Decision Engineering

Springer-Verlag Berlin Heidelberg GmbH

Series Editor

Dr. Rajkumar Roy
Department of Enterprise Integration
School of Industrial and Manufacturing Science
Cranfield University
Cranfield
Bedford
MK43 0AL
UK

Other titles published in this series

Cost Engineering in Practice
John McIlwraith
Publication due August 2003

Yann Collette · Patrick Siarry

Multiobjective Optimization

Principles and Case Studies

Springer

PhD Yann Collette
rue Laurent Mermet 14
77500 Chelles
France

Professor Patrick Siarry
Université de Paris XII
L.E.R.I.S.S.
av. du Général de Gaulle 61
94010 Créteil
France

Original french edition published by Groupe Eyrolles, 2002

1st ed. 2003. Corrected 2nd. printing

Cataloging-in-Publication Data applied for
Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie;
detailed bibliographic data is available in the Internet at <http://dnb.de>

ISBN 978-3-642-07283-3 ISBN 978-3-662-08883-8 (eBook)
DOI 10.1007/978-3-662-08883-8

Originally published by Springer-Verlag Berlin Heidelberg New York in 2004
Softcover reprint of the hardcover 1st edition 2004

Typesetting: Digital data supplied by authors
Cover-Design: medio Technologies AG, Berlin
Printed on acid-free paper 62/3020Rw 5 4 3 2 1 0

Contents

Preface

Engineers daily face technological problems of growing complexity in various domains such as image processing, the development of mechanical systems, operational research and electronics. The problem to be solved can frequently be expressed as an *optimization problem* in which we define one objective function, or cost function (sometimes more than one), to be minimized considering all the parameters of the problem. For example, in the famous traveling salesman problem, we try to minimize the length of a journey made by a salesman who has to visit a fixed number of towns before coming back to the starting town. The definition of an optimization problem is often completed by some *constraints*: all the parameters (or *decision variables*) of the proposed solution must respect these constraints to be feasible.

A lot of "classical" optimization methods exist to solve such problems. These methods can be used providing that certain mathematical conditions are satisfied: thus, *linear programming* efficiently solves problems where the objective function and constraints are linear with respect to the decision variables. Unfortunately, the situations encountered in practice often include one or more difficulties which make these methods inapplicable: for example, the objective function may be nonlinear, or even may have no analytic expression in terms of the parameters; or, alternatively, the problem may have contradictory objective functions.

To clarify this situation, we can consider two domains, *multiobjective optimization* and *hard optimization*, which are related to two sources of complications. The first domain, to which this book is dedicated, deals with concurrent objective functions, while the second domain deals with the difficulties inherent in the properties of the objective function. We shall see that these aspects are often linked in practice. Before we proceed to our main topic of multiobjective optimization, it is necessary to say some more about the environment of hard optimization.

Hard Optimization

In reality, we can discern two kinds of optimization problems: combinatorial problems (or "discrete" problems) and continuous-variable problems. For the sake of clarity, let us give some examples: amongst combinatorial problems, we find the traveling salesman problem mentioned above, and also the problem of scheduling the tasks belonging to a project. Two examples of continuous problems are the following: the identification of the parameters of an electronic device from experimental data char-

acterizing this device, and the optimal renovation of an image starting from a blurred one.

This discrimination is necessary to limit the domain of hard optimization. Indeed, two kinds of problem are labeled with this name, which is not strictly defined (it is linked, in fact, to the state of the art in terms of optimization):

- Some combinatorial optimization problems, for which we do not know any *fast* exact algorithms. This is the case, particularly, for problems labeled "NP-hard", that is to say, briefly, for which an exact solution is not possible in a computing time proportional to N^n, where N is the number of unknown parameters of the problem and n is an integer.
- Some continuous-variable problems, for which we do not know any algorithms able to locate a global optimum with certainty in a finite number of computations.

Efforts have been made, separately, to solve these two kinds of "difficult" problem. In the domain of continuous optimization, there exists a vast number of classical methods labeled "global optimization methods", but these techniques are often inefficient if the objective function does not possess some particular structural properties, such as convexity. In the domain of combinatorial optimization, a vast number of *heuristics*, which produce nearly optimal solutions, have been developed; but most of these heuristics have been tailored specifically to a unique problem.

Metaheuristics

The appearance of a new kind of method, called *metaheuristics*, has allowed both of the preceding domains to become friends again: indeed, these methods can be applied to various sorts of combinatorial optimization problems and can also be adapted to continuous problems. These methods, which include the simulated annealing method, genetic algorithms, tabu search and ant colony algorithms, have been developed since 1980 with a common aim: to solve difficult optimization problems in the best way possible. They have in common, furthermore, the following characteristics:

- They are, at least partly, *stochastic*: this approach can handle the combinatorial explosion of possibilities.
- Their origins are combinatorial: they have the advantage, crucial in the continuous case, of being *direct*, which means they do not need to compute the derivatives of the objective function.
- They are inspired by *analogies*: with physics (simulated annealing, simulated diffusion, etc.), with biology (genetic algorithms, tabu search, etc.) or with ethology (ant colony and particle swarm methods, etc.).
- They are able to guide, in a particular task, another specialized search method (for example, another heuristic or a *local* exploration method).
- They share the same drawbacks: difficulties in *tuning* the parameters of the method, and the high *computation time.*

Metaheuristics are not mutually exclusive. In the current state of research, it is often impossible to predict with certainty the efficiency of a method when it is applied to a problem. The current tendency is towards the use of *hybrid methods,* which try to exploit the specific advantages of different approaches by combining them. We shall see that metaheuristics play an important role in this book, because they have contributed to the renewal of multiobjective optimization.

To close these introductory considerations on difficult mono-objective optimization, we shall briefly analyze the source of the efficiency of metaheuristics, and then outline some extensions of these methods. This study allows us to sketch a general classification of mono-objective optimization methods.

Source of efficiency of metaheuristics

For the sake of clarity, let us take a simple example of an optimization problem: the positioning of the components of an electronic device. The objective function to be minimized is the total length of the wires, and the decision variables are the positions of the components of the circuit. The form of the objective function of this problem can be sketched as in Fig. 0.1, and it depends on the "configuration": each configuration corresponds to a particular placement of components, associated with a choice of value for each of the decision variables.

When the space of possible configurations shows a complicated structure, it is difficult to locate the global minimum c_*. We explain below the failure of a "classical" iterative algorithm, before commenting on the efficient behavior of metaheuristics.

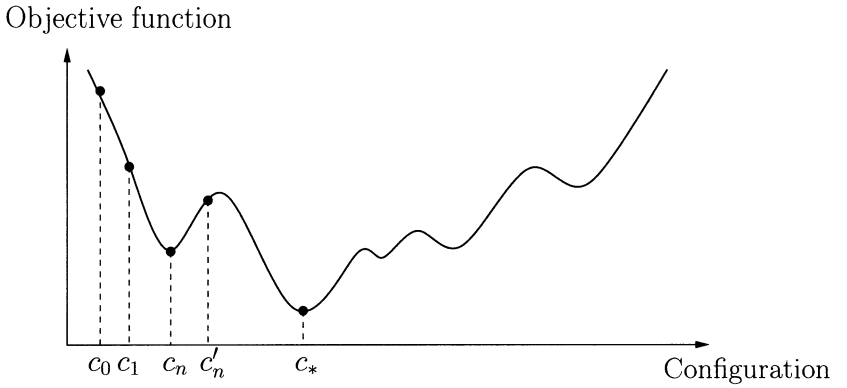


Fig. 0.1. Shape of the objective function of a difficult optimization problem depending on the "configuration".

Trapping of a "classical" iterative algorithm in a local minimum

The principle of a classical algorithm of "iterative improvement" is the following: we start with an initial configuration c_0, which can be randomly chosen, or – for example, in the case of the positioning of the components of an electronic device – be a configuration drawn by the designer. We then try to perform an elementary modification, often called a "*movement*" (for example, we permute two randomly chosen components, or we translate one of them), and we compare the values of the objective function before and after this modification. If the change brings about a decrease in the objective function, it is accepted, and the configuration obtained c_1, which is "close" to the previous configuration, will become the reference point for a new try. Otherwise, we come back to the previous configuration before making a new

trial. This process is iterated until each new modification brings about an increase in the objective function. Figure 0.1 shows that this iterative improvement algorithm (also called the *descent method*) does not allow one, in general, to locate the global minimum, but only a local minimum c_n, which corresponds to the best accessible solution with respect to initial hypothesis.

To improve the efficiency of the method, we can, of course, apply this method many times, with different initial conditions arbitrarily chosen, and take as the final solution the best of the local minima computed by the method; however, this process greatly increases the computing time of the algorithm, and is not guaranteed to discover the optimal configuration c_*.

Ability of metaheuristics to escape from a local minimum

To overcome the obstacle of local minima, another idea has been very fruitful, so much so that it has become a basis for all the *neighborhood* metaheuristics (simulated annealing and tabu search): the goal is to allow, from time to time, a *climbing* movement, that is to say, to allow a temporary degradation of the situation when the current configuration is changed. This is the case if we change from c_n to $c_n^{'}$ (Fig. 0.1). A degradation control mechanism, specific to each metaheuristic, allows one to avoid divergence of the process. It then becomes possible to escape from trapping at a local minimum, to explore another fruitful "valley".

"Distributed" metaheuristics (such as genetic algorithms) have mechanisms that allow them to escape from trapping at local minima of objective functions. These mechanisms (such as *mutation* in genetic algorithms) that modify a solution raise illustrate the collective mechanisms that fought against local minima represented by the parallel control of a "population" of solutions.

Extensions of metaheuristics

We shall now describe some of the extensions proposed to deal with some particular features of optimization.

Adaptation to continuous-variable problems

These problems, by far the most common in engineering, attract the attention of few computer scientists. Most of the metaheuristics have combinatorial origins and have been adapted to the continuous case; they have recourse to a strategy of discretizing the variables. The discretization step must be adjusted during the course of the optimization to guarantee regularity in the progress toward the optimum and accuracy of the result.

Parallelization

Multiple ways of parallelizing various metaheuristics have been proposed. Some of these techniques are general; some others, on the other hand, take advantage of particular features of the problem. Thus, in the problem of component positioning, tasks can be naturally divided between many processors: each one takes care of performing optimization inside a certain area, and information are periodically shared between neighboring processors.

Multimodal optimization

The task, now, is to construct a set of optimal solutions instead of just one solution. Genetic algorithms are particularly well adapted to this task, owing to their distributed nature. Some variants of the "multipopulation" type use many populations in parallel, each population trying to locate all the optima.

Hybrid methods

After the triumphalism of the masters of such metaheuristics in the early days, the time has come to draw up the balance sheet and to accept the complementarity between these methods and other approaches: hence the appearance of hybrid methods.

General classification of mono-objective optimization methods

To attempt to sum up the preceding considerations, we propose in Fig. 0.2 a general classification of mono-objective optimization methods. We find, in this figure, the main distinctions outlined above:

- We make a distinction, in the first place, between combinatorial optimization and continuous optimization.
- For combinatorial optimization, we use approximate methods when we face a difficult problem; in this case, we can make a choice between a "specialized" heuristic entirely dedicated to the problem and a metaheuristic.
- For continuous optimization, we can quickly make a separation between the linear case (which can be handled by linear programming for example) and the nonlinear case, where we can find a difficult optimization framework; in this case, a pragmatic solution may be to use repeatedly a local method which either exploits or does not exploit, derivatives of the objective function. If there are too many local minima, we must use a global method: we then find metaheuristics, which are a good alternative to classical global optimization methods which need restrictive mathematical properties with respect to the objective function.
- Amongst metaheuristics, we can make a distinction between "neighborhood" metaheuristics which improve one solution at a time (simulated annealing, tabu search, etc.) and "distributed" metaheuristics, which manipulate in parallel a whole population of solutions (genetic algorithms, etc.).
- Lastly, hybrid methods often combine a metaheuristic and a local search method. This combination can be in the form of a switching between the metaheuristic and the local search method, which is responsible for the improvement of the current solution. But the two approaches can be combined in very complex ways.

Let us cite some examples that exemplify these categories:

- Exact methods:
  these methods search for the global optimum. To do this, they enumerate the solutions in the search space. For example, the "branch and bound" method (see [Hillier et al. 95]) in which we execute the following steps:
  - we divide the search space into subspaces,

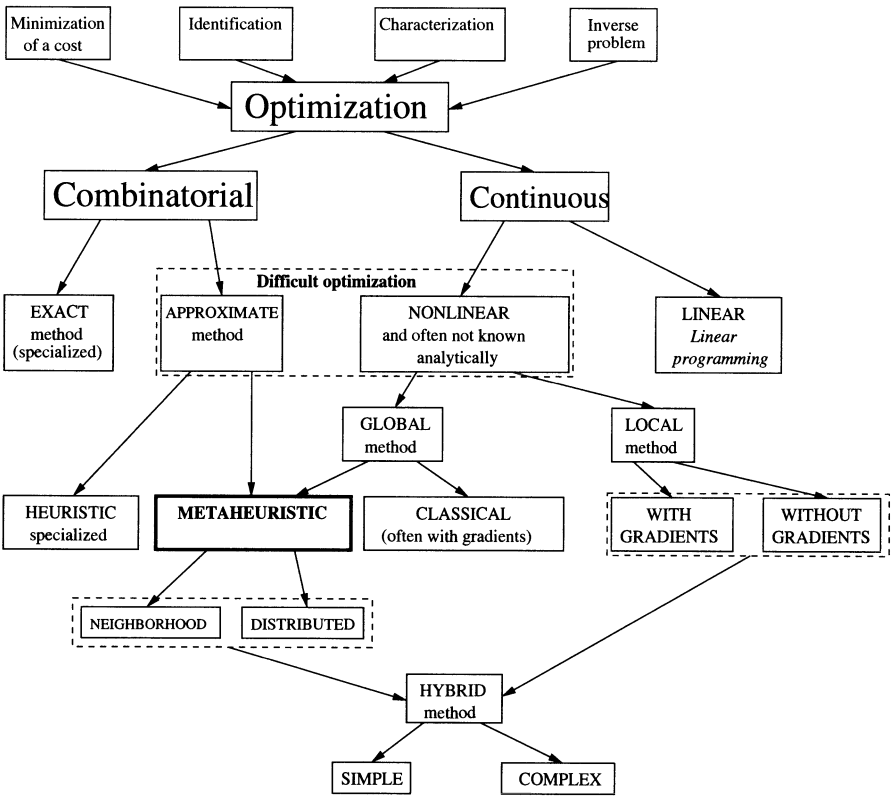


Fig. 0.2. General classification of monobjective optimization methods.

- we look for a minimum boundary associated with each subspace for the objective function,
- we remove each "bad" subspace,
- we repeat the preceding steps until we find the global optimum.

This method allows us to find the global optimum exactly, in some kinds of problems. It is not possible to apply this method to problems for which the size of the search space is too high.

- Approximate methods:
  - Heuristic methods: these have been developed to solve particular types of problems. They work only for the problem for which they have been developed, for example (see [Ausiello et al. 99]) the "first fit" algorithm dedicated to the "bin packing" problem.
  - Metaheuristic methods: these are based on a general idea (the imitation of a natural process, such as the annealing of a metal in the case of simulated annealing). They can be adapted to all kinds of problem and give good results. For example, simulated annealing and the genetic algorithms, which will be presented later in this book.

Nonlinear methods:

- Global:

- Metaheuristics: the driving idea on which a metaheuristic is built is sufficiently general to be easily transposed to continuous optimization problems.
- Classical: these methods can be considered as "foundation" methods in optimization. In this family, we find the gradient descent method. This method, when applied to a convex function, allows one to find the global optimum.

- Local:
  - With objective function derivatives: these methods use the derivatives to perform the search for the optimum. When this technique is applied to a general continuous optimization problem, it is not always possible to find the global optimum of the problem. In this case, we find a local optimum.
  - Without objective function derivatives: this family gathers together a diversity of methods. Amongst these methods, we can cite:
    - Constructive methods: we construct the solution one variable after another, and we forbid the modification of a variable already modified. For example, the greedy method belongs to the family of constructive methods.
    - Stochastic methods: these are based on a random process. For example, we can cite the random walk method. In this method, we randomly choose a solution in the neighborhood of the current solution and this solution becomes the current solution whatever the value of the objective function is.

Multiobjective optimization

The main difficulty that we encounter in mono-objective optimization comes from the fact that modeling a problem with just one equation can be a very difficult task. The goal of modeling the problem using just one equation can introduce a bias during the modeling phase.

Multiobjective optimization allows a degree of freedom which is lacking in mono-objective optimization. This flexibility is not without consequences for the method used to find an optimum for the problem when it is finally modeled. The search will give us not a unique solution but a set of solutions. These solutions are called *Pareto solutions*, and the set of solutions that we find at the end of the search is called the *tradeoff surface.*

After having found some solutions of the multiobjective optimization problem, we come up against some difficulties: we must select a solution from this set. The solution selected by the user will reflect tradeoffs performed by the user with respect to the various objective functions.

The decision maker is "human"; he/she will make choices, and one of the goals of multiobjective optimization is to model the choices of the decision maker, or rather, his/her preferences. To model these choices, we can apply two overall theories:

- multi-attribute utility theory (see [Keeney et al. 93]);
- multicriteria decision aid theory (see [Roy et al. 93]).

These two theories have different approaches.

The first approach models the preferences of the decision maker, postulating that, implicitly, each decision maker will try to maximize a function called the *utility function.* This approach does not accept, at the end of the selection phase, any solutions of equal ranks.

The second approach tries to reproduce the selection process of one or more decision maker. To do so, we use tools allowing us to select solutions from a set of solutions. This approach accepts solutions of equal ranks.

Another difficulty which we have to face before performing an optimization is to select an optimization method. We can divide multiobjective optimization methods into three families. These families are defined using the stage in process when we perform tradeoff with respect to the objective functions. We have the following families:

- *A priori* optimization methods:
  In these methods, the tradeoff that we would like to perform with respect to the objective functions is determined before the execution of the optimization method. To obtain the solution to our problem, we have to perform one search and, at the end of the optimization, the solution we obtain will reflect the tradeoff that we have performed between the objective functions before starting the search.
  This method is interesting in the sense that we have to perform only one search to find the solution. However, we must not forget the work we have to do to model the tradeoff before finding this result. We must not forget, also, that the decision maker is "human" and will be capable of noticing if the solution obtained at the end of the optimization is not totally satisfactory; in this case the decision maker might like, after examination of the solution, a solution which performs another tradeoff between objective functions.
- *Progressive* optimization methods:
  In these methods, we ask questions of the decision maker during the optimization such that he/she can reorient the search toward areas capable of containing solutions which perform the tradeoff between objective functions that the decision maker would like to perform now.
  These methods, although applying original techniques to model preferences of the decision maker, have the following drawback: they require the attention of the decision maker during the whole optimization process.
  This drawback is not a great one when we face an optimization problem for which the duration needed to evaluate an objective function is not too large. For other problems, the use of such a method can be tricky. It is difficult to monopolize the attention of a decision maker during a period which can be as long as several hours by asking him/her questions every ten minutes or every hour.
- *A posteriori* optimization methods:
  In these methods, we search for a set of solutions that are regularly spaced in the solution space. The immediate aim is to show these solutions to the decision maker so as to allow him/her to select the solution that satisfies him/her best by judging the various proposed solutions.
  Here, it is not necessary anymore to model the preferences of the decision maker. We merely produce a set of solutions which will be proposed to the decision maker. There is a nonnegligible time profit relative to the preference-modeling phase of the a priori family of methods.
  The main drawback that we must emphasize is that now we have to generate a set of regularly spaced solutions. This task not only is difficult, but also can require a prohibitive execution time.

In comparison with multiobjective optimization, we notice that we have gained some more degrees of freedom to model our problem. On the other hand, while the selection of a mono-objective optimization method is already hard owing to the great diversity of available methods, we face, in multiobjective optimization, a greater diversity.

This new kind of optimization, however, allows us to achieve success in various areas of optimization, for example:

- sizing of networks,
- structural optimization,
- scheduling problems.

The dynamism of the multiobjective optimization community shows that this domain is expanding:

- new conferences dedicated to multiobjective optimization are being hold;
- the number of publications related to multiobjective optimization is growing quickly;
- industrial interest in multiobjective optimization is obvious.

In this book, we shall present a detailed but not exhaustive description of the state of the art of multiobjective optimization methods, and a description of the related theory and of some applications.

Structure of the book

Part I: Principles of multiobjective optimization methods

Chapter 1: Introduction: multiobjective optimization and domination.
We describe the various ideas related to multiobjective optimization through some simple examples. Reading of this key chapter is necessary to understand the content of the rest of the book.

Chapter 2: Scalar methods.
In this chapter, the best-known methods in multiobjective optimization are described. We have tried to sketch these methods using some simple analytical multiobjective problems. If the solution of an example does not allow one to gain a deeper understanding of the method (because of heavy calculation, for example), the computation is not treated in details.

Chapter 3: Interactive methods.
The methods described in this chapter are certainly the hardest to understand. They are based on sequences using the methods of Chap. 2 and interact with the decision maker in a way which is not always simple. These methods are used less often today in the domain of engineering sciences, and a reading of this chapter is not essential. Nevertheless, readers interested in decision aids should read this chapter because, in this field, interactive methods are still used.

