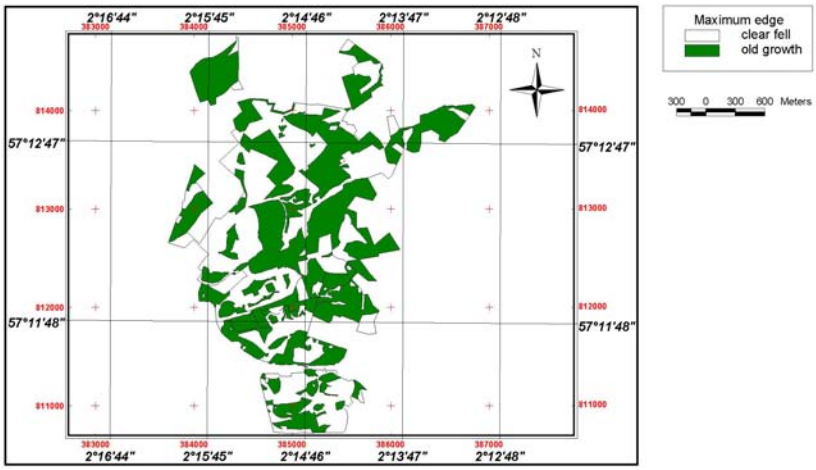


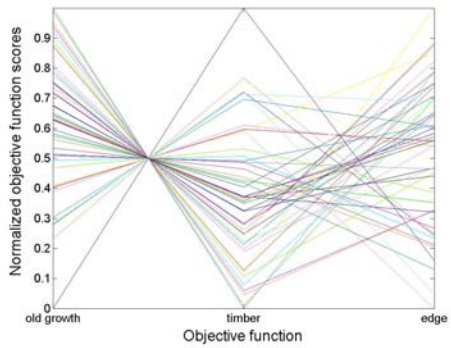
Figure 10.7: Harvesting pattern for a maximum abundance of edge-dependent species

Due to the lack of spread, there is not much difference in actual output levels between the three extreme solutions found in this run (Table 10.1), but even for the small differences noted, large differences in spatial configuration are found. In Table 10.1 under a maximum scenario of timber production, the timber volume amounts to 288.6 m^3 and in Fig. 10.5 is shown that there are more clear felled stands. For a maximal old growth objective many more stands are left (Table 10.1 and Fig. 10.6). However, this is not very pronounced, either due to insufficient population sizing or because of the linkage learning problem mentioned in the single objective case. The solution maximising the abundance of edge-dependent species resembles a checkerboard pattern (Fig. 10.7) and the total area that is clear felled is approximately half (277 ha) of the total area of all stands (454 ha). The number of edge-dependent species counted as badgers setts is 85.8 setts/454 ha or 18.9 setts/100 ha. This is also within the boundaries found for the abundance of badgers in the British Isles (26 mean used setts/100 ha), as well as for the land class found in mid-east Scotland (lowlands with variable land use, but mainly arable), where the mean used sett density is 7.4 setts/100 ha and the total sett density is 21 setts/ha (Cresswell et al., 1990). The results show that including the spatial information leads to very different results in terms of spatial layout. Because the spatial data can be included during the optimisation process, the combination of genetic algorithms and GIS opens possibilities for a spatial decision support system.

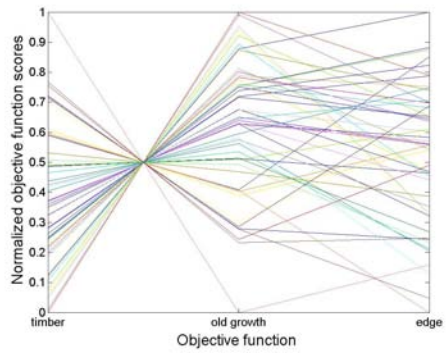
Because there are more than two objectives, other methods for visualising the solutions and their relationships are needed. Fonseca (1995) introduced the method of parallel coordinates to show possible conflicts between objectives. These graphs depict each non-dominated solution, with on the x-axis an integer

representing the objective function and on the y-axis their rescaled objective function scores. These objective functions are rescaled to the interval [0,1]. If the connecting lines between the objectives are parallel, the objectives are not conflicting but in harmony, and the problem can be recast into a single objective one. If the lines are crossing, the objectives are conflicting and only a multiple objective approach can produce solutions. Purshouse and Fleming (2003) provided a mathematical foundation for the conflict, harmony and independence relationships between the objectives.

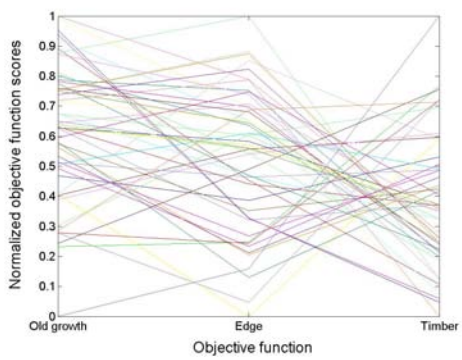
There is a clear inverse linear relationship between the objective of timber production and leaving old growth stands (Figs. 10.8(a) and 10.8(b)). The relationship between the objective of edge and timber on the one hand (Figs. 10.8(a) and 10.8(c)) and old growth on the other hand (Figs. 10.8(b) and 10.8(c)) is not inverse, but is on some occasions in harmony while on other occasions there is a conflict. This is logical because the Pareto-front has a maximum for the edge objective function when the level of old growth species and timber production is average and is minimised where the other two objectives are maximal.



(a) Timber vs old growth and edge



(b) Old growth vs timber and edge



(c) Edge vs old growth and timber

Figure 10.8: Visualisation of the conflicts between the three objectives: timber volume, abundance of old growth species and abundance of edge-dependent species

10.4 Conclusions

The integration of GA and GIS offers clear potential for spatial decision support systems. The functionality of both modules fit together and can remain in two entities. This ensures flexibility for the SDSS.
Solving a simple spatial problem is not efficient because the problem is too easy to solve and in the initial population most of the solutions are already generated. However, when the size of the problem increases, the advantage of genetic algorithms becomes apparent: it is able to generate solutions closer to the Pareto-front in a short time. Solving a real-world application again shows that the discreteness of the forest management problem causes the lack of spread along the Pareto-front. The solutions are all centred in the middle. If the extreme solutions are taken as alternative, the plans do appear valid: for the timber scenario, clearly more stands are scheduled for clear felling, whilst for the old growth scenario, more stands are left. For the edge-dependent problem, the alternative resembles a checkerboard pattern, and half of the forest is cut. Clearly the solution is not optimal yet, and it is thought that this is caused either by insufficient population sizing or the linkage problem. These two problems will be handled in detail in Chapter 12.

Part III

Advanced genetic algorithms

Chapter 11

Estimation of distribution algorithms

11.1 Deceptiveness in genetic algorithms

In simple genetic algorithms the building blocks are processed implicitly. This has several consequences: (1) the encoding strategy must be appropriate for the problem at hand, and (2) the operators that are used during the evolutionary optimisation process should preserve the building blocks. If the decision variables are independent of each other or when the defining length of the schemata of the building blocks is low, then the classical crossover operators can properly mix the building blocks to get Pareto-optimal solutions.

For some optimisation problems though, decision variables are dependent on other variables. Under those conditions, it may occur that the partial schemata fitness is lower than the fitness of the complete schema. Suppose the following schemata with length l (Goldberg, 1989):

$$
\begin{aligned}
&**0**1** \\
&**0**0** \\
&**1**0** \\
&**1**1**
\end{aligned}
$$

and suppose that the global optimum contains the schema 11. Solutions containing the other schemata are suboptimal solutions. This means that the fitness of schema 11, f_{11} is larger than that of schema 01, f_{01}, schema 10, f_{10} and than that of schema 00, f_{00}. If however for the partial schema $0*$, $*0$, $1*$ and $*1$ holds that $f(0*) > f(1*)$ and $f(*0) > f(*1)$, then the genetic algorithm might be fooled because of the 'wrong' sampling information. If $f_{00} \geq f_{01}$ and the initial proportion of $00 \gg 01$ then the population converges on a local optimum.

This type of problem is referred to as a deceptive problem, because the partial fitness of the schemata deceives the genetic algorithm and inhibits convergence towards global optima. This also occurs in nature and is referred to as *epistasis* by biologists. Epistasis occurs when two genes are evaluated separately and are 'useless' that way but when they are evaluated as a whole their function is suddenly extremely important to the organism.

Epistasis leads to disturbed genetic algorithm behaviour especially when the order or the defining length of the schemata showing deceptiveness is long. When there is a higher order building block, the bits that make up the optimal schema will never be linked under the influence of a standard crossover operator, and thus the optimal solution cannot be found.

Epistasis is called the 'linkage problem' in the domain of evolutionary algorithms (Mitchell, 1996; Goldberg, 1989; Falkenauer, 1998). Bosman and Thierens (1999) define linkage for the binary representation as 'the structural cohesion of the bits in the coding string with respect to the search space'. The aim of linkage learning is to arrange the genes of which the building block is composed, together so that the crossover operator does not break the building blocks.

11.2 Research rationale

In real-world applications, schemata are more often than not loosely coupled (Harik, 1997). Theoretical studies (Thierens & Goldberg, 1993) have shown that genetic algorithms ignoring possible linkage patterns reduce the exploitation of the problem structure (Bosman & Thierens, 1999).

The abundance of edge-dependent species was maximised using simple single (Chapter 6) and multiple objective genetic algorithms (Chapter 10) and in both cases suboptimal solutions were derived. This might be due to loosely coupled building blocks. Because of the spatial relationship between the data, the genes that belong together might be far apart in the chromosome. This was also noted by Van Dijk et al. (2000): they designed a specialised crossover operator for a map-labelling problem in GIS. The relation of one stand to another stand can be either a one-to-one relationship, indicating complete independence, a one-to-many relationship, indicating epistasis or a many-to-one relationship, which arises when several parameters are represented by one building block (Falkenauer, 1998). If there is complete independence, the problem can be solved using simple genetic algorithm strategies. In the other two cases, the simple approach might fail. This chapter addresses the problem of a one-to-many relationship and answers the following research question: "Is there a one-to-many relationship between the genes for forestry related problems?"

11.3 A short review of linkage learning

11.3.1 Using tailor-made representations or operators

In linkage learning there are two lines of research. In the first class of techniques various reordering operators are designed in order to bring the genes of the building blocks together.

Techniques to preserve linkage information have been proposed quite early. Both Holland (1975) and Goldberg (1989) propose the use of an inversion and a reordering operator to bring the composing genes of the building blocks together. This inversion operator chooses two positions in the chromosome at random and switches the allele values. Through random chance, it may occur that the defining length of a schema is shortened and therefore preserved after crossover. This operator, however, is not very successful, as selection drives the population towards convergence long before the genes are brought together (Pelikan et al., 2000b).

A second method is the messy genetic algorithm (messy GAs) (Goldberg et al., 1992, 1993). Messy genetic algorithms do not work using a fixed-length chromosome but instead use a variable length chromosome where each gene is defined by its position (locus) and its value (allele). In each chromosome it is possible for some loci to be missing (underspecification) and others to be specified more than once (overspecification), hence the name 'messy'. There are two phases: the primordial phase where initially the population is formed through enumeration of all possible schemata of the order k, which has to be known or guessed in advanced. In the juxtapositional phase the chromosomes are recombined taking into account the under and overspecification. Mitchell (1996) points out that enumerating all schemata of order k is infeasible. Goldberg et al. (1993) and Kargupta (1995) replaced the complete enumeration with a probabilistically complete initialisation, but Mitchell (1996) shows that even with this initialisation phase the initial population size grows exponentially with the order of the building blocks.

A third approach in this framework is the Linkage Learning Genetic Algorithm (LLGA) (Harik, 1997). Harik (1997) designed a specialised crossover operator that uses the linkage information for the recombination phase and a probabilistic expression method to supplement the messy GA. The LLGA performs well for exponentially scaled problems but Pelikan et al. (2000b) mention that it is not very efficient from a theoretical point of view for uniformly scaled building blocks.

11.3.2 Probabilistic modelling

The setting and choice of crossover and mutation operators itself is a difficult task. This together with a need to better understand the behaviour of a genetic algorithm led to the creation of so-called breeder algorithms (Mühlenbein & Schlierkamp-Voosen, 1994) where no recombination operators *senso stricto*

are used. The group of these probabilistic genetic algorithms is referred to as Estimation of Distribution Algorithms (EDAs) (Larranaga & Lozano, 2002).

In EDA, the population is seen as a sample from the solution space and through the use of probability density functions, the relationship between the genes is explicitly formulated. These relationships ensure that the genes of each building block are kept together during recombination.

A distinction between the different probabilistic models can be made according to the degree of interaction between the genes they allow. The simplest models, such as by Harik et al. (1998) and by Baluja and Caruana (1995), do not include any interaction. These probabilistic models assume that all decision variables are independent and build the density function across the population for each gene independently. In the compact GA (cGA) (Harik et al., 1998) the selection step is the same as in the regular evolutionary algorithm. After selection a probability vector is created. This vector represents the probability that the gene on that position has an allele value of 1. The probability is calculated by scanning across the population and counting the number of times a 1 is encountered on that position. The basic idea is that, for a single objective problem, the population converges and that ultimately the probability vector will consist only of probability zero and probability one. According to these probabilities new individuals are created. The behaviour of the cGA is in essence that of a simple genetic algorithm with uniform crossover. As these models are designed for univariate relationships, they do not perform well when there are interactions between the variables (Pelikan et al., 2000b). Bivariate models, such as by Baluja and Davies (1997) allow interactions between two genes and can effectively solve problems with order 2, but are insufficient to solve higher order problems. Multivariate models allow multiple interactions and can be represented by trees, clusters or Bayesian networks (Mühlenbein & Mahnig, 1999; Harik, 1999; Pelikan et al., 2000b).

In the context of geographical information systems, the use of discrete multivariate models seems appropriate. The discrete nature of a stand translates naturally into a discrete model. The spatial linkage between one stand and the others can be described as a one-to-many relationship. This relationship might be expressed by multivariate probabilistic models. It is hypothesised that the probability density structure of the problem can be represented either (1) by clusters of several genes forming building blocks that do not have any further interaction between them or (2) by a network if there are conditional dependencies between the clusters (Fig. 11.1). The second research question addressed in this chapter is thus:

- if there is epistasis, can the relationship be explained by using clusters or by a full network structure?

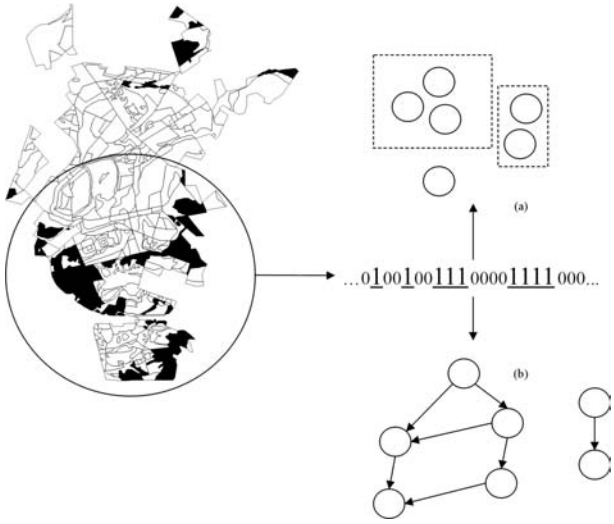


Figure 11.1: Possible probabilistic structures for the representation of the spatial structure of the forest: (a) as in the Extended Compact GA and (b) as in the Bayesian Optimization Algorithm

11.4 Probabilistic models used for EDAs

Two probabilistic models will be investigated: the Extended Compact Genetic Algorithm (ECGA) by Harik (1999) because it represents multivariate relationships as clusters and the Bayesian optimization algorithm (Pelikan et al., 2000b) that allows a full network structure.

11.4.1 The Extended Compact Genetic Algorithm

The Extended Compact GA (ECGA) was designed by Harik (1999) and extends the compact GA by Harik et al. (1998) because it allows multivariate interactions whereas the compact GA does not allow gene interaction. The procedure of the ECGA is as follows: first the initial population is generated and the mating pool is selected. Based on the mating pool, the probabilistic model is built. Because there is a trade-off between the goodness-of-fit of a model to represent the data set on the one hand and the model size on the other hand, different measures to determine the best model can be used. The size of the model in ECGA is determined by the memory space needed and is called the model complexity (Eq. 11.1). The compressed population complexity (Eq. 11.2), which is based on the Shannon-entropy (Shannon, 1948) (Eq. 11.4), corresponds to how much the data can be compressed and thus how good the model represents the data set. The Combined Complexity (CC) (Eq. 11.3) is the sum of the two previous

scoring functions and is the same as the Minimum Description Length (MDL) (Rissanen, 1989). The CC thus favours the least complex models that fit the data best.

$$MC = (\log_2(N+1)) \times \sum_i (2^{S_i} - 1) \tag{11.1}$$

$$CPC = N \times \sum_i \text{entropy}(M_i) \tag{11.2}$$

$$CC = MC + CPC \tag{11.3}$$

where

$$\text{entropy}(M_i) = -\sum_i p_i \times \log_2 p_i \tag{11.4}$$

where N is the population size, p_i is the probability of having an allele value of 1 on locus i, S_i is the cardinality of subset i and M_i is the marginal distribution of this subset.

Initially, the simplest model, one not including any interaction, is generated and the CC is calculated. Then the genes are combined on a second level by making one group of two bits and all the other bits are left independent, and again the CC is calculated for each of these models. According to the steepest descent search, the model that shows the largest decrease of CC is retained for further expansion. All genes are added to the model in this way until there is no further decrease in CC. Once the probabilistic model is generated, a crossover-like operator is used. This operator shuffles the subsets between all parents and generates new offspring (Fig. 11.2). Note that in this way no mutation operator is applied. When the size of all subsets is equal to one, then this operator is the same as a uniform crossover over the complete population (Alg. 2). An example of the ECGA is given in Appendix D.

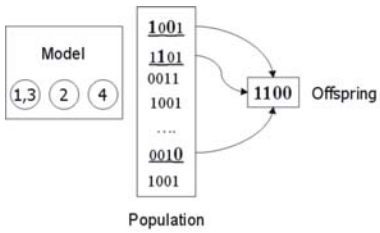


Figure 11.2: The crossover operator in the Extended Compact Genetic Algorithm: for each subset a parent is randomly selected from the mating population. The corresponding subset from the parent is transferred to the offspring.

Algorithm 2: The Extended Compact GA (Harik et al., 1999)

```
set t ← 0;
while termination criteria not met do
    select promising string S(t) from population P(t);
    create initial model M;
    calculate the CC;
    while CC decreases do
        create new model by expanding subsets from former model M;
        calculate CC;
    end
    generate a set of new strings O(t) according to best model M;
    create population P(t + 1) by replacing some strings from P(t) by
    O(t);
    set t ← t + 1;
end
```

11.4.2 The Bayesian Optimization Algorithm

A short introduction to Bayesians networks A Bayesian network is an efficient tool to build a model from a domain with a certain degree of uncertainty and has been used extensively in expert networks (Jensen, 1996). It is a directed acyclic graph (DAG) where each directed edge shows the conditional probability existing between the two connected nodes. A Bayesian network can encode a joint probabilistic distribution function formally (Heckerman et al., 1995).

In order to learn Bayesian networks semi-automatic methods are used to construct or modify them using prior knowledge. Two types of learning can be defined (Jensen, 1996):

- quantitative learning: determining the structure of the network;
- qualitative learning: determining the parameters of the network.

Batch learning is a technique where the learning process is based on a data set. Because the data set is finite, the probability distribution over the population does not always represent the true distribution. In that case, even if the model fits the database well, the model is still incorrect. A way around this is to determine the joint probability of a network given the database. This is usually assessed using Bayes' rule (Jensen, 1996) (Eq. 11.5).

$$P(B \mid D) = \frac{P(D \mid B) \cdot P(B)}{P(D)} \tag{11.5}$$

with B a prior hypothetical network and D the data set, and $P(B \mid D)$ the conditional probability that network B is present given data set D, $P(D \mid B)$ the conditional probability that the data set D is derived from network B, $P(B)$ and $P(D)$ the probability distribution over respectively network B and the data

set D. The joint probability of network B and data set D occurring together is then given by Eq. 11.6:

$$P(B, D) = P(D \mid B) \cdot P(B) \tag{11.6}$$

If there are only two categories, e.g. an experiment of tossing a fair or biased coin, the probability of having a head is π and the probability of having a tail is $1 - \pi$, with π some prior knowledge about whether the coin is fair or biased, and by how much the coin is biased. The probability mass function over n coin tossing trials is then a binomial distribution (Eq. 11.7)

$$P(x \mid \pi) = \begin{pmatrix} n \\ x \end{pmatrix} \pi^x \cdot (1 - \pi)^{n-x} \tag{11.7}$$

with x the number of heads and n the number of trials. Instead of assigning discrete values to π, it is also possible to characterise π as a probability density function, and use this function as prior knowledge. A distribution that is often used for binomial probabilities is the β-distribution, because if a β-distribution is used as prior in combination with a likelihood function, the posterior distribution is also a β-distribution. If the prior β-density function has two positive parameters (a, b) then the β-density function is written as in Eq. 11.8:

$$P(\pi) \propto \pi^{a-1} \cdot (1 - \pi)^b \tag{11.8}$$

The posterior β-distribution function, after n trials with x successes is then characterised by the parameters $(a + x, b + n - x)$ (Eq. 11.9):

$$P(\pi \mid x, n) \propto \pi^{a+x-1} \cdot (1 - \pi)^{b+n-x} \tag{11.9}$$

This can be extended to the multinomial case, where the number of categories $k > 2$. Under those conditions, the multinomial variant of the β-distribution is used, and this is known as the Dirichlet distribution (Congdon, 2001). Let $x_1, x_2, \ldots, x_k$ denote the counts from $k > 2$ categories of the outcome. Then the multinomial likelihood function specifies (Congdon, 2001):

$$p(\theta_1, \theta_2, \ldots, \theta_k \mid \alpha_1, \alpha_2, \ldots, \alpha_k) \propto \prod_{j=1}^{k} \theta_j^{x_j} \tag{11.10}$$

where the θ_j, the probabilities of belonging to one and one only of the k classes, sum to 1. The conjugate prior density function is a density for $\theta_1, \ldots, \theta_k$ specified in terms of positive parameters $\alpha_1, \cdots, \alpha_k$, namely

$$p(\theta_1, \theta_2, \cdots, \theta_k \mid \alpha_1, \alpha_2, \cdots, \alpha_k) \propto \prod_{j=1}^{k} \theta_j^{\alpha_j - 1} \tag{11.11}$$

Suppose the following initial values $c_1, \ldots, c_k$ are assigned to $\alpha_1, \alpha_2, \ldots, \alpha_k$ then the posterior density of the $\theta_1, \cdots, \theta_k$ is again a Dirichlet distribution with

parameters $c_1+x_1, c_2+x_2, \cdots, c_k+x_k$. From the properties of the Dirichlet distribution, the posterior means of the multinomial probabilities, for $i = 1, \cdots, k$ are then

$$\frac{x_i + c_i}{X + C} \tag{11.12}$$

with $X = \sum_{i=1}^{k} x_i$ and $C = \sum_{i=1}^{k} c_i$. Often the c_i are assumed equal to each other, i.e. $c_i = C/k$ for all i (Bishop, Fienberg, & Holland, 1980).

This allows to incorporate knowledge to various degrees by changing the expected cases for each category. Imagine there is a previous study of a similar problem, then the information from that study can be used as prior values for c_j. The smaller the ratio between the expected cases c_i and the observed cases x_i the more weight is attached to the prior knowledge. If however these values are unknown or the information from the study has a high degree of uncertainty then setting c_j equal to 1 and $C = k$, will ensure that although the information is used to some extent, it can be swamped by the new information from the current data set.

One measure based on this rule is the Bayesian Dirichlet measure (Heckerman et al., 1995) (Eq. 11.13). This measure combines prior knowledge and information from the data set in order to determine what the posterior probability of the presence of a certain model is together with the data set.

$$P(D, B \mid \zeta) = P(B \mid \zeta) \cdot \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(N_{ij}^{'})}{\Gamma(N_{ij}^{'} + N_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(N_{ijk}^{'} + N_{ijk})}{\Gamma(N_{ijk}^{'})} \tag{11.13}$$

$$\Gamma(n) = (n-1)! \tag{11.14}$$

with ζ some background information, r_i is the number of states node X_i can take and q_i the number of states the parents $\mathbf{Pa_i}$ of node X_i can take. The jth instance of the parents of X_i is written as $\mathbf{Pa_{ij}}$. N_{ij} is the number of cases in the data set D that the parents are in instance j. N_{ijk} denotes the number of cases in the data set D where $\mathbf{Pa_{ij}} = k$, in other words the number of cases where the node X_i is in state k given that the parent set is in its jth state. $N_{ij}^{'}$ and similarly $N_{ijk}^{'}$ denote the prior information about the number of instances of resp. $\mathbf{Pa_{ij}}$ and $\mathbf{Pa_{ij}} = k$. Finally, the gamma function (Eq. 11.14) is the discrete gamma function for positive integers. This gamma function is sometimes called the factorial function (Spiegel, 1980).