Chapter 4: Fuzzy methods.
Fuzzy logic has had a lot of success in multiobjective optimization. It allows one to "model a decision maker" using a less prescriptive method than traditional tools of multiobjective optimization and decision aids. The ideas required for the understanding of these methods are presented in the first part of this chapter.

Chapter 5: Multiobjective methods using metaheuristics.
The chapter really needs a wider presentation which should occupy several volumes. We have chosen to present some elements essential to the understanding of this subject. We have also presented the most widespread

approaches in the engineering sciences domain. The interested reader will find in the bibliography a list of references with, for some papers, internet links that allow the reader to download those papers.

Chapter 6: Decision aid methods.
This is again a subject which needs several volumes for an exhaustive presentation. We have chosen, once again, to present the most representative methods in this domain. The presentation of these methods is based on some reference books. The interested reader can consult these books for a more detailed presentation of these methods.

Part II: Evaluation of methods, and criteria for choice of method

Chapter 7: Performance measurement.
We have collected together in this chapter some tools that allow measurement of the performances of multiobjective optimization methods. The behavior of these tools is sketched with simple examples.

Chapter 8: Test functions for multiobjective optimization methods.
As with mono-objective optimization, multiobjective optimization needs problems that allow one to evaluate the efficiency of the optimization methods. First, we present the family of Deb test problems. These test problems, which are widely used, allow one to tune various parameters of the search space and illustrate nicely some specific difficulties that we can encounter in multiobjective optimization. We also present a family of constrained multiobjective test problems which is, at present, the one and only coherent family of multiobjective constrained test problems.

Chapter 9: An attempt to classify multiobjective optimization methods.
The diversity of methods is an asset. Nevertheless, it leads to the following question: how to choose a suitable method for a given problem? This chapter presents some techniques that "partially" solve this difficulty.

Part III: Case studies

Chapter 10: Case study 1: qualification of scientific software.
Here, we begin a presentation of applications illustrating some concrete uses of multiobjective optimization methods in various domains. This chapter deals with the qualification of scientific software, that is to say, the way to tune the parameters of scientific software so that the results computed by this software are closer to reality. The problem has been solved in the space of decision variables, and not in the space of objective functions as done classically.
This chapter was written in collaboration with M. Dumas of CEA.

Chapter 11: Case study 2: study of the extension of a telecommunication network.
The application illustrated in this chapter deals with the sizing of a telecommunication network. This is a major field of application of multiobjective optimization and is particularly important at present.
This chapter was written by C. Decouchon-Brochet of France Télécom Research and Development.

Chapter 12: Case study 3: multicriteria decision tools to deal with bids.
In this chapter, an application of a multicriteria decision aid method (the ELECTRE TRI method) is presented. This application deals with the sorting of calls for bids into various categories:

- calls for bids that one is sure to win,

- calls for bids with an uncertain outcome,
- calls for bids that one is sure to lose.

This chapter was written by J.-M. Contant, J.-L. Bussière and G. Piffaretti of EADS Launch Vehicles.

Conclusions: To conclude, we sum up the assets and difficulties of multiobjective optimization, before we consider the future evolution of the field.

Reading advices

Many ways of reading this book are possible:

- A linear reading of the book.
  We advise you to follow this way if you want to learn about multiobjective optimization.
- Read specific parts of the book.
  If you are interested in some specific part of multiobjective optimization, such as interactive methods or multiobjective optimization methods using a metaheuristic, it is possible to follow a "shorter way" to read this book. Some chapters are, however, needed to understand other chapters. We have presented all the possible shorter ways to read this book in Fig. 0.3.

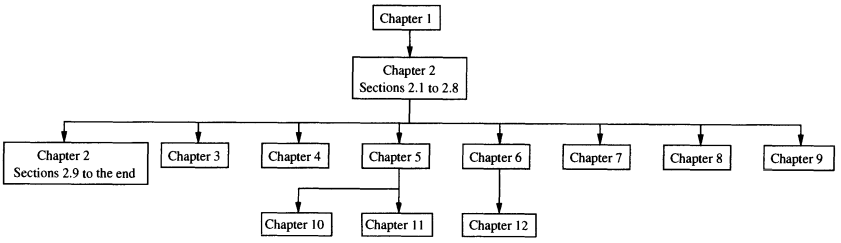


Fig. 0.3. Ways to read this book.

Acknowledgments

The authors of this book would like to thank Electricité de France Research and Development, which has supported Yann Collette in the preparation of his PhD thesis, and which has authorized the publication of this book. We would also like to thank the people who participated in the writing of this book through the three industrial applications that they wrote about to illustrate it: specifically, M. Dumas of CEA, for his contribution to Chap. 10; C. Decouchon-Brochet of France Télécom, for the complete writing of Chap. 11; and J.-M. Contant, J.-L. Bussière and G. Piffaretti of EADS Launch Vehicles, for their complete writing of Chap. 12.

Yann Collette would like to thank Isabelle, Marie and Inès Collette, who have constantly supported him and have accepted with good humor his long hours dedicated

to the book. He would like to thank also his colleagues from the SINETICS/I29 team at Electricité de France Research and Development Clamart, for their constant good humor and their always relevant comments.

Cachan
Paris

Yann Collette
Patrick Siarry

Part I

Principles of multiobjective optimization methods

1

Introduction: multiobjective optimization and domination

1.1 What is a multiobjective optimization problem ?

An optimization problem is defined as the search for a minimum or a maximum (the optimum) of a function. We can also find optimization problems for which the variables of the function to be optimized are constrained to evolve in a precisely defined area of the search space. In this case, we have a particular kind of optimization called constrained optimization problem.

The need for optimization comes from the necessity of an engineer to give the user a system that fulfills the user's needs. This system must be calibrated so that:

- it occupies the minimum volume needed for its good working (cost of raw materials),
- it uses the least possible energy (working cost),
- it fulfills the user's needs (terms and conditions).

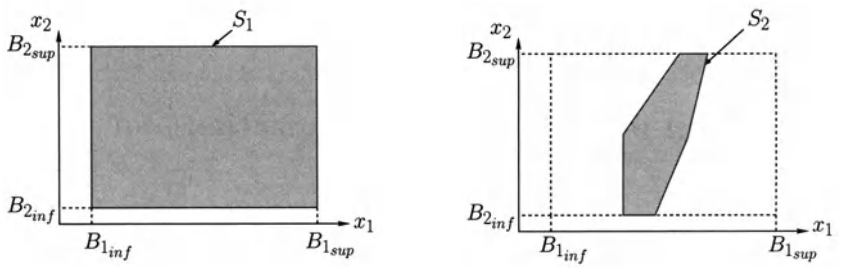
Mathematically speaking, an optimization problem has the following form:

$$\begin{array}{|ll} \text{minimize } f(\vec{x}) & \text{(function to be optimized)} \\ \text{with} \quad \vec{g}(\vec{x}) \leq 0 & (m \text{ inequality constraints}) \\ \text{and} \quad \vec{h}(\vec{x}) = 0 & (p \text{ equality constraints}) \end{array}$$

We alse have $\vec{x} \in \mathbb{R}^n$, $\vec{g}(\vec{x}) \in \mathbb{R}^m$ and $\vec{h}(\vec{x}) \in \mathbb{R}^p$. Here, the vectors $\vec{g}(\vec{x})$ and $\vec{h}(\vec{x})$ represent m inequality constraints and p equality constraints, respectively. This set of constraints delimits a restricted subspace to be searched for the optimal solution.

Usually, we can find two types of inequality constraints:

- Constraints of the type $B_{i_{inf}} \leq x_i \leq B_{i_{sup}}$: values of $\vec{x}$ which respect these constraints define the "search space ". This space is represented in Fig. 1.1a ($n = 2$).
- Constraints of the type $c(\vec{x}) \leq 0$ or $c(\vec{x}) \geq 0$: values of $\vec{x}$ which respect these constraints define the "feasible search space". This space is represented in Fig. 1.1b ($n = 2$).



(a) Search space.

(b) Feasible search space.

Fig. 1.1. The various search spaces.

1.2 Vocabulary and definitions

Definition 1. ***Objective function***
This is the name we give to the function f (we can also call it the ***cost function*** *or* ***optimization criterion****). It is this function that the optimization algorithm will try to "optimize" (finding an optimum).*

Except where it is stated explicitly to the contrary, we shall suppose in the rest of this book that we have to minimize the objective function.

Definition 2. ***Decision variables***
These are gathered together in the vector $\vec{x}$. It is by modifying this vector that we perform our search for an optimum of the function f.

Definition 3. ***Global minimum***
A "point" $\vec{x}^*$ is a global minimum of the function f if we have $f(\vec{x}^*) < f(\vec{x})$ for any $\vec{x}$ such that $\vec{x}^* \neq \vec{x}$. This definition corresponds to point M_3 in Fig. 1.2.

Definition 4. ***Strong local minimum***
A "point" $\vec{x}^*$ is a strong local minimum of the function f if we have $f(\vec{x}^*) < f(\vec{x})$ for any $\vec{x} \in V(\vec{x}^*)$ and $\vec{x}^* \neq \vec{x}$, where $V(\vec{x}^*)$ defines a "neighborhood" of $\vec{x}^*$. This definition corresponds to points M_2 and M_4 in Fig. 1.2.

Definition 5. ***Weak local minimum***
A "point" $\vec{x}^*$ is a weak local minimum of the function f if we have $f(\vec{x}^*) \leq f(\vec{x})$ for any $\vec{x} \in V(\vec{x}^*)$ and $\vec{x}^* \neq \vec{x}$, where $V(\vec{x}^*)$ defines a "neighborhood" of $\vec{x}^*$. This definition corresponds to point M_1 in Fig. 1.2.

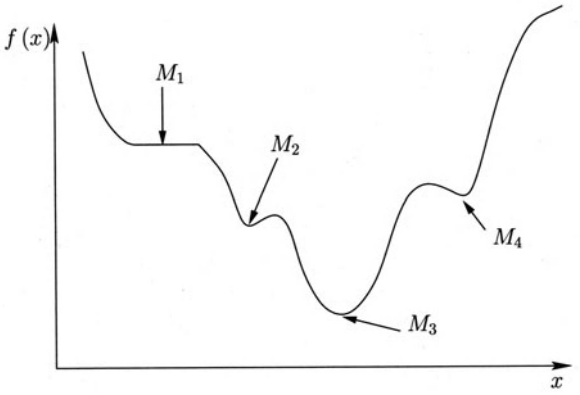


Fig. 1.2. Various minima.

1.3 Classification of optimization problems

We can classify the various optimization problems that are currently found using their characteristics:

1. Number of decision variables:
   - One ⇒ monovariable.
   - Several ⇒ multivariable.
2. Type of decision variable:
   - Continuous real number ⇒ continuous.
   - Integer number ⇒ integer or discrete.
   - Permutations on a set of numbers of finite size ⇒ combinatorial.
3. Type of the objective function:
   - Function which is linear with respect to the decision variables ⇒ linear.
   - Function which is quadratic with respect to the decision variables ⇒ quadratic.
   - Function which is nonlinear with respect to the decision variables ⇒ nonlinear.
4. Form of the problem:
   - With constraints ⇒ constrained.
   - Without constraints ⇒ unconstrained.

Let us look at an example of how to use this classification. If we consider the refueling pattern of a reactor, we can say this problem is:

- Nonlinear ⇒ we do not have any analytical representation of the problem. We must use simulation software to compute as precisely as possible the various values to be optimized.
- Constrained ⇒ fuel elements cannot be placed everywhere in the reactor.
- Multivariable ⇒ a nuclear reactor contains a number of fuel elements and each element has different characteristics.
- Combinatorial ⇒ to perform a modification to the state of the reactor we perform a permutation between some fuel elements.

1.4 Multiobjective optimization

The preceding form of problem was related to a problem in which we look for an optimum for just one objective function (f in the preceding problem).

Nevertheless, when we model a problem, we often try to satisfy multiple objectives. For example, we would like to have an efficient system and we would like this system to have a small power consumption. In this case, we call this problem a multiobjective optimization problem (or a multicriterion optimization problem). This problem has the following form:

$$\begin{array}{ll} \text{minimize} & \vec{f}(\vec{x}) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{array}$$

where $\vec{x} \in R^n$, $\vec{f}(\vec{x}) \in \mathbb{R}^k$, $\vec{g}(\vec{x}) \in \mathbb{R}^m$ and $\vec{h}(\vec{x}) \in \mathbb{R}^p$.

We shall call this problem the "*P problem*" in the rest of the book.

As we can see here, we no longer have one and only one objective to fulfill, but k objective functions (the vector $\vec{f}$ gathers together k objective functions).

The goal we want to reach during the solution of the multiobjective optimization problem is to minimize "in the best way possible" the various objective functions. As we shall see in the next section, in a multicriterion optimization problem, we often have contradictory objectives. Two objectives are contradictory when a decrease in one objective leads to an increase in the other objective.

1.5 Multiplicity of solutions

When we look for an optimal solution to a multiobjective optimization problem, we hope to find one and only one solution. In fact, this is an ideal case. Most of the time, we find a set of solutions, owing to the contradictory objectives.

If we consider the problem of the sizing of a beam which has to support a certain load, we would like to have the smallest section possible for the beam, and the smaller distortion possible, when the load is applied in the middle of the beam. In this example (represented in Fig. 1.3), we can see intuitively that solving the objective "beam with

the smallest section" does not correspond to a solution that solves the objective "beam with the smallest distortion" (see [Coello et al. 99] for the mathematical modeling of this problem).

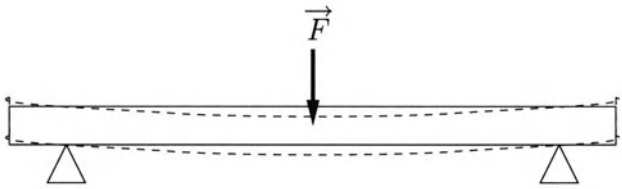


Fig. 1.3. Distortion of a beam under a load.

When we solve this multiobjective optimization problem, we shall find a huge number of solutions. These solutions, as we can expect, will not be optimal, in the sense that they will not minimize all the objectives of the problem.

One interesting idea, which allows us to define the solutions to be computed, is that of tradeoff. Indeed, the solutions that we find when we solve this problem are tradeoff solutions. They minimize some objectives while increasing some other objectives.

1.6 Domination

1.6.1 Introduction and definitions

When we have solved the multiobjective optimization problem, we will have found a multitude of solutions. Only a small subset of these solutions will be of interest. For a solution to be interesting, there must exists a domination relation between the solution considered and the other solutions, in the following sense:

Definition 6. ***domination relation***
We say that a vector $\vec{x_1}$ dominates a vector $\vec{x_2}$ if:
$\vec{x_1}$ is at least as good as $\vec{x_2}$ for all the objectives, and
$\vec{x_1}$ is strictly better than $\vec{x_2}$ for at least one objective.

Solutions which dominate the others but do not dominate themselves are called optimal solutions in the Pareto sense (or non dominated solutions). We define local optimality and global optimality in the Pareto sense as follows:

Definition 7. ***Local optimality in the Pareto sense***
A vector $\vec{x} \in \mathbb{R}^n$ is locally optimal in the Pareto sense if there exists a real $\delta > 0$ such that there is no vector $\vec{x}'$ which dominates the vector $\vec{x}$ with $\vec{x}' \in \mathbb{R}^n \cap B(\vec{x}, \delta)$, where $B(\vec{x}, \delta)$ represents a bowl of center $\vec{x}$ and of radius δ.

A vector $\vec{x}$ is locally optimal in the Pareto sense if it is optimal in the Pareto sense with a restriction on the set $\mathbb{R}^n$. This definition is illustrated in Fig. 1.4.

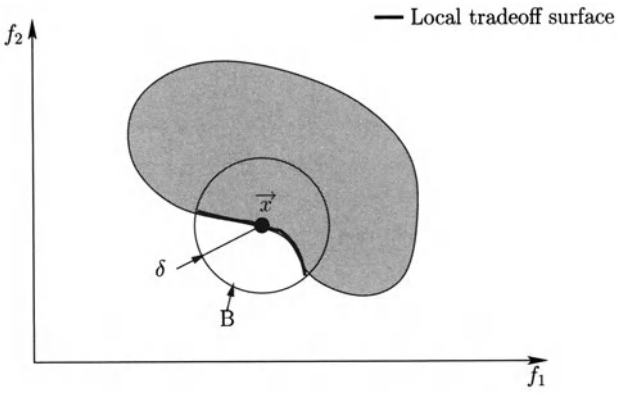


Fig. 1.4. Local optimality in the Pareto sense.

Definition 8. ***Global optimality in the Pareto sense***
A vector $\vec{x}$ is globally optimal in the Pareto sense (or optimal in the Pareto sense) if there does not exist any vector $\vec{x}'$ such that $\vec{x}'$ dominates the vector $\vec{x}$.

The main difference between this definition and the definition of local optimality lies in the fact that we do not have a restriction on the set $\mathbb{R}^n$ anymore.

A "graphical" version of the preceding definition uses the contact theorem.

Definition 9. ***Negative cone***
A negative cone is defined in $\mathbb{R}^k$ in the following way:

$$C^- = \left\{ \vec{x} \mid \vec{f}(\vec{x}) \in \mathbb{R}^k \text{ and } \vec{f}(\vec{x}) \leq 0 \right\}$$

Definition 10. ***Contact theorem***
A vector $\vec{x}$ is optimal in the Pareto sense for a multiobjective optimization problem if

$$(C^- + \vec{x}) \cap F = \{\vec{x}\}$$

where F corresponds to the feasible subspace.

The way to use this theorem is illustrated in Fig. 1.5.

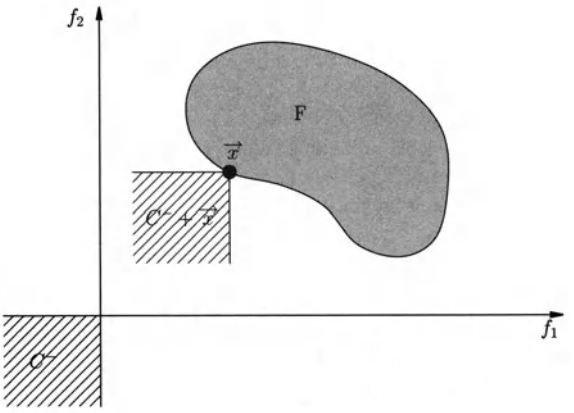


Fig. 1.5. Contact theorem.

When we apply the definition of domination, we can define four areas. We can also associate a preference level with each area. These areas are represented in Fig. 1.6. This figure uses a splitting defined by use of the negative cone introduced above and spreads it out over the whole space.

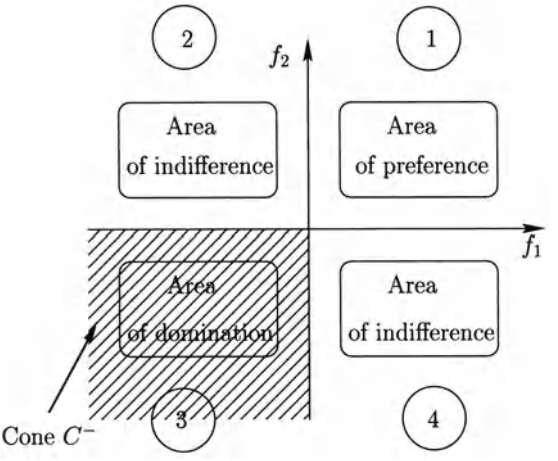


Fig. 1.6. Preference level and domination relation.

For example, if this figure is centered on solution A and we compare this solution with solution B, we have the following possibilities:

- if solution B belongs to area 1, then solution A is preferred to solution B;
- if solution B belongs to area 3, then solution A is dominated by solution B;
- if solution B belongs to area 2 or 4, then we cannot say if we prefer solution A in comparison with solution B or if we prefer solution B in comparison with solution A.

1.6.2 Example

For a better understanding of what is a domination relation, we shall work with an example (see [Deb 99]).

We consider a problem with two objectives: to maximize f_1 and to minimize f_2. For this problem, we find a set of solutions. This set of solutions is represented in table 1.1. We represent these points in a plane with f_1, f_2 as the axes (see in Fig. 1.7). We present comparisons between various solutions in the table 1.2. Let us be precise about the notation:

Table 1.1. Set of solutions of a problem with two objectives.

Point	Objective f_1	Objective f_2
A	8	5
B	9	2
C	12	1
D	11	2
E	16	2

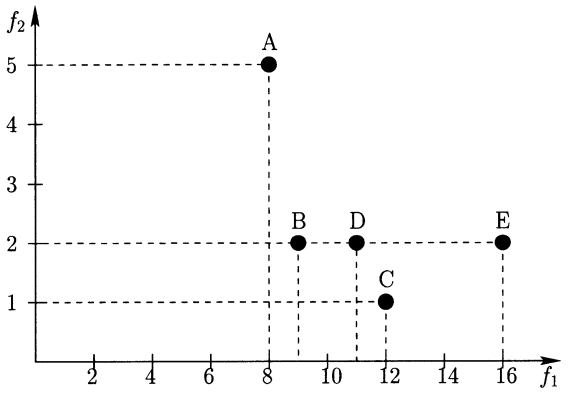


Fig. 1.7. Representation of the solutions in the plane f_1, f_2.

The comparison between two solutions, say P and Q, is represented by a pair of symbols at the intersection of row P and column Q. This pair is made up of two symbols, associated with the two objectives f_1 and f_2; each of these symbols can take three values, $+$, $-$ or $=$, according whether P is better than, worse than or equal to Q, considering the objective with which it is associated.

We recall that we want to maximize f_1 and minimize f_2. Therefore, a point P is better than a point Q considering objective f_1 if $f_1(P)$ has a higher value than $f_1(Q)$; a point P is better than a point Q considering objective f_2 if $f_2(P)$ has a smaller value than $f_2(Q)$.

The following is an example of how to proceed with points A and B of Table 1.1. Let us perform the comparisons:

- A is worse than B considering objective f_1. Therefore, the first element of the pair in the cell $[A, B]$ is the sign $-$.

- A is worse than B considering objective f_2. Therefore, the second element of the pair in the cell $[A, B]$ is the sign $-$.

If we refer to the preceding definition, we can conclude that solution B dominates solution A.

Table 1.2. Classification of solutions.

	A	B	C	D	E
A		(−,−)	(−,−)	(−,−)	(−,−)
B	(+,+)		(−,−)	(−,=)	(−,=)
C	(+,+)	(+,+)		(+,+)	(−,+)
D	(+,+)	(+,=)	(−,−)		(−,=)
E	(+,+)	(+,=)	(+,−)	(+,=)	

Table 1.3. Classification of solutions of rank 2.

	A	B	D
A		(−,−)	(−,−)
B	(+,+)		(−,=)
D	(+,+)	(+,=)	

Note: Obviously, pairs in the cells of table 1.2 which are symmetric about the principal diagonal are "complementary".

We shall now try to extract nondominated solutions.

1. Consider point A.
   - This point is dominated by the following points:
     B (pair (−,−) at the intersection of row A and column B),
     C (pair (−,−) at the intersection of row A and column C),
     D (pair (−,−) at the intersection of row A and column D) and
     E (pair (−,−) at the intersection of row A and column E).
2. Consider point B.
   - This point is dominated by the following points:
     C (pair (−,−) at the intersection of row B and column C),
     D (pair (−,=) at the intersection of row B and column D) and
     E (pair (−,=) at the intersection of row B and column E).
   - It dominates point A (pair (+,+) at the intersection of row B and column A).

   We can say that point A does not belong to the set of nondominated solutions of rank 1, because it is possible to find a point (B in this case) which is better than the point A for all objectives.
3. Consider point C.
   - This point is not dominated by point E (pair (+,−) at the intersection of row E and column C);
     and it does not dominate point E either (pair (−,+) at the intersection of row C and column E);
     points C and E are nondominated.

- It dominates the following points:
  A (pair (+,+) at the intersection of row C and column A),
  B (pair (+,+) at the intersection of row C and column B) and
  D (pair (+,+) at the intersection of row C and column D).

We can say that points A, B and D do not belong to the set of nondominated solutions of rank 1, because it is possible to find a point (C in this case) which is better than these two points for all objectives.

4. The point D is dominated, so it is not necessary to compare it with other points.
5. Consider point E.
   - This point is not dominated by point C (pair (−,+) at the intersection of row C and column E),
     and it does not dominate point C either (pair (+,−) at the intersection of row E and column C).
   - It dominates the following points:
     A (pair (+,+) at the intersection of row E and column A),
     B (pair (+,=) at the intersection of row E and column B) and
     D (pair (+,=) at the intersection of row E and column D).

   We can say that points E and C are nondominated. These points dominate points A, B and D but do not dominate themselves.

Now, we take these two points (E and C) and remove them from the table: they belong to the set of non dominated points. We can sort these solutions using the rank of domination. In our example, we attribute rank 1 (because we have finished the first comparison) to points E and C, because they dominate all the other points but do not dominate themselves. So, these points are Pareto optimal solutions of rank 1.

We now go back to the start and apply this rule again to the remaining elements of the table. The remaining solutions after points E and C have been removed are represented in Table 1.3. This process stops when the set of points to be compared is empty. Fig. 1.8 represents the various points and their ranks.

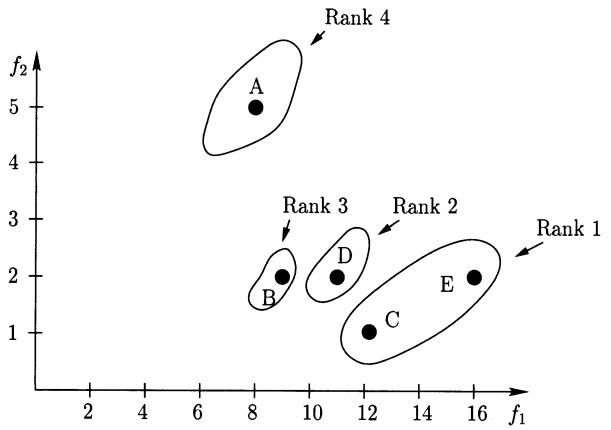


Fig. 1.8. Solutions and their Pareto ranks.

The pseudo-code of the Pareto rank assignment function is represented in Algorithm 1. In this algorithm, the variable N corresponds to the number of points of the set on which we perform the comparisons.

Algorithm 1 Assignment of the Pareto rank.

```
CurrentRank = 1
m = N
while N ≠ 0 do
  For i = 1 to m do
    If X_i is Nondominated
      Rank(X_i, t) = CurrentRank
  End For
  For i = 1 to m do
    If Rank(X_i, t) = CurrentRank
      Store X_i in a temporary set
      N = N − 1
  End For
  CurrentRank = CurrentRank + 1
  m = N
End While
```

1.6.3 Sizing of a beam

1.6.3.1 Full beam with a square section

Now, we shall consider in detail our introductory example: the sizing of a beam of a given length (1 m), represented in Fig. 1.9.

We are searching for the value of the size a of the square section which allows us to minimize both the weight of the beam and the distortion of the beam when its center of gravity is subjected to a 1000 N load.

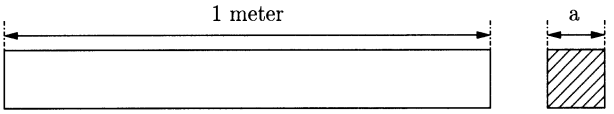


Fig. 1.9. A full beam with square section.

As minimizing the volume of a beam is equivalent to minimizing the perimeter of its section, we shall express the section and the maximum distortion of the beam with respect to the length a as follows:

$$\begin{cases} S(a) = a^2 \\ d(a) = 1000 + \frac{1 \cdot 10^{-3}}{192 \cdot 2 \cdot 10^5 \cdot \frac{a^4}{12}} \\ a \leq 0.1 \end{cases}$$

(the characteristics of the beam are given in [Spenle et al. 96])

The one and only one constraint we impose is that the length a does not exceed 10 cm.

To show the shape of the set of feasible solutions, we choose at random some values for a, and then we compute the values of the two objectives and plot these values in the (S, d) plane. This shape in represented in Fig. 1.10.

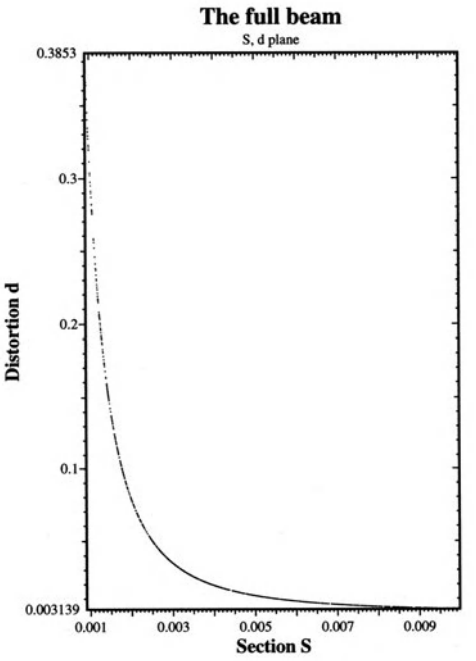


Fig. 1.10. Set of values of the objective functions for a square-section beam.

The first thing we may notice in this figure is the fact that the objective functions are contradictory. Minimization of the value of one objective function involves the maximization of the value of the other objective function. However, in this figure, we can see that all solutions are nondominated. This arises from the fact that this optimization problem, as we have modeled it, is equivalent to a simple parametric function.

The second thing we may notice is a reason for using multiobjective optimization rather than mono-objective optimization. Indeed, if we had performed a mono-objective optimization on this sizing problem, we would have tried to minimize the volume of the beam or to minimize the distortion of this beam. In both cases, we would find a "stupid" solution:

- A null value for the beam section (thus, the volume is overminimized);
- A huge value for the beam section in comparison with the length of the beam, in the case of optimization of the distortion of the beam. Indeed, the larger the section, the less the distortion of the beam.

This sizing problem shows that, in some cases, an optimization problem cannot be modeled as a single-objective optimization problem.

1.6.3.2 Hollow beam with a square section

To show some important aspects of multiobjective optimization, we now slightly modify our problem by adding a new parameter to it. We now consider a hollow beam with a square section. The interior dimension is specified by the variable b. The various representative variables are illustrated in Fig. 1.11.

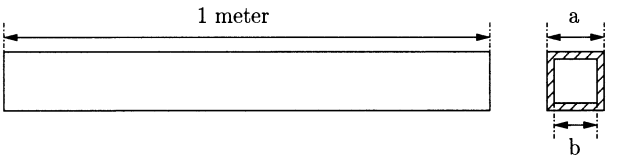


Fig. 1.11. A hollow beam with a square section.

$$\begin{cases} S(a,b) = a^2 - b^2 \\ d(a,b) = 1000 + \frac{1 \cdot 10^{-3}}{192 \cdot 2 \cdot 10^5 \cdot \frac{a^4 - b^4}{12}} \\ a \leq 0.1 \\ b + 0.04 \leq a \end{cases}$$

To show the shape of the set of feasible solutions, we choose at random some values for the pair of parameters (a,b), with $a,b \in [0,0.1]$, and then we compute the values of the two objective functions and plot them in the (S,d) plane. The result is represented in Fig. 1.12.

For this example, we notice that the situation is now more complicated. All the solutions are not gathered together on a simple parametric curve anymore; they are gathered together on a "true" surface. A difference from the preceding example is that now, all the solutions do not have the same efficiency. One set of solutions dominates the others. To extract these solutions, we apply the domination relation and remove all the dominated solutions from the set. We obtain the set represented in Fig. 1.13.

We notice that, after having removed the nondominated solutions, we find a boundary of the starting set. This is the tradeoff surface, and it is on the tradeoff surface that we find the "best" solutions, the ones for which we cannot improve any objective functions.

1.7 Illustration of the reason for using multiobjective optimization

We can illustrate the gain obtained from using multiobjective optimization relative to mono-objective optimization by visualizing the relative improvement of one objective function against another on a simple test problem.

The test problem is the following:

$$\begin{array}{ll} \text{minimize } f_1(\theta,x) = 1 - \cos(\theta) + x \\ \text{minimize } f_2(\theta,x) = 1 - \sin(\theta) + x \\ \text{with} \quad 0 \leq \theta \leq \frac{\pi}{2} \\ \text{and} \quad 0 \leq x \leq 1 \end{array}$$

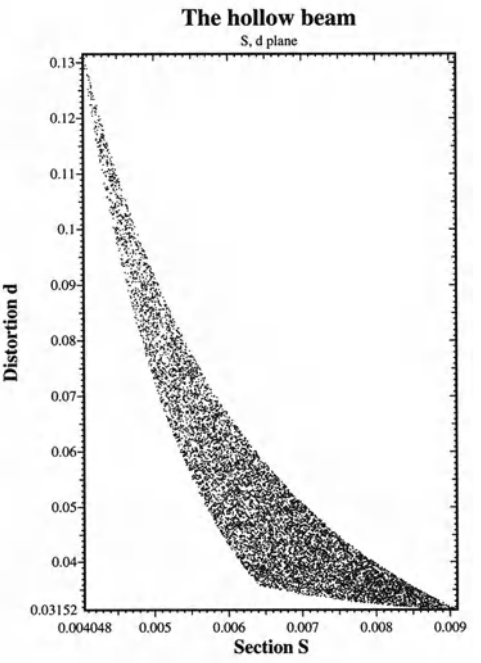


Fig. 1.12. Set of feasible solutions for a hollow beam with square section.

We can find the tradeoff surface by replacing the variable x by 0. So, we obtain an analytical expression for the tradeoff surface:

$$\begin{vmatrix} f_1(\theta) = 1 - \cos(\theta) \\ f_2(\theta) = 1 - \sin(\theta) \\ 0 \leq \theta \leq \frac{\pi}{2} \end{vmatrix}$$

The set of values of the objective function and the tradeoff surface are represented in Fig. 1.14.

For this tradeoff surface, we now plot the relative improvement for each objective function with respect to the position of the solution on the tradeoff surface. To do so, we divide the tradeoff surface into ten pieces (the angle varies from 0 to 90° in steps of 9°). The values of the objective functions at the corresponding points are listed in Table 1.4. The relative gap is measured against the total excursion of the objective function (the value of the excursion is 1 for the two objective functions).

As we can see in Fig. 1.15a, depending on the position of the solution on the tradeoff surface, we can hope to have a good decrease of the level of one objective function without having a huge increase for the other objective function. This is the case, for example, for the objective function f_1 for values of θ close to 0. For this position, if we increase the value of θ by 0.2 rad (and thus θ varies from 0 rad to 0.2

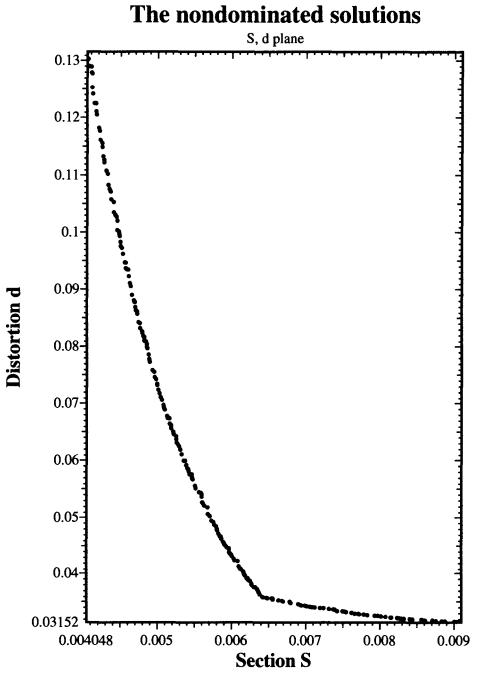
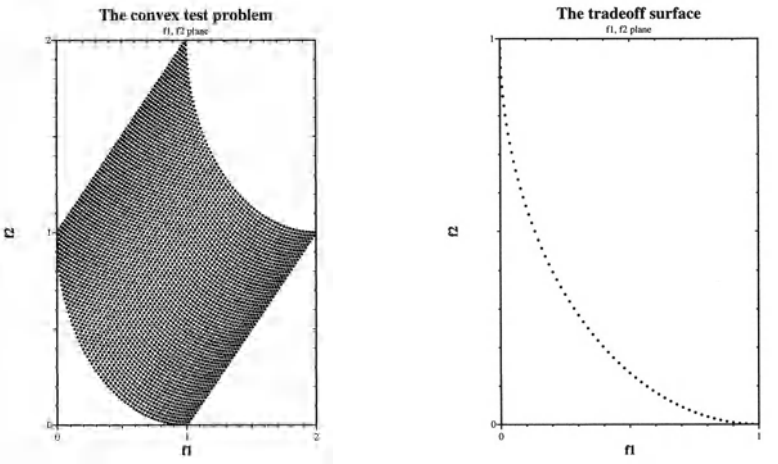


Fig. 1.13. Nondominated solutions for the problem of sizing a hollow beam with a square section.

Table 1.4. Values of the objective functions.

θ	$f_1(\theta)$	$f_2(\theta)$
0	0	1
9	0.0123	0.843
18	0.0489	0.691
27	0.109	0.546
36	0.191	0.412
45	0.293	0.293
54	0.412	0.191
63	0.546	0.109
72	0.691	0.0489
81	0.843	0.0123
90	1	0



(a) The set of values of the objective function.

(b) The tradeoff surface.

Fig. 1.14. A trigonometric test problem.

rad), the value of the objective function f_1 varies from 1 to 0.843 while the value of the objective function f_2 varies from 0 to 0.0123.

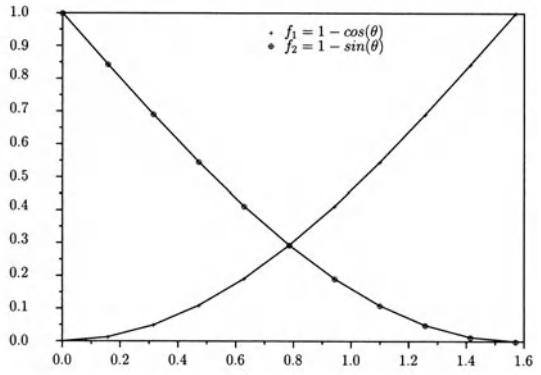
The gain we can hope to obtain can be represented by the derivative of the objective function with respect to θ at that point. A positive derivative means an improvement for the objective function, while a negative derivative means a deterioration for the objective function.

This typical behavior of multiobjective optimization gives a fairly good illustration of the goal we are trying to achieve in this domain: to minimize a group of objective functions in such a way that the optimum value of one objective function does not deteriorate too much when another objective function is minimized.

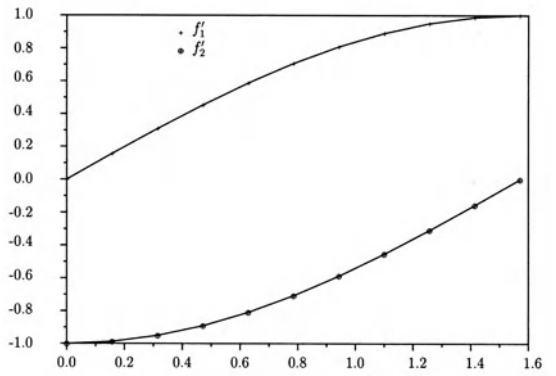
1.8 Relations derived from domination

1.8.1 Introduction

The domination relation does not offer too many degrees of freedom in its definition. For example, it is not possible to include in the definition of the domination relation a preference for one objective function against another one. To overcome this lack of flexibility, relations derived from the domination relation have been developed. Any solutions we can find with these derived relations are always optimal in the Pareto sense. The major change that we find with these relations is that the set of solutions that we obtain with the derived relations is a subset of the set of solutions obtained using the domination relation.



(a) Absolute evolution.



(b) Relative evolution.

Fig. 1.15. Evolution of the values of the objective functions on the tradeoff surface.

In this section, the set S^k represents the set of feasible solutions of an optimization problem with k objective functions.

1.8.2 Lexicographic optimality

This definition of optimality allows one to include preferences with respect to objective functions [Ehrgott 97].

Definition 11. ***Optimality in the lexicographic sense***
A solution $\overrightarrow{x^*} \in S^k$ *is optimal in the lexicographic sense if* $\overrightarrow{x^*} \leq_{lex} \overrightarrow{x}, \forall \overrightarrow{x} \in S^k - \left\{\overrightarrow{x^*}\right\}$.

If $\overrightarrow{x}, \overrightarrow{y} \in S^k$, we say that $\overrightarrow{x} \leq_{lex} \overrightarrow{y}$ if there exists an index value q^* such that $x_q = y_q$ for $q = 1, \cdots, (q^* - 1)$ and $x_{q^*} < y_{q^*}$. The relations between x_q and y_q for $q \geq q^*$ are not taken into account because we stop at index q^* (the first index for which $x_q < y_q$).

The definition involves the requirement that the decision maker has sorted the objective functions by increasing preference. The comparison between two solutions is done using this sorting of objective functions.

Let us illustrate this relation using an example.

Take two points A and B:

$$A = (1, 2, 3, 4, 5, 6)$$
$$B = (1, 2, 3, 9, 4, 9)$$

For these two points, we have $A \leq_{lex} B$ because, until the third position, we have $A_i = B_i$, $i = 1, 2, 3$ and, for the fourth position, we have $4 < 9$. We can conclude that solution A dominates solution B lexicographically.

1.8.3 Extremal optimality

As with the lexicographic optimality relation, this relation allows one to add preferences between objective functions. This preference is modeled using weights. The more important an objective function, the heavier the weight [Ehrgott 97].