When applying this measure there is a large degree of freedom to set the level of incorporating the prior knowledge. $N_{ij}^{'}$ and $N_{ijk}^{'}$ together with $P(B \mid \zeta)$ specify the current knowledge about the domain. The specification of these values, however, is very difficult for all instances of i,j and k. If uninformative exponents are used by setting $N_{ijk}^{'} = 1$ and since $N_{ij}^{'} = \sum_{k=1}^{r_i} N_{ijk}^{'}$ then $N_{ij}^{'} = k$. The weight attached to the knowledge from the data set is much higher than the weight attached to the prior knowledge. If the number of cases N_{ij} and N_{ijk} is much larger than $N_{ij}^{'}$ and $N_{ijk}^{'}$ then the data will tend to swamp the

prior knowledge (Congdon, 2001). If all exponents are set to 1, and the prior distribution over the networks is uniform a special case of the BD-measure, the K2-measure (Eq. 11.15), is derived.

$$P(D, B \mid \zeta) = P(B \mid \zeta) \cdot \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(k)}{\Gamma(k + N_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(1 + N_{ijk})}{\Gamma(1)} \tag{11.15}$$

and combining Eqs. 11.15 and 11.14 leads to Eq. 11.16:

$$P(D, B \mid \zeta) = P(B \mid \zeta) \cdot \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(k-1)!}{(k-1+N_{ij})!} \prod_{k=1}^{r_i} (N_{ijk})! \tag{11.16}$$

This expression can also be written in a logarithmic form and then all products become sums. Each change in network structure can then be evaluated rather quickly.

An example of probabilistic modelling at hand Suppose the following population of chromosomes (Table 11.1) and the network structure (Fig. 11.3):

Table 11.1: An example population with 11 individuals and two possible classes

0	0	1	0
1	0	1	0
1	1	0	1
1	0	1	1
0	0	1	1
1	0	1	1
1	1	1	1
0	0	0	1
1	0	0	0
1	0	0	0
0	0	0	0

Node or gene 1 is a top node and has no parents (Fig. 11.3). As all the chromosomes in the population can be represented by either schema $1 * **$ or by schema $0 * **$, the number of cases N_{ij} for the empty parent set of node 1 is $N_{\mathbf{Pa_1}=\emptyset} = 11$. The number of cases where $X_1 = 0$ and the parent set is empty is then : $N_{X_1=0 \wedge \mathbf{Pa_1}=\emptyset} = 4$ and the number of cases where $X_1 = 1$ and the parent set is empty is $N_{X_1=1 \wedge \mathbf{Pa_1}=\emptyset} = 7$. Node 2 is also a top node and therefore $N_{\mathbf{Pa_2}=\emptyset} = 11$. The number of cases where $X_2 = 0$ and the parent set is empty is: $N_{X_2=0 \wedge \mathbf{Pa_2}=\emptyset} = 9$ and the number of cases where $X_2 = 1$ and there is an empty parent set is $N_{X_2=1 \wedge \mathbf{Pa_1}=\emptyset} = 2$. Node 3 is conditionally dependent by the

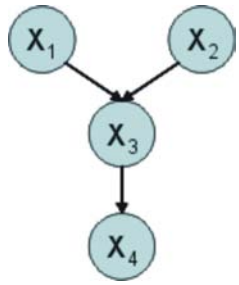


Figure 11.3: An example network with four nodes and the conditional dependencies between the nodes

parent set $\mathbf{Pa_3} = \{X_1, X_2\}$ and consequently there are 4 possible instances of the parents set: $\{0,0\}, \{0,1\}, \{1,0\}, \{1,1\}$. For these four instances the number of cases where $X_3 = 0$ and $X_3 = 1$ has to be counted:

$$\begin{aligned}
N_{X_3=0\wedge\mathbf{Pa_3}=\{0,0\}} &= 2\\
N_{X_3=0\wedge\mathbf{Pa_3}=\{0,1\}} &= 0\\
N_{X_3=0\wedge\mathbf{Pa_3}=\{1,0\}} &= 2\\
N_{X_3=0\wedge\mathbf{Pa_3}=\{1,1\}} &= 1\\
N_{X_3=1\wedge\mathbf{Pa_3}=\{0,0\}} &= 2\\
N_{X_3=1\wedge\mathbf{Pa_3}=\{0,1\}} &= 0\\
N_{X_3=1\wedge\mathbf{Pa_3}=\{1,0\}} &= 3\\
N_{X_3=1\wedge\mathbf{Pa_3}=\{1,1\}} &= 1
\end{aligned}$$

so that $N_{X_3\wedge\mathbf{Pa_3}=\{0,0\}} = 4$, $N_{X_3\wedge\mathbf{Pa_3}=\{0,1\}} = 0$, $N_{X_3\wedge\mathbf{Pa_3}=\{1,0\}} = 5$ and $N_{X_3\wedge\mathbf{Pa_3}=\{1,1\}} = 2$. Node 4 has node 3 in the parent set $\mathbf{Pa_4} = \{X_3\}$ and there are two possible instances of this parent set: $\{0\}, \{1\}$. The counts are then:

$$\begin{aligned}
N_{X_4=0\wedge\mathbf{Pa_4}=\{0\}} &= 3\\
N_{X_4=0\wedge\mathbf{Pa_4}=\{1\}} &= 2\\
N_{X_4=1\wedge\mathbf{Pa_4}=\{0\}} &= 2\\
N_{X_4=1\wedge\mathbf{Pa_4}=\{1\}} &= 4
\end{aligned}$$

and $N_{X_4\wedge\mathbf{Pa_4}=\{0\}} = 5$ and $N_{X_4\wedge\mathbf{Pa_4}=\{1\}} = 6$. Since for all nodes there all only two states $k = 2$ the K2-measure 11.15 reduces to:

$$P(D, B \mid \zeta) = P(B \mid \zeta) \cdot \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{1}{(1 + N_{ij})!} \prod_{k=1}^{r_i} (N_{ijk})! \tag{11.17}$$

So for this example

$$P(D, B \mid \zeta) \propto \frac{7! \cdot 4!}{12!} \cdot \frac{2! \cdot 9!}{12!} \cdot \frac{2! \cdot 2!}{5!} \cdot \frac{2! \cdot 3!}{6!} \cdot \frac{1! \cdot 1!}{3!} \cdot \frac{3! \cdot 2!}{6!} \cdot \frac{2! \cdot 4!}{7!}$$
$$\propto 5.62\ 10^{-15}$$

This indicates the probability for network B to be jointly present with data set D. If the network is changed by deleting the arc between node 1 and node 3, and inserting an arc between node 1 and node 2, then $P(D, B \mid \zeta) = 6.10\ 10^{-14}$ and thus it is more likely that the second network represents the conditional relationships better than the first network.

The Bayesian Optimization Algorithm The Bayesian Optimization Algorithm uses a Bayesian network to represent the conditional dependencies between the genes. Because the search for the optimal network is NP-complete (Chickering et al., 1995), Pelikan et al. (2000b) implement an additional parameter constraining the number of incoming edges in the DAG. This reduces the search space and simplifies the construction of new networks. The initial network is empty and the new networks are created by adding, deleting or inverting the arcs. Using the logarithmic form of the BD-measure, each change is evaluated quickly. The search for new networks can be based on any heuristic search technique and in the Bayesian Optimization Algorithm a simple greedy search technique is used. Once the model is built, the marginal probabilities for each gene can be calculated and used to generate new values for each of the genes according to these probabilities. A general description of the Bayesian Optimization Algorithm is given in Alg. 3. In Appendix D an example of the network construction and creation of new offspring is given.

Algorithm 3: The Bayesian Optimization Algorithm (Pelikan et al., 2000b)

```
set t ← 0;
while termination criteria not met do
    select promising string S(t) from the population P(t);
    construct the network B using a chosen measure and constraints;
    generate a set of new strings O(t) according to the joint distribution
    encoded by B;
    create a new population P(t+1) by replacing some strings from P(t)
    by O(t);
    set t ← t + 1;
end
```

Chapter 12

Case study: Maximising the abundance of badgers with EDAs and GIS

12.1 Research rationale

Because in the previous case studies the best solutions after optimising the abundance of edge-dependent species were suboptimal, the effect of the estimation of distribution algorithms on this problem is investigated. It is assumed that loose linkage is (partially) responsible for the suboptimal solutions. Loose linkage cannot be detected unless the application of specialised algorithms or operators results in better performance than simple genetic algorithms and standard crossover operators.

12.2 Methodology

The original source code of Extended Compact GA (ECGA) (Lobo & Harik, 1999) and the Bayesian Optimization Algorithm (BOA) (Pelikan, 2000) was adapted for the real-world application. ECGA was also implemented in Java, but as a Java-implementation is not so fast in execution time as a version in C the original source code was used.

In order to sample the population properly, large population sizes are required, otherwise wrong correlations may be built into the model. For both ECGA and BOA, the population size was fixed to 1000. Larger population sizes were not considered because fitness evaluations are very costly in terms of

computing time. For BOA it was assumed that the maximum number of incoming edges is 5, consequently the order of the building blocks is at most 5. For both algorithms, the crossover probability was 1 and mutation probability was zero. In ECGA, tournament selection was used with a comparison set of size 16. As the selective pressure was high, no elitism is needed. In BOA, truncation selection was applied with $\tau = 50\%$, this means that half of the population is replaced by the offspring. Because of the lower selective pressure, elitism was used in this case. These values are reported as suitable settings for the two algorithms (Harik, 1999; Pelikan et al., 2000b).

12.3 Results and discussion

12.3.1 EDAs versus simple GA

Both EDAs perform significantly better than the simple GA with elitism ($p = 0 < 0.05$) (Table 12.1). This suggests that the building blocks cannot properly mix using a simple GA and a standard crossover operator. Therefore advanced algorithms taking into account the linkage information are needed to solve this type of real-world applications more efficiently. Implicit processing of building blocks does not guarantee that the correct building blocks are found in the population.

Table 12.1: The mean number of badger setts and the variance for simple GA (sGA), the Extended Compact GA (ECGA) and the Bayesian Optimization Algorithm (BOA).

Algorithm	Mean	Std Dev
sGA	72.15718	0.4295
ECGA	104.73810	0.3052
BOA	97.25322	0.5845

The first research question is thus answered:

- for this type of real-world applications, building blocks have a higher order;
- and there is loose linkage or epistasis.

This means that linkage learning is helpful to find optimal solutions when maximising the abundance of edge-dependent species.

12.3.2 ECGA versus BOA

To address the second question: 'How can the relationship between the genes be represented and what structure leads to the highest objective function value?', the underlying structure from the two algorithms, either a cluster or a network, and the objective function values are analysed. As a good solution within acceptable time limits is more preferable than the optimal solution where one has to wait a long time, the maximum number of generations was used as stopping criterion and the output after 40 generations was used to compare the algorithms.

ECGA A solution to which the population converges in one of the repetitions is given in Fig. 12.1. This solution has a perimeter length of 83895 m and this corresponds to 95.34 badger setts. The average abundance in the region of Aberdeen is 21 setts/100 ha (Cresswell et al., 1990) and for Kirkhill forest this results in 104.16 setts. The value obtained with the genetic algorithm is thus very similar to the average value for the region of Aberdeenshire. The area of the forest that is clear felled amounts to 237.1451 ha which is approximately half of the forest.

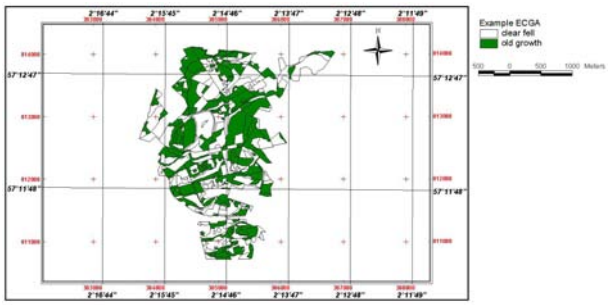


Figure 12.1: An example solution found by ECGA

ECGA clusters the stands in 99 groups. The average number of stands in each cluster is initially around 4 (Fig. 12.2). The size of the cluster increases a little during the first 5 generations but quickly declines afterwards over the number of generations. This is caused by population convergence: as the individuals are more and more similar due to selection, the model cannot detect any correlations. The model found in the first generation for the previous solution is shown in Fig. 12.3.

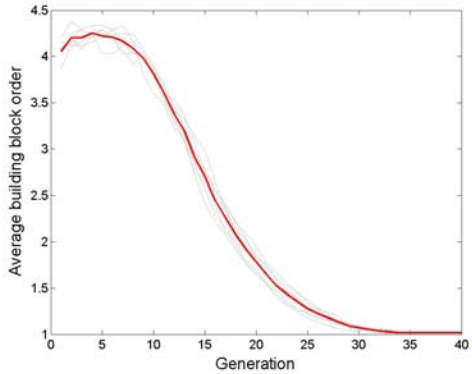


Figure 12.2: Evolution of average size of the clusters or building blocks for ECGA over the number of generations: the average cluster size decreases as the population converges

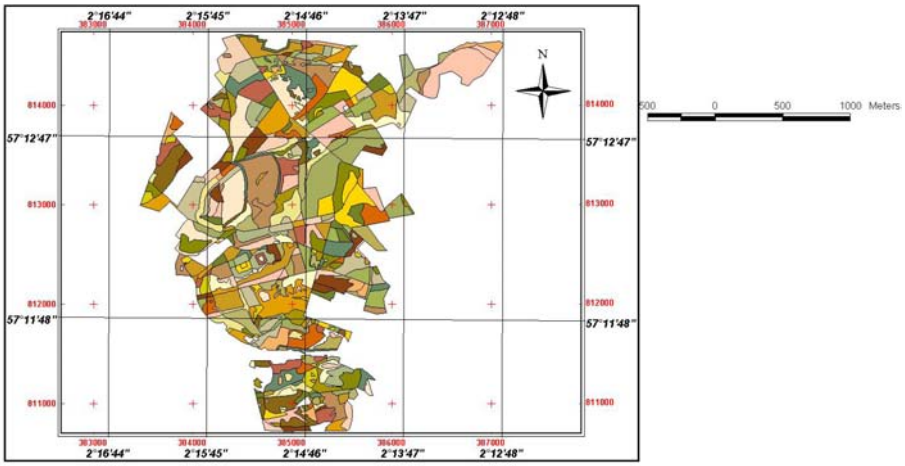


Figure 12.3: An example of a linkage model found in the first generation. There are 99 different clusters, with an average building block size of 4.1 genes

The chromosomes from the best solutions over the different repetitions are different from each other. Upon closer examination, it appears that parts of the chromosomes are complementary. For example, stands 22 and 23 have a different value in the final solutions from all repetitions and are each other's complement.

	Repetitions						
	1	2	3	4	5	6	7
22	0	1	1	1	0	1	1
23	1	0	0	0	1	0	0

Next to stands 22 and 23, other stands feature the same phenomenon: the 89 stands that are a complement to their neighbour are coloured in red in Fig. 12.4. Assuming that for these stands their linkage is only with their neighbour, the propagation of the building blocks for 22% of the genes of the chromosome would pose no problem because the genes are already closely coupled. The rest of the chromosome will have linkage with stands further away inducing loose linkage and this implies that specialised operators or algorithms are required indeed. These complementary chromosomes were also encountered in the grid problem: there the best local optimal solution was the complement of the global optimal solution. This implies that for the real-world application, as for the grid problem, the fitness surface has local suboptima.

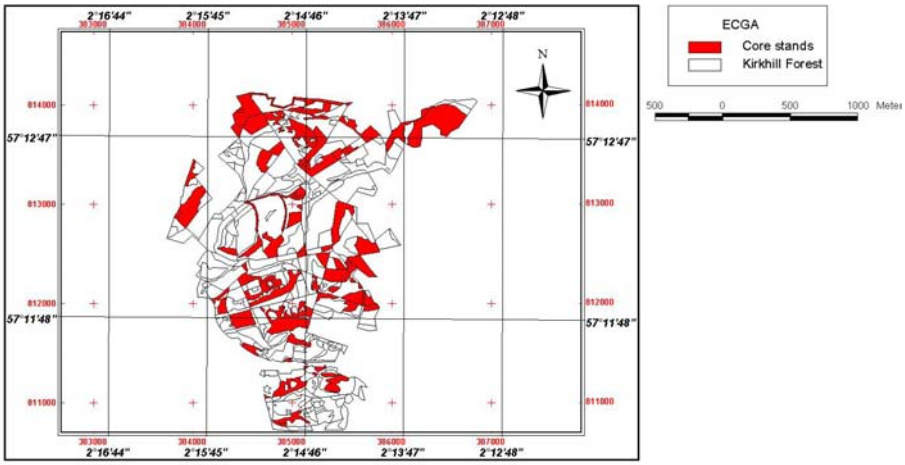


Figure 12.4: Core stands after running ECGA

BOA The best solution found in one of the repetitions after 40 generations is depicted in Fig. 12.5. The perimeter length for this solution is 77243 m, and this amounts to 97.44 badger setts in Kirkhill forest, again close to the average number of badger setts reported for Aberdeenshire. The total area cut is 195 ha, which is much less than the area clear felled with ECGA (237 ha).

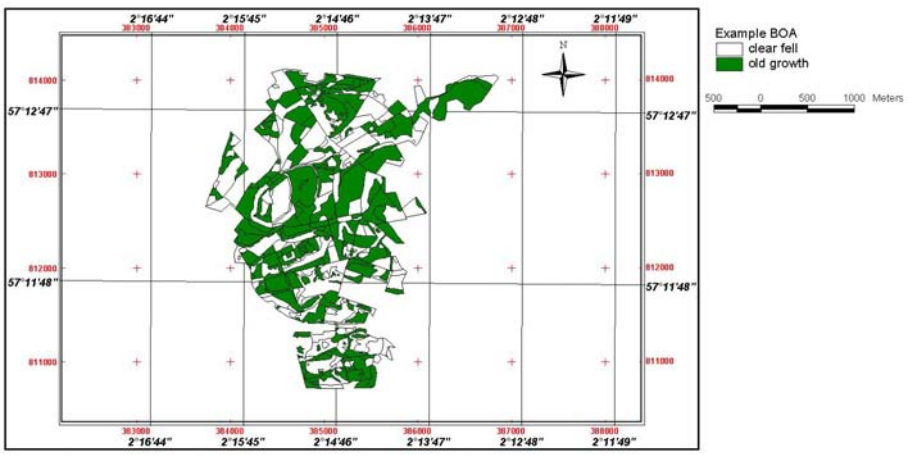


Figure 12.5: An example solution found by BOA

The structure found by BOA is a complex network (Fig. 12.6).

$0 \leftarrow 2, 99, 368, 321$
$1 \leftarrow 54, 365, 363, 58$
$2 \leftarrow 367, 145, 353, 31$
$3 \leftarrow$
$4 \leftarrow$
$5 \leftarrow 71, 15$
$6 \leftarrow$
$7 \leftarrow 317, 113, 291, 41$
$8 \leftarrow$
$9 \leftarrow 216, 261, 347, 316$
$10 \leftarrow 21, 204, 5, 305$
$11 \leftarrow 8$
$12 \leftarrow 19, 45, 97, 321$
$13 \leftarrow 170, 298, 109, 11$
$14 \leftarrow 334, 161, 15, 277$
$15 \leftarrow 59$
$16 \leftarrow 163, 363, 194, 158$
...

Figure 12.6: Several forest stands are grouped together to form a building block. In this figure for example stands nos. 54, 58, 363, 365

Because the network structure is complex, it is very difficult to present a physical interpretation of the structure. Still, as in the case of ECGA, complementary stands are present (Fig. 12.7). In BOA this number of complementary stands is larger than for ECGA: there are 105 complements in total.

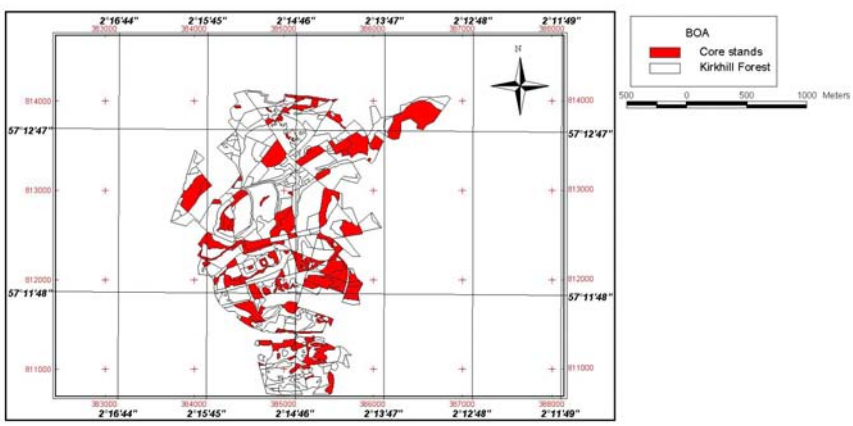


Figure 12.7: Core stands after running BOA

BOA detects building blocks up to order k, this being the number of incoming edges of a node. This parameter needs setting because finding the optimal

network is NP-complete (Chickering et al., 1995) and by using this parameter the search space is drastically reduced. However, for a real-world problem this can be a serious limitation, as usually the size of the building blocks is not known *a priori*. If the size is limited up to a certain order k and building blocks of higher order are present, then the population size needs to grow exponentially (Pelikan et al., 2001). The average number of incoming edges in the network is initially around 5 (Fig. 12.8) and declines to an average number of 4.5 incoming edges over time. It can be put forward that the size of the building blocks is not larger than 5 because ECGA uses no prior information and found no building blocks larger than 5. This makes it very likely that the assumption for the order of the building block for BOA was valid. The decline in size of the building blocks is slower than in the case of ECGA because the selective pressure in ECGA is higher and this drives the population much quicker to convergence (Fig. 12.9). This convergence probably leads to suboptimal solutions but has the advantage that better solutions are found in a shorter time span.

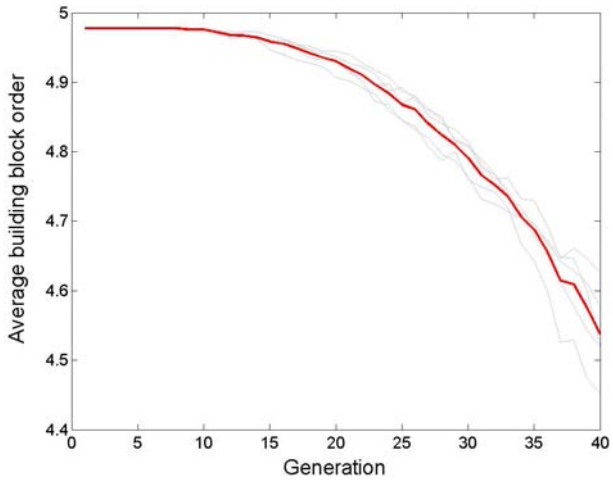


Figure 12.8: Evolution of average size of the clusters or building blocks for ECGA over the number of generations: the average cluster size decreases as the population converges

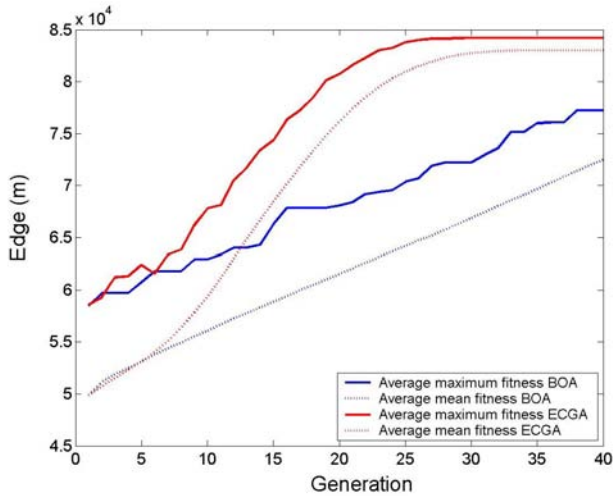


Figure 12.9: Mean maximum and average values over all repetitions for ECGA and BOA

Discussion The results for ECGA are better than for BOA (Fig. 12.9). After statistical analysis it was found that ECGA is significantly better ($p = 0.05$) than BOA. From Fig. 12.9 can be seen that the best individual and the average individual have not yet converged at generation 40 in the case of BOA. ECGA on the other hand converged already around generation 25.

Three possible causes can be put forward for the slow convergence of BOA. First of all, it is possible that the number of generations is not large enough, but this can be rejected on the basis of calculations by Pelikan et al. (2001). They calculated that for a problem with 396 bits, the time until convergence is around 35 generations. A second reason might be that the selective pressure in BOA is too low (oral comm. Pelikan, 2002) because the original settings advised by Pelikan (2000) were used.

Finally the population size could be too small (Pelikan et al., 2001). They calculated that the population size required for a problem with a length of 396 bits and building blocks of order 5 is around 10000 individuals when using BOA. In order to increase the population size two options are open: the population size can be set arbitrarily high, thereby using a lot of computer time, but this might cause even slower convergence as Lobo (2000) showed in his experiments. Another option that can be pursued is to use a parameter-less GA (Lobo, 2000). In that case one does not need to worry about the population size at all. One possible disadvantage is that this approach needs interaction with the user, who needs to decide when the evolution has to terminate.

One of the main problems that needs to be addressed before attempting any of these approaches is the reduction of the time span of the fitness evaluations. As 1 fitness evaluation takes up to 5 s on a 1.3 Ghz AMD-Athlonprocessor, extremely large population sizes should be avoided. This problem will be addressed in Chapter 13.