Definition 12. ***Extremal optimality***
A solution $\vec{x^*} \in S^k$ is extreme-optimal if, given a weight vector $\vec{\lambda} \in \mathbb{R}^k$ such that $\sum_i \lambda_i = 1$, $\vec{x^*}$ is an optimal solution of the monocriterion minimization problem with the following objective function:

$$\sum_{i=1}^{k} \lambda_i \cdot \vec{x_i} \tag{1.1}$$

so,

$$\sum_{i=1}^{k} \lambda_i \cdot \vec{x_i^*} \leq \sum_{i=1}^{k} \lambda_i \cdot \vec{x_i}, \forall \vec{x} \in S^k - \left\{\vec{x^*}\right\} \tag{1.2}$$

Let us illustrate the notion of extremal optimality using the preceding example. Here, we consider objective 6 as a reference objective. We also assume that objectives 1, 3 and 5 are 20% more important than the reference objective and that objectives 2 and 4 are equivalent to the reference objective. We now compute the weight of each objective:

$$\begin{aligned} &w_1 = w_3 = w_5 = 1.2 \cdot w_6 \\ &w_2 = w_4 = w_6 \\ &\textstyle\sum_{i=1}^{6} w_i = 1 \end{aligned}$$

Solving these equations gives us

$$\begin{aligned} w_1 = w_3 = w_5 &= \frac{0.2}{1.1} = 0.18 \\ w_2 = w_4 = w_6 &= \frac{1}{6.6} = 0.15 \end{aligned}$$

$\sum_{i=1}^{6} w_i \cdot A_i = 3.45$ and $\sum_{i=1}^{6} w_i \cdot B_i = 5.39$ so, the point A extreme-dominates point B.

1.8.4 Maximal optimality

This relation, unlike the preceding relations, does not allow one to add preferences between objective functions [Ehrgott 97].

Definition 13. ***Maximal optimality***
A solution $\vec{x^*} \in S^k$ is max-optimal if the value of the worst objective is as small as possible:

$$\max_{q \in \{1,\cdots,k\}} x_q^* \leq \max_{\substack{q \in \{1,\cdots,k\} \\ \vec{x} \in S^k - \left\{\vec{x^*}\right\}}} x_q \tag{1.3}$$

Let us illustrate this relation by using the preceding example. We can say that solution A max-dominates solution B because

$$\max A = 6 < \max B = 9$$

1.8.5 Cone optimality

Another sorting relation exists. This is the cone-domination relation. This relation has the advantage, relative to the domination relation, of being "tunable".

Definition 14. ***A cone***
A cone of slope λ is defined in the following way:
If $0 < \lambda < 1$ (see Fig. 1.16a):

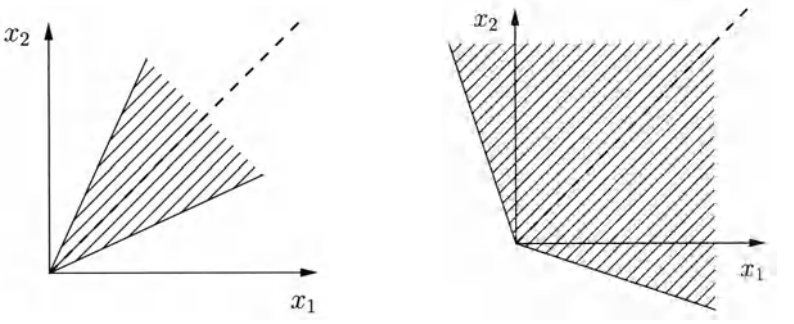
$$C_\lambda(x_1, x_2) = \left\{ \vec{x} \in \mathbb{R}^2 \,|\, x_1 \geq 0,\, x_2 \geq 0,\, \lambda \cdot x_1 \leq x_2 \leq \frac{1}{\lambda} \cdot x_1 \right\}$$

If $\lambda < 0$ (see Fig. 1.16b):

$$C_\lambda(x_1, x_2) = \left\{ \vec{x} \in \mathbb{R}^2 \,|\, \lambda \cdot x_1 \leq x_2 \text{ and } \lambda \cdot x_2 \leq x_1 \right\} \tag{1.4}$$

If $\lambda = 0$:

$$C_\lambda(x_1, x_2) = \left\{ \vec{x} \in \mathbb{R}^2 \,|\, x_1 \geq 0 \text{ and } x_2 \geq 0 \right\}$$



(a) λ positive. (b) λ negative.

Fig. 1.16. Shape of a cone for positive and negative values of λ.

This relation can be extended to the multidimensional case. We will find in [Deb 01] a different relation which allows one to tune the shape of the domination cone.

Definition 15. ***Multidimensional cone***
A cone of slope λ is defined in the following way:

$$C_\lambda = \begin{array}{l}\{\overrightarrow{x} \in \mathbb{R}^n \,|\, \forall i \in \{1, \cdots, n\}, \forall j \in \{1, \cdots, n\}, j > i, \\ x_i, x_j \in C_\lambda(x_i, x_j)\}\end{array} \tag{1.5}$$

The projection of a multidimensional cone on various planes is represented in Fig. 1.17.

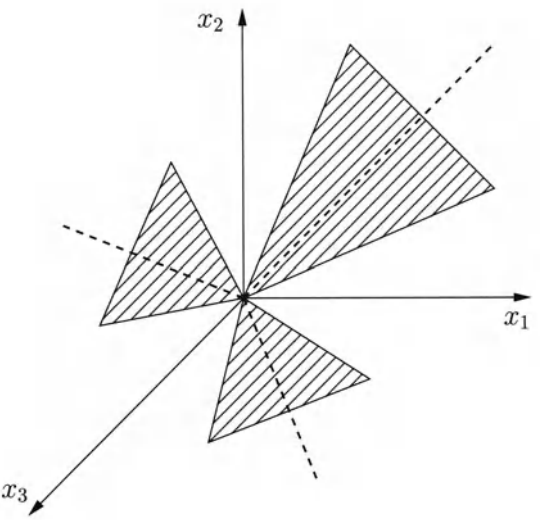


Fig. 1.17. Projection of the multidimensional cone.

Definition 16. ***Cone-domination***
Let a cone C_λ be given. A vector $\overrightarrow{x} \in \mathbb{R}^n$ cone-dominates a vector $\overrightarrow{y} \in \mathbb{R}^n$ (this relation is written $\overrightarrow{x} \leq_{C_\lambda} \overrightarrow{y}$) if $\overrightarrow{y} - \overrightarrow{x} \in C_\lambda$ and $\overrightarrow{x} \neq \overrightarrow{y}$ (or, again, if $\overrightarrow{y} - \overrightarrow{x} \in C_\lambda - \{0\}$).

The cone-domination relation is represented in Fig. 1.18. We have also represented two examples of the application of this relation in Fig. 1.19. In both figures, we have represented the "domination cone". This corresponds to the area inside which the origin point (the vector $\overrightarrow{x^*}$ in the definition of cone-domination) dominates all the points which belong to this area (the vectors $\overrightarrow{x}$ in the definition of cone-domination).

The cone-domination relation is equivalent to the domination relation described earlier when $\lambda = 0$.

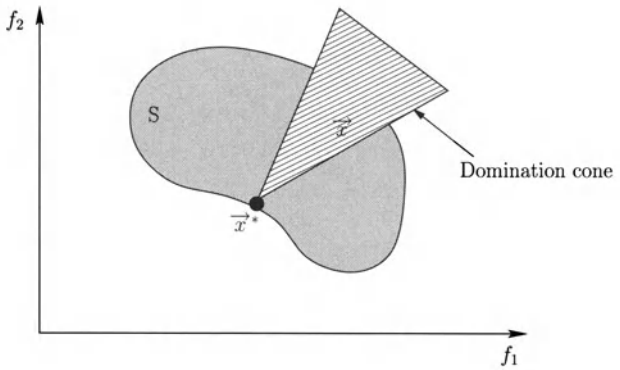
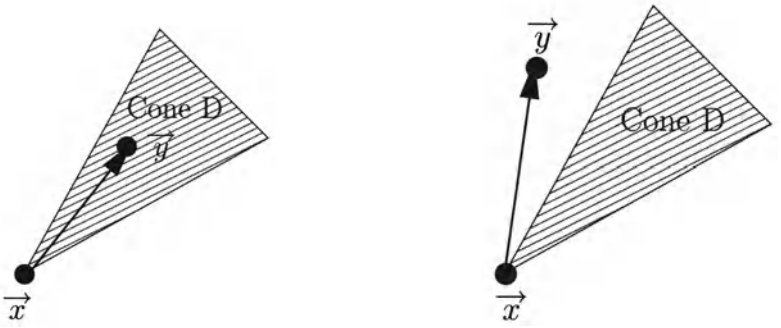


Fig. 1.18. The cone-domination relation.



(a) Vector $\vec{x}$ dominates vector $\vec{y}$ with respect to the cone D.

(b) Vector $\vec{x}$ does not dominate vector $\vec{y}$ with respect to the cone D.

Fig. 1.19. Two examples of the application of the cone-domination relation.

1.8.6 a-domination

The definition of this type of domination was introduced in [Othmani 98].

Definition 17. ***a-domination***
A solution $\vec{x^*} \in S^k$ a-dominates a solution $\vec{x} \in S^k$ if there exists a set of combinations of $k+1-a$ criteria (we denote by $I_{(k+1-a)}$ the set of indices corresponding to the set of combinations of these criteria) such that:

1. $\vec{x_j^*} \leq \vec{x_j}$ *for all* $j \in I_{(k+1-a)}$, *and*
2. $\vec{x_j^*} < \vec{x_j}$ *for at least one* $j \in I_{(k+1-a)}$.

To illustrate this definition, let us take an example. Let us consider a family of three criteria (C_1, C_2 and C_3) and two solutions $\overrightarrow{x_A}$ and $\overrightarrow{x_B}$, and establish the 2-domination relation.

To establish the 2-domination relation, we must test the domination relationship between the vectors $\overrightarrow{x_A}$ and $\overrightarrow{x_B}$ on all combinations of $3 + 1 - 2 = 2$ criteria. These families are the following:

- $F_1 = \{C_1, C_2\} \Rightarrow I_1' = \{1, 2\}$
- $F_2 = \{C_1, C_3\} \Rightarrow I_2' = \{1, 3\}$
- $F_3 = \{C_2, C_3\} \Rightarrow I_3' = \{2, 3\}$

So, $I_2 = \left\{I_1', I_2', I_3'\right\} = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$.

Let us now consider two points A and B, for which the values of the criteria are listed in Table 1.5.

Table 1.5. Values of the criteria for points A and B.

	C_1	C_2	C_3
A	1	2	3
B	1	1	2

The point A 2-dominates the point B because it dominates the point B on each family of criteria:

- A dominates B if we consider criteria 1 and 2,
- A dominates B if we consider criteria 1 and 3,
- A dominates B if we consider criteria 2 and 3.

So, A 2-dominates B because A dominates B if we consider the combinations of criteria $\{C_1, C_2\}$, $\{C_1, C_3\}$ and $\{C_2, C_3\}$.

1.8.7 Domination in the Geoffrion sense

A last important form of domination in the realm of multiobjective optimization is domination in the Geoffrion sense (see [Ehrgott 00] and [Miettinen 99]). Optimal solutions obtained using this kind of domination are called proper Pareto optimal solutions.

Definition 18. ***Domination in the Geoffrion sense***
A solution $x^* \in S$ is called a proper Pareto optimal solution if:

- *it is Pareto optimal,*
- *there exists a number $M > 0$ such that $\forall i$ and $\forall x \in S$ satisfying $f_i(x) < f_i(x^*)$, there exists an index j such that $f_j(x^*) < f_j(x)$ and:*

$$\frac{f_i(x^*) - f_i(x)}{f_j(x) - f_j(x^*)} \leq M$$

This relation is almost never used as is. In general, we use a result which is a consequence of this definition. Indeed, the interpretation given in [Ehrgott 00] of this theorem is the following: "Proper Pareto optimal solutions have bounded tradeoff considering their objectives."

A theorem related to the weighted sum of objective functions using this result is the following:

Theorem 1.
Let the following weighted sum of objective functions be given:

$$f_{eq}(\vec{x}) = \sum_{i=1}^{N} \omega_i \cdot f_i(\vec{x})$$

Suppose that $\forall i = 1, \cdots, N$, $\omega_i > 0$ *with* $\sum_{i=1}^{N} \omega_i = 1$.
If x^* *is an optimal solution obtained using the preceding aggregation method, then this solution is also a proper Pareto optimal solution.*

A weighted sum of objective functions, with weights respecting the above relations, allows one to obtain solutions with bounded tradeoffs.

1.8.8 Conclusion

These various domination relations allow one some degrees of freedom to choose a relation which reproduces best the behavior of an engineer or a decision maker. For example, the lexicographic relation allows one to reproduce the behavior of a decision maker who chooses a solution from N solutions by considering criteria sorted by order of importance. When comparing two solutions, this relation allows one to compare the values of objectives until one objective allows the user to determine a preference between the two solutions considered. Once the preference between the two solutions is determined, we do not consider the remaining objectives. The comparison stops here.

Domination is a tool, one amongst many others, that allows us to reproduce the process of searching for an optimal solution. It is not the only one, of course, but it is an important tool for solving an optimization problem.

1.9 Tradeoff surface

The small number of solutions of rank 1 that we have selected using the sorting rule based on the definition of domination produces what we call the tradefoff surface (or Pareto front).

Let us imagine that we have a problem with two objective functions (minimize f_1 and minimize f_2 under the constraints $\vec{g}(\vec{x}) \leq 0$ and $\vec{h}(\vec{x}) = 0$).

- We denote by S the set of values of the pair $(f_1(\vec{x}), f_2(\vec{x}))$ when $\vec{x}$ respects the constraints $\vec{g}(\vec{x})$ and $\vec{h}(\vec{x})$.
- We denote by P the tradeoff surface.

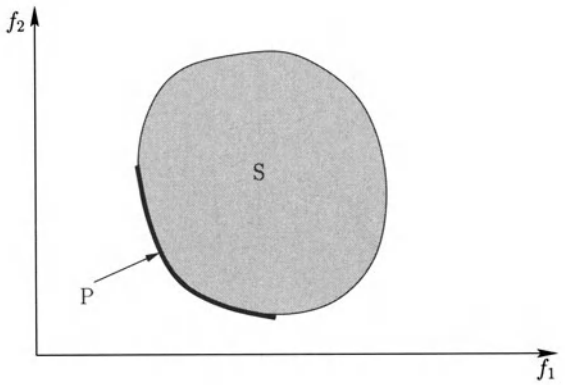


Fig. 1.20. Representation of the tradeoff surface.

We represent S and P in Fig. 1.20.

A noticeable property is that we obtain certain shapes for the tradeoff surface, depending on the type of problem we are dealing with. These common shapes of tradeoff surface are shown in Fig. 1.21. These shapes of tradeoff surfaces are typical of a multiobjective problem with a convex solution set (for a definition of convexity, see Sect. 1.10). This is the kind of solution set we often face.

We notice two characteristic points associated with a tradeoff surface:

Definition 19. ***Ideal point***
The coordinates of this point are obtained by minimizing each objective function separately.

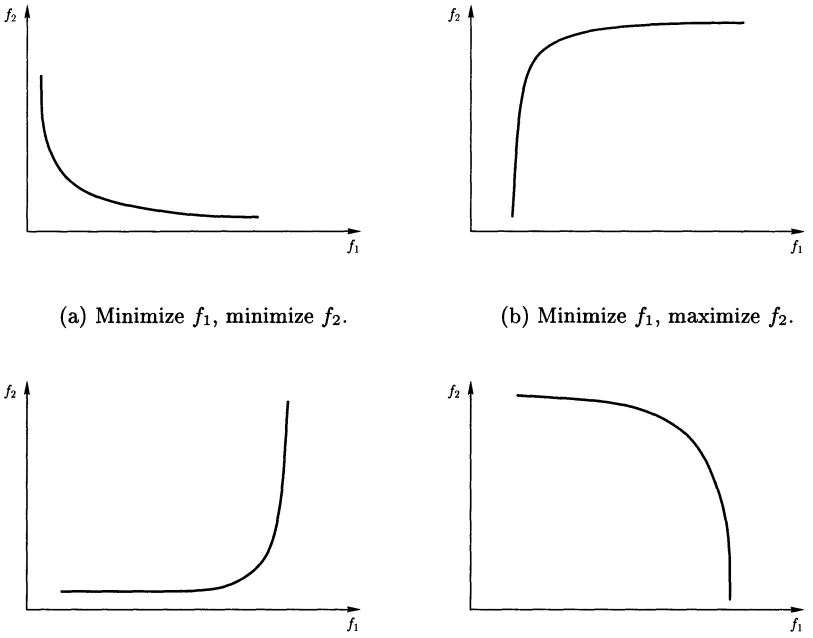
Definition 20. ***Nadir point***
The coordinates of this point correspond to the worst values obtained for each objective function when the solution set is restricted to the tradeoff surface.

The ideal point is used in a lot of optimization methods as a reference point. The nadir point is used to restrict the search space; it is often used in interactive optimization methods.

These two definitions are illustrated in Fig. 1.22.

1.10 Convexity

As we shall see below, some multiobjective optimization methods require some hypothesis to be respected. Often, a method needs to work on a set S of values of $\overrightarrow{f}$ which is convex.



(a) Minimize f_1, minimize f_2.

(b) Minimize f_1, maximize f_2.

(c) Maximize f_1, minimize f_2.

(d) Maximize f_1, maximize f_2.

Fig. 1.21. Common shapes of tradeoff surface considering two objective functions.

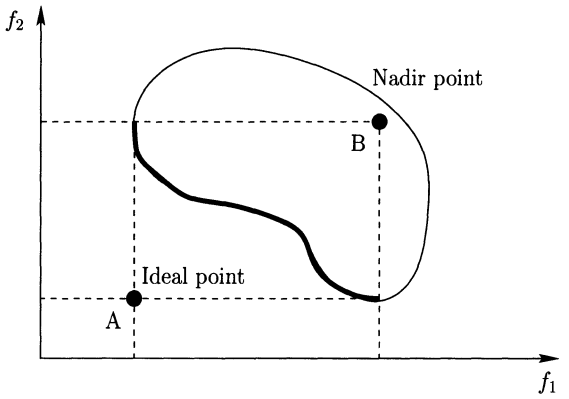
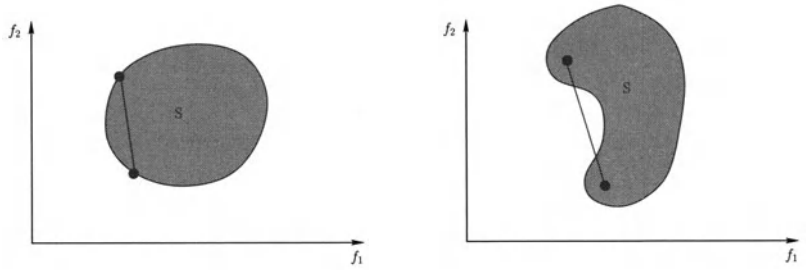


Fig. 1.22. Representation of the ideal point and the nadir point.

Definition 21. *Convexity*
A set S is convex if, given two distinct points in this set, the segment which links these two points lies in the set S.

An example of a convex set and an example of a nonconvex set are represented in Fig. 1.23.



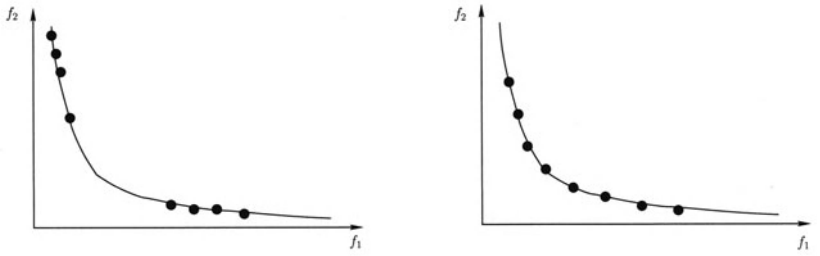
(a) A convex set.

(b) A nonconvex set.

Fig. 1.23. Examples of a convex and a nonconvex set.

1.11 Representation of a tradeoff surface

All representations of a tradeoff surface, for a given problem, are not equivalent. An ideal representation of the tradeoff surface must be made up of regularly spaced solutions on the tradeoff surface (see Fig. 1.24).



(a) A bad representation of the tradeoff surface.

(b) A good representation of the tradeoff surface.

Fig. 1.24. Representation of the tradeoff surface.

In the first case illustrated in Fig. 1.24, the points representing the tradeoff surface are not regularly spaced. The decision maker will not have a very useful set of solutions in his/her hands. Indeed, if the decision maker decides finally that a solution that he/she has chosen is not efficient, the choice of another solution will produce huge variations in all the objectives, and this solution will not convince the decision maker either. It is likely that the solution which offers the "best" tradeoff will belong to an area which is not well approximated by the solutions.

The determination of a good representation of a tradeoff surface is a criterion used in the choice of a multiobjective optimization method.

1.12 Solution methods of multiobjective optimization problems

There exist various methods of optimization, and we have classified them into five sets:

- scalar methods,
- interactive methods,
- fuzzy methods,
- methods which use a metaheuristic,
- decision aid methods.

The methods in these five sets can be sorted into three families of multiobjective optimization methods [Van Veldhuizen 99]:

- *a priori* preference methods:
  With these methods, the decision maker defines the tradeoff to be applied (the decision maker shows his/her preferences) before running the optimization method. In this family, we put most of the aggregative methods (where the objective functions are gathered into one objective function).
- *progressive* preference methods:
  With these methods, the decision maker improves the tradeoff to be applied during the running of the optimization method. In this family, we find interactive methods.
- *a posteriori* preference methods:
  With these methods, the decision maker chooses the solution by examining solutions computed by the optimization method. Methods which belong to this family produce, at the end of the optimization, a tradeoff surface.

There exist some multiobjective optimization methods which do not fit exclusively into one family. For example, we can use an a priori preference method with preferences computed at random. The result will be a huge number of solutions, which can be presented to the decision maker to choose the tradeoff solution. This combination will produce an a posteriori preference method.

So, this classification just gives us an idea of the approach we have to follow to produce a result. A "finest" classification is presented in Chap. 9.

1.13 Annotated bibliography

[Ehrgott 97] This article presents various domination relations. It uses a very mathemathical presentation but is a good starting point for an overview of these relations.

[Hillier et al. 95] This is an introductory book about operational research. All the domains of operational research are introduced through a very didactic presentation (linear programming, game theory and nonlinear programming). Many well-commented examples illustrate the presentation of each method.

[Miettinen 99] This book, as indicated by the title, is entirely dedicated to multiobjective optimization. The basic concepts are presented through a mathematical discussion. This book deals mostly with a priori and progressive methods. The main mathematical characteristics of the methods are presented.

2

Scalar methods

2.1 The weighted-sum-of-objective-functions method

2.1.1 Principle

This approach to multiobjective optimization problem solving is the most obvious. We also call this method the "naive approach" to multiobjective optimization [Coello 98]. The goal, here, is to transform our problem so that it turns into a mono-objective optimization problem, for which there exist various methods of solution. The simplest way to proceed is to take each objective function, associate a weight with the objective function and then take a weighted sum of objective functions. Hence we obtain a new, unique objective function.

2.1.2 Presentation of the method

We start from the P problem (see page 18). We transform the P problem in the following way:

$$\begin{array}{|ll} \text{minimize} & f_{eq}(\vec{x}) = \sum_{i=1}^{k} w_i \cdot f_i(\vec{x}) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{array}$$

We have $\vec{x} \in \mathbb{R}^n$, $\vec{g}(\vec{x}) \in \mathbb{R}^m$ and $\vec{h}(\vec{x}) \in \mathbb{R}^p$.

Frequently, the weights must respect the following relation: $w_i \geq 0$ for all $i \in \{1, \cdots, k\}$ and

$$\sum_{i=1}^{k} w_i = 1 \tag{2.1}$$

We can obtain a graphical representation of the behavior of the method for a problem with two objective functions. The problem is the following:

$$\begin{array}{|ll} \text{minimize} & f_1(\vec{x}) \\ \text{minimize} & f_2(\vec{x}) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{array}$$

Our new objective function has the following analytical form:

$$f_{eq}(\vec{x}) = w_1 \cdot f_1(\vec{x}) + w_2 \cdot f_2(\vec{x}) \tag{2.2}$$

This is the analytical expression for a line in the f_1, f_2 plane. Indeed, if we try to minimize $f_{eq}(\vec{x})$, in fact we look for a constant C of the following line equation, which must be as small as possible:

$$f_2(\vec{x}) = -\frac{w_1}{w_2} \cdot f_1(\vec{x}) + C \tag{2.3}$$

This equation corresponds to an isovalue curve of the equivalent objective function.

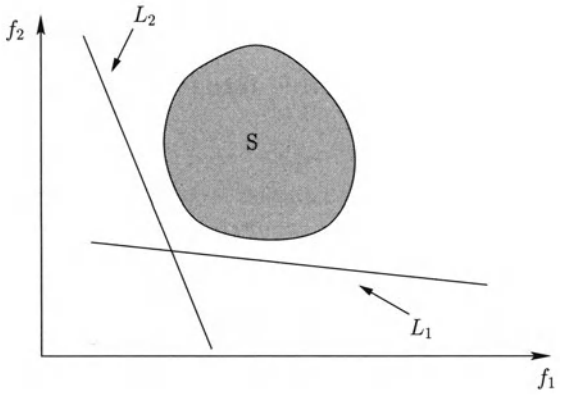


Fig. 2.1. Weighted-sum-of-objective-functions method.

In Fig. 2.1, the set S corresponds to the set of values of the pair (f_1, f_2) which respect the constraints defined by $\vec{g}(\vec{x})$ and $\vec{h}(\vec{x})$. Lines L_1 and L_2 correspond to two distinct pairs of weights (w_1, w_2).

This method consists of making lines L_1 and L_2 "tangential" to the set S. The tangent point corresponds to the solution sought.

If we repeat the process for several weight values, the solutions we find will produce a tradeoff surface.

This method can only be applied to sets S that are convex, otherwise it does not allow one to discover all the solutions on the tradeoff surface. For example, the area on the tradeoff surface marked with a bold line in Fig. 2.2 cannot be reached by this method.

2.1.3 Two concrete examples

Example 1: Schaffer's F2 problem

Let us consider the following problem with two objective functions:

$$\begin{vmatrix} \text{minimize } f_1(x) = x^2 \\ \text{minimize } f_2(x) = (x-2)^2 \\ \text{with} \qquad x \in [0,2] \end{vmatrix}$$

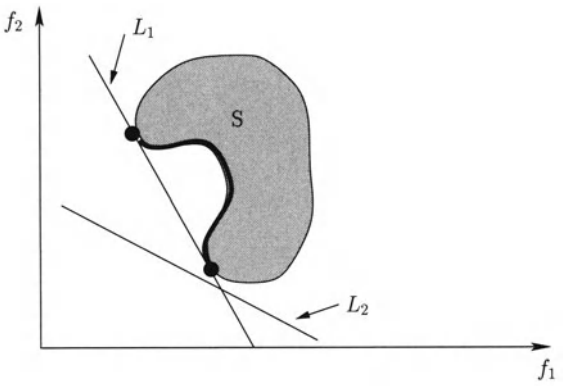


Fig. 2.2. An insurmountable difficulty for the weighted-sum-of-objective-functions method.

This is a test problem called Schaffer's F2 problem.

The representation of the solution set of this problem is different from the representation that we used earlier to illustrate the various definitions related to multiobjective optimization. If we express f_2 in terms of f_1, we obtain the equation of an ellipse:

$$\begin{vmatrix} f_1(\vec{x}) = x^2 \\ f_2(\vec{x}) = (x-2)^2 = x^2 - 4 \cdot x + 4 = f_1(\vec{x}) + 4 - 4 \cdot x \end{vmatrix}$$

Finally, we have

$$(f_2(\vec{x}) - f_1(\vec{x}) - 4)^2 = (4 \cdot x)^2$$
$$f_1^2 + f_2^2 - 2 \cdot f_1 \cdot f_2 - 8 \cdot f_1 - 8 \cdot f_2 + 16 = 0$$

Moreover, as we shall see during the optimization process, the tradeoff surface (which usually corresponds to a real surface) is a part of a curve here, and the set of the pairs (f_1, f_2) is also a curve (an ellipse in this example).

We have chosen this atypical example because it allows us to illustrate numerically and easily the behavior of some multiobjective optimization methods. In example 2 we shall deal with a less atypical multiobjective optimization problem. In the remainder of this book, however, we shall use example 1 rather than example 2 for the sake of simplicity, knowing that this kind of approach can be adapted to a classical multiobjective optimization problem.

The expression for the new objective function is

$$f_{eq}(x) = w_1 \cdot f_1(x) + w_2 \cdot f_2(x) \tag{2.4}$$

The weight coefficients w_1 and w_2 respect the following conditions:

w_1 and $w_2 \geq 0$,
$w_1 + w_2 = 1$

We seek a minimum of the objective function $f_{eq}(x)$. The conditions needed for the existence of a minimum are

$$\frac{df_{eq}(x)}{dx} = 0$$
$$\frac{d^2 f_{eq}(x)}{dx^2} > 0$$

Let us seek points of $f_{eq}(x)$ which satisfy these conditions:

$\frac{df_{eq}(x)}{dx} = 2 \cdot x \cdot (w_1 + w_2) - 4 \cdot w_2$

$\frac{d^2 f_{eq}(x)}{dx^2} = 2 \cdot (w_1 + w_2) > 0,$
because we have chosen $w_1 + w_2 = 1$.

So, the minimum is located at

$x^* = \frac{2 \cdot w_2}{w_1 + w_2} = 2 \cdot w_2$

Now, we compute four points on the tradeoff surface. We choose four values for w_1 between 0.2 and 0.8 using a step of 0.2. We deduce the values of w_2, and then we compute x^*, $f_1(x^*)$ and $f_2(x^*)$. We notice that the values of x^* respect the constraint; these values are listed in Table 2.1.

Table 2.1. Recapitulatory table.

w_1	0.2	0.4	0.6	0.8
w_2	0.8	0.6	0.4	0.2
x^*	1.6	1.2	0.8	0.4
$f_1(x^*)$	2.56	1.44	0.64	0.16
$f_2(x^*)$	0.16	0.64	1.44	2.56

Now, we plot these values in the f_1, f_2 plane and link them together (see Fig. 2.3). We notice that, in this example, we have found a typical tradeoff surface of the kind that we have when we minimize all the objective functions. In Fig. 2.4, we have represented the set of feasible solutions with and without the constraint $x \in [0, 2]$.

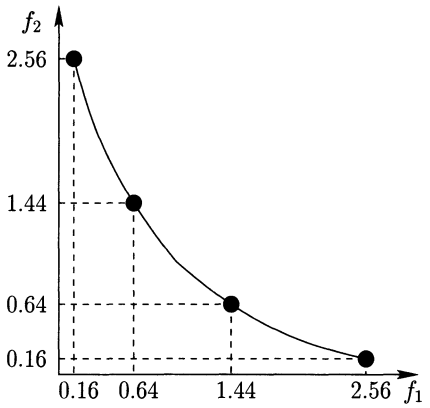


Fig. 2.3. The tradeoff surface obtained for Schaffer's F2 problem.

Example 2: The Binh problem

Let us consider the following problem with two objective functions:

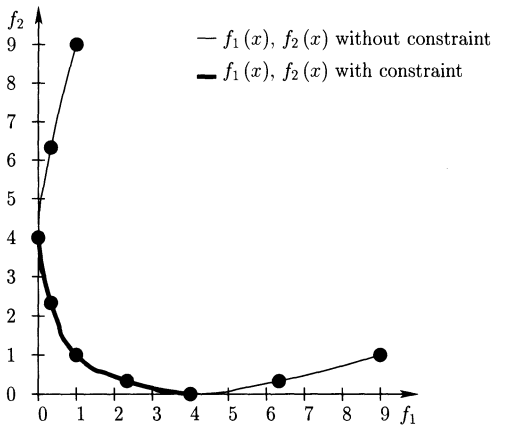


Fig. 2.4. The whole tradeoff surface, with and without the constraint $x \in [0, 2]$.

$$\begin{vmatrix}\text{minimize } f_1(x_1, x_2) = x_1^2 + x_2^2 \\ \text{minimize } f_2(x_1, x_2) = (x_1 - 5)^2 + (x_2 - 5)^2 \\ \text{with} \qquad -5 \leq x_1 \leq 10,\ -5 \leq x_2 \leq 10\end{vmatrix}$$

This problem is called the Binh problem. Some of the solutions of this problem are represented in Fig. 2.5.

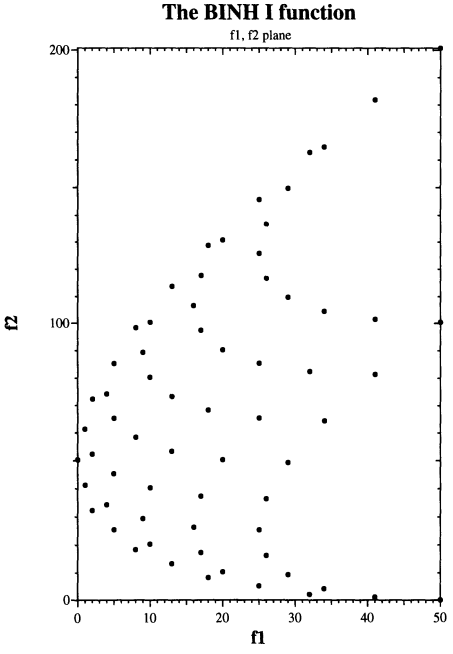


Fig. 2.5. Solution set of the Binh problem.

We begin by constructing a new objective function:

$$f_{eq}(x_1, x_2) = w_1 \cdot (x_1^2 + x_2^2) + w_2 \cdot \left((x_1 + 5)^2 + (x_2 + 5)^2\right)$$

with $w_1 + w_2 = 1$ et $w_1, w_2 \geq 0$. So, we have

$$f_{eq}(x_1, x_2) = x_1^2 + x_2^2 + 10 \cdot w_2 \cdot (x_1 + x_2) + 50 \cdot w_2$$

Now, we look for the optimum of this objective function:

$$\frac{\partial f_{eq}(x_1, x_2)}{\partial x_1} = 2 \cdot x_1 + 10 \cdot w_2 \tag{2.5}$$

$$\frac{\partial f_{eq}(x_1, x_2)}{\partial x_2} = 2 \cdot x_2 + 10 \cdot w_2 \tag{2.6}$$

$$\frac{\partial^2 f_{eq}(x_1, x_2)}{\partial x_1 \partial x_2} = \frac{\partial^2 f_{eq}(x_1, x_2)}{\partial x_1 \partial x_2} = 2 > 0 \tag{2.7}$$

Equation 2.7 shows that a search for the points at which equations 2.5 and 2.6 are equal to zero will give us a minimum of this objective function. So:

$2 \cdot x_1^* + 10 \cdot w_2 = 0 \Rightarrow x_1^* = -5 \cdot w_2$
$2 \cdot x_2^* + 10 \cdot w_2 = 0 \Rightarrow x_2^* = -5 \cdot w_2$

We find the following objective functions:

$f_{eq}(w_2) = 50 \cdot w_2 \cdot (1 - w_2)$
$f_1(w_2) = 50 \cdot w_2^2$
$f_2(w_2) = 50 \cdot (1 - w_2)^2$

We can compute some solutions of the tradeoff surface for some values of the w_2 weight. These results are gathered in table 2.2. The shape of the tradeoff surface is represented in Fig. 2.6.

Table 2.2. Recapitulatory table.

w_2	$f_{eq}(w_2)$	$f_1(w_2)$	$f_2(w_2)$
0	0	0	50
0.1	4.5	0.5	40.5
0.2	8	2	32
0.3	10.5	4.5	24.5
0.4	12	8	18
0.5	12.5	12.5	12.5
0.6	12	18	8
0.7	10.5	24.5	4.5
0.8	8	32	2
0.9	4.5	40.5	0.5
1	0	50	0

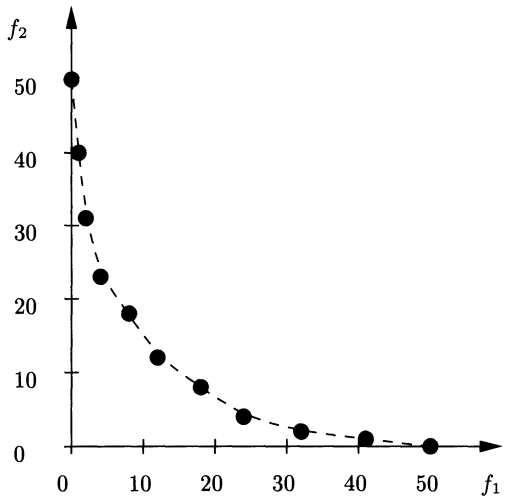


Fig. 2.6. Tradeoff surface for the Binh problem.

2.1.4 Discussion

This method was the first one used. From an algorithmic point of view, it is very efficient. We can, by varying the value of the weights, approximate the tradeoff surface, if the feasible space of values of the objective function is convex.

Nevertheless, this method cannot discover solutions hidden in concavities. In some structural optimization problems, it can behave catastrophically [Koski 85].

2.2 Keeney–Raiffa method

2.2.1 Principle

This method uses a product of objective functions to construct an equivalent mono-objective function. The approach used here is similar to the one used with the weighted-sum-of-objective-functions method. The objective function we obtain is called the Keeney–Raiffa utility function.

We find this function in the multiattribute utility theory presented in [Keeney et al. 93] (MAUT, multiattribute utility theory). This theory, which came from the decision sciences, deals with the properties of a "utility function" and ways to create it.

2.2.2 Presentation of the method

We start with the P problem (see page 18). We transform the "objective" part using the following function:

$$f_{eq}(\vec{x}) = K \cdot \prod_{i=1}^{k} (k_i \cdot u_i(f_i(\vec{x})) + 1) \tag{2.8}$$

In this expression, k_i is a normalization coefficient (between 0 and 1), and $u_i(\vec{y})$ is a strictly nondecreasing function with respect to y which can incorporate nonlinearities. For example, $u_i(y) = y^2$. This is a strictly nondecreasing function with respect to y if $y \in \mathbb{R}^+$.

If we take $u_i(y) = y^2$, $K = 1$ and $k_i = 1$, the P problem is transformed the following way:

$$\begin{vmatrix} \text{minimize} & f_{eq}(\vec{x}) = \prod_{i=1}^{k} \left((f_i(\vec{x}))^2 + 1 \right) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{vmatrix}$$

2.3 Distance-to-a-reference-objective method

2.3.1 Principle

This method allows one to transform a multiobjective optimization problem into a mono-objective one.

The sum that we use here corresponds to a distance (for example $\sqrt[k]{\sum()^k}$, or the kth root of the sum of elements to the power of k). In some cases, the elements are normalized with respect to some value before computing the sum [Azarm 96, Miettinen 99].

2.3.2 Presentation of the method

We start again from the P problem.

In the following definitions, the vector $\vec{F}$ (of components F_i, $i \in \{1, \cdots, k\}$) is a vector whose components correspond to an ideal objective (or a reference objective) in a sense defined by the decision maker (it can be an ideal point, or a point which represents best the preferences of the decision maker). The vector $\vec{F}$ is chosen by the decision maker.

We can, for example, use the absolute distance. The definition of this distance is

$$L_r\left(\vec{f}(\vec{x})\right) = \left[\sum_{i=1}^{k} |F_i - f_i(\vec{x})|^r\right]^{\frac{1}{r}} \tag{2.9}$$

with $1 \leq r < \infty$.

For example:

- $r = 1$:

$$L_1\left(\vec{f}(\vec{x})\right) = \sum_{i=1}^{k} |F_i - f_i(\vec{x})| \tag{2.10}$$

- $r = 2$:

$$L_2\left(\vec{f}(\vec{x})\right) = \left[\sum_{i=1}^{k} |F_i - f_i(\vec{x})|^2\right]^{\frac{1}{2}} \tag{2.11}$$

- $r \to \infty$:

$$L_\infty\left(\overrightarrow{f}\left(\overrightarrow{x}\right)\right) = \max_{i\in\{1,\cdots,k\}}\left(F_i - f_i\left(\overrightarrow{x}\right)\right) \tag{2.12}$$

For this last form, the multiobjective method is called the min–max method or the Tchebychev method.

We can also use the relative distance. The definition of this distance is

$$L_r^{Rel}\left(\overrightarrow{f}\left(\overrightarrow{x}\right)\right) = \left[\sum_{i=1}^{k}\left|\frac{F_i - f_i\left(\overrightarrow{x}\right)}{F_i}\right|^r\right]^{\frac{1}{r}} \tag{2.13}$$

with $1 \leq r < \infty$.

For example, the "relative min–max" method is obtained for $r \to \infty$:

$$L_{minmax}\left(\overrightarrow{f}\left(\overrightarrow{x}\right)\right) = \max_{i\in\{1,\cdots,k\}}\left(\frac{F_i - f_i\left(\overrightarrow{x}\right)}{F_i}\right) \tag{2.14}$$

Lastly, we can use weight coefficients in the distance formula. These weights allow one to look for a point in a certain area of the search space, as with the weighted sum of objective functions. We obtain the following expression:

$$L_r^W\left(\overrightarrow{f}\left(\overrightarrow{x}\right)\right) = \left[\sum_{i=1}^{k}\left|w_i \cdot \left(F_i - f_i\left(\overrightarrow{x}\right)\right)\right|^r\right]^{\frac{1}{r}}$$

with $1 \leq r \leq \infty$.

For the min–max method, we obtain

$$L_{minmax}\left(\overrightarrow{f}\left(\overrightarrow{x}\right)\right) = \max_{i\in\{1,\cdots,k\}}\left(w_i \cdot \left(F_i - f_i\left(\overrightarrow{x}\right)\right)\right)$$

Let us apply the distance method to the P problem.

To transform our problem, we shall use the distance L_2. We have the following equivalent problem:

$$\begin{array}{|ll} \text{minimize} & f_{eq}\left(\overrightarrow{x}\right) = \sqrt{\sum_{i=1}^{k}\left|F_i - f_i\left(\overrightarrow{x}\right)\right|^2} \\ \text{with} & \overrightarrow{g}\left(\overrightarrow{x}\right) \leq 0 \\ \text{and} & \overrightarrow{h}\left(\overrightarrow{x}\right) = 0 \end{array}$$

We emphasize that, when we use these methods to search for Pareto optimal solutions, it is not necessary to include the $\sqrt{}$ operator in the $f_{eq}\left(\overrightarrow{x}\right)$ expression, because this operator is monotonic. This implies a gain with respect to the computing time of the equivalent objective function.