12.4 Conclusion

It can be concluded that the use of linkage learning for the forest management problem offers opportunities. The two linkage learning algorithms are significantly better than the simple GA. The real-world application suffers from loose linkage: the genes that make up a building block are not close together and the implicit processing of the building blocks does not guarantee optimality.
ECGA processes building blocks with an average size of 4.5 at the beginning of the evolution and the size decreases to 1 over time due to population convergence. The size of the building block is only determined by the probabilistic model. ECGA quickly converges due to its high selective pressure. Already at generation 25 the population converges. When the linkage model is physically interpreted, it is concluded that the number of badger setts found by ECGA approximates the average number of badger setts in the Aberdeenshire region. The cutting pattern in the forest resembles a checkerboard and a little more than half of the forest is clear felled. The solutions found by the different repetitions are not always the same. On the contrary, 22% of the bits form a group with their neighbour and are complementary to the same bits in other solutions. This indicates that for the other bits groups are formed with stands located further away and that linkage learning is necessary.
BOA finds very complicated network structures with on average 5 incoming edges at the beginning of the evolutionary process and 4.5 near the end of the evolution. This is the same size as in ECGA. BOA did not converge before the end of the evolution and the solution quality is still inferior to that of ECGA, probably due to insufficient population sizing and a lower selective pressure. The physical interpretation is much more difficult, but as for ECGA the genes from the chromosomes from different repetitions are complementary to some parts of the chromosomes.
In order to investigate the behaviour of the algorithms further the fitness evaluation time should be reduced as this currently is a major impediment for using larger population sizes. This will be dealt with in Chapter 13.

Chapter 13

Fitness inheritance

13.1 Introduction to fitness inheritance

In many real-world applications of evolutionary algorithms, the fitness of an individual has to be derived using complex models and time-consuming computations. Especially in the case of multiple objective optimisation problems, the time needed to evaluate these individuals increases exponentially (Chen et al., 2002) due to 'the curse of dimensionality' (Bellman, 1961) and this leads to a slower convergence of the evolutionary algorithms. For example, the determination of the edge between the cut and old growth area takes up to 5 s on a 1.3 Ghz AMD-Athlonprocessor. It is not feasible to use such time-consuming models with large population sizes unless the time to evaluate the objective functions is reduced.

Fitness inheritance is an efficiency enhancement technique that was originally proposed by Smith et al. (1995) to improve the performance of genetic algorithms. Sastry et al. (2001) and Chen et al. (2002) have developed analytical models for fitness inheritance. In fitness inheritance an offspring receives a fitness value that is inferred from the fitness values of its parents instead of through full fitness evaluation. If inferring the fitness value is faster than evaluating the model needed to determine the objective value, then fitness inheritance would speed up the optimisation process, unless the noise resulting from the fitness inheritance procedure is influencing the search behaviour of the genetic algorithm.

13.2 Theoretical foundation of fitness inheritance

13.2.1 Single objective fitness inheritance

Smith et al. (1995) were the first to introduce the technique of fitness inheritance. They point out that for some (and probably most) real-world optimisation problems, genetic algorithms cannot be applied because the cost of determining the fitness values for an entire population is too high. Earlier on, Grefenstette and Fitzpatrick (1985) tried to reduce the evaluation time by partially evaluating each individual instead of completely evaluating it and by allowing the genetic algorithm run for more generations. They concluded that it is more effective to evaluate fast noisy fitness functions for more generations than to evaluate the slow but exact functions for fewer generations. Smith et al. (1995) attempted a similar procedure, but instead of evaluating parts of the individuals, they evaluated only part of the population. In order to derive a fitness value for the offspring that is not evaluated they proposed two methods: the average inheritance, where the offspring's fitness is calculated as the average value from both parents, and the proportional inheritance where the average is weighed according to the similarity between the offspring and their parents.

Smith et al. (1995) applied the schema theory to explain why the genetic algorithm is not disturbed by the noisy fitness function and base their calculations on two characteristics of genetic algorithms:

- a child can inherit schemata common to both parents and in that case, the schemata fitness values are correctly determined. The update of the genetic algorithm then reflects the average fitness of the common schemata in the individuals;
- a child can inherit schemata that are only present in one parent and this leads to an approximate fitness value of those schemata.

In the case of fitness inheritance, the fitness values of those schemata are either deemed constant in the case of average fitness inheritance, or linearly related to the number of bits from the parents for proportional fitness inheritance.

For the bit-counting problem (One Max, page 58) they compared the behaviour of the conventional genetic algorithm with that of the inheritance techniques. When all individuals were evaluated, the optimum (all ones) is reliably reached after 10200 evaluations. In the case of the inheritance approaches this was attained after only 2640 function evaluations. They argued that this might be caused by the simplicity of the One Max problem and applied the same techniques to an aircraft routing problem. For this real-world application, the length of the flying route has to be minimised, while the aircraft has to avoid detection by a threat along the route. Each time the aircraft was detected, a penalty was added to the objective function. Their preliminary results were that the best results were obtained if only one individual at each generation is evaluated provided elitism is applied. This showed that fitness inheritance had the potential to improve GA performance.

Even though this approach seemed promising, it took another six years before this work was continued. Sastry et al. (2001) investigated the time to convergence, population sizing and the optimal proportion of inheritance for the One Max problem. The optimal proportion is the amount of inheritance that can be used in such a way that the number of function evaluations is minimised. They found that the time until population convergence is:

$$t_{conv} = \frac{\pi}{2I}\sqrt{\frac{l}{1-p_i}} \tag{13.1}$$

with I the selection intensity, i.e. a measure for the selective pressure, l the length of the chromosome and p_i the fraction of the individuals that inherit their value. The population size n can be written as:

$$n = -\frac{2^{k-1}\log(\psi)\sqrt{\pi}}{(1-p_i^3)}\sqrt{\sigma_f^2} \tag{13.2}$$

where k is the size of the building block, ψ is the failure rate or the rate that one accepts for not reaching the optimal value and σ_f^2 is the variance of the noisy fitness functions. This noise is caused by the incorrect fitness value for the inheritance individuals. Finally, p_i is once more the proportion of individuals that inherit fitness values. They also determined that the optimal proportion of inheritance p_i^* lies between $54\% - 55.8\%$. This is considerably less than what Smith et al. (1995) indicated: they calculated that for the One Max problem the proportion of inheritance could be raised up to 90% without loss of optimality. By building these analytical models, Sastry et al. (2001) were the first to provide a strong theoretical foundation for fitness inheritance.

13.2.2 Multiple objective fitness inheritance

Chen et al. (2002) extended the analytical model provided by Sastry et al. (2001) to multiple objective problems. They included an extra parameter M to account for the number of niches in the multiple objective problem. The problem for which the population sizing model and the time to convergence was derived is the bi-objective One Max problem. This problem is defined by :

$$\text{Maximise} \begin{cases} f_1(s, x_1) = l - d(s, x1) \\ f_2(s, x_2) = l - d(s, x2) \end{cases} \tag{13.3}$$

where string s is the string to be evaluated, x_1 and x_2 are two fixed reference strings, the string length is l and $d(s, x_i)$ is the Hamming distance between string s and string x_i. Chen et al. (2002) used as reference string x_1 all ones and as reference string x_2 all ones except for the first four bits. Their model for convergence time is:

$$t_{conv} = \frac{\pi}{2I}\sqrt{\frac{l}{1-p_i}}\sqrt{1+\frac{M-1}{l}} \tag{13.4}$$

The population size can be determined as follows:

$$n = -\frac{2^{k-1}\log(\psi)M\sqrt{\pi}}{(1-p_i^3)}\sqrt{\sigma_f^2+\sigma_N^2} \tag{13.5}$$

where σ_N^2 is the noise variance from the other niches. These models are very similar to those by Sastry et al. (2001) and if $M = 1$ they reduce to the single objective models.

Both models were experimentally tested on the bi-objective One Max problem. They found that when the inheritance proportion is smaller than 0.7, the results fit the predicted convergence and population-sizing model, but that for large inheritance proportions the models were no longer valid.

13.3 Research rationale

Other than the preliminary results for the aircraft routing problem by Smith et al. (1995), there are no real-world applications that use fitness inheritance. The question arises whether the efficiency enhancement technique can cope with real-world applications. To investigate this, both the average and proportional inheritance techniques will be tested initially on a test suite of functions provided by Zitzler (1999) and Zitzler et al. (2000). These functions pose different problems for genetic algorithms and have been used extensively in literature. Finally the proposed techniques will be applied to the case study of the harvest scheduling problem in Chapter 14.

13.4 Methodology

13.4.1 Zitzler's test suite

Zitzler (1999) and Zitzler et al. (2000) presented a test suite of problems that pose certain difficulties to multiple objective genetic algorithms. This test bed has been used extensively for the comparison of new algorithms and is regarded as a benchmark in many papers. Hence it is used here to examine the behaviour of genetic algorithms with and without fitness inheritance. Three functions from the test suite are used because they represent three different categories of test functions: convex, non-convex and discontinuous functions. The other three functions from the test suite also fall under one of these three categories. The three functions are listed below and are also depicted in Figs. 13.1 to 13.3.

- Test function 1 (h_1) has a convex Pareto-optimal front:

$$\begin{aligned} f_1(x_1) &= x_1 \\ g_1(x_2, \ldots, x_n) &= 1 + \frac{9 \cdot (\sum_{i=2}^{n} x_i)}{n-1} \\ h_1(f_1, g_1) &= 1 - \sqrt{\frac{f_1}{g_1}} \end{aligned} \tag{13.6}$$

where $n = 30$ and $x_i \in [0, 1]$. The Pareto-optimal front is formed when g_1 equals 1.

- Test function 2 (h_2) is the non-convex counterpart of test function 1:

$$\begin{aligned} f_2(x_1) &= x_1 \\ g_2(x_2, \ldots, x_n) &= 1 + \frac{9 \cdot (\sum_{i=2}^{n} x_i)}{n-1} \\ h_2(f_2, g_2) &= 1 - \left(\frac{f_2}{g_2}\right)^2 \end{aligned} \tag{13.7}$$

where $n = 30$ and $x_i \in [0, 1]$. The Pareto-optimal front is formed when g_1 equals 1.

- Test function 3 (h_3) tests whether a genetic algorithm is able to cope with discreteness in the Pareto-optimal front:

$$f_3(x_1) = x_1$$

$$g_3(x_2, \ldots, x_n) = 1 + \frac{9 \cdot (\sum_{i=2}^{n} x_i)}{n-1} \tag{13.8}$$

$$h_3(f_3, g_3) = 1 - \sqrt{\frac{f_1}{g_1}} - \left(\frac{f_1}{g_1}\right) \sin(10\pi f_1) \tag{13.9}$$

where $n = 30$ and $x_i \in [0, 1]$. The Pareto-optimal front is formed when g_1 equals 1. The sine function introduces discontinuity in the Pareto-optimal front but not in the objective space.

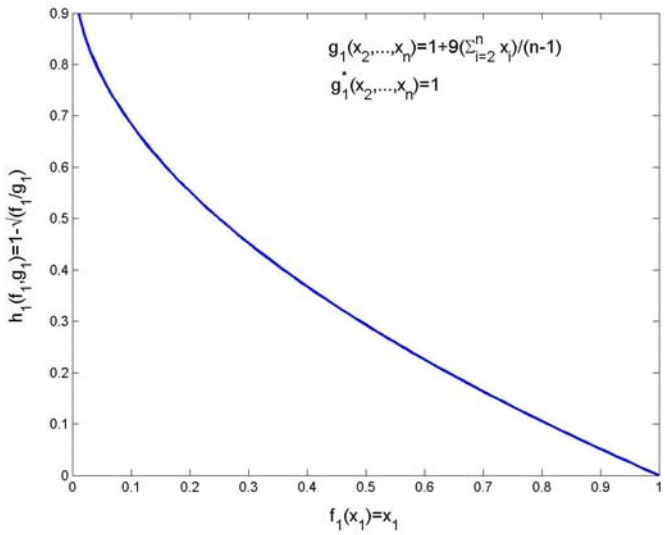


Figure 13.1: Test function 1 is a convex function

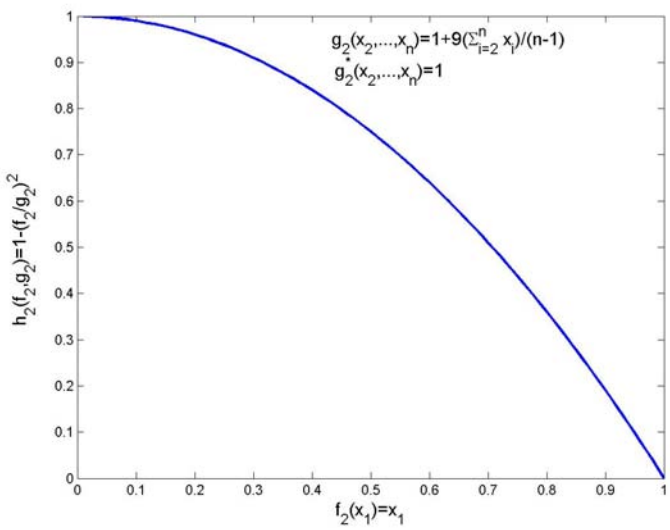


Figure 13.2: Test function 2 is the non-convex counterpart of the first test function

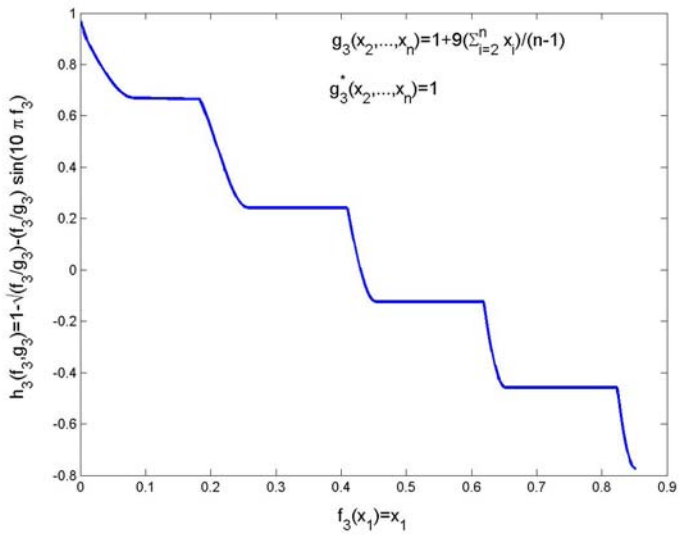


Figure 13.3: Test function 3 is a discontinuous function. The discontinuity is only present in the objective space

13.4.2 Parameter settings

The experiments were performed using a genetic algorithm with binary tournament selection, one-point crossover with crossover probability of 0.8 and a uniform mutation rate of 0.01. The population size was set to 100 and the number of generations was set to 200. These settings are the same as in Zitzler (1999) with as sole difference the number of generations. The encoding of the decision vector was also the same as in Zitzler (1999): an individual is represented as a bit vector where each parameter x_i is represented by 30 bits. The crowding distance assignment procedure was used for sharing. The fitness assignment applied was proposed by Deb et al. (2000) and equals the number of solutions that dominate the current solution. All experiments were repeated 10 times. Since the same settings as in Zitzler (1999) were chosen, it is possible to check the implementation of the genetic algorithm implemented in this dissertation as well as to test the behaviour of the fitness inheritance strategies. The optimal proportion of 0.54 is used to test the behaviour of the genetic algorithms. The maximum number of fitness evaluations for the inheritance approach should be the same as for the non-inheritance approach, therefore the number of generations is doubled to 400 generations to allow fair comparison.

13.5 Solving the test functions with fitness inheritance

For the analysis of the fitness inheritance, the three previously described test functions are used. The Pareto-fronts achieved by the evolutionary algorithm without fitness inheritance, as well as the Pareto-fronts for the two genetic algorithms with average or proportional fitness inheritance, will be presented. The Pareto-optimal front is drawn for comparison purposes. The unary measures generational distance, error ratio, spacing, spread and hypervolume measure (Section 8.4.2 on p. 100) are used to compare the different approaches. These measures are calculated for 100 function evaluations for the non-inheritance approach and every 50 function evaluations for the inheritance techniques. For all three algorithms this is at every generation. The difference in means of the above measures over time is determined by a One Way ANOVA test if the data is normally distributed and homoscedastic, otherwise a non-parametric Kruskal-Wallis test is applied. All tests are at a significance level of 95%.

13.5.1 Convex functions

Visual interpretation The standard genetic algorithm is capable of finding a Pareto-front very close to the optimal Pareto-front for the first convex test function (Fig. 13.4). Both inheritance strategies approximate the Pareto-optimal front well. If there is no inheritance the variance between the different points is low; on the other hand the points from the proportional inheritance approach are much more scattered. The solutions from the two inheritance approaches are also much more concentrated near the point $(f_1, h_1) = (0, 1)$.

Generational distance The evolution of generational distance over the function evaluations for the standard genetic algorithm is very similar to that of the two inheritance techniques (Fig. 13.5). In all three cases, the decrease in generational distance is initially strong but as the population converges this becomes much smaller. The data was statistically analysed using a One Way ANOVA test because on all occasions the data was normally distributed and homoscedastic. All p-values for the relevant tests of normality (Shapiro-Wilk), homoscedasticity (Levene's test) and One Way ANOVA are listed in Table 13.1. Before 1800 function evaluations the generational distance is significantly lower for the inheritance techniques than for the regular genetic algorithm ($p = 0.000$). From then on, there is no difference anymore ($p = 0.067 - 0.052$) until 16800 function evaluations. After that, the generational distance for the non-inheritance approach is significantly lower than for the proportional inheritance approach but not than the average inheritance technique ($p = 0.037$) according to Tukey's posthoc-test.

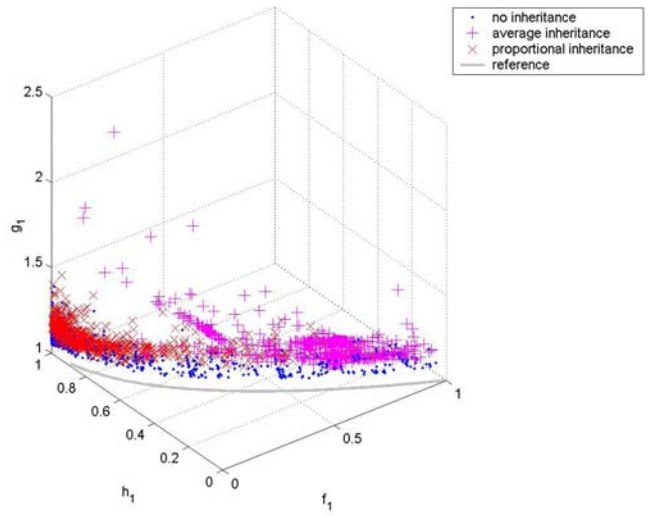


Figure 13.4: The overall Pareto-front for test function 1. On the x-axis $F_1 = f_1$, on the y-axis $F_2 = h_1$ and on the z-axis $F_3 = g_1$

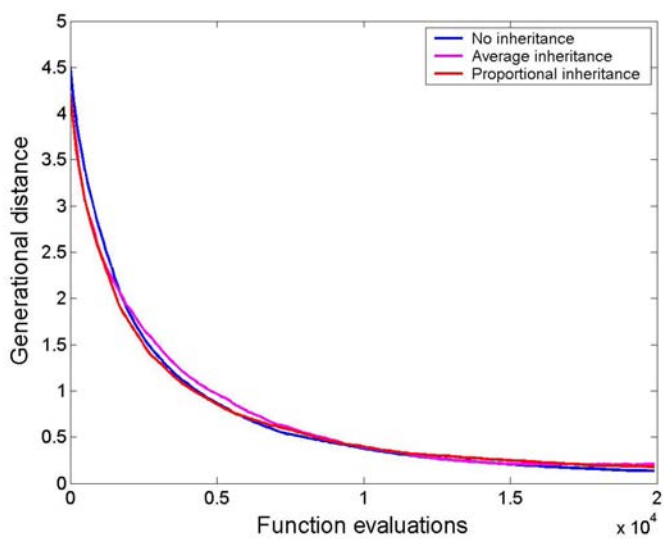


Figure 13.5: Evolution of generational distance over the number of function evaluations for the first test function

Table 13.1: p-values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for generational distance (non = no inheritance, average = average inheritance and proportional = proportional inheritance)

Evaluations	Group	Shapiro-Wilk	Levene	One Way ANOVA
100	non	0.971	0.7761	0.000
	average	0.662		
	proportional	0.380		
1800	non	0.688	0.776	0.067
	average	0.885		
	proportional	0.695		
16800	non	0.482	0.919	0.052
	average	0.984		
	proportional	0.493		
16900	non	0.470	0.902	0.037
	average	0.910		
	proportional	0.407		

Error ratio The previous findings are confirmed by the error ratio: initially this is 1 for all algorithms, but after 10000 function evaluations the error ratio declines quickly. The error ratio of the inheritance techniques is higher than that of the standard genetic algorithms, especially near the end of the evolution. Apparently the convergence towards the front is hampered by the fitness inheritance. The effect of the inheritance on the error ratio was also investigated statistically: for each 100 function evaluations, the Kruskal-Wallis test was performed because the data was not normally distributed for all cases, and the data was not homoscedastic. According to this test, there is no significant difference present between the three approaches, but near the end of the evolution, the test statistic p approximates 0.05. This indicates that if the number of function evaluations increases even more, the inheritance techniques might perform worse than the regular technique.

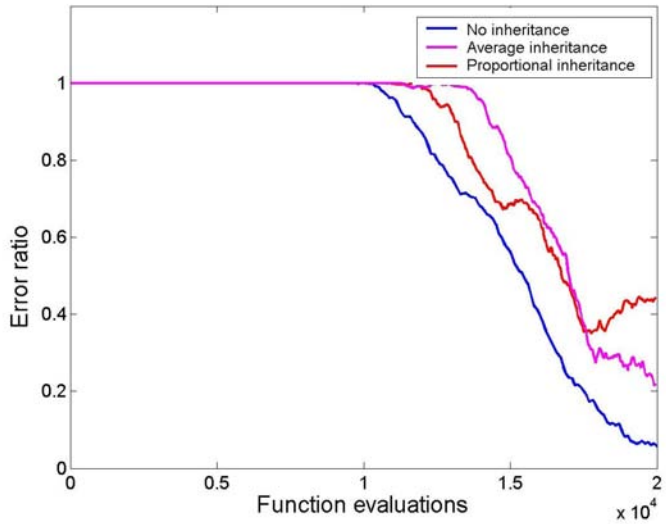


Figure 13.6: Evolution of error ratio over the number of function evaluations for the first test function

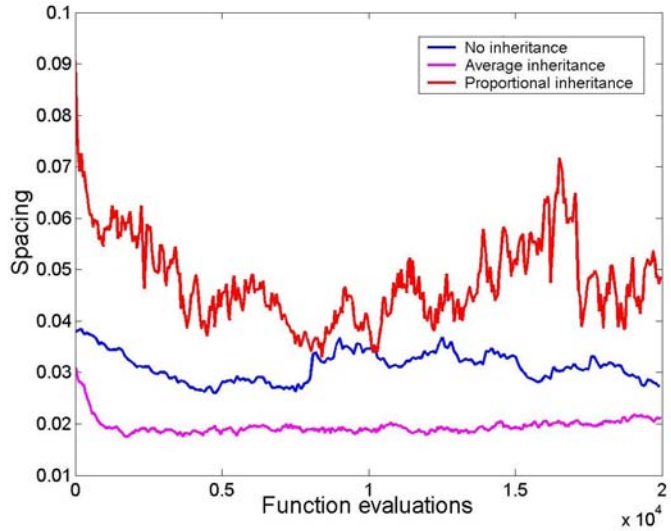


Figure 13.7: Evolution of spacing over the number of function evaluations for the first test function

Spacing and spread Spacing and spread on the other hand are not so much influenced by the inheritance techniques: in all cases the crowding distance operator ensures that the spacing and spread remain more or less equal over the complete evolution. The effect of the inheritance techniques was again tested using a Kruskal-Wallis test as once more the data was not normally distributed and homoscedastic (Table 13.2). Initially the spacing measure for the non-inheritance approach is significantly lower than for the other two approaches ($p = 0.1 - 0.2$), but after 175000 function evaluations there is no longer a difference between non-inheritance and average inheritance. In the last phase, it is inconclusive whether there is a difference or not: $p = 0.047 \approx 0.05$.

Table 13.2: p-values for normality, homoscedasticity and Kruskal-Wallis for different numbers of function evaluations for spacing for the first test function (non = no inheritance, average = average inheritance and proportional = proportional inheritance)

Evaluations	Group	Shapiro-Wilk	Levene	Kruskal-Wallis
17500	non	0.377	0.481	0.422
	average	0.010		
	proportional	0.010		

For spread, the Kruskal-Wallis test shows clearly that in all cases there are significant differences. For all cases $p = 0$ even though this is not clear from Fig. 13.8. All p-values for the relevant tests of normality (Shapiro-Wilk), homoscedasticity (Levene's test) and One Way ANOVA and Kruskal-Wallis are listed in Table 13.3 . Initially all groups are significantly different from each other ($p = 0.000$), and the standard genetic algorithm performs the worst in terms of spread. At 6000 generations, they have more or less the same performance but still the Kruskal-Wallist test indicates significant differences ($p = 0.002$). Even at the end of the evolution there is still a significant difference between the proportional inheritance and no-inheritance strategy ($p = 0.000$) according to Tukey's posthoc test. It seems that the amount of spacing and spread is highly determined by the initial spacing in the population. Especially spacing does not change much over the course of the genetic algorithm. Spread is lowered somewhat more than spacing but still remains much the same. This might indicate that the crowding distance measure is more a 'diversity preserving' measure than a 'diversity stimulating' measure.

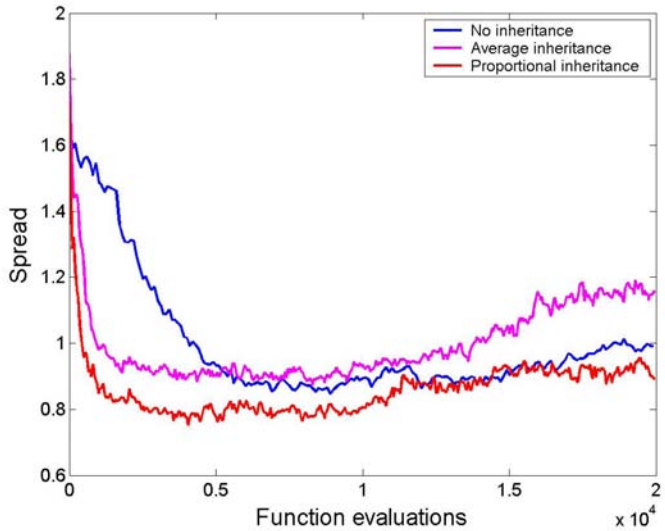


Figure 13.8: Evolution of spread over the number of function evaluations for the first test function

Table 13.3: p-values for normality, homoscedasticity and Kruskal-Wallis for different numbers of function evaluations for spacing for the first test function, for 20000 function evaluations One Way ANOVA is used (non = no inheritance, average = average inheritance and proportional = proportional inheritance)

Evaluations	Group	Shapiro-Wilk	Levene	Kruskal-Wallis
3200	non	0.200	0.010	0.000
	average	0.193		
	proportional	0.644		
6000	non	0.390	0.216	0.002
	average	0.017		
	proportional	0.636		
10000	non	0.442	0.004	0.000
	average	0.455		
	proportional	0.583		
20000	non	0.428	0.058	0.000
	average	0.483		
	proportional	0.713		

Hypervolume The fact that a higher error ratio is present with the inheritance techniques is confirmed by the hypervolume measure: this value is for both inheritance techniques lower than for the standard genetic algorithm (Fig. 13.9). All p-values for the relevant tests of normality (Shapiro-Wilk), homoscedasticity (Levene's test) and One Way ANOVA are listed in Table 13.4. Up until 4000 function evaluations this difference is not significant. Between 4000 and 16000 there is a difference between the non-inheritance and the average approach according to Tukey's posthoc-test. Beyond this point this posthoc-test indicates a significant difference between the non-inheritance approach on the one hand and the two inheritance strategies on the other hand.

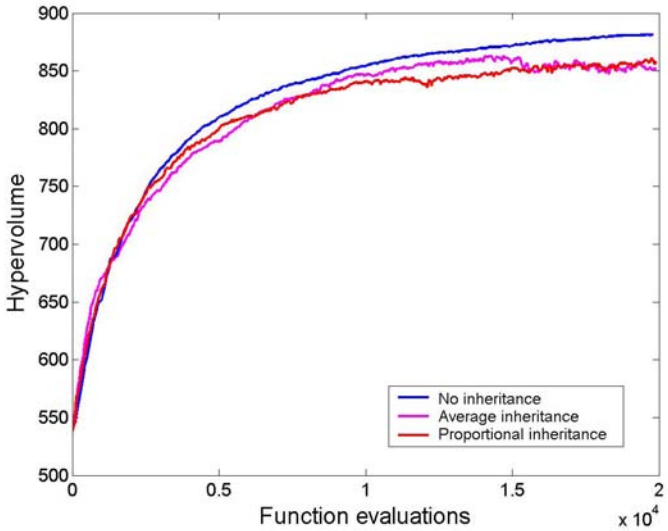


Figure 13.9: Evolution of hypervolume over the number of function evaluations for the first test function

Table 13.4: p-values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for hypervolume for the first test function (non = no inheritance, average = average inheritance and proportional = proportional inheritance)

Evaluations	Group	Shapiro-Wilk	Levene	One Way ANOVA
4000	non	0.960	0.180	0.000
	average	0.482		
	proportional	0.773		
16000	non	0.930	0.719	0.002
	average	0.482		
	proportional	0.773		

Conclusion Between the two fitness inheritance techniques there is little difference in performance. The generational distance, error ratio and hypervolume are not always significantly different, and in most cases very close to the values of the non-inheritance approach. Spacing and spread are even better for the average fitness inheritance than for the standard genetic algorithm. Overall fitness inheritance can speed up the optimisation process for convex functions while maintaining the same behaviour as the regular genetic algorithm.

13.5.2 Non-convex functions

Visual interpretation The algorithms based on fitness inheritance are not suitable for solving non-convex optimisation problems (Fig. 13.10). The points found by the inheritance approaches are much further away from the Pareto-optimal front. Furthermore, there are many points near the origin of the x-axis. This phenomenon was also observed in the first test function.

Generational distance The generational distance is for both inheritance techniques higher than for the non-inheritance approach and this is true for the complete evolution (Fig. 13.11). Between the two inheritance techniques, however, there is visually no difference. The data was statistically analysed using a One Way ANOVA, all p-values for the relevant tests of normality (Shapiro-Wilk), homoscedasticity (Levene's test) and One Way ANOVA are listed in Table 13.5. At 600 function evaluations, the ANOVA test value shows that there are no significant differences between the three algorithms ($p = 0.21$). At 700 function evaluations however this changes, there is no conclusion possible as $p = 0.050$. At 800 evaluations however the p-value is 0.018. Tukey's posthoc-test indicates that there is a significant difference between the non-inheritance and the average inheritance approach, but is inconclusive for the proportional approach. This remains the same until 1600 function evaluations. At that point, Tukey's posthoc-test shows that there is a significant difference between the non-inheritance approach on one hand and both inheritance techniques on the other hand.

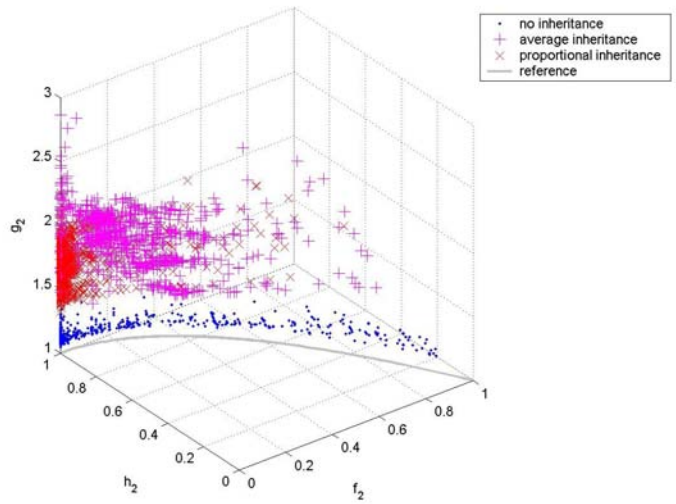


Figure 13.10: The overall Pareto-front for test function 2. On the x-axis $F_1 = f_2$, on the y-axis $F_2 = h_2$ and on the z-axis $F_3 = g_2$

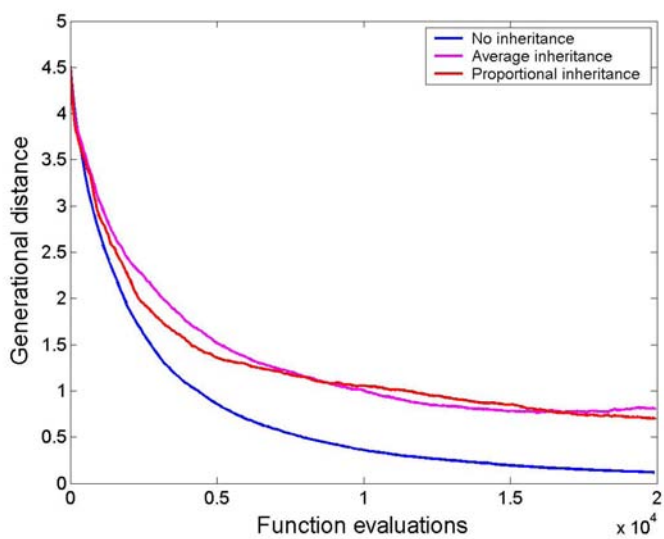


Figure 13.11: Evolution of generational distance over the number of function evaluations for the second test function

Table 13.5: p-values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for generational distance for the second test function(non = no inheritance, average = average inheritance and proportional = proportional inheritance)

Evaluations	Group	Shapiro-Wilk	Levene	One Way ANOVA
600	non	0.344	0.098	0.210
	average	0.491		
	proportional	0.511		
700	non	0.668	0.111	0.050
	average	0.761		
	proportional	0.487		
800	non	0.779	0.183	0.018
	average	0.989		
	proportional	0.595		
1600	non	0.135	0.497	0.001
	average	0.499		
	proportional	0.357		

Error ratio The error ratio of the two inheritance techniques is much worse than for the non-inheritance approach. Whereas the error ratio drops after 10000 evaluations for the regular genetic algorithm, the error ratio for the two inheritance approaches stays 1 for the complete duration of the genetic algorithm. As their error ratio remains constant, no statistical analysis is possible. Still, from Fig. 13.12 follows clearly that the inheritance techniques do not perform well in terms of error ratio, and that they fail to reach the Pareto-optimal front for a non-convex function.

Spacing and spread Spacing and spread are again not influenced by the inheritance techniques. The crowding distance operator ensures once more that the spacing and spread remains more or less equal over the complete evolution (Figs. 13.13 and 13.14). The spacing of the non-inheritance approach is less than that of the two inheritance approaches, and the spacing of the average inheritance is better than that of the proportional inheritance technique (Fig. 13.13). Before 8000 function evaluations the difference between the spacing is significant only between the group of non-inheritance and proportional inheritance. Beyond this point, there is a difference between the group of non-inheritance and the two inheritance approaches (Table 13.6).

In the case of spread, initially the non-inheritance approach performs worse than the two other approaches. This difference is not significant with respect to the average inheritance approach but is significant with respect to the proportional inheritance approach until 6000 function evaluations ($p = 0.000$).

After 6000 function evaluations, there is no longer a difference between the non-inheritance and inheritance approach but there is a significant difference between the two inheritance procedures: according to Tukey's posthoc test, average inheritance has a significantly higher spread than the proportional inheritance ($p = 0.013$).

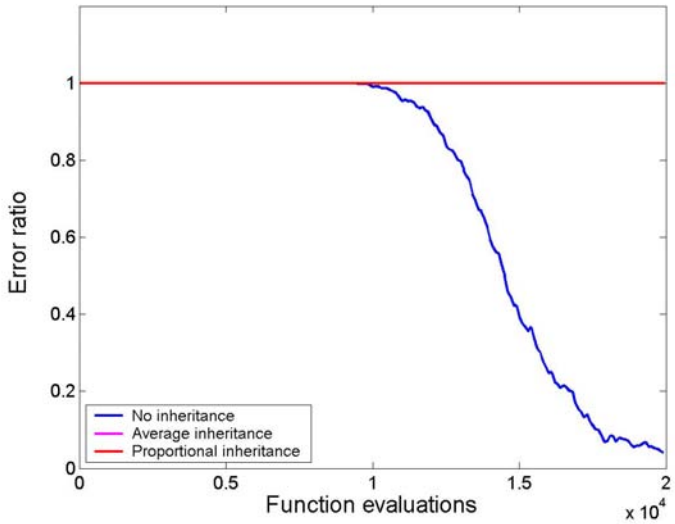


Figure 13.12: Evolution of error ratio over the number of function evaluations for the second test function

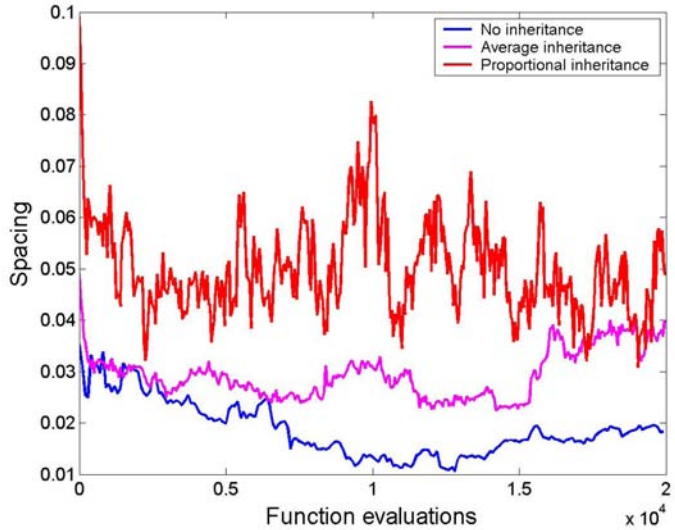


Figure 13.13: Evolution of spacing over the number of function evaluations for the second test function

Table 13.6: p-values for normality, homoscedasticity and Kruskal-Wallis for different numbers of function evaluations for spacing for the second test function (non = no inheritance, average = average inheritance and proportional = proportional inheritance)

Evaluations	Group	Shapiro-Wilk	Levene	Kruskal-Wallis
4800	non	0.010	0.151	0.021
	average	0.655		
	proportional	0.438		
8000	non	0.010	0.288	0.003
	average	0.203		
	proportional	0.073		
16000	non	0.010	0.632	0.012
	average	0.648		
	proportional	0.048		

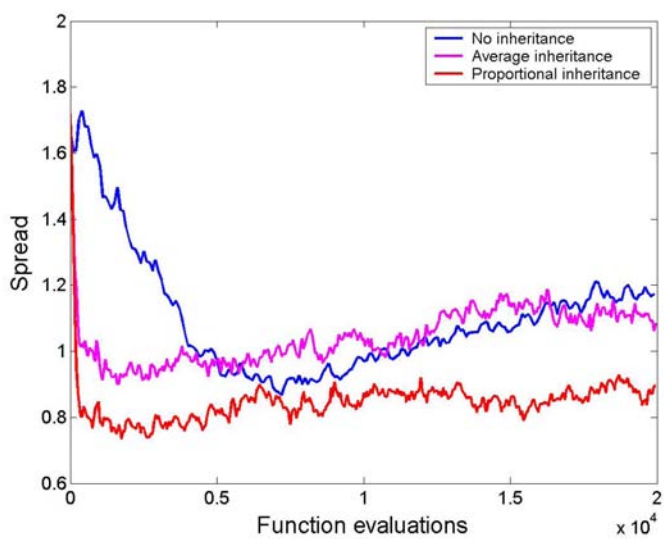


Figure 13.14: Evolution of spread over the number of function evaluations for the second test function

Table 13.7: p-values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for spread for the second test function (non = no inheritance, average = average inheritance and proportional = proportional inheritance

Evaluations	Group	Shapiro-Wilk	Levene	One Way ANOVA
4000	non	0.247	0.422	0.000
	average	0.527		
	proportional	0.728		
6000	non	0.175	0.645	0.013
	average	0.468		
	proportional	0.696		
10000	non	0.764	0.626	0.0000
	average	0.424		
	proportional	0.448		

Hypervolume The hypervolume measure again confirms the findings of the error ratio and the generational distance: this value is lower for both inheritance techniques than for the standard genetic algorithm. Up until 1400 function evaluations this difference is not significant between the three groups ($p = 0.054$) (Table 13.8). Between 1400 and 1800 function evaluations, the non-inheritance approach is only significantly different ($p = 0.000$) from the average approach, but later on Tukey's posthoc-test indicates that all groups are significantly different from each other.

Conclusion It can be concluded that for non-convex functions, after a few fitness evaluations, the inheritance techniques converge slower to the Pareto-optimal front than the non-inheritance approach. Maintaining spread or spacing is again not affected by the inheritance. This means that for non-convex functions, inheritance techniques are less suitable to use.

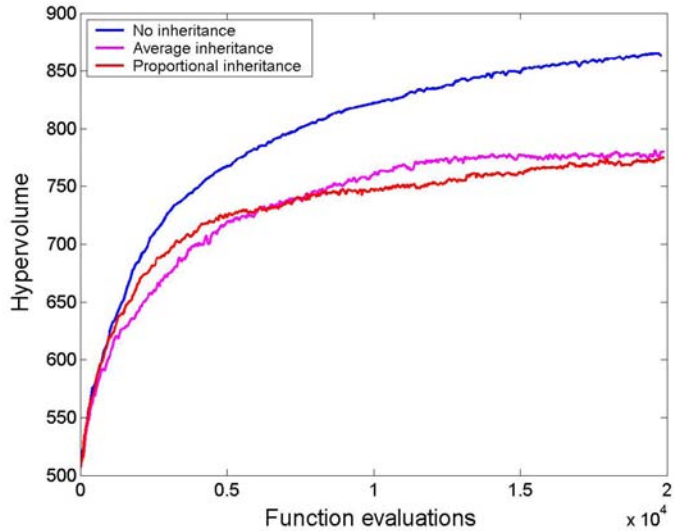


Figure 13.15: Evolution of hypervolume over the number of function evaluations for the second test function

Table 13.8: p-values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for hypervolume for the second test function (non = no inheritance, average = average inheritance and proportional = proportional inheritance)

Evaluations	Group	Shapiro-Wilk	Levene	One Way ANOVA
1400	non	0.692	0.269	0.054
	average	0.349		
	proportional	0.871		
1800	non	0.951	0.324	0.000
	average	0.378		
	proportional	0.560		

13.5.3 Discontinuous functions

Visual interpretation The inheritance techniques are unable to solve the third discontinuous problem (Fig. 13.16). Both inheritance techniques yield the same result but their Pareto-fronts approximate the Pareto-optimal front in a linear way. This is particularly pronounced in the case of the average fitness inheritance. The solutions are also unevenly distributed along the fronts. Apparently, the noise resulting from the linear interpolation of the fitness values of the offspring results in a high disturbance of the genetic algorithm.

Generational distance The generational distance is similar for all approaches for the complete evolution (Fig. 13.17). Between the two inheritance techniques, however, there is visually no difference. The data was statistically analysed using a One Way ANOVA, all p-values for the relevant tests of normality (Shapiro-Wilk), homoscedasticity (Levene's test) and One Way ANOVA are listed in Table 13.9. Early in the evolution, there is not much difference between the approaches, but this according to Tukey's posthoc-test becomes significant among all groups after 10000 function evaluations.

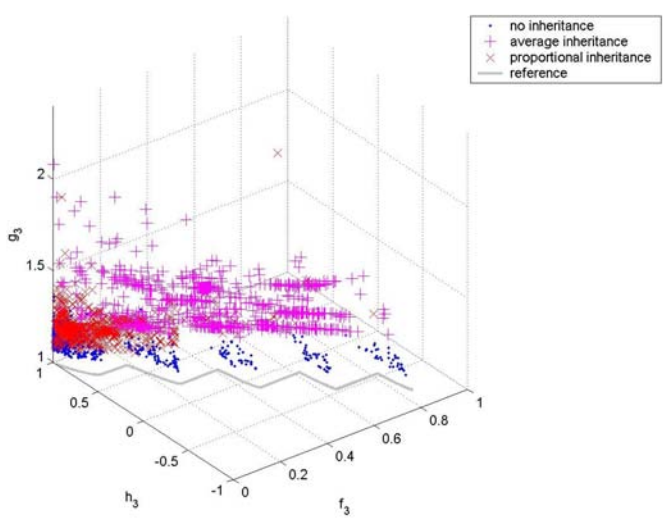


Figure 13.16: The overall Pareto-front for test function 3. On the x-axis $F_1 = f_2$, on the y-axis $F_2 = h_2$ and on the z-axis $F_3 = g_2$

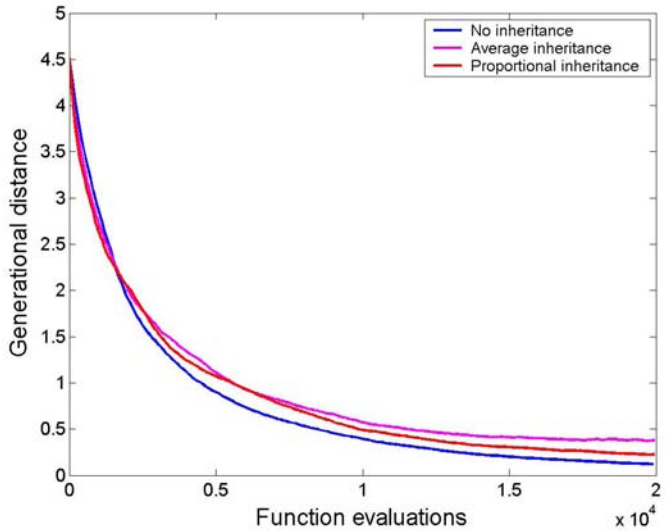


Figure 13.17: Evolution of generational distance over the number of function evaluations for the third test function

Table 13.9: p-values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for generational distance for the third test function (non = no inheritance, average = average inheritance and proportional = proportional inheritance)

Evaluations	Group	Shapiro-Wilk	Levene	One Way ANOVA
10000	non	0.348	0.846	0.000
	average	0.828		
	proportional	0.516		

Error ratio The error ratio of the two inheritance techniques is much worse than for the non-inheritance approach. Whereas the error ratio drops after 10000 evaluations for the regular genetic algorithm, the error ratio for the average inheritance approaches stays 1 for the complete duration of the genetic algorithm. The error ratio of the proportional inheritance technique does decline after 16000 function evaluations but much less than in the case of non-inheritance approach.

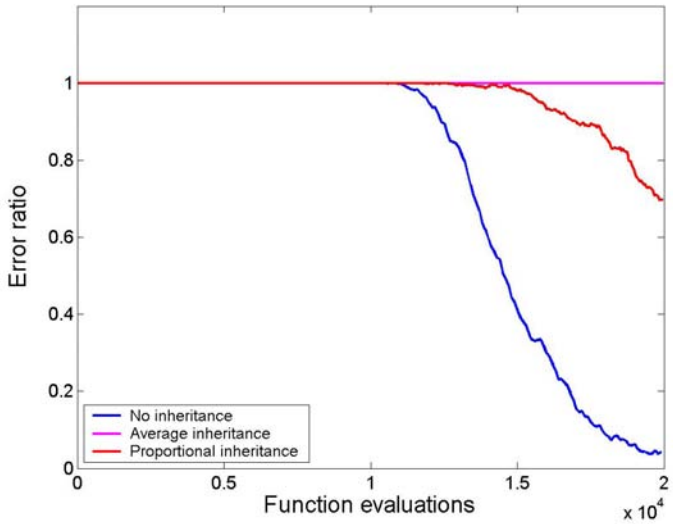


Figure 13.18: Evolution of error ratio over the number of function evaluations for the third test function

Spacing and spread As was the case with the two other functions, the spacing and spread are not affected by the inheritance. The spacing of the average inheritance is again much lower than that of the others. From Fig. 13.16 can be concluded that once more the points from the average inheritance are well distributed, but that proportional inheritance shows a higher degree of scatter. This is confirmed by the spread measure: the spread measures for the three approaches are similar and the values for the average technique are now in the range of the other two procedures. However, because the distance to the extreme points is taken into account, this compensates for the low values of the spacing measure.

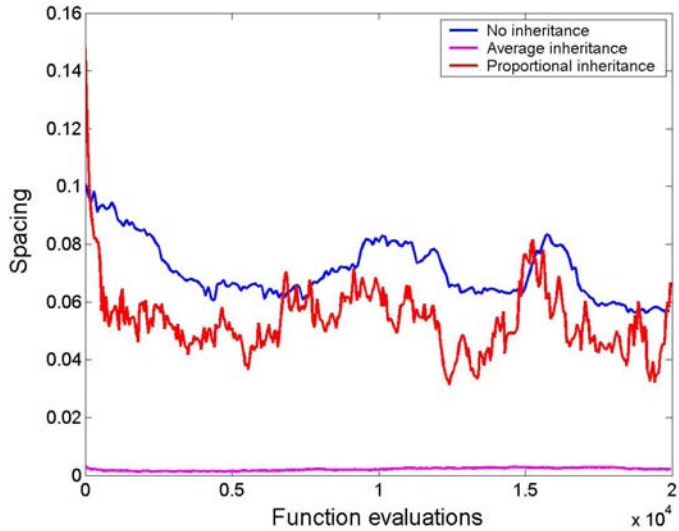


Figure 13.19: Evolution of spacing over the number of function evaluations for the third test function

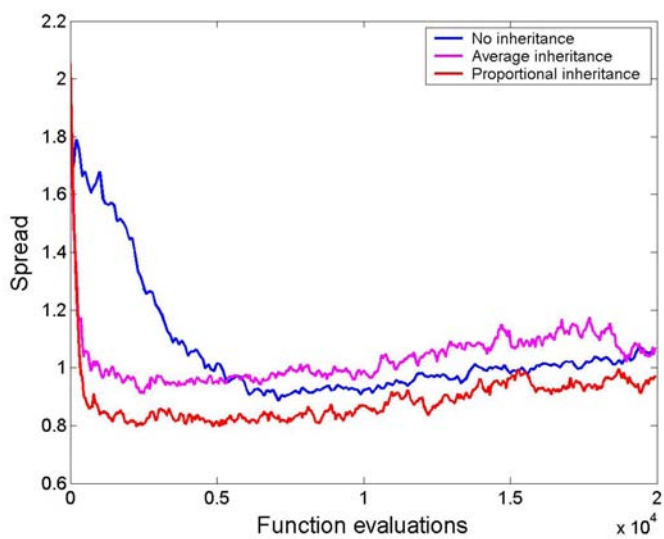


Figure 13.20: Evolution of spread over the number of function evaluations for the third test function

Hypervolume Finally the hypervolume measure confirms the findings with the error ratio and the generational distance (Fig. 13.21): this value is lower for both the inheritance techniques than for the standard genetic algorithm. Up until 2700 function evaluations, this difference is not significant between the three groups (Table 13.10). After 2700 there is a difference between the non-inheritance and the two inheritance approaches, but these two do not differ from each other until the very end of the evolution.