Figure 2.7 illustrates the behavior of this method. In this figure, we have represented the reference point (which is different from the ideal point) using a black dot. The tradeoff surface is represented using a bold line. As we can see in Fig. 2.10, the choice of a reference point has a lot of influence on the "quality" of the result. This method allows us to approximate the tradeoff surface, but the surface is "seen" from a reference point.

2.3.3 Isovalue curves

To plot an isovalue curve for these aggregation-of-objective-functions method, we must proceed the same way as with the weighted-sum-of-objective-functions method:

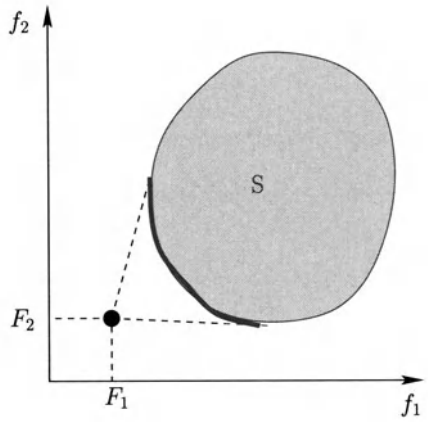


Fig. 2.7. Distance method.

minimizing $f_{eq}(\vec{x})$ is equivalent to finding a value for the variable $\vec{x}$ such that the constant C of the following expression is minimal:

$$C = w_1 \cdot (f_1(\vec{x}) - F_1)^2 + w_2 (f_2(\vec{x}) - F_2)^2 \tag{2.15}$$

This is the equation of an ellipse of center (F_1, F_2), with minor axis $2 \cdot \sqrt{\frac{C}{w_1}}$ and major axis $2 \cdot \sqrt{\frac{C}{w_2}}$.

So, the goal is to find the smallest C such that the ellipse is "tangential to" the solution set S. This isovalue curve is represented in Fig. 2.8.

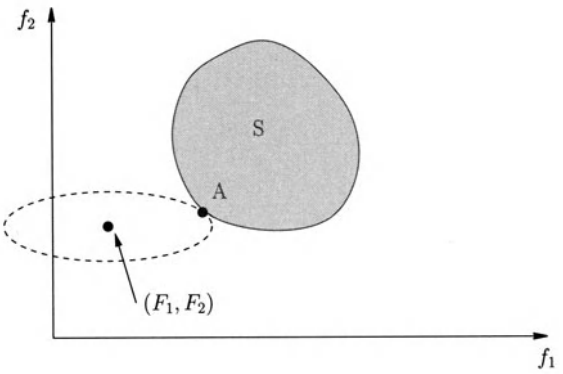


Fig. 2.8. An isovalue curve for the distance method.

For the Tchebychev method, the equation of the isovalue curve is much simpler:

$$\begin{cases} f_1(\vec{x}) = C \text{ if } w_1 (f_1(\vec{x}) - F_1) > w_2 (f_2(\vec{x}) - F_2) \\ f_1(\vec{x}) = C \text{ if } w_1 (f_1(\vec{x}) - F_1) < w_2 (f_2(\vec{x}) - F_2) \end{cases} \tag{2.16}$$

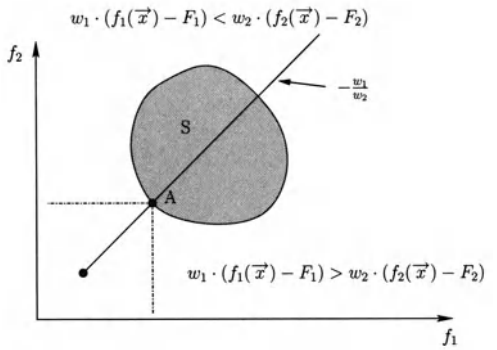


Fig. 2.9. An isovalue curve for the Tchebychev method.

This curve is represented in Fig. 2.9.

2.3.4 Discussion

For this method to work well, we must be able to find a good reference point. Otherwise, the solutions that we find will not be "optimal", in the sense that they will not necessarily correspond to the initial point.

Figure 2.10 illustrates a defective behavior of this method when the reference objective is badly chosen. We are trying to minimize two objective functions (f_1 and f_2); a good choice for the reference point is illustrated in Fig. 2.7.

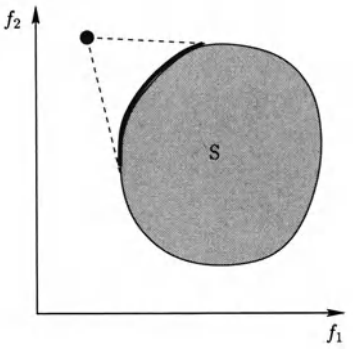
- In Fig. 2.10a, the fact that we have chosen the reference point in the upper left corner gives us a tradeoff surface equivalent to the one that we obtain when solving a multiobjective problem where we minimize objective function f_1 and maximize objective function f_2.
- In Fig. 2.10b, the fact that we have chosen the reference point in the upper right corner gives us a tradeoff surface equivalent to the one that we obtain when solving a multiobjective problem where we maximize both objective functions (f_1 and f_2).
- In Fig. 2.10c, the fact that we have chosen the reference point in the lower right corner gives us a tradeoff surface equivalent to the one that we obtain when solving a multiobjective problem where we maximize objective function f_1 and we minimize objective function f_2.
- Lastly, in Fig. 2.10d, the fact that we have chosen the reference point inside the feasible set of objective function values (f_1, f_2) gives us a unique solution which corresponds neither to a maximization nor to a minimization of the two objective functions.

However, as with the weighted sum of objective functions, this method allows one to discover solutions hidden inside concavities under some conditions, as presented in [Messac et al. 00].

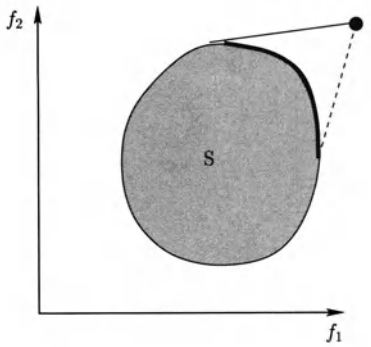
2.4 Compromise method

2.4.1 Principle

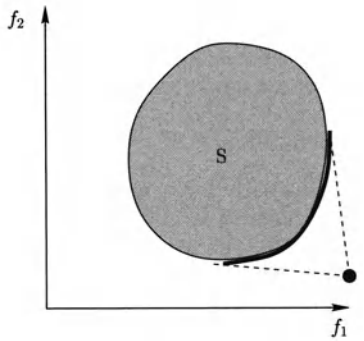
Having studied methods which allow us to merge objective functions into one function (we call this transformation "aggregation" of objective functions), we shall



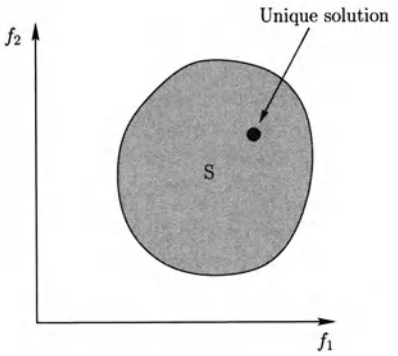
(a) Maximization of f_2 and minimization of f_1.



(b) Maximization of f_2 and maximization of f_1.



(c) Minimization of f_2 and maximization of f_1.



(d) Unique solution..

Fig. 2.10. Difficulty in choosing a reference point.

study methods which allow us to transform a multiobjective optimization problem into a mono-objective optimization problem with additional constraints.

The approach is the following:

- we choose an objective function to be optimized with high priority;
- we choose a vector of initial constraints;
- we transform the problem by conserving the objective function to be optimized and transforming all other objective functions into inequality constraints.

This method is also called "method of the ε constraint" [Miettinen 99].

2.4.2 Presentation of the method

We start from the P problem. We suppose that the objective function with a high priority has the index 1. We choose a constraint vector ε_i, $i \in \{2, \cdots, k\}$, $\varepsilon_i \geq 0$. We transform the P problem as follows:

$$\begin{array}{ll} \text{minimize} & f_1(\vec{x}) \\ \text{with} & f_2(\vec{x}) \leq \varepsilon_2 \\ & \vdots \\ & f_k(\vec{x}) \leq \varepsilon_k \\ & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{array}$$

In Fig. 2.11, we have illustrated the behavior of this method for a problem with two objective functions.

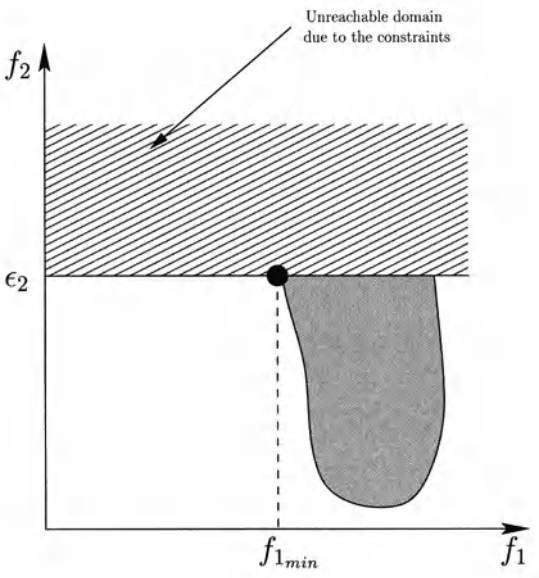


Fig. 2.11. The compromise method.

2.4.3 A concrete example

We take again the example of Schaffer's F2 problem.

By applying the preceding transformation process, we obtain the following problem:

$$\begin{array}{ll} \text{minimize} & f_1(x) = x^2 \\ \text{with} & f_2(x) = (x-2)^2 \leq \varepsilon \\ \text{and} & x \in [0, 2] \end{array}$$

We now decide to solve this problem by choosing four values for the variable ε: 0, 1, 2, 3. We rewrite the constraint as

$$(x-2)^2 \leq \varepsilon \Leftrightarrow 2-\sqrt{\varepsilon} \leq x \leq 2+\sqrt{\varepsilon}$$

With the values of ε that we have chosen,

$$0 < 2-\sqrt{\varepsilon} \leq 2$$

When we merge this expression with $x \in [0,2]$, we have the following new constraint:

$$x \in [2-\sqrt{\varepsilon}, 2]$$

Table 2.3. Solutions for Schaffer's F2 problem.

ε	Constraint	Solution x^*	$f_1(x^*)$	$f_2(x^*)$
0	$x \in [2,2]$	2	4	0
1	$x \in [1,2]$	1	1	1
2	$x \in [2-\sqrt{2},2]$	$2-\sqrt{2}$	$6-4\cdot\sqrt{2}$	2
3	$x \in [2-\sqrt{3},2]$	$2-\sqrt{3}$	$7-4\cdot\sqrt{3}$	3

The resulting solutions are listed in Table 2.3. We can represent the various solutions in the (f_1, f_2) plane (see Fig. 2.12).

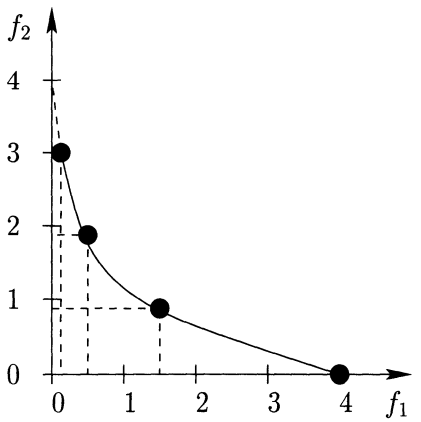
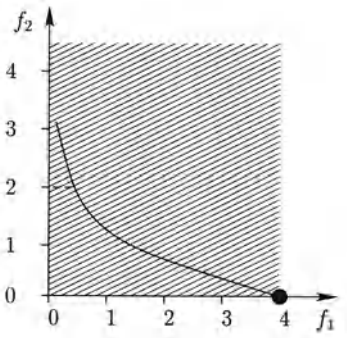


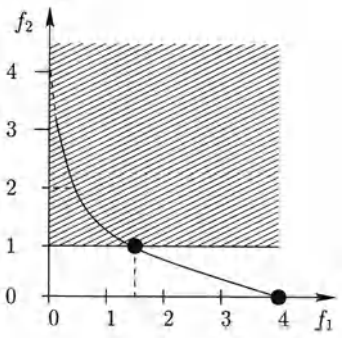
Fig. 2.12. Representation of the tradeoff surface for Schaffer's F2 problem, using 0, 1, 2 and 3 for the variable ε.

In Fig. 2.13, we have illustrated the approach we have followed to solve this test problem. We see in this figure the effect of changing the bound of our new constraint. Moreover, we can obtain, step by step, solutions of our problem and, in the same way, the tradeoff surface.

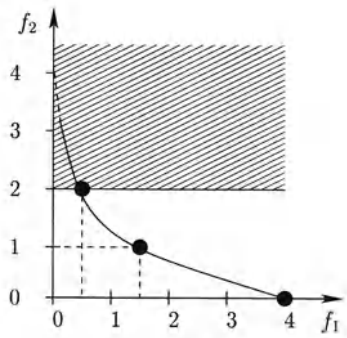
For simple problems, a choice of values for ε_i can provide a good spread of solutions on the tradeoff surface. Nevertheless, in most concrete cases, an arbitrary choice of



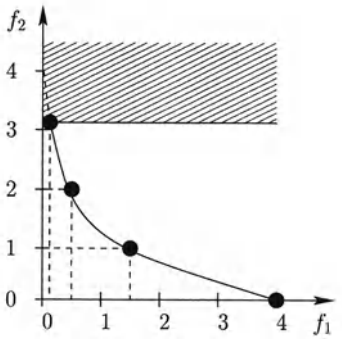
(a) Bound of the constraint set to 0.



(b) Bound of the constraint set to 1.



(c) Bound of the constraint set to 2.



(d) Bound of the constraint set to 3.

Fig. 2.13. The step-by-step solution of the test problem.

values for ε_i does not allow us to obtain a good spread of solutions on the tradeoff surface (this is the case in our example because a solution is missing between values 2 and 3 for the f_1 coordinate). In such cases we cannot obtain a sufficient number of solutions, and meet with many difficulties in extrapolating the shape of the tradeoff surface.

2.4.4 Discussion

In [Eschenauer et al. 90], it is shown that this method can be transformed into the weighted-sum-of-objective-functions method by making a slight transformation.

The main drawbacks of this method are the following: it consumes a large amount of computing time, and the programming of the algorithm can be very hard to do if there are too many objective functions. Nevertheless, the relative simplicity of the equations of the method has made it popular.

2.5 Hybrid methods

2.5.1 Principle

As we shall see, it is possible to combine several methods into a new one. In that case, we have a "hybrid method".

2.5.2 Presentation of a hybrid method

The best-known hybrid method is the Corley method. This method uses the weighted-sum-of-objective-functions and compromise methods.

We start from the P problem. We modify it the following way:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{k} w_i \cdot f_i(\vec{x}) \\ \text{with} & f_j(\vec{x}) \leq \varepsilon_j,\ j = 1, \cdots, k \\ & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{array}$$

In Fig. 2.14, we have illustrated the behavior of this method for a problem with two objective functions.

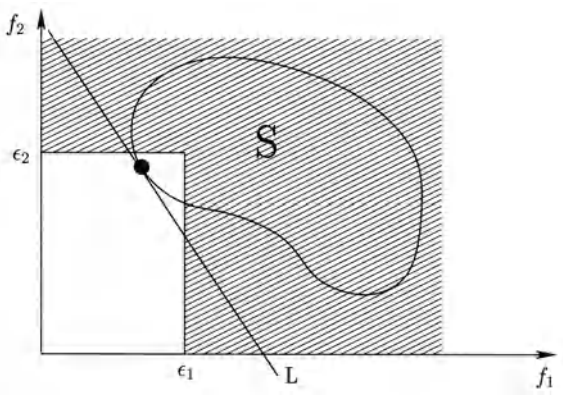


Fig. 2.14. The Corley hybrid method.

In this figure, we can see that adding the two constraints ($f_1(\vec{x}) \leq \varepsilon_1$ and $f_2(\vec{x}) \leq \varepsilon_2$) allows us to restrict the search space. We then apply the weighted-sum-of-objective-functions method in the restricted search space. For a weight vector $\vec{w} = (w_1, w_2)$ which corresponds a line L, searching for an optimal value is equivalent to making the line L "tangential" to the restricted search space.

2.5.3 Discussion

This method allows one to combine the advantages of two methods described in the preceding sections (the weighted-sum-of-objective-functions method and the compromise method). So, it is efficient in various kinds of optimization problems, whether convex or not. Nevertheless, a difficulty arises: the number of parameters to

be tuned has been multiplied by two. It is harder for the decision maker to express his/her preferences by adjusting the whole set of parameters. For more information about this method, see [Miettinen 99].

2.6 Goal attainment method

2.6.1 Principle

Unlike the weighted-sum-of-objective-functions method, this method does not require us to work with a convex feasible set.

The approach followed is:

- we choose an initial vector of ideal objective function values $\overrightarrow{F}$;
- we choose a search direction $\overrightarrow{w}$ (in a sense, we provide weights, as in the weighted-sum-of-objective-functions method);
- next, we try to minimize a scalar coefficient λ which represents the gap relative to the initial objective $\overrightarrow{F}$ that we have set up.

2.6.2 Presentation of the method

We start from the P problem. We choose an initial vector of objective functions $\overrightarrow{F} \in \mathbb{R}^k$ and a set of weights w_i, $i \in \{1, \cdots, k\}$ and λ designates a scalar variable that we will try to minimize. The P problem becomes:

$$
\begin{array}{ll}
\text{minimize } \lambda & \\
\text{with} & f_1(\overrightarrow{x}) - w_1 \cdot \lambda \leq F_1 \\
& \vdots \\
& f_k(\overrightarrow{x}) - w_k \cdot \lambda \leq F_k \\
& \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\
\text{and} & \overrightarrow{h}(\overrightarrow{x}) = 0
\end{array}
$$

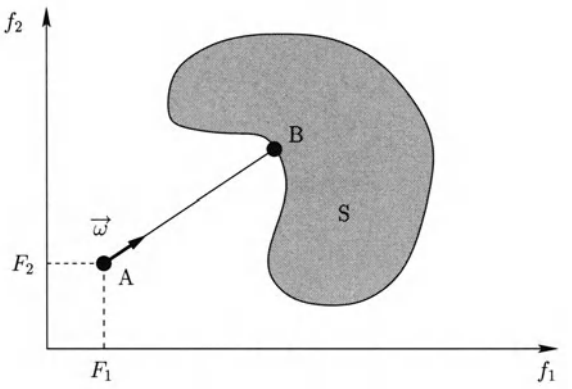


Fig. 2.15. The goal attainment method, for a problem with two objective functions.

In Fig. 2.15 we represent the behavior of this method for a problem with two objective functions. In this figure, point A represents the reference objective, the vector $\overrightarrow{w}$ represents a search direction and point B represents the goal to be reached.

The aim of this method is to find a value for λ (which represents the distance between the point A and the point B) such that the point B lies on the boundary of the set S and the distance between the point A and the point B is minimized. We can easily see in this figure that this method is efficient when we are working with nonconvex sets.

Often, the variable λ is called the degree of overattainment or underattainment of the objective $\overrightarrow{F}$.

To obtain the tradeoff surface, we must choose, as before, several search directions (and maybe even several distinct initial vectors of objective functions).

For more information about this method, see [Pentzaropoulos et al. 93].

2.6.3 A concrete example

Let us consider again Schaffer's F2 problem. We transform this problem the following way:

$$\begin{array}{ll} \text{minimize } \lambda & \\ \text{with} & x^2 - w_1 \cdot \lambda \leq 1 \\ & (x-2)^2 - w_2 \cdot \lambda \leq 1 \\ \text{and} & x \in [0,2] \end{array}$$

Our knowledge of this problem leads us to choose $F_1 = 1$ and $F_2 = 1$. In addition, we impose $w_1 + w_2 = 1$ and $w_1, w_2 \geq 0$.

Now, we shall try to find a λ_{min} such that the constraints are respected. For a better illustration of our aim we represent in Fig. 2.16 the functions $(g_1(f_1, f_2, \lambda)$, $g_2(f_1, f_2, \lambda)$, $\lambda)$ in the (f_1, f_2, λ) plane. We use the following notation:

- Δ_1 represents the line $g_1(f_1, f_2, \lambda) = f_1(x) - w_1 \cdot \lambda$,
- Δ_2 represents the line $g_2(f_1, f_2, \lambda) = f_2(x) - w_2 \cdot \lambda$.