Conclusion As was the case for non-convex functions, the inheritance techniques are not useful for discontinuous functions. Their behaviour in terms of generational distance, error ratio and hypervolume is significantly worse than for the non-inheritance approach over the complete evolution. Again, spacing and spread are not affected by the fitness inheritance.

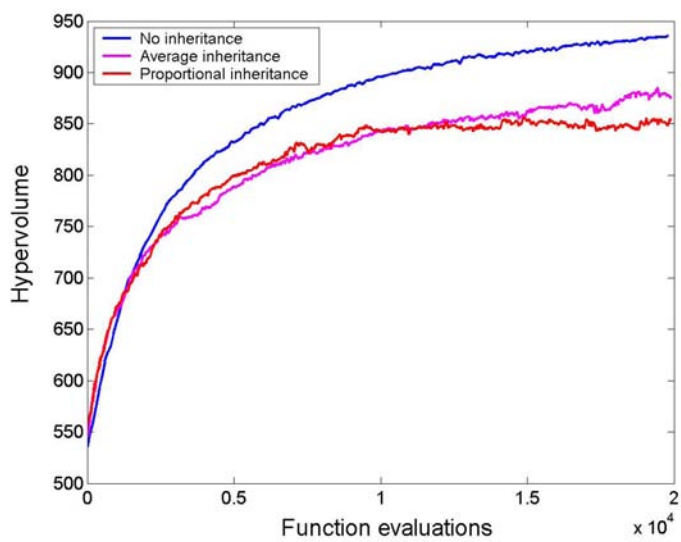


Figure 13.21: Evolution of hypervolume over the number of function evaluations for the second test function

Table 13.10: p-values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for hypervolume for the third test function (non = no inheritance, average = average inheritance and proportional = proportional inheritance)

Evaluations	Group	Shapiro-Wilk	Levene	One Way ANOVA
2700	non	0.485	0.068	0.040
	average	0.491		
	proportional	0.444		

13.5.4 General conclusion

Fitness inheritance efficiency enhancement techniques can be used in order to reduce the number of fitness evaluations provided that the Pareto-front is convex and continuous. If the surface is not convex, the fitness inheritance strategies fail to reach the Pareto-optimal front. As to the inheritance strategy, it is safer to choose the proportional approach, because in most cases the proportional inheritance performs better in terms of generational distance, error ratio and hypervolume.
If real-world practitioners want to use fitness inheritance, it is advisable to check beforehand what the nature (convex, non-convex,...) of the Pareto-front is. This can be achieved by solving the problem with a classical multiple objective GA for a low number of generations and then switch to fitness inheritance techniques if there is sufficient indication that the Pareto-front is convex and continuous. They should also consider other techniques for speeding up the optimisation process. These might include local fitness recalculation, where the fitness value of the individuals is updated from the value from the parents by recalculating only where the children differ from the parents. As the harvest scheduling problem is convex, the proportional fitness inheritance technique will be applied in the final chapter.

Chapter 14

Case study: Fitness inheritance for harvest scheduling

As a final case study, fitness inheritance is used to speed up the optimisation process for the bi-objective harvest scheduling problem. As this problem is convex, fitness inheritance should be a feasible approach.

14.1 Methodology

The same input data and Forestry Commission production tables as in Chapters 5 and 9 are applied. The population size is set to 100, the number of generations without fitness inheritance to 200, and with proportional inheritance to 400. Average inheritance was not tested because from the test function followed that its behaviour was either similar or worse than that of proportional inheritance. One-point crossover is used with a probability of 0.8 and uniform mutation with a probability of 0.01. Integer encoding is used, as this proved to be the best encoding strategy for the harvest scheduling problem. Binary tournament selection was used and the crowding distance operator ensured sharing.

14.2 Results and discussion

From Fig. 14.1 follows that after the same number of function evaluations, the attainment surface from the inheritance approach equals that of the non-inheritance approach. This is confirmed by calculating the hypervolume measure. The data is normally distributed ($p = 0.99 > 0.05$) and homoscedastic ($p = 0.685$). From the Student t-test test statistic follows that there is no significant difference between the two groups ($p = 0.209$).

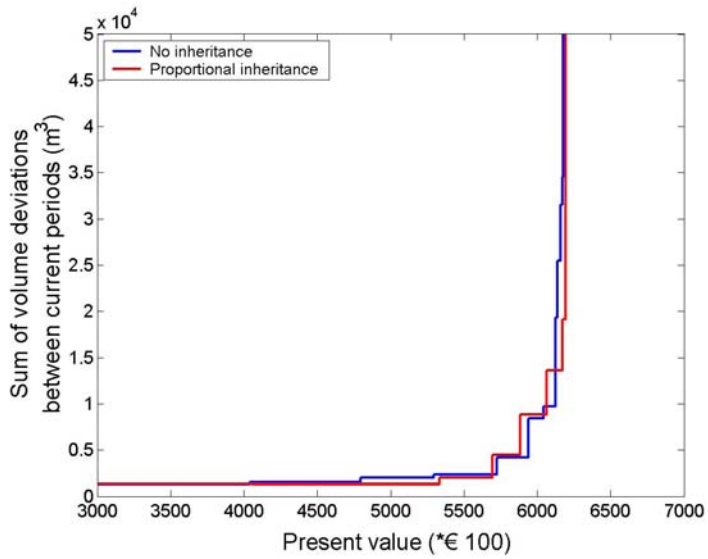


Figure 14.1: Attainment surfaces for the harvest scheduling problem for non-inheritance and proportional inheritance approaches

14.3 Conclusions

The behaviour of the inheritance approach is similar to that of the standard genetic algorithm. However, this should be relativised because in reality the same number of function evaluations are necessary to obtain the same Pareto-front.

Part IV

Conclusion

Chapter 15

Summary and conclusions

15.1 Research question 1

A forest management problem can be formulated as a Type I Model or a Type II Model. Both of these models have their merits for timber harvest and activity scheduling. If one considers other objectives than purely economic ones, a Type I Model has a slight advantage in comparison with the other model, because the management unit can be easily tracked from the beginning to the end of the planning horizon. Still, the Model I formulation as originally conceived is insufficient to include ecological models or any other models that need spatial data as input, because it is stratum-based and thus the link between the management unit and its location is not straightforward. This traditional model has been used extensively to solve harvest scheduling problems and is translated very easily into a linear programming format.

In order to guarantee spatial integrity, the decision variables that were originally continuous are transformed into integer variables. This leads to very large combinatorial problems that cannot be solved using exact techniques such as integer or mixed-integer programming for real-world problems, even with today's computer power. Heuristics have been proposed as a means to handle these complicated optimisation problems, and are indeed capable of solving them.

The integration of the optimisers together with a geographical information system to include the spatial information in an explicit way has also been advocated. Until now, GIS is only used for classification purposes prior to the optimisation process or for visualisation of the alternatives after the process. An online integration of GIS and optimisers in forest management has not yet been reported.

All the previously described techniques also require that there is only one objective function. The multiple objectives have to be combined using for example the weighing method or the constraint method. This is difficult to achieve if the

objectives are incommensurable. If a production possibility frontier is required, then for these approaches the optimiser has to be run repeatedly. Until today, there has been no use of a truly multi-objective optimiser in forest management.

15.2 Research question 2

15.2.1 Harvest scheduling as a single objective problem

A Type I harvest scheduling problem is initially solved using a single objective genetic algorithm. Because the Type I problem has two objectives (maximisation of present value and minimisation of the deviations between the successive harvesting periods) these objectives were combined using the weighing method prior to the optimisation. Because the Pareto-optimal front is searched for, the weights were linearly distributed on the interval $]0, 1]$. Using these weights did not lead to evenly spaced solutions along the Pareto-front and very little information could be gained. Therefore two additional weights were chosen: $w = 0.01$ and $w = 0.05$. When the present value was 100 times more important, the slope of the Pareto-front became very steep. This implies that an unknowing user investigating the effect of the weights on the two objectives might lose a considerable amount of information on the production possibility frontier if the weights are linearly distributed on a small interval.

The age distribution of the forest is normalised due to the implicit volume control of the even-flow objective. The genetic algorithm produces harvesting plans enforcing a balanced age distribution. The present value obtained with the relaxed even-flow constraints amounts to 72.9% of the total maximum attainable present value. The average volume and the volume harvested per year is also affected by the even-flow objective: it declines as this objective becomes more important.

A practical drawback from using weights is that it is very cumbersome. Rerunning the genetic algorithm or any single objective optimiser for several weight combinations, is a tedious job and requires large amounts of computing time.

15.2.2 Extension to multiple objectives

A comparative study Before the Type I harvest scheduling problem was solved as a bi-objective problem, two genetic algorithms commonly used in the domains of control or civil engineering, the Multiple Objective Genetic Algorithm (MOGA) and the Non-dominated Sorting Algorithm-II (NSGA-II), were compared on a benchmark problem with a known Pareto-optimal front. Their performance was also compared with that of a random search strategy. The objectives for this benchmark problem were: (1) maximise timber revenue and (2) maximise the recreational value. The Pareto-optimal front is non-convex, and therefore it cannot be solved using the weighing method.

Both MOGA and NSGA-II have a better performance than a random search strategy. They both approximate the Pareto-optimal front well, but suffer from a lack of spread. Especially the NSGA-II is not capable of finding the more extreme solutions. Because of this lack of spread, the implementation of the two algorithms was tested on another non-convex function. From this experiment followed that both algorithms have a very good spread over the complete Pareto-front and thus the lack of spread is only caused by the nature of the forest management problem.

NSGA-II is capable of approximating the Pareto-optimal front faster than MOGA and the solutions found by NSGA-II are more evenly spaced along the front. If the distance from the extreme solutions in the Pareto-front to the most extreme Pareto-optimal ones are included, MOGA scores better than NSGA-II but not significantly. This highlights that MOGA finds more solutions near the extremes than NSGA-II does. The variance between the several runs in generational distance is smaller for NSGA-II than for MOGA and this shows that NSGA-II is more robust than MOGA in terms of approximation of the Pareto-optimal front.

When the algorithms are compared in terms of both spread and closeness, the hypervolume indicates that the NSGA-II dominates a higher portion of the solution space than MOGA does. Using the attainment surface similar conclusions can be drawn: the Mann Whitney test procedure shows that NSGA-II dominates MOGA in the larger portion of the search space. Given these results, NSGA-II was chosen to solve the harvest scheduling problem.

Harvest scheduling as a multiple objective problem The Type I harvest scheduling problem is now solved as a bi-objective problem using NSGA-II. Using such an algorithm clearly speeds up the optimisation process: in order to find evenly spaced solutions along the Pareto-front a single run suffices. The attainment surface of NSGA-II is similar to the attainment surface obtained with the single-objective optimiser, but in order to reach this only one fifth of the function evaluations is necessary. The minimal population size needed to solve this problem is 750, increasing it even more does not improve the approximation of the Pareto-optimal front.

Validity of the harvest scheduling plans For both optimisers the effect of the plans on the age structure of the forest is the same: if the even-flow objective becomes more important, the age structure resembles that of a normal forest. This is caused by the volume control, even though this is not explicitly mentioned in the objective functions. If the even-flow objective is relaxed, the harvesting of the stands is postponed to later harvesting periods. The Pareto-front is very steep indicating that forest managers have to design their plans carefully to meet the objective of even-flow.

15.3 Research question 3

Single objective optimisation In order to answer this research question, the abundance of an edge-dependent species, namely the badger, was maximised. The abundance of the badger depends on the spatial configuration of clear felled and old growth stands. Here it was assumed that the abundance is linearly related to both the internal edge (between clear felled and old growth blocks) as well as the external edge (between forest and the surrounding agricultural fields). Before the single objective genetic algorithm was applied to Kirkhill Forest, it was used to solve a similar problem from literature. For this problem the aim is to maximise the internal and external edge for a 3-by-3 grid problem. Because the search space for this problem is so small (512 different solutions), the genetic algorithm was stopped after 600 function evaluations. A simple genetic algorithm was not always capable of finding the global optimum, only in 90.5% of the repetitions for a population size of 30 the global optimum is attained before the stopping criterion was reached.

Reformulating the problem as an adjacency problem leads to better results: for a population size of 30 the optimum is reached in 99.5% of the cases. The difference in performance of the genetic algorithm is due to the existence of local optima for the edge maximisation problem. For the edge-dependent formulation, the local optimum is the complement of the global optimum. For the adjacency formulation, however, both solutions are global optima. This explains why the second formulation is much easier to solve than the first one.

Multiple objective optimisation In the previous case study, the spatial constraints were created by hand. This is not very efficient for larger problems and almost impossible for an irregular configuration of forest stands. This implies that the non-linear techniques can only solve relatively small problems and that they cannot scale up to real-world applications. The integration of GA and GIS offers great potential for spatial decision support systems (SDSS). The functionality of both modules fits together and can remain in two entities and this ensures flexibility for the SDSS. Therefore, the genetic algorithm and GIS were loosely coupled. This has the advantage of being platform independent and ensures an optimal task distribution between the two modules. Originally ArcView GIS® was chosen as GIS module but this led to many broken connections between the genetic algorithm and the GIS. Therefore, in a later stage the open source GRASS GIS was used instead.

The single objective problem from the previous case study is now extended to a multiple objective one. Three objectives have to be optimised simultaneously: (1) timber revenue, (2) abundance of old growth species and (3) abundance of edge-dependent species. In literature, this is solved on a three-by-three grid. Because the problem is too easy to solve, the initial population already contains most of the Pareto-optimal solutions and therefore using a genetic algorithm for this problem is not efficient. When the scale of the problem increases from a three-by-three grid to a nine-by-nine grid, the advantage of genetic algorithms

becomes apparent: it approximates the Pareto-optimal front in a short time.

The genetic algorithm is then used to optimise the same objectives on Kirkhill forest. This introduces an extra difficulty because the stands are now in an irregular configuration and the size of the search space becomes very large. The results show that the discreteness of the forest management problem again causes a lack of spread along the Pareto-front: most of the solutions are centred in the middle of the Pareto-front. If the extreme solutions are taken as alternatives, the plans do appear valid. The harvest pattern for the scenario of maximum of timber production allocates 288 ha for clear felling. This is much lower for the scenario of maximum old growth objective, where only 184 ha are clear felled. For the scenario of maximum abundance of the badger, the sum of internal and external edge equals 68 km and this corresponds to 85 badger setts. This is less than the average for Aberdeenshire and this might indicate that for this objective the solutions are far from the optimum.

15.4 Research question 4a

The Type I harvest scheduling problem was solved using three different encoding strategies: (1) binary codes which are used in general in genetic algorithms, (2) gray codes that are supposed to enhance genetic algorithm performance because they reduce the discontinuity and (3) integer codes because they are the most natural representation for the forest management problem. The encoding strategy does not affect the genetic algorithm behaviour for the single but does affect the behaviour of the multiple objective genetic algorithm. For the multiple objective genetic algorithm the approximation of the Pareto-optimal front is best when either gray or integer codes are used.

15.5 Research question 4b

Because of the two-dimensional nature of spatial information, the linear representation of the decision variable vectors might violate the basic assumption of genetic algorithms: building blocks should be short and low-order pieces of the chromosome. In order to test whether a forest management problem employing spatial data needs more advanced genetic operators, the single objective maximisation problem of the abundance of edge-dependent species was optimised using two Estimation of Distribution Algorithms (EDAs). These algorithms should detect the building blocks in the chromosome through probabilistic modelling and process them explicitly during recombination. When two of these EDAs, the extended compact GA (ECGA) and the Bayesian Optimization Algorithm (BOA), are applied to Kirkhill forest, their performance is significantly better than of a simple genetic algorithm. This means that the real-world application suffers from loose linkage: the genes that make up a building block are not close together and the implicit processing of the building blocks does not guarantee optimality.

ECGA processes building blocks with an average size of 4.5 at the beginning of the evolution and the size decreases to 1 over time due to population convergence. The size of the building block is only determined by the probabilistic model. ECGA quickly converges due to its high selective pressure. Already at generation 25 the population converges. When the linkage model is physically interpreted, it is found that the number of badger setts found by ECGA approximates the average number of badger setts in the Aberdeenshire region. The cutting pattern in the forest resembles a checkerboard and a little more than half of the forest is clear felled. The solutions found by the different repetitions are not always the same. On the contrary, 22% of the bits form a group with their neighbour and are complementary with the same bits in other solutions. This indicates that the other building blocks consists of stands far away and that linkage learning is necessary.

BOA finds very complicated network structures with on average 5 incoming edges at the beginning of the evolutionary process and 4.5 near the end of the evolution. This is the same size as in ECGA. BOA did not converge before the end of the evolution and the solution quality at that point is still inferior to that of ECGA, probably due to insufficient population sizing and a lower selective pressure. The physical interpretation is much more difficult, but as for ECGA the genes of the chromosomes from different repetitions are complementary in some parts.

In short, using a cluster-based probabilistic model ensures that better solutions can be found than in the case of the simple genetic algorithm in a shorter time span than a network-based probabilistic model.

15.6 Research question 5

Two fitness inheritance techniques have been proposed in literature as efficiency enhancement techniques. Because little was known about their applicability for multiple objective problems, they were first evaluated on three functions: a convex, a non-convex and a discontinuous one. Their performance was tested using several performance indices and from the results followed that fitness inheritance can only be used in order to reduce the number of fitness evaluations provided that the Pareto-optimal front is convex and continuous. If the surface is non-convex or discontinuous, the fitness inheritance strategies fail to approximate the Pareto-optimal front.

If real-world practitioners want to use fitness inheritance, it is advisable to check beforehand what the nature (convex, non-convex,...) of the Pareto-optimal front is. The problem should first be solved using a classical multiple objective GA for a low number of generations and given sufficient indication that the Pareto-optimal front is convex and continuous, one could switch to fitness inheritance. If a decision about which inheritance strategy must be taken, it is safer to choose the proportional approach, because in most cases the proportional inheritance shows better performance in terms of generational

distance, error ratio and hypervolume.

When the proportional fitness inheritance technique is applied for the convex harvest scheduling problem, it can speed up the optimisation process. According to performance indices and the attainment surfaces, fitness inheritance can approximate the Pareto-optimal front.

15.7 Critical notes and indications for future research

Parameter settings Parameter settings for the different algorithms are always susceptible to criticism. Here they were set to the best suitable ones that were found in literature. However, if for some reasons they were not set to these values, because for example a large population size is not feasible, then the best alternative settings were chosen.

Extension of spatial models The spatial models that were used to determine the abundance of the badger are not really realistic, even though realistic sett numbers were obtained. Extension of these models and inclusion of other models is certainly necessary for operational applications. Moreover, until now the use of extensive spatial modelling is not feasible in terms of computing time. Real-world practitioners should consider other techniques for speeding up the optimisation process. These might include local fitness recalculation, where the fitness value of the individuals is updated from the value from the parents by recalculating only part of the fitness function. Another possibility is instead of simply averaging the fitness values, is to use advanced statistical techniques such as kriging.

Testing the acceptability of the harvest plans Even though the plans seemed valid, it might be useful to present them to forest managers. This might indicate some logistic problems for the implementation of the plans. This might also increase the acceptability of the plans.

Extension of EDAs to multiple objectives The EDAs might be extended for multiple objectives, thus making it possible to solve even more complex optimisation problems.

New algorithms A full examination of new algorithms such as by Jaszkiewicz (2000) might be worth studying, because they might provide a faster way of finding Pareto-optimal solutions.

15.8 Main contributions of this dissertation

This dissertation has tried to make contributions to both the forest management as well as to the genetic algorithm domain. These contributions are listed below.

To the forest management community

- a comparative study was performed to assess which genetic algorithm is best suited for solving forest management problems (Ducheyne et al., 2001)
- the effect of the encoding technique for harvest scheduling problems was analysed
- the plans generated by the single and multiple objective genetic algorithms were analysed for their validity (A1-paper submitted)
- the proposed genetic algorithm is coupled with a GIS to allow the online evaluation of spatial models (A1-paper in progress)

To the genetic algorithm community

- estimation of distribution algorithms were applied to a real-world application; the probabilistic models were physically interpreted (Ducheyne et al., 2002)
- in order to investigate the usefulness of fitness inheritance, both the average and proportional fitness inheritance techniques were extensively analysed both on a test suite of functions and a real-world application (A1-paper) (Ducheyne et al., 2003)

Chapter 16

Samenvatting en conclusies

Bij de aanvang van dit onderzoek werden drie objectieven vooropgesteld:

1. De ontwikkeling van een optimalisatietechniek voor bosbeheer die alternatieve plannen genereert voor meerdere objectieven.
2. Bij een dergelijk beheersinstrument moet de integratie van ruimtelijke gegevens en GIS-functionaliteit mogelijk zijn.
3. Dit hulpmiddel is bij voorkeur efficiënt en snel.

Er werden vijf onderzoeksvragen geformuleerd die ervoor moesten zorgen dat de drie hoofdobjectieven werden bereikt:

1. Welke optimalisatietechnieken worden heden ten dage gebruikt bij bosbeheer met meerdere objectieven, en wat zijn hun tekortkomingen?
2. Welke nieuwe optimalisatietechnieken uit operationeel onderzoek of computationele intelligentie zijn beschikbaar die meerdere objectieven zonder aggregatie kunnen optimaliseren?
3. Welke van deze technieken zijn voldoende flexiebel om integratie met GIS mogelijk te maken?
4. Wat zijn de basisveronderstellingen bij deze methodes? Als deze niet vervuld zijn in het geval van bosbeheerstoepassingen, hoe kunnen deze methodes worden aangepast? Deze vraag werd opgesplitst in twee deelvragen:
   a. Wat is de invloed van de coderingsstrategie van de beslissingsvariabelen op de kwaliteit van de oplossingen?
   b. Bemoelijken bosbeheersproblemen het optimalisatie-en zoekproces?

5. Hoe kunnen de technieken worden verbeterd zodat goede oplossingen sneller worden bekomen?

In de volgende paragrafen wordt het antwoord op deze vijf vragen kort toegelicht.

16.1 Onderzoeksvraag 1

Elk bosbeheersprobleem kan worden geformuleerd als een Type I of een Type II Model. Deze modellen hebben hun nut bewezen bij het plannen van kapregeling of andere beheersactiviteiten. Deze modellen worden eenvoudig vertaald in een lineair programma. Een Type I formulering is te verkiezen boven de tweede formulering als er andere objectieven aanwezig zijn, omdat dan de beslissingsvariabelen toelaten om een beheerseenheid te volgen van het begin tot het einde van de beslissingshorizon. Toch is ook dit model ongeschikt om bijvoorbeeld ecologische modellen te integreren tijdens het optimalisatieproces. Hiervoor zijn ruimtelijke data nodig en omdat beide modellen stratum-gebaseerd zijn is de link tussen de beheerseenheid en de locatie verbroken.

Om toch ruimtelijke integriteit te bekomen werden de beslissingsvariabelen, die in de vorige formulering continu waren, gediscretiseerd. Dit leidde tot zeer grote combinatorische problemen die voor reële toepassingen niet kunnen opgelost worden met behulp van exacte technieken zoals *integer* en *mixed-integer* programmeren. Verschillende heuristieken werden voorgesteld om deze complexe optimalisatieproblemen op te lossen. Zo zijn bijvoorbeeld *simulated annealing* en *tabu search* in staat om dit te realiseren op een reële schaal.

In plaats van de ruimtelijke gegevens als beslissingsvariabelen te formuleren, biedt de integratie van een optimalisatietechniek met een GIS veel mogelijkheden. Tot op heden werd GIS echter enkel voor classificatiedoeleinden vóór het optimalisatieproces of voor visualisatie na dit proces gebruikt. De online samenwerking tussen beiden werd echter nog niet gerapporteerd binnen bosbeheer.

De vorige technieken veronderstellen verder dat er slechts één enkele objectieffunctie is. Meerdere objectieven moeten omgevormd worden tot één enkele functie bijvoorbeeld met de gewichtenmethode of de voorwaardenmethode. Dit is vooral moeilijk wanneer de objectieven in verschillende eenheden uitgedrukt zijn. Als er dan ook nog een Pareto-front tussen de objectieven gezocht wordt, moet men de optimalisatietechniek meermaals toepassen. In bosbeheer is er echter tot nog toe geen echte optimalisatietechniek in gebruik die meervoudige objectieven aankan, zonder dat die op voorhand moeten samengevoegd worden.

16.2 Onderzoeksvraag 2

16.2.1 Het opstellen van kapschema's als een enkelvoudig objectiefprobleem

Een optimaal Type I kapschema wordt eerst gezocht met behulp van een genetisch algoritme met een enkelvoudig objectief. Vermits de Type I formulering twee objectieven heeft (de maximalisatie van de tegenwoordige waarde en de minimalisatie van de afwijking in kapvolume tussen de kapperiodes) werden deze twee objectieven voor het optimalisatieprocess gecombineerd met behulp van de gewichtenmethode. Om het Pareto-optimale front te vinden werden de gewichten lineair gekozen over het halfopen interval $]0, 1]$. Het gebruik van deze gewichten leverde echter geen gelijk verdeelde oplossingen over het Pareto-front op. Er kon dan ook weinig informatie uit het bekomen Pareto-front gehaald worden en daarom werden nog twee extra gewichten gekozen: $w = 0.01$ en $w = 0.05$. Als het objectief van de tegenwoordige waarde 100 keer belangrijker werd dan dat van gelijke houtopbrengst, werd de helling van het Pareto-front zeer steil. Dit heeft als gevolg dat indien een gebruiker die gewichten moet kiezen als hij weinig kennis over het probleem heeft, veel informatie over het Pareto-front kan verliezen.

Het bekomen gemiddeld houtvolume wordt sterk beïnvloed door het objectief van gelijk volume. Dit houtvolume neemt af naarmate dit objectief belangrijker wordt. De leeftijdsverdeling van het bos wordt door een impliciete volumecontrole genormaliseerd en een genetisch algoritme genereert kapplannen met een gebalanceerde leeftijdsdistributie. Als er weinig belang is voor even-flow wordt een tegenwoordige waarde bekomen die tot 73% van de maximum te bereiken tegenwoordige waarde bedraagt.

Een praktisch nadeel van de gewichtenmethode is dat de gewichtenmethode omslachtig is: het meermaals laten lopen van het genetisch algoritme om meerdere punten op het Pareto-optimaal front te vinden vergt veel tijd.

16.2.2 Uitbreiding naar meerdere objectieven

Een vergelijkende studie Vooraleer het Type I kapprobleem werd opgelost als een tweevoudig objectiefprobleem werden twee meervoudig objectief genetische algoritmes met elkaar vergeleken. Deze twee genetische algoritmes werden reeds vaak toegepast, onder meer in het domein van regeltechniek en mechanica. Deze twee algoritmes zijn het Multiple Objective Genetic Algorithm (MOGA) en het Non-dominated Sorting Algorithm-II (NSGA-II). Ze werden vergeleken op een referentieprobleem waarvan het Pareto-optimale front gekend is en ze werden tevens vergeleken met een random zoekmethode. De objectieven voor dit probleem waren: (1) maximaliseer de houtopbrengst en (2) maximaliseer het nut dat mensen bekomen van het bos. Het Pareto-optimale front is niet convex en kan niet worden opgelost met behulp van de gewichtenmethode.

Zowel MOGA als NSGA-II doen het beter dan de random zoektocht. Beide

zijn in staat om het Pareto-optimale front goed te benaderen, maar vertonen weinig spreiding van de oplossingen. Vooral NSGA-II kan de meer extreme oplossingen niet vinden. Om uit te sluiten dat dit gebrek aan spreiding werd veroorzaakt door de implementatie van de algoritmes, werden ze extra getest op een niet convexe functie waarvoor de performantie gekend is. Hieruit volgde dat ze voor die functie wel degelijk een goede spreiding vertonen en dat de hogervermelde gebrekkige spreiding te wijten is aan de aard van het bosbeheersprobleem.

NSGA-II benadert het Pareto-optimale front sneller dan MOGA en de oplossingen liggen op een gelijke afstand langs het front. Het feit dat MOGA meer extreme oplossingen vindt wordt kwantitatief bevestigd met behulp van de spreidingsmaat, maar het verschil in spreidingsmaat tussen de twee algoritmes is niet significant. De variantie in generatie-afstand tussen de verschillende herhalingen is verder kleiner voor NSGA-II dan voor MOGA en dit toont aan dat NSGA-II robuuster is tegen de randominitialisatie van de populatie dan MOGA.

De hypervolume-maat toont tenslotte aan dat NSGA-II een groter deel van de zoekruimte domineert en als de *attainment*-functie wordt gebruikt kan een gelijkaardige conclusie worden getrokken. Uit de Mann-Whitney testprocedure volgt dat NSGA-II MOGA domineert in het grootste deel van de zoekruimte.

Het opstellen van kapschema's als een meervoudig objectief probleem Het Type I kapprobleem kan nu worden opgelost als een tweevoudig objectiefprobleem met NSGA-II. Eén enkel run volstaat om equidistante oplossingen langs het Pareto-front te vinden. Bovendien wordt de stap van de gewichten vermeden. Het bekomen Pareto-front van het NSGA-II is vergelijkbaar met dat van het enkelvoudig genetisch algoritme en hieruit blijkt dat een dergelijk algoritme het optimalisatieproces kan versnellen in vergelijking met de enkelvoudige versie.

Validatie van de kapschema's Het effect op de leeftijdsverdeling is analoog als bij de enkelvoudige formulering: als het objectief van gelijk kapvolume belangrijker wordt dan lijkt de leeftijdsverdeling van het bos op dat van een normaal bos. Dit wordt veroorzaakt door een impliciete volumecontrole. Als het objectief van gelijk kapvolume minder belangrijk is dan dat van de tegenwoordige waarde, dan wordt het kappen van de bestanden verdaagd naar latere kapperiodes, en dit zorgt ervoor dat de leeftijdsstructuur van een normaal bos nog beter wordt benaderd. Het Pareto-front is verder zeer steil, en dit toont aan dat bosbeheerders hun kapschema's goed moeten analyseren wanneer ze gelijke kapvolumes wensen.

16.3 Onderzoeksvraag 3

Enkelvoudige optimalisatie Om deze onderzoeksvraag te kunnen beantwoorden wordt de abundantie van een grensafhankelijke diersoort, namelijk de das, gemaximaliseerd. De abundantie van de das is afhankelijk van de configuratie van gekapte en niet-gekapte bestanden. In dit onderzoek werd verondersteld dat het aantal dassen lineair gerelateerd is met zowel de interne grens (tussen de gekapte en niet-gekapte bestanden) als de externe grens (tussen bos en landbouwvelden). Voordat het genetisch algoritme werd toegepast op het studiegebied, werd het gebruikt om een gelijkaardig probleem uit de literatuur op te lossen. Bij dit probleem is het doel de interne en externe grens voor een raster van 3-bij-3 cellen te maximaliseren. Omdat de zoekruimte beperkt is (slechts 512 verschillende oplossingen), werd het genetisch algoritme gestopt na 600 functie-evaluaties. Het genetisch algoritme was in slechts 90.5% van de herhalingen in staat om het globale optimum te vinden voor een populatiegrootte van 30.

Als het probleem wordt omgevormd tot een *adjacency*-probleem worden betere resultaten bekomen. Met een populatiegrootte van 30 wordt het optimum in 99.5% van de gevallen bekomen. Het verschil in performantie van het genetisch algoritme kan verklaard worden door de aanwezigheid van lokale optima bij de oorspronkelijke formulering. Dit lokale optimum is het complement van het globale optimum, terwijl voor het tweede geval beide oplossingen globale optima zijn.

Meervoudig objectief optimalisatie In het voorgaande voorbeeld werden de vergelijkingen voor de ruimtelijke informatie met de hand opgesteld. Dit is niet efficiënt voor grote problemen en bijna onmogelijk voor een onregelmatige configuratie van de bestanden. Dit impliceert dat de voorgaande methode niet kan toegepast worden op reële toepassingen. De integratie van GA and GIS daarentegen biedt mogelijkheden voor dergelijke ruimtelijke problemen. De functionaliteit van beide modules is complementair en de modules kunnen afzonderlijk blijven bestaan. Het genetisch algoritme en het GIS werden in dit werk los met elkaar gekoppeld zodat de toepassing platformonafhankelijk was en de taakdistributie zo efficiënt mogelijk tot stand werd gebracht. Initieel werd ArcView GIS® gebruikt als GIS-module maar de communicatie tussen ArcView en het GA verliep niet optimaal. Daarom werd in een later stadium besloten om het *open source* GRASS GIS te gebruiken.

Het enkelvoudig objectief probleem van de vorige *case study* werd uitgebreid tot een meervoudig probleem. Drie objectieven waren: (1) maximalisatie van houtopbrengst, (2) maximalisatie van de abundantie van een oude-bos diersoort en (3) de abundantie van een grensafhankelijke diersoort. In de literatuur werd dit opgelost op een fictief bos met een rasterlayout van drie-bij-drie cellen. De initiële populatie bevat reeds veel Pareto-optimale oplossingen en het gebruik van een genetisch algoritme in dergelijke kleine problemen is dan ook niet efficiënt. Als het probleem wordt vergroot tot een raster van negen-bij-

negen cellen dan zijn de voordelen van het genetisch algoritme wel duidelijk: het genetisch algoritme is in staat het Pareto-optimale front snel te bereiken.

Het genetische algoritme wordt vervolgens aangewend om hetzelfde probleem in Kirkhill op te lossen. Dit introduceert een extra moeilijkheid omdat de bestanden in een onregelmatig patroon staan en bovendien wordt de zoekruimte door het aantal bestanden zeer groot. Opnieuw wordt aangetoond dat het discrete karakter van het bosbeheersprobleem een gebrekkige spreiding van de oplossingen veroorzaakt. De meeste oplossingen liggen in het centrum van het Pareto-front. De drie meest extreme oplossingen worden gebruikt om de plannen te valideren. Het kapplan dat het meeste houtvolume oplevert, alloceert 288 ha van het bos voor kappen. Dit is veel minder in het geval waar de abundantie van de oude-bos soort wordt gemaximaliseerd: 184 ha wordt gekapt. Om de maximale abundantie van de das te bekomen bedraagt de som van de interne en externe grens 68 km. Dit komt overeen met 85 dassenburchten over Kirkhill en is minder dan het gemiddelde voor Aberdeenshire. Dit toont aan dat de oplossingen nog ver verwijderd zijn van de optimale oplossingen.

16.4 Onderzoeksvraag 4a

Het kapprobleem werd opgelost met behulp van drie verschillende coderingstechnieken voor de beslissingsvariabelen: (1) binaire codering, de meest gebruikte codering voor genetische algoritmes, (2) gray codering, die verondersteld wordt de discontinuïteiten tussen gehele getallen te verminderen en (3) gehele getallen die de meest natuurlijke voorstelling zijn van de beslissingsvariabelen bij een kapprobleem. De codering heeft enkel effect op het gedrag van het meervoudig objectief genetisch algoritme. Bij meervoudige objectieven is zowel de benadering van het Pareto-optimale front het best met integer en gray codering.

16.5 Onderzoeksvraag 4b

Omdat ruimtelijke gegevens een twee-dimensioneel karakter hebben, kan het zijn dat een lineaire voorstelling van de beslissingsvariabelen niet voldoet aan de basisveronderstelling van een genetisch algoritme, namelijk dat *building blocks* kort en van lage orde moeten zijn. Om na te gaan of er bij een bosbeheersprobleem met ruimtelijke gegevens meer geavanceerde operatoren dan de eenvoudige recombinatieoperatoren vereist zijn, werd teruggegrepen naar het enkelvoudig objectief probleem waarbij de abundantie van de das werd gemaximaliseerd. Er was immers al vastgesteld dat de oplossing voor dit objectief zowel in het enkelvoudig als in het meervoudige geval niet optimaal is.

Dit probleem werd dan opgelost met behulp van twee distributieschattingsalgoritmes: het Extended Compact Genetic Algorithm (ECGA) en het Bayesian Optimization Algorithm (BOA). Deze algoritmes zijn in staat om *building blocks* in een chromosoom te detecteren door probabilistische correlaties op te sporen

in de populatie. Als de *building blocks* zijn gevonden, kunnen ze tijdens de recombinatiefase expliciet worden verwerkt. Wanneer deze probabilistische schattingsmethodes worden toegepast op Kirkhill, blijkt hun performantie significant beter te zijn dan deze van een standaard genetisch algoritme. Dit bewijst dat voor dit reële probleem, de genen van de *building blocks* ver uit elkaar liggen en dat de geavanceerde technieken nut hebben. Het standaard genetisch algoritme is niet in staat om de *building blocks* impliciet te gaan verwerken.

ECGA detecteert *building blocks* met een gemiddelde grootte van 4.5 genen bij het begin van de evolutie en deze grootte daalt tot 1 op het einde van het optimalisatieproces. Dit is een gevolg van de populatieconvergentie. Deze convergentie treedt bij ECGA vrij snel op (reeds op generatie 25) als gevolg van de hoge selectiedruk die nodig is om het algoritme te doen werken.

Wanneer het *linkage model* fysisch wordt geïnterpreteerd, blijkt dat het aantal dassenburchten gevonden door ECGA overeenkomt met het gemiddelde voor de Aberdeenshire regio. Het kappatroon benadert ook in vrij hoge mate een dambordpatroon en iets meer dan de helft van het bos wordt gekapt. De oplossingen van de herhalingen zijn niet altijd dezelfde. Het blijkt dat 22% van de genen een groepje vormt met zijn buur en dat deze groepjes complementair zijn tussen de verschillende herhalingen. Deze complementariteit was ook aanwezig bij het rasterprobleem en duidt waarschijnlijk weer op lokale minima. De andere bits zijn gelinkt met bits die niet de buur zijn en duiden erop dat de genen van de *building blocks* inderdaad ver van elkaar verwijderd zijn. Dit verklaart waarom een klassiek genetisch algoritme faalt op dit probleem.

BOA vindt een zeer complexe netwerkstructuur tussen de bestanden. Initieel is de grootte van de groepjes 5 en dit vermindert tot 4.5 op het einde van de evolutie. Dit is dezelfde orde die ook ECGA weergaf. De populatieconvergentie is bij BOA nog niet opgetreden na 40 generaties, en de waarde van de beste oplossingen is nog altijd lager dan bij ECGA. Dit is te wijten aan ofwel een te kleine populatie ofwel een te lage selectiedruk.

De fysische interpretatie is door de complexe netwerkstructuur moeilijker dan bij ECGA, maar net zoals bij ECGA worden complementaire stukjes in de chromosomen van de verschillende herhaling waargenomen.

Voor een bosbeheersprobleem kan dus besloten worden dat een model gebaseerd op clusters voldoende is om betere oplossingen te genereren dan een klassiek genetisch algoritme en dat de tijdsduur hiervoor beperkter is dan voor BOA.

16.6 Onderzoeksvraag 5

Het bepalen van de fitnesswaarden voor nieuwe individuen door overerving werd voorgesteld in de literatuur als een manier om de efficiëntie van een genetisch algoritme te verbeteren. Omdat er weinig gekend is omtrent de toepasbaarheid van deze technieken voor problemen met meervoudige objectieven werden ze eerst getest op drie testfuncties: een convexe functie, een concave functie en

een discontinue functie. De performantie werd getest met behulp van verschillende indicatoren en hieruit blijkt dat fitness-erving enkel mogelijk is indien het Pareto-optimale front convex en continu is. Indien dit niet het geval is, dan kan het genetisch algoritme met fitness-erving het Pareto-optimale front niet benaderen.

Als men fitness-erving wil gebruiken voor reële problemen, dan is het aan te raden om op voorhand na te gaan wat de vorm van het Pareto-front is. Indien blijkt dat dit convex is dan kan men eventueel fitness-erving toepassen. Een haalbare werkwijze is eerst het probleem op te lossen voor een beperkt aantal generaties en om de vorm van het Pareto-front na te gaan en daarna over te schakelen op fitness-erving. Als men moet beslissen welke ervingstechniek gebruikt moet worden, dan is het beter om proportionele fitness-erving te kiezen omdat die voor de meeste indices de beste score heeft in vergelijking met de gemiddelde fitness-erving.

Indien men het optimalisatieprocess wil versnellen, kan men ook andere technieken inschakelen zoals bijvoorbeeld een lokale herberekening van de fitnesswaarde. Dit is vooral interessant als er al enige graad van convergentie aanwezig is omdat dan de nakomelingen heel sterk op de ouders zullen lijken.

Als tenslotte proportionele fitness-erving wordt toegepast voor het convexe kapprobleem, blijkt dat de ervings-techiek in staat is om het optimalisatieproces te versnellen. Fitness-erving is in staat om het Pareto-optimale front te benaderen.

16.7 Bemerkingen en indicaties voor toekomstig onderzoek

Parameterwaardes Parameterwaardes voor de verschillende algoritmes kunnen altijd worden bekritiseerd. In dit onderzoek werd getracht om ze zoveel mogelijk op die waarden te zetten zodat de algoritmes hun beste perfomantie bereiken. Indien dit niet mogelijk was, bijvoorbeeld omdat een zeer grote populatie nodig was dan werden ze op de best mogelijke waardes gezet.

Uitbreiding van de ruimtelijke modellen De gebruikte ruimtelijke modellen om de abundantie van de das te bepalen zijn zeer simplistisch. Hoewel toch een realistisch aantal dassenburchten werd bekomen, zijn uitgebreidere modellen nodig voor operationeel gebruik. Tot nog toe is ook het gebruik van complexe ruimtelijke modellen niet echt haalbaar omdat het te veel computertijd vergt. Daarom is het aan te raden om technieken die het proces versnellen te gebruiken. Omdat fitness-erving niet werkt met eenvoudige lineaire schatting van de fitnesswaarden, kan bijvoorbeeld overwogen worden om meer geavanceerde schattingstechnieken zoals kriging te gebruiken. Verder kan ook lokale herberekening van de fitness-waarden zoals bij *tabu search* een mogelijkheid zijn.

Validiteit van de kapplannen Hoewel de planning geldig lijken, is het zeker nuttig om ze voor te leggen aan bosbeheerders. Deze kunnen vervolgens wijzen op de logistieke problemen die kunnen ontstaan door de implementatie van de plannen. Een dergelijke interactie zou natuurlijk ook de aanvaardbaarheid van de plannen verhogen.

Uitbreiding van de probabilistisch schattingsmodellen voor meerdere objectieven De probabilistische schattingsmodellen zouden kunnen uitgebreid worden voor meerdere objectieven. Dit zou het mogelijke maken om meervoudige objectief problemen met ruimtelijke gegevens op te lossen.

Nieuwe algoritmes Dit doctoraatsonderzoek spitste zich enkel op genetisch algoritmes toe omdat ze heel gemakkelijk meervoudige objectieven kunnen optimaliseren door hun populatie-gebaseerde aanpak. Deze aanpak heeft ook het nadeel dat er veel individuen moeten geëvalueerd worden en dit kan het zoekproces vertragen. Een onderzoek naar het gebruik van nieuwere algorithms zoals bijvoorbeeld van Jaszkiewicz (2000) is dan ook de moeite waard omdat ze misschien sneller Pareto-optimale oplossingen kunnen genereren.

16.8 Wetenschappelijke bijdrage van dit werk

Tijdens dit onderzoekswerk werd getracht om een bijdrage te leveren tot zowel de kennis in het domein van bosbeheer als in het domein van genetische algoritmes. Deze zijn:

Tot het domein van bosbeheer

- Een vergelijkende studie werd uitgevoerd om na te gaan welk genetisch algoritme geschikt is voor een bosbeheersprobleem (Ducheyne et al., 2001).
- Het effect van de codering voor een kapprobleem werd geanalyseerd.
- De kapplannen die gegenereerd werden door zowel enkelvoudige als meervoudige objectief genetische algoritmes werden gevalideerd (A1-artikel ingezonden).
- Het voorgestelde genetisch algoritme werd gekoppeld aan een GIS om een online evaluatie van ruimtelijke modellen mogelijk te maken (A1-artikel in voorbereiding).

Tot het domein van genetische algoritmes

- Probabiliteitsschattingsalgoritmes werden toegepast op een reëel probleem, de probabilistische modellen werden fysisch geïnterpreteerd (Ducheyne et al., 2002).

- Het nut van gemiddelde en proportionele fitness-erving werd uitgebreid getest zowel op een drietal testfuncties als op een reëel probleem (Ducheyne et al., 2003).

Part V

Appendices

Appendix A

Real prices for standing volume published in 2000

Table A.1: Real prices for standing volume (Anonymous, 2000)

	Prices (€) per m^3 for conifers							
Diameter at 1.5 m (cm)	20-39	40-59	60-69	70-89	90-119	120-149	150-179	> 180
Norway spruce	-	6	15	27.5	40	47.5	47.5	47.5
Larch	-	2.5	7.5	10	21.25	26.25	28.75	28.75
Scots pine	-	2.5	7.5	10	21.25	26.25	28.75	28.75
Douglas fir	-	2.5	7.5	10	21.25	26.25	28.75	28.75

	Prices (€) per m^3 for broadleaves						
Diameter at 1.5 m (cm)	100-119	120-149	150-179	180-199	200-219	220-249	> 250
Oak	10	23.50	35	50	55	150	155
Beech	15.50	30	52.50	70	80	85	85

Appendix B

Java documentation files for the single and multiple objective genetic algorithm

```
/*Class Parameters*/
// Imports
import java.lang.String;
import java.io.BufferedReader;
import java.lang.Exception;

public class Parameters {

  // Fields
  private int numberOfGens;
  private int popSize;
  private int numberOfReps;
  private String outputprefix;
  private int maxInt;
  private char elitisme;
  private int numberOfElites;
  private int tDom;
  private char crossOverType;
  private char selectionType;
  private double pCross;
  private double pMut;
  private int numberOfObjectives;
  private int numberOfGenes;
  private int functionNumber;
```

```
  private double pDecode;
  private double weight;

  // Constructors
  public Parameters() { }
  public Parameters(int p0, int p1, int p2, String p3, \\
  int p4, char p5, int p6, int p7, char p8, char p9, \\
  double p10, double p11, double p12, int p13, int p14,\\
  int p15, double p16) throws Exception { }

  // Methods
  public int getGenerations() { }
  public int getPopsize() { }
  public int getNumberOfRepetitions() { }
  public String getOutputFile() { }
  public int getMaxInt() { }
  public char getElitisme() { }
  public int getNumberElites() { }
  public int getTDom() { }
  public char getCrossOverType() { }
  public char getSelectionType() { }
  public double getPCross() { }
  public double getPMut() { }
  public double getPDecode() { }
  public int getNumberOfObjectives() { }
  public int getNumberOfGenes() { }
  public String getOutputprefix() { }
  public int getFunctionNumber() { }
  public double getWeight() { }
  public static Parameters readParams(BufferedReader p0)\\
   throws Exception { }
  public String toString() { }
}
/*Class Chromosome*/
public class Chromosome {

  // Fields
  private int[] chrom;
  private int maxInt;
  private int numberOfGenes;

  // Constructors
  public Chromosome(Parameters p0) { }
  public Chromosome(Object p0) throws Exception { }
```

```
  // Methods
  public int getBit(int p0) { }
  public void setBit(int p0, int p1) throws Exception { }
  public String toString() { }
  public boolean equals(Object p0) { }
}
/*Class Individual*/
public class Individual {

  // Fields
  private Chromosome chrom;
  private double[] objectiveValues;
  private double decisionVar;
  private int rank;
  private int number;
  private double fitness;
  private double distance;
  private double violations;
  private double[] deviation;
  private final double EPS = 9.999994179233909E-5;
  private boolean feasible;

  // Constructors
  public Individual(Parameters p0) { }
  public Individual(Parameters p0, Chromosome p1) \\
  throws Exception { }

  // Methods
  public Chromosome getChrom() { }
  public double getObjectiveValues(int p0) { }
  public int getRank() { }
  public double getFitness() { }
  public double getDev(int p0) { }
  public double getDecisionVar() { }
  public double getDistance() { }
  public int getNumber() { }
  public boolean getFeasible() { }
  public void setChrom(Object p0) throws Exception { }
  public void setRank(int p0) throws Exception { }
  public void setNumber(int p0) throws Exception { }
  public void setDistance(double p0) throws Exception { }
  public void setObjectiveValues(int p0, double p1) { }
  public void setDecisionVar(double p0) { }
  public void setFitness(double p0) { }
  public void setDev(int p0, double p1) { }
```

```
  public void setFeasible(boolean p0) { }
  public void setViolations(double p0) { }
  public boolean constrained_dominates(Object p0) \\
  throws Exception { }
  public boolean dominates(Object p0) throws Exception { }
  private boolean equals(Individual p0) { }
  public boolean greaterThen(Object p0) throws Exception { }
  public String toString() { }
  public static void decode(Individual p0, int p1, \\
  int p2, int p3, double p4) throws Exception { }
  public static int convertGrayToInt(int[] p0) { }
  public static int convertBinaryToInt(int[] p0) { }
}
/*Class Population*/
public class Population {

  // Fields
  private int popsize;
  private double[] max;
  private double[] min;
  private Individual[] pop;
  private int maxInt;
  private int numberOfObjectives;
  private double minFitness;
  private double maxFitness;
  private double averageFitness;

  // Constructors
  public Population(Parameters p0) { }
  public Population(int p0, Parameters p1) { }

  // Methods
  public void setIndiv(int p0, Individual p1) \\
  throws Exception { }
  public Individual getIndiv(int p0) { }
  public static void initialPop(Population p0, \\
  Parameters p1, Random p2) \\
  throws Exception { }
  public static void decodePop(Population p0, \\
  Parameters p1, Hashtable p2) \\
  throws Exception { }
  public static void sort(Population p0, \\
  Parameters p1) throws Exception { }
  public static Population select(Population p0, \\
  Parameters p1, Random p2) throws Exception { }
```

```
private static Population tournamentSelectionWith\\
DominationSet
(Population p0, Parameters p1, Random p2) throws Exception { }
private static Population tournamentSelection \\
(Population p0, Parameters p1, Random p2, int p3) throws Exception { }
private static Population TournamentSelectionECGA\\
(Population p0, Parameters p1, Random p2) throws Exception { }
private static int[] shuffle(int p0) { }
private static Population binaryTournamentSelection\\
(Population p0, Parameters p1, Random p2) throws Exception { }
private static Population rouletteWheelSelection\\
(Population p0, Parameters p1, Random p2) throws Exception { }
public static void calcRank(Population p0) throws Exception { }
private static void shareByRank(Population p0) { }
private static boolean[] dominatedCompare\\
(Population p0, Population p1) throws Exception { }
private static boolean[] dominatedCompare\\
(Population p0) throws Exception { }
private static double[] share(Population p0, \\
Population p1, int p2, Population p3) { }
private static void calcMaxMin(Population p0) { }
public static int recombine(Population p0, \\
Population p1, Parameters p2, Random p3, int p4) throws Exception { }
private static int uniformCrossover(Population p0,\\
 Population p1, Parameters p2, Random p3, int p4) throws Exception { }
private static int onepointCrossover(Population p0,\\
Population p1, Parameters p2, Random p3,int p4, Hashtable p5)\\
 throws Exception { }
private static boolean flip(double p0) { }
private static int mutate(int p0, Parameters p1, Random p2) { }
public String toString() { }
public static void printStats(Population p0, PrintWriter p1,\\
int p2, int p3) throws Exception { }
public static void merge(Population p0, Population p1,\\
 Population p2) throws Exception { }
public static void copyPop(Population p0, Population p1,\\
int p2) throws Exception { }
public static void copyPop(Population[] p0, Population p1,\\
 int p2, Parameters p3) throws Exception { }
public static Population[] fastNonDominatedSort(Population p0,\\
Population[] p1, Parameters p2) throws Exception { }
static Population[] changeSize(Population[] p0, int p1, \\
Parameters p2) throws Exception { }
private static void crowdingDistanceAssignment(Population p0,\\
 Parameters p1) throws Exception { }}
```

```
/*Main Class FORGA*/
public class FORGA {

  // Fields
  static BufferedReader in;
  static PrintWriter uit;

  // Constructors
  public FORGA() { }

  // Methods
  public static void main(String[] p0) { }
}
```

Appendix C

Matlab documentation files for performance indices and bootstrapping method

```
function [snijpuntenX,snijpuntenY] = findAttainment (fileprefix)
```

Function findAttainment requires as input the outputfiles from the genetic algorithm and yields the crosssections and the median attainment surfaces both numerically and graphically.