The interesting point in this figure is A. For the corresponding value of λ, λ_A, the constraints of our problem are respected whatever the value of x. This point is defined by

$$\begin{cases} g_1(f_1, f_2, \lambda_A) = 1 \\ g_2(f_1, f_2, \lambda_A) = 1 \end{cases}$$

We solve this problem as follows (we eliminate λ_A and find x):

$$\begin{cases} x^2 - w_1 \cdot \lambda_A = 1 \\ (x-2)^2 - w_2 \cdot \lambda_A = 1 \end{cases}$$

$$(x-2)^2 - \frac{w_2}{w_1} \cdot \left(x^2 - 1\right) = 1$$

$$x^2 \cdot \left(1 - \frac{w_2}{w_1}\right) - 4 \cdot x + 3 + \frac{w_2}{w_1} = 0$$

So, we have

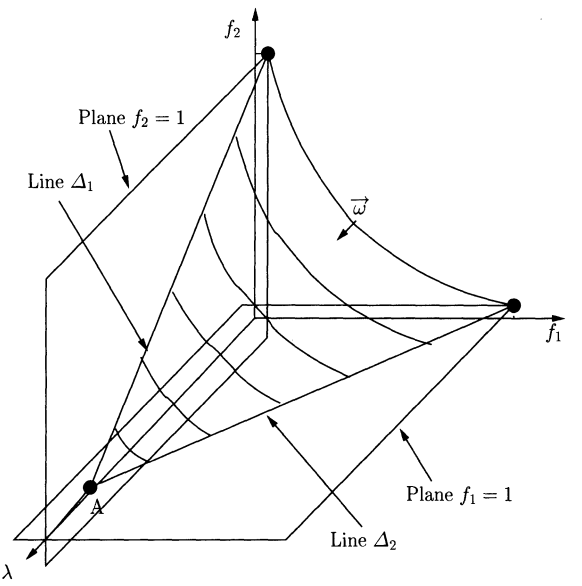


Fig. 2.16. Representation of the problem in the (f_1, f_2, λ) plane.

- $\Delta = 4 + 8 \cdot \frac{w_2}{w_1} + 4 \cdot \left(\frac{w_2}{w_1}\right)^2$ or

  $\Delta = 4 \cdot \left(1 + \frac{w_2}{w_1}\right)^2$, so $\sqrt{\Delta} = 2 \cdot \left(1 + \frac{w_2}{w_1}\right)$

- $x_1 = \frac{4+\sqrt{\Delta}}{2 \cdot \left(1 - \frac{w_2}{w_1}\right)}$, $x_2 = \frac{4-\sqrt{\Delta}}{2 \cdot \left(1 - \frac{w_2}{w_1}\right)} = 1, \forall w_1, w_2$

Finally, we have to fill in table 2.4.

In this table, the variables have the following meaning

w_1 and w_2: the weight coefficients.

Δ: polynome determinant.

x: values of x_1 and/or x_2 which respect the constraint $x \in [0, 2]$. For the first nine values, x_1 has been chosen, and for the remaining values, x_2 (only one value) has been chosen. In this way, we can guarantee some diversity in the spread of points on the tradeoff surface.

$f_1(x)$: the value of the objective function f_1 as a function of x.

$f_2(x)$: the value of the objective function f_2 as a function of x.

$\lambda_A(x)$: the value of λ_A as a function of x. This value is computed using the following formula:

$$\lambda_A = \frac{x^2 - 1}{w_1}$$

We represent the first ten solutions that we have obtained in the (f_1, f_2) plane in Fig. 2.17).

As we can see in this example, the representation of the solutions allows us to plot the tradeoff surface of the optimization problem. Nevertheless, we may notice that the solutions were obtained using a restricted range for the weights w_1, w_2.

Table 2.4. Results of the concrete example.

w_1	w_2	Δ	x_1	x	f_1	f_2	λ_A
0.05	0.95	1600	1.78	1.78	3.17	0.04	43.37
0.1	0.9	400	1.5	1.5	2.25	0.25	12.5
0.15	0.85	178	1.14	1.14	1.3	0.74	1.99
0.2	0.8	100	0.67	0.67	0.44	1.77	−2.75
0.21	0.79	90.7	0.55	0.55	0.30	2.10	−3.32
0.22	0.78	82.6	0.42	0.42	0.17	2.49	−3.74
0.23	0.77	75.6	0.29	0.29	0.08	2.92	−3.98
0.24	0.76	69.44	0.15	0.15	0.02	3.42	−4.07
0.25	0.75	64	0	0	0	4	−4
0.3	0.7	44.44	−1	1	1	1	0
0.4	0.6	25	−6	1	1	1	87.5
0.5	0.5	16	×	1	1	1	×
0.6	0.4	11.11	14	1	1	1	325
0.7	0.3	8.16	9	1	1	1	114
0.8	0.2	6.25	7.33	1	1	1	65.9
0.9	0.1	4.93	6.5	1	1	1	45.83

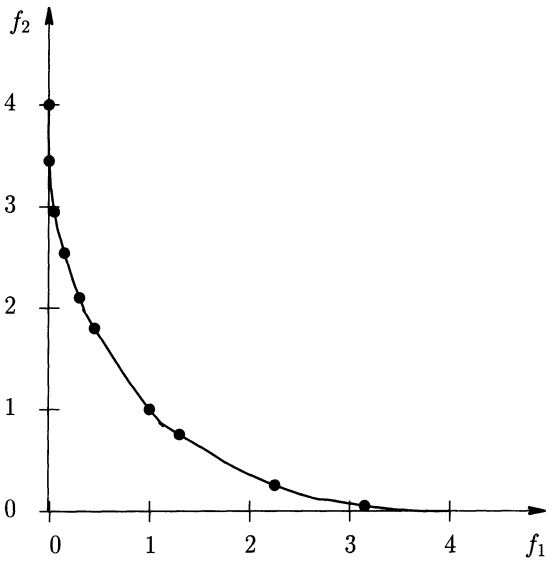


Fig. 2.17. Representation in the (f_1, f_2) plane of the solutions obtained using the goal attainment method.

Relative to the weighted-sum-of-objective-functions method, we can say that the goal attainment method is more sensitive to the choice of the weights that we use to compute a solution. The problem with this method in that the weights do not reflect the preference of the decision maker as with the weighted-sum-of-objective-function method. The weights do not have any "physical" significance, as with the weighted-sum-of-objective-functions method.

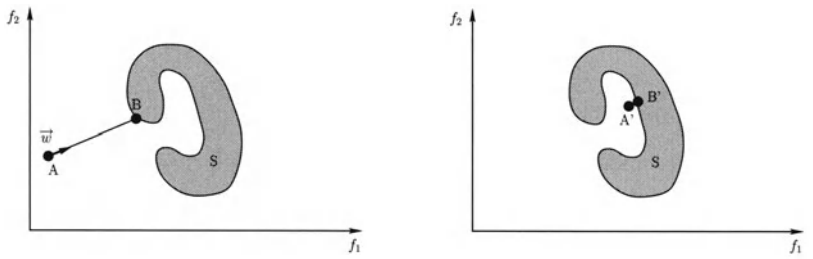
The choice of weights uniformly distributed in the interval $[0, 1]$ does not allow us to obtain an uniform representation of the solutions on the tradeoff surface. To obtain an uniform representation, we have to proceed the following way: we process a first "pass" so as to locate an area of interest, then, we focalize on the area of interest during a second "pass", which allows us to fine-tune the weights, so as to obtain a better representation of the solutions on the tradeoff surface.

2.6.4 Discussion

The main advantage of this method is that it allows us to work with a set of solutions S which is not necessarily convex. Nevertheless, there can exist some shapes of the search space for which the value of the solution depends heavily on the value of the reference point. In Fig. 2.18b we have represented such a space. Fortunately, this is an extreme case.

This conclusion can be drawn for all the methods using a distance.

Moreover, in the case of this example, if we do not fine-tune our weights during a second optimization phase, we encounter a classical difficulty in multiobjective optimization: nonuniformity of the representation of the solutions on the tradeoff surface.



(a) A common case. (b) An extreme case.

Fig. 2.18. A difficulty for the goal attainment method. A and A': initial objectives. B and B': optimal solutions. $\vec{w}$ and $\vec{w}'$: search directions.

2.7 Goal programming method

2.7.1 Principle

This method is close to the goal attainment method. The main difference is that, after having transformed the optimization problem, we have equality constraints instead of inequality constraints. The approach is the following:

- We choose an initial vector of objective functions $\overrightarrow{F}$.
- With each objective, we associate two new "slack" variables related to the initial vector of objective functions that we have chosen.
- Next, we minimize one of the two variables. The choice of the variable is made using the type of overstep we want (over or under the objective we have chosen).

2.7.2 Presentation of the method

We start from the P problem. We choose an initial vector of objective functions $\overrightarrow{F} \in \mathbb{R}^k$. We also associate a set of slack variables d_i^+ and d_i^- with each objective function $f_i(\overrightarrow{x})$, $i \in \{1, \cdots, k\}$.

So, we have the following problem:

$$\begin{array}{|ll} \text{minimize} & \left(d_1^+ \text{ or } d_1^-, \cdots, d_k^+ \text{ or } d_k^-\right) \\ \text{with} & f_1(\overrightarrow{x}) = F_1 + d_1^+ - d_1^- \\ & \vdots \\ & f_k(\overrightarrow{x}) = F_k + d_k^+ - d_k^- \\ & \overrightarrow{h}(\overrightarrow{x}) = 0 \\ \text{and} & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \end{array}$$

The slack variables that we try to minimize must respect some constraints:

d_i^+ and $d_i^- \geq 0$,
$d_i^+ \cdot d_i^- = 0$ with $i \in \{1, \cdots, k\}$.

We can try to minimize various combinations of the coefficients d_i^- and d_i^+, depending on the way in which we wish to reach the goal $\overrightarrow{F}$. These combinations are listed in Table 2.5.

Table 2.5. Various kinds of combinations of slack variables.

Type	Value of the slack	Variable
We want to reach the objective i using higher values	Positive	d_i^+
We want to reach the objective i using smaller values	Negative	d_i^-
We want to reach perfectly the objective	Null	$d_i^+ + d_i^-$

For example, if we want to reach the objective using higher values, we obtain the following problem:

$$\begin{array}{ll} \text{minimize} & \left(d_1^+, \cdots, d_k^+\right) \\ \text{with} & f_1(\vec{x}) = F_1 + d_1^+ \\ & \vdots \\ & f_k(\vec{x}) = F_k + d_k^+ \\ & \vec{h}(\vec{x}) = 0 \\ \text{and} & \vec{g}(\vec{x}) \leq 0 \end{array}$$

We see that this method allows us to transform our multiobjective optimization problem into the minimization of a vector. To minimize this vector, we can proceed in various ways:

- We can minimize a weighted sum of slack variables. For example,

$$\min\left(d_1^+, d_2^-, d_3^+, d_4^+ + d_4^-\right)$$

  may become

$$\min\left(2 \cdot d_1^+ + 3 \cdot d_2^- + 2 \cdot d_3^+ + 1 \cdot \left(d_4^+ + d_4^-\right)\right)$$

  The various weights define the preferences of the decision maker relative to the objective functions.
- We can minimize lexicographically the coordinates of the vector. We call this method "lexicographic optimization". If we take the above example again, we can proceed by the following steps:
  - minimize d_1^+,
  - minimize d_2^- while keeping d_1^+ constant,
  - minimize d_3^+ while keeping d_2^- and d_1^+ constant,
  - minimize $d_4^+ + d_4^-$ while keeping d_3^+, d_2^- and d_1^+ constant.

2.7.3 Discussion

The goal programming method is subject to the same criticisms as the goal attainment method. In some cases, we can fail to discover solutions hidden in concavities.

For more information, see [Azarm 96], [Coello 98], [MOPGP] and [Miettinen 99].

2.8 Lexicographic method

2.8.1 Principle

This method is very intuitive. It consists in considering objective functions one after the other and minimizing a mono-objective optimization problem while the problem is completed, gradually with constraints [Coello 98].

2.8.2 Presentation of the method

We start from the P problem. We proceed in k steps (as many steps as there are objective functions). Let us begin with the first objective function. We solve

$$\begin{array}{ll} \text{minimize} & f_1(\vec{x}) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{array}$$

We denote by f_1^* the solution of this problem.

Next, we transform the first objective function into an equality constraint. Then, we take the second objective function and solve the following problem:

$$\begin{array}{ll} \text{minimize} & f_2(\overrightarrow{x}) \\ \text{with} & f_1(\overrightarrow{x}) = f_1^* \\ & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\ \text{and} & \overrightarrow{h}(\overrightarrow{x}) = 0 \end{array}$$

We repeat this approach until we reach objective function number k. Then, lastly, we will have:

$$\begin{array}{ll} \text{minimize} & f_k(\overrightarrow{x}) \\ \text{with} & f_1(\overrightarrow{x}) = f_1^*, \cdots, f_1(\overrightarrow{x}) = f_{k-1}^* \\ & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\ \text{and} & \overrightarrow{h}(\overrightarrow{x}) = 0 \end{array}$$

The last value of $\overrightarrow{x}$ is the one which minimizes all the objective functions.

2.8.3 Discussion

This method has a drawback: the user needs to choose the sequence of objective functions to be minimized. This choice is somewhat arbitrary, as with the choice of weights in the weighted-sum-of-objective-functions method. Two distinct lexicographic optimizations with distinct sequences of objective functions do not produce the same solution.

2.9 Proper-equality-constraints method

2.9.1 Principle

This method has been presented in [Lin 76]. This method is also called strict-equality-constraints method. It is very cumbersome from a mathematical point of view, but it does not need any hypotheses about linearity and/or convexity.

2.9.2 Presentation of the method

We start from the P problem. We transform it the following way:

$$\begin{array}{ll} \text{minimize} & f_k(\overrightarrow{x}) \\ \text{with} & f_1(\overrightarrow{x}) = \alpha_1 \\ & \vdots \\ & f_{k-1}(\overrightarrow{x}) = \alpha_{k-1} \\ & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\ \text{and} & \overrightarrow{h}(\overrightarrow{x}) = 0 \end{array}$$

Up to now, this method is similar to the compromise method. Before we continue, we present some notation.

- We define $\overrightarrow{\alpha} = \{\alpha_1, \cdots, \alpha_{k-1}\}$. This is the vector of the bounds of the constraints.

- We define by X the feasible domain.
- We define $X_\alpha = \{\vec{x} \in X \mid f_1(\vec{x}) = \alpha_1, \cdots, f_{k-1}(\vec{x}) = \alpha_{k-1}\}$.
- We define $D = \{\vec{\alpha} \in \mathbb{R}^{k-1} \mid X_\alpha \neq \emptyset\}$. This is the set of values of $\vec{\alpha}$ such that the additional constraints (obtained after transformation of the problem) are not too restrictive and there exist values in the set X_α.
- We define $\phi(\vec{\alpha}) = \inf\{f_k(\vec{x}) \mid \vec{x} \in X_\alpha\}$. This is the minimal value of the objective function with respect to the constraints.
- We define $B = \{\vec{\alpha} \in D \mid \phi(\vec{\alpha}) < \infty \text{ and } f_k\left(\vec{\hat{x}}\right) = \phi(\vec{\alpha}),\ \vec{\hat{x}} \in X_\alpha\}$. B is a restriction of D.

These various definitions are illustrated in Fig. 2.19.

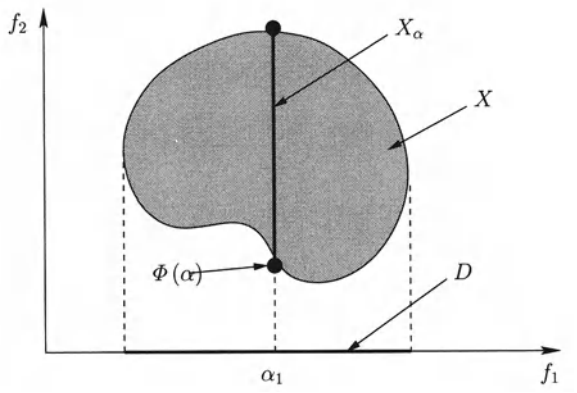


Fig. 2.19. Representation of the notation used with the proper-equality-constraint method in the case $k = 2$.

This method has five steps:

Step 1: Transform all the objectives but one into parametric equality constraints. We obtain the following problem:

$$\begin{array}{ll} \text{minimize} & f_k(\vec{x}) \\ \text{with} & f_1(\vec{x}) = \alpha_1, \cdots, f_{k-1}(\vec{x}) = \alpha_{k-1} \\ & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{array}$$

Step 2: Solve this problem and determine $\phi(\vec{\alpha})$, D and $\hat{x}(\vec{\alpha})$. The significance of the last parameter is as follows. For each $\vec{\alpha} \in D$ there corresponds an $\vec{\hat{x}} = \vec{\hat{x}}(\vec{\alpha})$ which is not necessarily in X nor in X_α. This $\vec{\hat{x}}$ is such that $f_k\left(\vec{\hat{x}}(\vec{\alpha})\right) = \phi(\vec{\alpha})$. We call this value a "supremum" solution of the related problem.

Step 3: Determine the set B and the optimal solution of the related problem $\vec{\hat{x}}(\vec{\alpha})$ with $\vec{\alpha} \in B$.

Step 4: Remove from B all the constraint vectors $\vec{\alpha}$ which are not proper (the following two propositions show how to determine whether a vector is proper).

Step 5: Determine the final remaining members of B, which we denote by B^*.

We now present two propositions which allow us to determine whether a vector is effectively proper.

Proposition 1.
A vector $\overrightarrow{\alpha_0}$ is proper if one of the following assertions is true:

1. *ϕ is increasing on D and $\phi(\overrightarrow{\alpha}) \neq \phi(\overrightarrow{\alpha_0})\ \forall \overrightarrow{\alpha} \in B$ such that $\overrightarrow{\alpha} \geq \overrightarrow{\alpha_0}$.*
2. *ϕ is increasing on D and is absolutely increasing on the right of $\overrightarrow{\alpha_0}$ on B.*
3. *ϕ is increasing on D and is absolutely increasing on B.*
4. *ϕ is absolutely increasing on the right of $\overrightarrow{\alpha_0}$ on D.*
5. *ϕ is absolutely increasing on the subset $D(\overrightarrow{\alpha_0}) = \{\overrightarrow{\alpha} \in D \mid \overrightarrow{\alpha} \geq \overrightarrow{\alpha}\}$.*
6. *ϕ is absolutely increasing on D.*
7. *ϕ is absolutely increasing on $D(\overrightarrow{\alpha_0})$ and is locally absolutely increasing on the right of $\overrightarrow{\alpha_0}$.*

Proposition 2.
A vector $\overrightarrow{\alpha_0} \in B$ is not locally proper if $\frac{\partial^+ \phi(\overrightarrow{\alpha_0})}{\partial \alpha_i} < 0$ for at least one i such that $1 \leq i \leq k-1$.

The main aim of these propositions can be described as follows (see Fig. 2.20):

1. We make the hypothesis that we have found a set B (Fig. 2.20a).
2. We apply one of these propositions to filter the set B and remove all the vectors which are not proper (Fig. 2.20b).
3. The set B^* now contains the tradeoff surface.

For a better understanding of this method, we shall use an example.

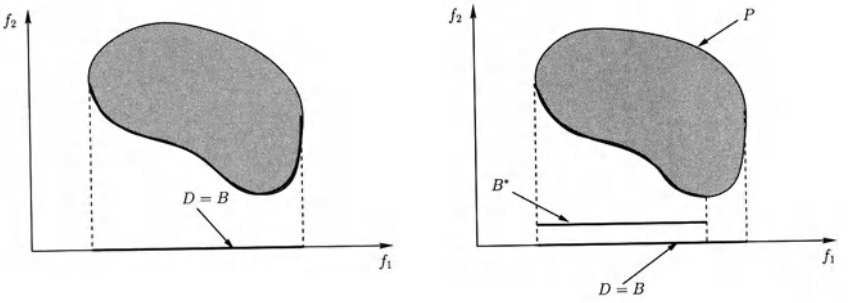
2.9.3 A concrete example

Let us take again our Schaffer's F2 test problem:

$$\begin{vmatrix} \text{minimize } f_1(x) = x^2 \\ \text{minimize } f_2(x) = (x-2)^2 \\ \text{with} \quad x \in [0,2] \end{vmatrix}$$

We choose the second objective function (f_2) as our main objective function. We change the other objective function into a parametric equality constraint. So we apply step 1. We have the following problem:

$$\begin{vmatrix} \text{minimize } f_2(x) = (x-2)^2 \\ \text{with} \quad f_1(x) = x^2 = \alpha \\ \text{and} \quad 0 \leq x \leq 2 \end{vmatrix}$$



(a) Before the application of a proposition.

(b) After the application of a proposition.

Fig. 2.20. Changes between B and B^* due to the application of a proposition.

We now determine D, $\phi(\alpha)$ and $\hat{x}(\alpha)$. These determinations correspond to step 2 of the method.

- To find D, we try to merge the two constraints into one. We notice that $x = \sqrt{\alpha}$. So, we deduce that $0 \leq \alpha \leq 4$. Finally, we have $D = \{\alpha \in \mathbb{R} \mid 0 \leq \alpha \leq 4\}$.
- To find $\phi(\alpha)$, we search for a relation of the type $f_2(x)$. We easily obtain this relation: $\phi(\alpha) = (\sqrt{\alpha} - 2)^2 = f_2(x)$ because $x = \sqrt{\alpha}$.
- To find $\hat{x}(\alpha)$, we search for a relation of the type $\hat{x} = f(\alpha)$ (where f is any function of the variable α). We easily obtain this relation: $\hat{x}(\alpha) = \sqrt{\alpha}$.