First, the Pareto-optimal solutions are extracted from the data set. These solutions are then sorted. This is followed by the determination of the intersections between an arbitrary diagonal line and the attainment surface for each run. As the intersections are on a line, it is possible to draw the frequency distribution, and hence to determine the median attainment surface.

```
function [ER, GD] = GD(fileprefix)
```

Function GD requires as input the outputfiles from every repetition from the genetic algorithm and the reference file containing the Pareto-optimal solutions. This function yields tab-delimited files containing the error ratio and generational distance for each repetition.

The Pareto-optimal front is read from an inputfile, together with the Pareto-fronts of all repetitions. For the generational distance, the minimum distance from each point of the Pareto-front to the Pareto-optimal front is determined using the module *dsearchn*. Adding these distances together leads to the generational distance. Using the distances from *dsearchn* it is also possible to determine the error ratio: if the distance is larger than treshold δ than the solution is counted as error.

```
function [spacingvector] = spacing(fileprefix)
```

Function spacing requires as input the outputfiles from every repetition from the genetic algorithm and yields a tab-delimited outputfile containing the spacing for each repetition.

Initially, the solutions are sorted according to the first objective. Then the duplicate points are removed from the inputfiles. The distance between the points is then determined by taking the absolute value of the difference between the objectives. For each of the solutions, the minimum distance is determined. In the next step, the mean distance over all points is taken, and finally the difference for each point to the mean distance is calculated. This produces a vector with the spacing values for each of the repetitions.

```
function [spreadvector] = spread(fileprefix)
```

Function spread requires as input the outputfiles from every repetition from the genetic algorithm and yields tab-delimited outputfile containing the spacing for each repetition.

The procedure to calculate the spread is the same as for spacing. Next to the distance between the points on the Pareto-front, the distance from the extreme solutions on the Pareto-front to the most extreme solutions on the Pareto-optimal front is determined.

```
function [hypervolume] = hypervolume(fileprefix)
```

This function converts the outputfiles from the genetic algorithm into the suitable format for the hypervolume measure by Zitzler (1999). This hypervolume measure requires that all objectives are maximised.

```
function []=bootstrapping(alpha,fileprefix1,fileprefix2)
```

The bootstrapping module requires as input the confidence level alpha, and the fileprefix from two inputfiles for which the bootstrapping must be performed. This function yields bootstrapping graphs with the bounds of the confidence interval (in red) and the test value (in green).

First, the inputfiles are loaded. The mean performance index for each algorithm is calculated, together with the difference in means. In the following step, the data from the two algorithms are put together, permuted and redistributed over the two algorithms. The mean for the two algorithms is calculated again. This is repeated 5000 times. The 5000 resulting means are then sorted, and the lower and upper bound of the confidence interval are determined according to Eq. 7.9 (p. 92). The bootstrapping values are put in a histogram and displayed together with the confidence interval boundaries and the original mean value as test measure.

Appendix D

Examples of ECGA and BOA

D.1 ECGA

Imagine the following population of 8 individuals. The length of each chromosome is 4. After the selection step the mating pool is formed as in Table D.1 :

Table D.1: Initial population and mating pool

Initial population		Mating pool
0010		1001
0000		1101
1001		0111
1101	$\rightrightarrows$	1100
1001		0011
0111		0111
1000		1000
1000		1001

From the mating pool, it is now possible to calculate model complexity (MC) (Eq. D.1), the compressed population complexity (CPC) (Eq. D.2) and

the combined complexity (CC) (Eq. D.3).

$$MC = (\log_2(N+1)) \times \sum_i (2^{S_i} - 1) \tag{D.1}$$

$$CPC = N \times \sum_i \text{entropy}(M_i) \tag{D.2}$$

$$CC = MC + CPC \tag{D.3}$$

where

$$\text{entropy}(M_i) = -\sum_i p_i \times \log_2 p_i \tag{D.4}$$

where N is the population size, p_i is the probability of having an allele value of 1 on locus i, S_i is the cardinality of subset i and M_i is the marginal distribution of this subset.

For the first model M_1, it is assumed that there is no interaction between the genes. M_1 can thus be represented as: M_1: [1][2][3][4]. The model complexity is then with $N = 8$ and $S_i = 1\ \forall i = 1, \ldots, n$: $\log_2 9 \times (1+1+1+1) = 12.67$. The compressed population complexity is based on sum of the entropy for each gene i:

$$CPC \propto \left((-\frac{5}{8}\log_2\frac{5}{8}) + (-\frac{3}{8}\log_2\frac{3}{8})\right) + \left((-\frac{4}{8}\log_2\frac{4}{8}) + (-\frac{4}{8}\log_2\frac{4}{8})\right)$$
$$\left((-\frac{3}{8}\log_2\frac{3}{8}) + (-\frac{5}{8}\log_2\frac{5}{8})\right) + \left((-\frac{5}{8}\log_2\frac{5}{8}) + (-\frac{3}{8}\log_2\frac{3}{8})\right)$$

As N=8, $CPC = 29.76$ and therefore the combined complexity for M_1 equals $CC = 12.67 + 29.76 = 42.44$.

If we now combine gene 1 and gene 3 together in one group, then model 2 is represented by: [1,3],[2],[4]. The model complexity is with $S_1 = 2$ for the first group and $S_i = 1$ for $i = 2$ and $i = 3$: $\log_2 9 \times (3+1+1) = 15.8$. This is higher than for the first model, because we need to count four instances of the first subset, instead of two instances. The compressed population complexity in this case is 22.12, and the combined complexity is 37.92. This indicates that the second model represents the data better than the first model.

This procedure is continued using a greedy search. At each grouping level, the combined complexity is calculated. The model with the lowest CC is then used to combine the genes into groups of three. Assuming the second model was the best in the previous calculations, this would be used to expand the model in the next level until no further decrease of CC is possible.

After the best model (for example our model [1,3][2][4]) is found, the next generation of individuals is created. To this end, the parent population is shuffled randomly. If for example the parent numbers are as follows after shuffling: 3,2,4,5,1,8,7 and 6, then the first child will receive subset [1,3] from parent 3, the second child from parent 2, etc. After all children have received the first

subset of genes, the parent population is shuffled again. Now, subset [2] will be assigned to all offspring. Finally, the population is shuffled for a last time, and subset [4] will be assigned to the offspring. This procedure is nothing else but a uniform crossover with N parents.

D.2 BOA

Imagine the following population of 4 individuals. The length of each chromosome is 3. After the selection step the mating pool is formed as in Table D.2 :

Table D.2: Mating pool

Mating pool
100
110
000
000

Based on the mating pool, the possible networks are constructed in the following manner.

Empty network In the simplest case, all genes are independent, and this corresponds with an empty network. We proceed with calculating the coefficients N_{ij} and N_{ijk} as in Eq. 11.16 on p. 148 and as was shown in the example on p. 148. For the empty network, we find that $P(D, B \mid \zeta) = 0.0004$.

Adding one edge Starting from the empty network, we can now add an edge between the different genes. This produces the following joint probability (Table D.3):

Table D.3: Joint probability functions and scores for the networks with only one edge

$P(D, B \mid \zeta)$	Score
$p(x_1)p(x_3)p(x_2 \mid x_1)$	0.0004
$p(x_1)p(x_2)p(x_3 \mid x_1)$	0.0002
$p(x_2)p(x_3)p(x_1 \mid x_2)$	0.0017
$p(x_2)p(x_1)p(x_3 \mid x_2)$	0.0004
$p(x_2)p(x_3)p(x_1 \mid x_3)$	0.0004
$p(x_1)p(x_3)p(x_2 \mid x_3)$	0.0003

The network with joint probability function $p(x_2)p(x_3)p(x_1 \mid x_2)$ has the highest score and will be used for adding the following edge.

Adding the second edge There are only four different networks possible as loops are not allowed in a Bayesian network. The joint probability for these networks in given in Table D.4 :

Table D.4: Joint probability function and scores for network with a second edge added

$p(D, B \mid \zeta)$	Score
$p(x_2)p(x_1 \mid x_2)p(x_3 \mid x_2)$	0.0010
$p(x_2)p(x_1 \mid x_2)p(x_3 \mid x_1)$	0.0002
$p(x_2)p(x_3)p(x_1 \mid x_2, x_3)$	0.0004
$p(x_3)p(x_2 \mid x_3)p(x_1 \mid x_2)$	0.0017

No more edges can be added to the network without creating loops. The best network has the highest probability and the lowest number of edges. In this case, the best network is therefore the network with the joint probability function $p(x_2)p(x_3)p(x_1 \mid x_2)$. In the following stage it is possible to generate the offspring.

Creating new offspring Given the best joint probability function, it is now possible to calculate the marginal frequencies of having an allele value of 1 at each gene position. From Table D.2 follows that the marginal frequencies are:

Table D.5: Marginal probability for gene 2 and gene 3

Gene	marginal probability
$x_2 = 1$	1/4
$x_3 = 1$	0/4

Using a random generator is it possible to determine the allele values for gene 2 and 3. The outcome of gene 1 depends on the value received for gene 2. Imagine that gene 2=0 than follows that the $p(x_1 = 0 \mid x_2 = 0) = 0.666$ and $p(x_1 = 0 \mid x_2 = 1) = 0.333$. Using the random generator again will give the final allele value for gene 1.

References

Afdeling Bos en Groen. (2000). *Beheersvisie voor de domeinbossen, andere openbare bossen en bossen gelegen in het VEN.*

Afdeling Bos en Groen. (2001). *De Bosinventarisatie van het Vlaamse Gewest. Resultaten van de eerste inventarisatie 1997–1999.* Ministerie van de Vlaamse Gemeenschap.

Anonymous. (2000). Gemiddelde Prijzen van Hout op Stam. *Houthandel en nijverheid*, 5–5.

Armstrong, M. P., & Densham, P. (1990). Database Organization Strategies for Spatial Decision Support Systems. *International Journal of Geographical Information Science*, *4*(1), 3–20.

Avery, T. E., & Burkhart, H. E. (1994). *Forest Measurements* (Fourth ed.). New York, U.S.A.: McGraw-Hill International Editions.

Bäck, T. (1994). *Selective Pressure in Evolutionary Algorithms: A Characterization of Selection Mechanisms.*

Bäck, T., Fogel, D. B., Whitley, D., & Angeline, P. J. (1997). Mutation. In T. Bäck, D. B. Fogel, & Z. Michalewicz (Eds.), *Handbook of Evolutionary Computation* (pp. C3.2:1–C3.2:14). Bristol and Oxford: IOP Publishing Ltd and Oxford University Press.

Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In J. j. Grefenstette (Ed.), *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms* (pp. 14–21). Lawrence Erlbaum.

Baker, W. L., & Cai, Y. M. (1992). The R.Le. Programs for Multiscale Analysis of Landscape Structure Using the GRASS GIS Geographical Information System. *Landscape Ecology*, *7*(4), 291–302.

Baker, W. L., & Mladenoff, D. J. (1999). Progress and Future Directions in Spatial Modeling of Forest Landscapes. In D. J. Mladenoff & W. L. Baker (Eds.), *Spatial Modeling of Forest Landscape Change: Approaches and Applications* (pp. 333–349). Cambridge, UK: Cambridge University Press.

Baluja, S., & Caruana, R. (1995). Removing the Genetics from the Standard Genetic Algorithm. In A. Prieditis & S. Russel (Eds.), *The Int. Conf. on Machine Learning 1995* (pp. 38–46). San Mateo, CA: Morgan Kaufmann Publishers.

Baluja, S., & Davies, S. (1997). *Combining Multiple Optimization Runs with Optimal Dependency Trees* (Tech. Rep. No. CMU-CS-97-157). Carnegie Mellon University.

Barrett, T. M., Gilless, J. K., & Davis, L. S. (1998). Economic and Fragmentation Effects of Clearcut Restrictions. *Forest Science*, *44*(4), 569–577.

Baskent, E. Z., & Jordan, G. A. (1991). Spatial Wood Supply Simulation Modelling. *The Forestry Chronicle*, *67*(6), 610–621.

Berck, P. (1999). Why Are the Uses Multiple. In F. Helles, P. Holten-Andersen, & L. Wichmann (Eds.), *Multiple Use of Forests and Other Natural Resources: Aspects of Theory and Application* (pp. 3–14). Dordrecht, The Netherlands: Kluwer Academic Publishers.

Berck, P., & Bible, T. (1984). Solving and Interpreting Large-Scale Harvest Scheduling Problems by Duality and Decomposition. *Forest Science*, *30*(1), 173–182.

Bevers, M., & Hof, J. (1999). Spatially Optimizing Wildlife Habitat Edge Effects in Forest Management Linear and Mixed-Integer Programs. *Forest Science*, *45*(2), 249-258.

Bishop, Y. M. H., Fienberg, S. E., & Holland, P. W. (1980). *Discrete Multivariate Analysis.* England: MIT Press.

Blickle, T. (1997). Tournament Selection. In T. Bäck, D. B. Fogel, & Z. Michalewicz (Eds.), *Handbook of Evolutionary Computation* (pp. C2.3:1–C2.3:4). Bristol and Oxford: IOP Publishing Ltd. and Oxford University Press.

Booker, L. B., Fogel, D. B., Whitley, D., & Angeline, P. J. (1997). Recombination. In T. Bäck, D. B. Fogel, & Z. Michalewicz (Eds.), *Handbook of Evolutionary Computation* (pp. C3.3:1– C3.3:27). Bristol and Oxford: IOP Publishing Ltd and Oxford University Press.

Booty, W. G., Lam, D. C. L., Wong, I. W. S., & Siconolfi, P. (2001). Design and Implementation of an Environmental Decision Support System. *Environmental modelling and software*, *16*, 453–458.

Borges, J. G., & Hoganson, H. M. (1999). Assessing the Impact of Management Unit Design and Adjacency Constraints on Forestwide Spatial Conditions and Timber Revenues. *Canadian Journal of Forestry Research*, *29*, 1764-1774.

Bos, J. (1994). *STAGES: A System for Generating Strategic Alternatives for Forest Management.* Unpublished doctoral dissertation, Universiteit Wageningen.

Bosman, P. A. N., & Thierens, D. (1999). *Linkage Information Processing in Distribution Estimation Algorithms* (Tech. Rep. No. 10). Department of Computer Science, Utrecht University.

Boston, K., & Bettinger, P. (1999). An Analysis of Monte Carlo Integer Programming, Simulated Annealing, and Tabu Search Heuristics for Solving Spatial Harvest Scheduling Problems. *Forest Science*, *45*(2), 292–301.

Brack, C. L., & Marshall, P. L. (1996). A Test of Knowledge-Based Forest Operations Scheduling Processes. *Canadian Journal of Forestry Research*, *26*, 1193-1202.

Brown, J. R., & MacLeod, N. D. (1996). Integrating Ecology Into Natural Resource Management Policy. *Environmental Management*, *20*, 289–296.

Brumelle, S., Granot, D., Halme, M., & Vertinsky, I. (1998). A Tabu Search Algorithm for Finding Good Harvest Schedules Satisfying Green-Up Constraints. *European Journal of Operational Research*, *106*, 408–424.

Buongiorno, J., & Gilles, J. K. (1987). *Forest Management and Economics.* New York: MacMillan Publishing Company.

Cameron, A. (2000). *Inventory Data Kirkhill Forest.*

Carlsson, M. (1999). A Method for Integrated Planning of Timber Production and Biodiversity: A Case Study. *Canadian Journal of Forestry Research*, *29*, 1183–1191.

Carter, D. R., Vogiatzis, M., Moss, C. B., & Arvanitis, L. G. (1997). Ecosystem Management or Infeasible Guidelines? Implications of Adjacency Restrictions for Wildlife Habitat and Timber Production. *Canadian Journal of Forestry Research*, *27*, 1302–1310.

Chen, J.-H., Goldberg, D. E., Ho, S.-Y., & Sastry, K. (2002). Fitness inheritance in multi-objective optimization. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, & N. Jonoska (Eds.), *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference* (pp. 319–326). New York: Morgan Kaufmann Publishers.

Chertov, O. G., Komarov, A. S., & Karev, G. P. (1999). *Modern Approaches in Forest Ecosystem Modelling, European Forest Institute Research Report No. 8.* Leiden, The Netherlands: Brill.

Chickering, D., Geiger, D., & Heckerman, D. (1995). Learning Bayesian Networks: Search Methods and Experimental Results. In *Proceedings of Fifth Conference on Artificial Intelligence and Statistics, Ft. Lauerdale, Florida* (pp. 112–128). Society for Artificial Intelligence in Statistics.

Church, R., & Daugherty, P. J. (1999). Considering Intergenerational Equity in Linear Programming-Based Forest Planning Models with MAXMIN Objective Functions. *Forest Science*, *45*(3), 366–373.

Chuvieco, E. (1993). Integration of Linear Programming and GIS for Land-Use Modelling. *International Journal of Geographical Information Systems*, *7*(1), 71–83.

Clements, S. E., Dallain, P. L., & Jamnick, M. S. (1990). An Operational, Spatially Constrained Harvest Scheduling Model. *Canadian Journal of Forest Research*, *20*, 1438–1447.

Coello, C. A. C., Van Veldhuizen, D. A., & Lamont, G. B. (2002). *Evolutionary Algorithms for Solving Multi-Objective Problems.* Dordrecht, The Netherlands: Kluwer Academics Publisher.

Congdon, P. (2001). *Bayesian Statistical Modelling.* West Sussex, England: Wiley and sons.

Cresswell, P., Harris, S., & Jefferies, D. J. (1990). *The History, Distribution, Status and Habitat Requirements of the Badger in Britain.* Peterborough, UK: Nature Conservancy Council.

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

Davis, L. S., Johsnon, K. N., Bettinger, P. S., & Howard, T. E. (2001). *Forest Management.* Boston: Mc Graw Hill.

Davis, R. G., & Martell, D. L. (1993). A Decision Support System That Links Short-Term Silvicultural Operating Plans with Long-Term Forest-Level Strategic Plans. *Canadian Journal of Forest Research*, *23*, 1078–1095.

Deb, K. (1999). Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. *Evolutionary Computation*, *7*(3), 205–230.

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms.* Chichester, UK: Wiley and Sons.

Deb, K., Agrawal, S., Pratab, A., & Meyarivan, T. (2000). *A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II* (KanGAL report No. 200001). Indian Institute of Technology, Kanpur, India.

Deb, K., Agrawal, S., Pratab, A., & Meyarivan, T. (2002). A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197.

Deb, K., & Jain, S. (2000). *Running Performance Metrics for Evolutionary Multiobjective Optimization* (Tech. Rep. No. 200004). Kanpur Genetic Algorithms Laboratory.

Densham, P. J. (1991). Spatial Decision Support Systems. In D. J. Maguire, M. F. Goodchild, & D. W. Rhind (Eds.), *Geographical Information Systems* (pp. 403–413). London: Longman Scientific Technical.

Ducheyne, E. (1999). *Het Opstellen van een Prototype Beheerssysteem voor Lineaire Aanplantingen, in Samenwerking met het OCMW Brugge.* Unpublished master's thesis, Faculty of Agricultural and Applied Biological Sciences, UGent.

Ducheyne, E. I., De Baets, B., & Wulf, R. R. D. (2003). Is Fitness Inheritance Really Useful for Real-World Applications. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, & L. Thiele (Eds.), *Lecture Notes in Computer Science 2632: Evolutionary Multi-Criterion Optimization, Second International Conference, EMO03* (pp. 31–43). Berlin: Springer-Verlag.

Ducheyne, E. I., De Wulf, R. R., & De Baets, B. (2001). Bi-Objective Genetic Algorithms for Forest Management: A Comparative Study. In L. Spector (Ed.), *Late breaking papers at the genetic and evolutionary computation conference (GECCO-2001)* (pp. 63–66). San Francisco, CA: AAAI.

Ducheyne, E. I., De Wulf, R. R., & De Baets, B. (2002). Using linkage learning for forest management planning. In E. Cantú-Paz (Ed.), *Late breaking papers at the genetic and evolutionary computation conference (GECCO-2002)* (pp. 109–114). New York, NY: AAAI.

Duerr, W. A. (1993). *Introduction to Forest Resource Economics.* New York, USA: McGraw-Hill Inc.

ESRI. (1996). *Using Avenue. Customization and Application Development for ArcView.*

ESRI. (2003). *ESRI Forum.* URL: http://arcscripts.esri.com.

Faber, B. G., Wallace, W. W., & Johnson, G. E. (1998). Active Response GIS for Resource Management and Spatial Decision Support Systems. *Photogrammetric Engineering and Remote Sensing*, *62*(1), 7–11.

Falçao, A. O., & Borges, J. G. (2001). Designing an Evolution Program for Solving Integer Forest Management Scheduling Models: An Application in Portugal. *Forest Science*, *47*(2), 158–168.

Falkenauer, E. (1998). *Genetic Algorithms and Grouping Problems.* Chichester: Wiley and sons.

Feng, C., & Lin, J. (1999). Using a genetic algorithm to generate alternative sketch maps for urban planning. *Computers Environment and Urban Systems*, *23*, 91–108.

Fonseca, C. M., & Fleming, P. J. (1993). Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In S. Forrest (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms* (pp. 416–423). San Mateo, California: Morgan Kauffman Publishers.

Fonseca, C. M., & Fleming, P. J. (1995). An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation*, *3*(1), 1–16.

Fonseca, C. M., & Fleming, P. J. (1996). On the Performance Assesment and Comparison of Stochastic Multiobjective Optimizers. In H. M. Voigt, W. Eibeling, I. Rechenberg, & H. P. Swefel (Eds.), *Parallel Problem Solving from Nature (PPSN)-IV)* (pp. 584–593). Berlin, Germany: Springer-Verlag.

Fonseca, C. M., & Fleming, P. J. (1997). Multiobjective Optimization. In T. Bäck, D. Fogel, & Z. Michalewicz (Eds.), *Handbook of Evolutionary Computation* (p. C4.5:1C4.5:9). Oxford: IOP Publishing Ltd. and Oxford University Press.

Fonseca, C. M., & Fleming, P. J. (1998). Multiobjective Optimization and Multiple Constraint Handling Using Evolutionary Algorithms — Part II: Application Example. *IEEE Transactions on Systems, Man and Cybernetics — Part A: Systems and Humans*, *28*(1), 38–47.

Fonseca, C. M. M. (1995). *Multiobjective Genetic Algorithms with Application to Control Engineering Problems.* Unpublished doctoral dissertation, University of Sheffield.

Franklin, J. F. (1994). Developing Information Essential to Policy, Planning and Management Decision Making: The Promise of GIS. In V. A. Sample (Ed.), *Remote Sensing and GIS in Ecosystem Management* (pp. 18–25). Washington D.C., USA: Island Press.

Garcia, O. (1990). Linear Programming and Related Approaches in Forest Planning. *New Zealand Journal of Forestry Science*, *20*(3), 307–331.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning.* USA: Addison-Wesley.

Goldberg, D. E., Deb, K., & Horn, J. (1992). *Massive Multimodality, Deception and Genetic Algorithms* (Tech. Rep. No. 92005). Illionois Genetic Algorithms Laboratory.

Goldberg, D. E., Deb, K., Kargupta, H., & Harik, G. (1993). *Rapid, Accurate Optimization of Difficult Problems Using Fast Messy Genetic Algorithms* (Tech. Rep. No. 93004). Illinois Genetic Algorithms Laboratory.

Gong, P. (1992). Multiobjective Dynamic Programming for Forest Resource Management. *Forest Ecology and Management*, *48*, 43-54.

Goodchild, M. (1992). Integrating GIS and Spatial Data Base Analysis: Problems and Possibilities. *International Journal of Geographical Information Systems*, *6*(5), 407–423.

Grefenstette, J. (1997a). Proportional Selection and Sampling Algorithms. In T. Bäck, D. B. Fogel, & Z. Michalewicz (Eds.), *Handbook of Evolutionary Computation* (p. C2.2:1-C2.2:7). Bristol and Oxford: IOP Publishing Ltd and Oxford University Press.

Grefenstette, J. (1997b). Rank-Based Selection. In T. Bäck, D. B. Fogel, & Z. Michalewicz (Eds.), *Handbook of Evolutionary Computation* (pp. C2.4:1–C2.4:6). Bristol and Oxford: IOP Publishing Ltd. and Oxford University Press.

Grefenstette, J. J., & Fitzpatrick, J. M. (1985). Genetic Search with Approximate Function Evaluations. In J. J. Grefenstette (Ed.), *Proceedings of an International Conference on Genetic Algorithms and Their Applications* (p. 112-120). Hillsdale, USA: Lawrence Erlbaum.

Hamilton, G. J., & Christie, J. M. (1971). *Forest Management Tables. Forestry Commission Booklet No. 34.* London, U. K.: Her Majesty's Stationery Office.

Harik, G. (1997). *Learning Gene Linkage to Efficiently Solve Problems of Bounded Difficulty using Genetic Algorithms.* Unpublished doctoral dissertation, The University of Michigan.

Harik, G. (1999). *Linkage Learning Via Probabilistic Modeling in the ECGA* (Tech. Rep. No. 99010). Ilinois Genetic Algorithms Laboratory at Urbana-Champaign.

Harik, G., Goldberg, D. E., Cantú-Paz, E., & Miller, B. (1999). The Gambler's Ruin Problem, Genetic Algorithms and the Sizing of Populations. *Evolutionary Computation*, *7*(3), 231–253.

Harik, G. R., Lobo, F., & Goldberg, D. E. (1998). The Compact Genetic Algorithm. In *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation* (pp. 523–528). New York: IEEE press.

Heckerman, D., Geiger, D., & Chickering, D. M. (1995). *Learning Bayesian Networks: The Combination of Knowledge and Statistical Data* (Tech. Rep. No. MSR-TR-94-09). Microsoft Research Advanced Techology Division.

Hof, J. G., Bevers, M., Joyce, L., & Kent, B. (1994). An Integer Programming Approach for Spatially and Temporally Optimizing Wildlife Populations. *Forest Science*, *40*(1), 177-191.

Hof, J. G., & Joyce, L. A. (1992). Spatial Optimization for Wildlife and Timber in Managed Forest Ecosystems. *Forest Science*, *38*(3), 489–508.

Hof, J. G., & Joyce, L. A. (1993). A Mixed Integer Linear Programming Approach for Spatially Optimizing Wildlife and Timber in Managed Forest Ecosystems. *Forest Science*, *39*(4), 816–834.

Hoganson, H. M., & Rose, D. W. (1984). A Simulation Approach for Optimal Timber Management Scheduling. *Forest Science*, *44*(4), 526–538.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor: University of Michigan Press.

Hollander, M., & Wolfe, D. A. (1999). *Nonparametric Statistical Methods.* United States of America: Wiley and Sons.

Hwang, C., & Masud, A. (1979). *Multiple Objective Decision Making - Methods and Applications.* Berlin: Springer-Verlag.

Jacob, C. (1998). Stochastic Search Methods. In M. Berthold & D. J. Hand (Eds.), *Intelligent Data Analysis: An Introduction* (pp. 299–394). Berlin: Springer.

Jamnick, M. S., & Walters, K. R. (1993). Spatial and Temporal Allocation of Stratum-Based Harvest Schedules. *Canadian Journal of Forestry Research*, *23*, 402-413.

Janssens, F., & de Schuyter, J. (1990). *Het Bosdecreet.* Brugge, Belgium: Vanden Broele.

Jaszkiewicz, A. (2000). *On the Performance of Multiple Objective Genetic Local Search on the 0/1 Knapsack Problem. A Comparative Experiment* (Tech. Rep. No. RA-002/2000). Institute of computing science, Poznań University of Technology.

Jensen, F. V. (1996). *An Introduction to Bayesian Networks.* London, UK: UCL Press Limited.

Johnson, K. N., & Scheurman, H. L. (1977). Techniques for Prescribing Optimal Timber Harvest and Investment under Different Objectives — Discussion and Synthesis. *Forest Science Monographs*(18), 31.

Jones, G. J., Meneghin, B. J., & Kirby, M. W. (1991). Formulating Adjacency Constraints in Linear Optimization Models for Scheduling Projects in Tactical Planning. *Forest Science*, *37*(1), 1283–1297.

Kargupta, H. (1995). *SEARCH, Polynomial Complexity and the Fast Messy Genetic Algorithm.* Unpublished doctoral dissertation, Department of Computer Science, University of Illinois, Urbana, Illinois.

Knowles, J., & Corne, D. (2002). On Metrics for Comparing Nondominated Sets. In *Proceedings of the 2002 Congress on Evolutionary Computation Conference* (pp. 711–716). IEEE.

Knowles, J. D., & Corne, D. W. (2000). Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, *8*(2), 149–172.

Kolman, B., & Beck, R. E. (1995). *Elementary Linear Programming with Application.* San Diego, USA: Academic Press.

Korhonen, P., Moskwitz, H., & Wallenius, J. (1992). Multiple Criteria Decision Support - a Review. *European Journal of Operational Research*, *63*(3), 361–375.

Koza, J. R. (1993). *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Massachusetts: The MIT Press.

Koza, J. R. (1994). *Genetic Programming II: Automatic Discovery of Reusable Programs.* Massachusetts: The MIT Press.

Kurttila, M. (2001). The Spatial Structure of Forests in the Optimization Calculations of Forest Planning - a Landscape Ecological Perspective. *Forest Ecology and Management*, *142*, 129–142.

Lammerts van Bueren, E. M. (1983). Een Landevaluatiebenadering Toegepast Op Bossen. *Nederlands Bosbouwtijdschrift*, *55*(1), 14–22.

Larranaga, P., & Lozano, J. A. (2002). *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation.* Dordrecht, Nederland: Kluwer Academics Publishers.

Leuschner, W. A. (1984). *Introduction to Forest Resource Management.* United States of America: Wiley and Sons, Inc.

Leuschner, W. A. (1990). *Forest Regulation, Harvest Scheduling, and Planning Techniques.* United States of America: Wiley and Sons, Inc.

Lobo, F. (2000). *The Parameter-Less Genetic Algorithm: Rational and Automated Parameter Selection for Simplified Genetic Algorithm Operation.* Unpublished doctoral dissertation, Universidade de Lisboa.

Lobo, F., & Harik, G. (1999). *Extended Compact Genetic Algorithm in C++* (Tech. Rep. No. 99016). Illinois Genetic Algorithms Laboratory at University of Illinois.

Lockwood, C., & Moore, T. (1993). Harvest Scheduling with Spatial Constraints: A Simulated Annealing Approach. *Canadian Journal of Forest Research*, *23*, 468–478.

Loomis, J. B. (1993). *Integrated Public Land Management: Principle and Applications to National Forests, Parks, Wildlife Refuges, and BLM Lands.* Columbia University Press.

Lu, F., & Erikkson, L. O. (2000). Formation of Harvest Units with Genetic Algorithms. *Forest Ecology and Management*, *130*, 57–67.

Matthews, K. B., Craw, S., Elder, S., Sibbald, A. R., & MacKenzie, I. (2000). Applying Genetic Algorithms to Multi-Objective Land Use Planning. In D. Whitley, D. Goldberg, E. Cantu-Paz, L. Spector, I. Parmee, & H. Beyer (Eds.), *Proceedings of the 2001 Genetic and Evolutionary Computation Conference. San Francisco, California, July 2001* (pp. 613–620). Morgan Kauffman.

Matthews, K. B., Sibbald, A. R., & Craw, S. (1999). Implementation of a Spatial Decision Support System for Rural Land Use Planning: Integrating Geographic Information System and Environmental Models with Search and Optimisation Models. *Computer and Electronics in Agriculture*, *23*, 9–26.

Mendelsohn, R. (1996). An Economic-Ecological Model for Ecosystem Management. In W. L. Adamowicz, P. C. Boxall, M. K. Luckert, W. E. Phillips, & W. A. White (Eds.), (pp. 213–222). Wallingford, UK: CAB International.

Michalewicz, Z. (1999). *Genetic Algorithms + Data Structures = Evolution Programs* (Third, Revised and Extended ed.). Berlin: Springer.

Michalewicz, Z., & Fogel, D. B. (2002). *How to Solve It: Modern Heuristics.* Berlin, Germany: Springer-Verlag.

Misund, G., Johansen, B., & Hasle, G. (1995). Integration of Geographical Information Technology and Constraint Reasoning - A Promising Approach to Forest Management. In T. J. Bjorke (Ed.), *Proceedings of the 5th Research Conference on GIS* (pp. 42–56).

Mitchell, M. (1996). *An Introduction to Genetic Algorithms.* Cambridge, Massachusetts: The MIT Press.

Moore, C. T., Conroy, M. J., & Boston, K. (2000). Forest Management Decisions for Wildlife Objectives: System Resolution and Optimality. *Computers and Electronics in Agriculture*, *27*, 25-39.

Mühlenbein, H., & Mahnig, T. (1999). FDA - a Scalable Evolutionary Algorithm for the Optimization of Additively Decomposed Functions. *Evolutionary Computation*, *7*(4), 353–376.

Mühlenbein, H., & Schlierkamp-Voosen, D. (1994). The Science of Breeding and its Application to the Breeder Genetic Algorithm BGA. *Evolutionary Computation*, *1*(4), 335–360.

Murray, A. T. (1999). Spatial Restrictions in Harvest Scheduling. *Forest Science*, *45*(1), 45–52.

Murray, A. T., & Church, R. L. (1995). Measuring the Efficacy of Adjacency Constraint Structure in Forest Planning Models. *Canadian Journal of Forest Research*, *25*, 1416–1424.

Naesset, E. (1997a). A Spatial Decision Support System for Long-Term Forest Management Planning by Means of Linear Programming and a Geographical Information System. *Scandinavian Journal of Forest Research*, *12*, 77–88.

Naesset, E. (1997b). Geographical Information Systems in Long-Term Forest Management and Planning with Special Reference to Preservation of Biological Diversity: A Review. *Forest Ecology and Management*, *93*, 121–136.

Nalli, A., Nuutinen, T., & Paivinen, R. (1996). Site–Specific Constraints in Integrated Forest Planning. *Scandinavian Journal of Forestry Research*, *11*, 85-96.

Neal, E., & Cheeseman, C. (1996). *Badgers.* Cambridge: University Press.

Neteler, M., & Mitasova, H. (2002). *Open Source GIS: A GRASS GIS Approach.* Boston, USA: Kluwer Academic Publishers.

Ohman, K., & Eriksson, L. O. (1998). The Core Area Concept in Forming Contiguous Areas for Long-Term Forest Planning. *Canadian Journal of Forestry Research*, *28*, 1032–1039.

Olson, C. M., & Orr, B. (1999). Combining tree growth, fish and wildlife habitat, mass wasting, sedimentation, and hydrologic models in decision analysis and long-term forest planning. *Forest Ecology and Management*, *114*, 339–348.

Osyczka, A. (2002). *Evolutionary Algorithms for Single and Multicriteria Design Optimization.* Heidelberg, New York: Physica-Verlag.

Pelikan, M. (2000). *A C++ Implementation of the Bayesian Optimization Algorithm (BOA) with Decision Graphs* (Tech. Rep. No. 2000025). Illinois Genetic Algorithms Laboratory.

Pelikan, M., Goldberg, D. E., & Cantú-Paz, E. (2000a). *Bayesian Optimization Algorithm, Population Sizing and Convergence* (Tech. Rep. No. 2000001). Ilinois Genetic Algorithms Laboratory.

Pelikan, M., Goldberg, D. E., & Cantu-Paz, E. (2000b). Linkage Problem, Distribution Estimation, and Bayesian Networks. *Evolutionary Computation*, *8*(3), 311–340.

Pelikan, M., Goldberg, D. E., & Sastry, K. (2001). Bayesian Optimization Algorithm, Decision Graphs, and Occam's Razor. In L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezesh, M. H. Garzon, & E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2001 - July 7-11, 2001, San Francisco, California* (pp. 519–526). San Francisco: Morgan Kauffman.

Philip, M. S. (1994). *Measuring Trees and Forests* (Second ed.). Wallingford, U.K.: CAB International.

Pirlot, M. (1992). General Local Search Heuristics in Combinatorial Optimization: A Tutorial. *Belgian Journal of operations reseach, statistics and computer science*, *32*(1-2), 7–68.

Purshouse, R. C., & Fleming, P. J. (2002). Why Use Elitism and Sharing In a Multi-objective Genetic Algorithm? In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, & N. Jonoska (Eds.), *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference* (pp. 520–527). New York: Morgan Kaufmann Publishers.

Purshouse, R. J., & Fleming, P. J. (2003). Conflict, Harmony and Independence: Relationships in Evolutionary Multi-Criterion Optimization. In C. M. Fonseca, E. Zitzler, & P. J. Fleming (Eds.), *Proceedings of the Second International Conference on Evolutionary Multi-Criterion Optimization, EMO03* (pp. 16–30). Faro, Portugal: Springer - Verlag.

Rissanen, J. (1989). *Stochastic Complexity in Statistical Inquiry* (Vol. 15). World Scientific.

Roise, J. P., Cubbage, F. W., Abt, R. C., & Siry, J. P. (2000). Regulation of Timber Yield for Sustainable Management of Industrial Forest Plantations – Theory and Practice. In K. von Gadow, T. Pukkula, & M. Tomé (Eds.), *Sustainable Forest Management* (pp. 217–257). Dordrecht, The Netherlands: Kluwer Academic Publishers.

Sastry, K., Goldberg, D. E., & Pelikan, M. (2001). Don't Evaluate, Inherit. In L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezesh, M. H. Garzon, & E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO 2001 - July 7-11, 2001, San Francisco, California* (pp. 551–558). San Francisco: Morgan Kauffman.

Schott, J. R. (1995). *Fault Tolerant Design Using Single and Multi-Criteria Genetic Algorithms.* Unpublished master's thesis, Massachussets Institute of Technology.

Seffino, L. A., Medeiros, C. B., Rocha, J. V., & Yi, B. (1999). WOODSS - A Spatial Decision Support System Based on Workflows. *Decision Support Systems*, *27*, 105–123.

Shannon, C. E. (1948). A Mathematical Theory of Communications. *Bells Systems Technical Journal*, *27*.

Sherali, H. D., & Liu, C. (1990). Identification of a Network Substructure and Some Algorithmic Considerations for Large-Scale Harvest Scheduling Problems. *Forest Science*, *36*(3), 599–613.

Smith, R. E., Dike, B. A., & Stegmann, S. A. (1995). Fitness Inheritance in Genetic Algorithms. In *Proceedings of the 1995 ACM Symposium on Applied Computing, February 26-28.* Nashville, TN, USA: ACM.

Snyder, S., & Revelle, C. (1997). Dynamic Selection of Harvests with Adjacency Restrictions: The SHARe Model. *Forest Science*, *43*(2), 213–222.

Snyder, S., Tyrrell, L. E., & Haight, R. G. (1999). An Optimization Approach to Selecting Research Natural Areas in National Forests. *Forest Science*, *45*(3), 458-469.

Spears, W. M. (1998). *The Role of Mutation and Recombination in Evolutionary Algorithms.* Unpublished doctoral dissertation, George Mason University.

Speidel, G. (1972). *Planung im Forstbetrieb.* Berlin, Germany: Verlag.

Spiegel, M. R. (1980). *Theory and Problems of Probability and Statistics.* Singapore: McGraw-Hill Book Co.

Srinivas, N., & Deb, K. (1994). Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, *2*(3), 221–148.

Strange, N., Tarp, P., Helles, F., & Brodie, J. D. (1999). A four-stage approach to evaluate management alternatives in multiple-use forestry. *Forest Ecology and Management*, *124*, 79–91.

Syswerda, G. (1989). Uniform Crossover in Genetic Algorithms. In J. D. Schaffer (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms* (pp. 2–9). Morgan Kaufmann.

Tarp, P., & Helles, F. (1997). Spatial Optimization by Simulated Annealing and Linear Programming. *Scandinavian Journal of Forest Research*, *12*, 390–402.

Taylor, K., Walker, G., & Abel, D. (1999). A Framework for Model Integration in Spatial Decision Support Systems. *International Journal of Geographical Information Science*, *13*(6), 533–555.

Thierens, D., & Goldberg, D. E. (1993). Mixing in Genetic Algorithms. In *Proceedings of the Fifth International Conference on Genetic Algorithms* (p. 38-45). Morgan Kaufman.

Twery, M. J., Rauscher, H. M., Bennet, D. J., Thomasma, S. A., Stout, S. L., Palmer, J. F., Hoffman, R. E., DeCalesta, D. S., Gustafson, E., Cleveland, H., Grove, J. M., Nute, D., Kim, G., & Kullasch, R. P. (2000). NED-1: Integrated Analyses for Forest Stewardship Decisions. *Computers and Electronics in Agriculture*, *27*, 167–193.

Ungerer, M. J., & Goodchild, M. F. (2002). Integrating Spatial Data Analysis and GIS: An New Implementation Using the Component Object Model (COM). *International Journal of Geographical Information Sciences*, *16*(1), 41–53.

Van Deusen, P. C. (1999). Multiple Solution Harvest Scheduling. *Silva Fennica*, *33*(3), 207–216.

Van Deusen, P. C. (2001). Scheduling Spatial Arrangement and Harvest Simultaneously. *Silva Fennica*, *35*(1), 85–92.

Van Dijk, S., Thierens, D., & de Berg, M. (2000). *Using Genetic Algorithms for Solving Hard Problems in GIS* (Tech. Rep. No. 32). Department of Computer Science, Utrecht University.

Van Veldhuizen, D. A., & Lamont, G. B. (1999). Multiobjective Evolutionary Algorithm Test Suites. In J. Carroll, H. Haddad, D. Oppenheim, B. Bryant, & G. B. Lamont (Eds.), *Proceedings of the 1999 ACM Symposium on Applied Computing* (pp. 351–357). San Antonio, Texas: ACM.

Van Veldhuizen, D. A., & Lamont, G. B. (2000). Multiobjective Evolutionary Algorithms Analyzing the State-of-the-Art. *Evolutionary Computation*, *8*(2), 125–147.

Vanhaeren, R. (2002). *Het Bosdecreetboek* (Tweede, herziene ed.). Brugge: Vanden Broele.

Westra, T. (2002). *Integratie Van Genetisch Algorithmes En GIS Voor de Optimalisatie Van Ecologische En Economische Functies in Kirkhill Forest, Aberdeen, Schotland.* Unpublished master's thesis, Faculteit Landbouwkundige en Toegepaste Biologische Wetenschappen, Universiteit Gent.

Yoshimoto, A., & Brodie, J. D. (1994). Comparative Analysis of Algorithms to Generate Adjacency Constraints. *Canadian Journal of Forest Research*, *24*, 1277–1288.

Zar, J. H. (1999). *Biostatistical Analysis* (Fourth ed.). New Jersey, U.S.A.: Prentice-Hall Inc.

Zhang, Z., & Griffith, D. A. (1997). Developing User-Friendly Spatial Statistical Analysis Modules for GIS: An Example Using ArcView. *Computers, environment and urban systems*, *21*(1), 5–29.

Zitzler, E. (1999). *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications.* Unpublished doctoral dissertation, Institut für Technische Informatik und Kommunikationsnetze.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, *8*(2), 173–195.

Zitzler, E., Laumanns, M., Thiele, L., Foneseca, C. M., & Fonseca, V. G. da. (2002). Why quality assessment of multiobjective optimizers is difficult. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, & N. Jonoska (Eds.), *Gecco 2002: Proceedings of the genetic and evolutionary computation conference* (pp. 666–674). New York: Morgan Kaufmann Publishers.

Curriculum vitae

Personalia

Voornamen, naam	Els Inge Ducheyne
Geboortedatum	16/06/1976
Geboorteplaats	Oostende
Adres	Flamingostraat 7, 9000 Gent
Telefoonnummer	09-242 04 47
Nationaliteit	Belg
Burgerlijke staat	Gehuwd

Opleiding

1999-2002	Doctoraatsopleiding in het land-en bosbeheer, FL&TBW, Universiteit Gent
1994 – 1999	Bio-ingenieur, specialisatie Land- & Bosbeheer, FL&TBW, Universiteit Gent. Diploma behaald met grote onderscheiding. *Scriptie*: Het opstellen van een prototype beheerssysteem voor lineaire aanplantingen. Promotor: Prof. dr. ir. R. De Wulf
1992 -1994	Wiskunde-Industriële Wetenschappen, Atheneum I, Oostende
1988 - 1992	Latijn-Wiskunde, Atheneum I, Oostende.

Loopbaanoverzicht - werkervaring

1999 - 2003	Doctoraatsbursaal Bijzonder Onderzoeksfonds Universiteit Gent

Buitenlandse ervaring tijdens opleiding en werk

2002	Studieverblijf 2/11- 10/11, Universidade do Algarve, Faro, Portugal (Prof. Dr. C. Fonseca, Prof. Dr. F. Lobo)
2001	Studieverblijf 26/9 - 26/10, Universidade do Algarve, Faro, Portugal (Prof. Dr. C. Fonseca, Prof. Dr. F. Lobo)
2000	Verzameling dataset Aberdeen University, 5/12-10/12
1997-1998	Verblijf van 6 maanden aan Aberdeen University, Socratesprogramma

Congressen en studiedagen

2003	• ORBEL 17, 23-24 januari, Brussel • Starters in het bosonderzoek, 25 februari, Brussel • Evolutionary multicriterion optimization (EMO) 2003, April 8-11, Faro, Portugal
2002	• Genetic and Evolutionary Computation Conference (GECCO 2002) July 9-13, New York, NY • 8th PhD symposium, Faculteit Landbouwkundige en Toegepaste Biologische wetenschappen, 9 oktober, Gent • Seminario da ADEEC, November 5, Universidade do Algarve, Faro, Portugal
2001	• Genetic and Evolutionary Computation Conference (GECCO 2001), July 7-11, San Francisco, CA • Seminario da ADEEC, 11 oktober, Universidade do Algarve, Faro, Portugal
2000	• Geographic Information Systems and Remote Sensing For Sustainable Forest Management: Challenge and Innovation in the 21st century, Edmonton, Canada, February 23 - 25 • Genetic and Evolutionary Computation Conference (GECCO-2000), July 8-12, Las Vegas, NE

Begeleiding en promotor scripties

- Copromotor voor Annelies Sevenant, 2000-2001, Vegetatiekartering in een duingebied (Doornpanne, Koksijde) aan de hand van hoge resolutie false colour orthofoto's. Thesis ter behaling van de graad Bio-ir in het land- en bosbeheer.

- Copromotor voor Rafael Ramirez de Verger y Salas, 2000-2001, Recreation terrain suitability analysis in Kirkhill Forest (Scotland) using GIS technology. Socratesstudent.

- Copromotor voor Toon Westra, 2001-2002, Integratie van genetische algoritmes en GIS voor optimalisatie van ecologische en economische functies

in Kirkhill Forest, Aberdeen, Schotland. Thesis ter behaling van de graad Bio-ir in het land- en bosbeheer. Laureaat SOGESCI Thesis - AWARD 2002-2003.

Onderwijsactiviteiten: practica

- Teledetectie (2000-2001)
- LIS basis (1999-2000, 2000-2001)
- LIS vegetatie (2000-2001)
- Bosbeheersregeling (1999-2000)

Wetenschappelijke output

- Els I. Ducheyne, Robert R. De Wulf and Bernard De Baets. Bi-objective Genetic Algorithms for Forest Management: A Compartive Study. In. D. Whitley (Ed.): Late Breaking Papers at the Genetic and Evolutionary Computation Conference (GECCO 2001) (pp. 63-67). San Francisco, CA: AAAI
- Els I. Ducheyne, Robert R. De Wulf and Bernard De Baets. Using linkage learning for forest management. In. E. Cantú-Paz (Ed.): Late Breaking Papers at the Genetic and Evolutionary Computation Conference (GECCO 2002) (pp. 109-114). New York, NY:AAAI
- Els I. Ducheyne, Bernard De Baets and Robert R. De Wulf. Is fitness inheritance useful for real-world applications? EMO03, Faro, Portugal. Lecture Notes in Computer Sciences 2632, p. 31–43 (A1-paper)
- Koen Mertens, Lieven Verbeke, Els Ducheyne and Robert De Wulf, Introduction of genetic algorithms in sub-pixel mapping, International Journal of Remote Sensing, In Press (A1-paper)
- Els I. Ducheyne, Bernard De Baets , and Robert R. De Wulf. Probabilistic models for linkage learning in forest management*, Evolutionary Computation*. Submitted.
- Els I. Ducheyne, Robert R. De Wulf and Bernard De Baets. Single versus multiple objective genetic algorithms for solving the even-flow forest management problem, *Forest Ecology and Management*. Submitted.
- Els I. Ducheyne, Bernard De Baets and Robert R. De Wulf. Don't inherit, evaluate. In progress.
- Els I. Ducheyne, Robert R. De Wulf and Bernard De Baets. A spatial approach to forest management optimisation: linking GIS and multiple objective genetic algorithms. In progress.