Now, we go to step 3. We determine the set $B = \{\alpha \in D \mid f_2(\hat{x}(\alpha)) = \phi(\alpha)\}$, where $B = \{\alpha \in D \mid f_2(\hat{x}(\alpha)) = (\sqrt{\alpha} - 2)^2\}$.

Now, we perform the last three steps all at once. To do so, we verify that $\frac{\partial \phi(\alpha)}{\partial \alpha} < 0$. We compute this derivative: $\frac{d\phi(\alpha)}{d\alpha} = -\frac{1}{2 \cdot \sqrt{\alpha}} < 0, \forall \alpha \in \mathbb{R}^{+*}$. Therefore, we do not remove any point from B. So, lastly, we have $B^* = B$.

We represent graphically the tradeoff surface. To do so, we choose values of α between 0 and 4 using a step of 1. We have the results listed in Table 2.6.

Table 2.6. Solutions for the Schaffer's F2 problem with the proper-equality-constraints method.

α	$\hat{x}(\alpha)$	$\phi(\alpha)$	$f_1(\hat{x}(\alpha))$	$f_2(\hat{x}(\alpha))$
0	0	4	0	4
1	1	1	1	1
2	$\sqrt{2}$	$(\sqrt{2}-2)^2$	2	$(\sqrt{2}-2)^2$
3	$\sqrt{3}$	$(\sqrt{3}-2)^2$	3	$(\sqrt{3}-2)^2$
4	2	0	4	0

The shape for the tradeoff surface is shown in Fig. 2.21.

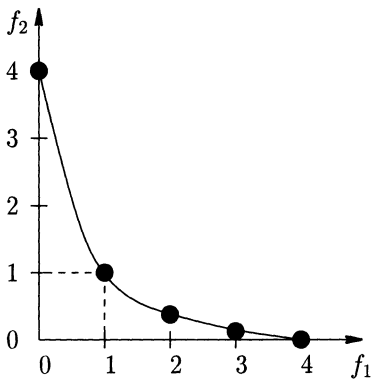


Fig. 2.21. Schaffer test problem solved using the proper-equality-constraints method.

2.9.4 Discussion

This method is an analytical method. It is not easy to implement. To do so, we must use two algorithms that will be presented in Sects. 2.11 and 2.12.

2.10 Proper-inequality-constraints method

2.10.1 Principle

This method was inspired very much by the preceding method and is presented in [Lin 77]. The main difference lies in the fact that it transforms the starting problem into a problem using inequality constraints rather than equality constraints.

2.10.2 Presentation of the method

We start from the P problem. We modify it the following way:

$$
\begin{array}{ll}
\text{minimize} & f_k(\vec{x}) \\
\text{with} & f_1(\vec{x}) \leq \alpha_1 \\
 & \cdots \\
 & f_{k-1}(\vec{x}) \leq \alpha_{k-1} \\
 & \vec{g}(\vec{x}) \leq 0 \\
\text{and} & \vec{h}(\vec{x}) = 0
\end{array}
$$

Let us introduce some notation. We define $z_i = f_i(\vec{x})$ and $\vec{\alpha} = (\alpha_1, \cdots, \alpha_{k-1})$. We also make the following definitions:

- $Z = \left\{ \vec{f}(\vec{x}) \mid \vec{g}(\vec{x}) \leq 0 \text{ and } \vec{h}(\vec{x}) = 0 \right\}$: this is the set of objective function values such that the constraints on $\vec{x}$ are satisfied.
- $\bar{Z}$ is the boundary of Z.
- $Z^{\alpha} = \left\{ \vec{z} \in \bar{Z} \mid z_i \leq \alpha_i \text{ with } i \neq k \right\}$: this is the set of objective function values such that they belong to the boundary of Z and such that $\vec{z} \leq \vec{\alpha}$.

Now, we define the sets A and P:

- $A = \{\overrightarrow{\alpha} \in \mathbb{R}^{k-1} \mid \overline{Z^{\alpha}} \text{ is not empty}\}$.
- $A(\overrightarrow{\alpha_0}) = \{\overrightarrow{\alpha} \in A \mid \overrightarrow{\alpha} \geq \overrightarrow{\alpha_0}\}$.
- $\psi(\overrightarrow{\alpha}) = \inf\{z_k \mid \overrightarrow{z} \in \overline{Z^{\alpha}}\}$: this is the minimum value of the objective function $f_k(\overrightarrow{x})$ when the other objective functions $(f_1(\overrightarrow{x}), \cdots, f_{k-1}(\overrightarrow{x}))$ have values on the boundary of Z^{α}.
- $P = \{(\overrightarrow{\alpha}, \psi(\overrightarrow{\alpha})) \mid \overrightarrow{\alpha} \in A\}$.

Once we have determined the set P, we just have to determine the set D which contains solutions of our problem. To do so, we take values from the set P and test whether they are supremum values. We have some criteria available to perform this test.

Theorem 2. ***Global quasi-supremality***
$(\overrightarrow{\alpha_0}, \psi(\overrightarrow{\alpha_0})) \in D$ if and only if $\psi(\overrightarrow{\alpha}) \neq \psi(\overrightarrow{\alpha_0})$ for all $\overrightarrow{\alpha} \in A$ such that $\overrightarrow{\alpha} \geq \overrightarrow{\alpha_0}$.

Theorem 3. ***Local quasi-supremality***
$(\overrightarrow{\alpha_0}, \psi(\overrightarrow{\alpha_0})) \in D$ if and only if, for a neighborhood $N(\overrightarrow{\alpha_0}, \delta)$, $\psi(\overrightarrow{\alpha}) \neq \psi(\overrightarrow{\alpha_0})$ for all $\overrightarrow{\alpha} \in N(\overrightarrow{\alpha_0}, \delta) \bigcap A(\overrightarrow{\alpha_0})$ such that $\overrightarrow{\alpha} \neq \overrightarrow{\alpha_0}$.

2.10.3 Discussion

This method has been used extensively and has given birth to some algorithms, which we shall study in Sects. 2.11 and 2.12.

2.11 Lin–Tabak algorithm

2.11.1 Principle

This algorithm [Tabak 79] is based on the preceding method.

2.11.2 Presentation of the method

We start from the P problem. We perform the following transformation:

$$\left|\begin{array}{ll} \text{minimize} & f_k(\overrightarrow{x}) \\ \text{with} & f_1(\overrightarrow{x}) \leq \varepsilon_1, \cdots, f_{k-1}(\overrightarrow{x}) \leq \varepsilon_{k-1} \\ & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\ \text{and} & \overrightarrow{h}(\overrightarrow{x}) = 0 \end{array}\right.$$

We define $\phi(\overrightarrow{\varepsilon}) = f_k(\overrightarrow{x_0}(\overrightarrow{\varepsilon}))$, where $\overrightarrow{\varepsilon} = [\varepsilon_1, \cdots, \varepsilon_{k-1}]^t$.
This algorithm is divided into five steps:

Step 1: Solve the problem with an initial value $\overrightarrow{\varepsilon^{(0)}}$, which gives us a global optimum $\phi\left(\overrightarrow{\varepsilon^{(0)}}\right) = \phi^{(0)}$.

Step 2: Make the modification $\varepsilon_1^{(0)} \rightarrow \varepsilon_1^{(0)} + \Delta\varepsilon_1 = \varepsilon_1^{(1)}$ with $\Delta\varepsilon_1 > 0$ and solve the problem again. When this is done, we obtain a global optimum $\phi^{(1)}$.

Step 3: Perform the following comparisons:

1. If $\phi^{(1)} = \phi^{(0)}$ then we must delete $\varepsilon_1^{(0)}$ because it is not proper.
2. If $\phi^{(1)} \neq \phi^{(0)}$ then we accept $\varepsilon_1^{(0)}$ and $\varepsilon_1^{(1)}$. In fact, we must have $\phi^{(1)} < \phi^{(0)}$. If we have a "greater than" inequality, then the solution of the preceding problem was merely local.

Step 4: Repeat steps 2 and 3 with increasing values of ε_1 until a boundary of the proper area in the ε_1 direction is reached. This condition will be clearly apparent from the outcome of part 1 of step 3.

Step 5: Repeat steps 2 to 4 for another ε_i, $i = \{2, \cdots, k-1\}$.

2.12 Lin–Giesy algorithm

2.12.1 Principle

This method [Giesy 78] is also based on the proper-inequality-constraints method. There exists a theorem which guarantees, in some cases, the convergence of the algorithm.

2.12.2 Presentation of the method

Let us introduce, first, some notation:

- We denote by $\overrightarrow{\varepsilon^{(0)}}$ the choice of the initial value of $\vec{\varepsilon}$, called the "satisfiability threshold". This value must be given by the decision maker. In this algorithm, the vector $\vec{\varepsilon}$ is of dimension k.
- We denote by $\overrightarrow{\varepsilon^{(i)}}$ the value of the vector $\vec{\varepsilon}$ reached at iteration i ($i = \{1, \cdots, k\}$). Here, k takes the same value as the above variable k.
- We denote by σ a permutation of the set $\{1, 2, \cdots, k\}$. For example, if $A = \{1, 2, 3\}$ then we can take $\sigma = \{2, 1, 3\}$, and $\sigma_1 = 2$, $\sigma_2 = 1$ and $\sigma_3 = 3$.
- At iteration i, we have the following description of the algorithm:

Step i_1: Solve the problem

$$\left|\begin{array}{ll} \text{minimize} & f_{\sigma_i}(\vec{x}) \\ \text{with} & f_j(\vec{x}) \leq \varepsilon_j,\ j = \{1, \cdots, k\},\ j \neq \sigma_i \\ & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{array}\right.$$

We denote the solution by $\overrightarrow{x^{(i)}}$. For iteration 1 only, if there are no solutions or if $f_{\sigma_1}\left(\overrightarrow{x^{(1)}}\right) \geq \varepsilon_{\sigma_1}^{(0)}$, then a less restrictive choice for $\overrightarrow{\varepsilon^{(0)}}$ must be made.

Step i_2: We define $\overrightarrow{\varepsilon^{(i)}} = \vec{f}\left(\overrightarrow{x^{(i)}}\right)$.

We state the following theorem about the convergence of the method.

Theorem 4. *Convergence*
If the algorithm finds a solution at iteration i_1 then it will find a result at each of k iterations, $\overrightarrow{x^{(k)}}$ is the optimal solution in the Pareto sense and $\overrightarrow{f}\left(\overrightarrow{x^{(k)}}\right) \leq \overrightarrow{\varepsilon^{(0)}}$.

2.13 Annotated bibliography

[Azarm 96] A website which presents the best-known multiobjective optimization methods, such as the weighted-sum-of-objective-functions method. The presentation of these methods is really "user" oriented. Few details are given about the mathematics but, on the other hand, full details are given about how to use the method to approximate the tradeoff surface.

[Ehrgott 00] A book which presents multiobjective optimization methods. A high level of mathematical knowledge is required to read this book. It is the reference book about the mathematical properties of the various multiobjective optimization methods.

3

Interactive methods

3.1 Introduction

Interactive methods allow the user to find one and only one solution. They belong to the family of progressive methods and allow the decision maker to fine-tune his/her preferences with respect to a tradeoff between objective functions during the running of the optimization method. These methods are to be compared with the following methods:

- A priori preference methods, where the decision maker chooses the tradeoff to be operated between objective functions before the optimization method is run.
- A posteriori preference methods, where the decision maker does not choose any tradeoff before the optimization method is run. The optimization method computes all the Pareto optimal solutions, and the decision maker can perform comparisons between these solutions and choose one.

We shall now present some interactive methods (or, rather, some principles of interactive methods).

3.2 Surrogate-worth tradeoff method

3.2.1 Principle

The surrogate-worth tradeoff (SWT) method has been used to solve a water resources optimization problem [Haimes et al. 75]. It is based on the compromise method, to which we add an interactive process to allow the method to converge to a solution which has a good chance of satisfying the decision maker.

3.2.2 Presentation of the method

We start from the P problem. This method is divided into seven steps:

Step 1: Find the minimum of the function f_j by solving

$$\left|\begin{array}{ll} \text{minimize} & f_j(\vec{x}) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{array}\right.$$

The solution of this problem becomes the jth coordinate of the vector $\overrightarrow{f}_{min}$, which gathers together the $k-1$ minimum values of f_j, $j=\{2,\cdots,k\}$. If possible, we search during this step for the coordinates of the vector $\overrightarrow{f}_{max}$, which gathers together the $k-1$ maximum values of f_j, $j=\{2,\cdots,k\}$, also.

Step 2: Choose the starting value of $\varepsilon_j \geq f_{j_{min}}$, $j=\{2,\cdots,k\}$.

Step 3: Solve the following problem:

$$\begin{array}{ll} \text{minimize} & f_1(\overrightarrow{x}) \\ \text{with} & f_2(\overrightarrow{x}) \leq \varepsilon_2, \cdots, f_k(\overrightarrow{x}) \leq \varepsilon_k \\ & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\ \text{and} & \overrightarrow{h}(\overrightarrow{x}) = 0 \end{array}$$

If some of the constraints of this problem cannot be respected, we set $\varepsilon_j = f_j(\overrightarrow{x})$ when j corresponds to the index of a constraint that is not respected (we loosen the constraints). We must start this step again. If the constraints are respected, we call $\overrightarrow{x}^*$ the solution vector and we start the next step.

Step 4: If the constraints are sufficiently loose, we go to step 5; otherwise, we choose a new value for $\varepsilon_j \geq f_{j_{min}}$, for all $j=\{2,\cdots,k\}$ and return to step 3. A method of selecting a new value for ε is to choose a very large value for ε and then to decrease each ε_j, for all $j=\{2,\cdots,k\}$, using a certain $\Delta_j > 0$, each time a constraint is not respected. If the constraints are not respected, we set $\varepsilon_j = f_j(\overrightarrow{x})$ where j is each index of the constraint that is not respected (here, $\overrightarrow{x}$ corresponds to the current value of the solution).

Step 5: Now, we ask the decision maker to choose the coefficients W_{1j}, for all $j=\{2,\cdots,k\}$. These coefficients represent the opinion of the decision maker about an increase of the function f_j by λ_{1j} units (the way we compute λ_{1j} is presented below). The cost is measured on a scale of -10 to $+10$, where -10 corresponds to an unfavorable opinion about this increase and $+10$ corresponds to a favorable opinion. 0 corresponds to indifference. This operation is repeated for j varying from 2 to k.

Step 6: We repeat step 5 until we find null values for the coefficients W_{1j}, $j=\{2,\cdots,k\}$. We denote by f_j^*, for all $j=\{2,\cdots,k\}$, the values of the objective functions corresponding to these coefficients.

Step 7: The preferred solution vector $\overrightarrow{x}^*$ is determined by solving the following problem:

$$\begin{array}{ll} \text{minimize} & f_1(\overrightarrow{x}) \\ \text{with} & f_2(\overrightarrow{x}) = f_2^*, \cdots, f_k(\overrightarrow{x}) = f_k^* \\ & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\ \text{and} & \overrightarrow{h}(\overrightarrow{x}) = 0 \end{array}$$

The most difficult thing in this method is to clearly understand the significance of the coefficient W_{1j}. The following is a process for evaluating W_{1j}.

We start from the following point:

- $f_2(\overrightarrow{x})$,
- $f_3(\overrightarrow{x})$ and

- $f_1(\vec{x})$.

all expressed in their own units. We then ask the following questions to the decision maker:

For W_{12}: At this point, would you be ready to increase f_2 by λ_{12} units of f_2 to decrease f_1 by 1 unit of f_1? Give your answer on a scale of -10 (do not agree) to $+10$ (agree).

For W_{13}: At this point, would you be ready to increase f_3 by λ_{13} units of f_3 to decrease f_1 by 1 unit of f_1? Give your answer on a scale of -10 (do not agree) to $+10$ (agree).

This method allows one, right from the start, to find an area of satisfaction (between two zeros of W_{12} or of W_{13}, or between a zero of W_{12} and a zero of W_{13}; see Figs 3.1 and 3.2). Once we have found an area of satisfaction, we can solve our problem using the compromise method, while restricting the values we use to the area of satisfaction.

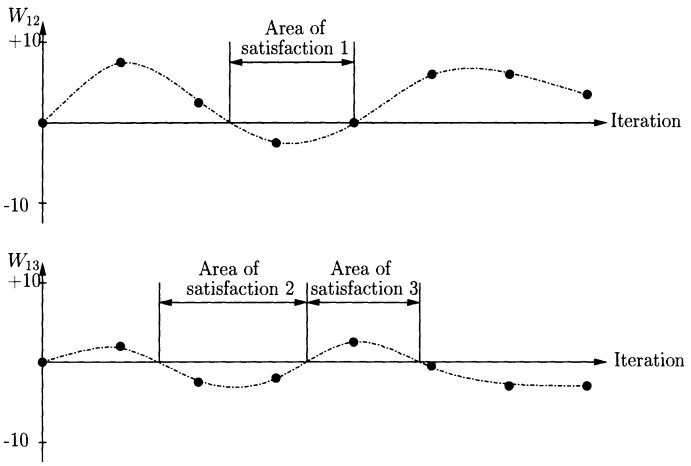


Fig. 3.1. Areas of satisfaction for the SWT method – example 1.

The values of the coefficient λ_{ij} are computed in the following way:

$$\lambda_{ij} = -\frac{\partial f_i}{\partial f_j} \tag{3.1}$$

In our case, we just compute

$$\lambda_{1j} = -\frac{\partial f_1}{\partial f_j} \tag{3.2}$$

because we always perform our comparisons relative to objective function f_1.

3.2.3 Discussion

In this method, it is not always easy to know how much we are willing to pay for an increase in the value of an objective function. In most cases, we may be tempted to answer more or less at random. This kind of answer can prevent the method from finding the optimum solution or the set of optimum solutions.

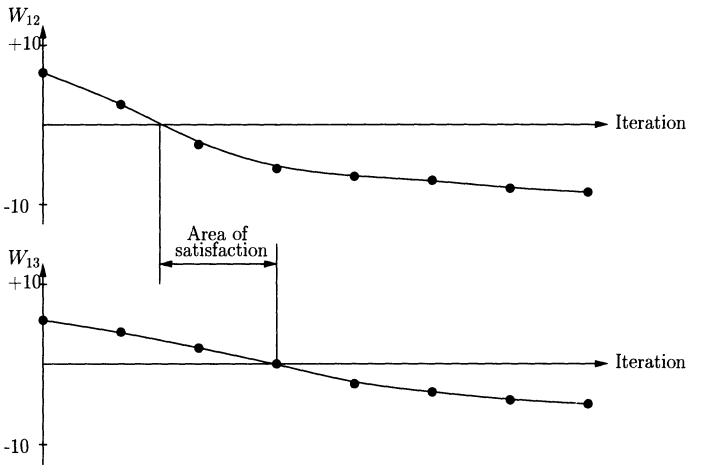


Fig. 3.2. Areas of satisfaction for the SWT method – example 2.

Moreover, this method is not applicable to problems for which the objective functions do not have any derivatives (owing to the coefficients λ_{1j}).

3.3 Fandel method

3.3.1 Principle

The aim of this method is to help the decision maker in the choice of the weights [Eschenauer et al. 90].

3.3.2 Presentation of the method

We start from the P problem. We modify this problem the following way:

$$\begin{vmatrix} \text{minimize} & \sum_{i=1}^{k} w_i \cdot f_i(\vec{x}) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{vmatrix}$$

We also have $w_i \geq 0$ and $\sum_{i=1}^{k} w_i = 1$. We find this condition in the weighted-sum-of-objective-functions method (see Sect. 2.1.2).

As with the weighted sum of objective functions, variation of the coefficients w_i allows us to determine points on the tradeoff surface. Nevertheless, in the Fandel method, we suppose that these coefficients are unknown. We also suppose that the decision maker is looking for a solution which is close to the ideal solution (the ideal solution is defined below).

To obtain these coefficients, we proceed as follows:

- In the first step, we compute the minimum value for each of the objective functions (which we denote by $\overline{f}_j$) while respecting the constraints. This is equivalent to solving the following problem:

$$\begin{vmatrix} \text{minimize} & f_j(\vec{x}) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{vmatrix}$$

with $j = \{1, \cdots, k\}$. We denote by $\vec{x}_j^*$ the solution of this problem. We denote by $\vec{f}^{*j} = \vec{f}\left(\vec{x}_j^*\right)$ the value of the vector of objective functions which corresponds to this solution. For $j = 4$, this vector has the following coordinates:

$$\vec{f}^{*4} = \left[f_1\left(\vec{x}_4^*\right), \cdots, f_3\left(\vec{x}_4^*\right), \bar{f}_4, f_5\left(\vec{x}_4^*\right), \cdots, f_k\left(\vec{x}_4^*\right)\right]$$

- Next, we construct the vector of the values of the ideal objective function:

$$\vec{\bar{f}} = \left(\bar{f}_1, \cdots, \bar{f}_k\right)^t \tag{3.3}$$

We can also construct the matrix B:

$$B = \left(\vec{f}^{*1}, \cdots, \vec{f}^{*k}\right)^t \tag{3.4}$$

This matrix contains the values of $\bar{f}_j$ on its diagonal. For a problem with two objective functions, we have the situation sketched in Fig. 3.3.

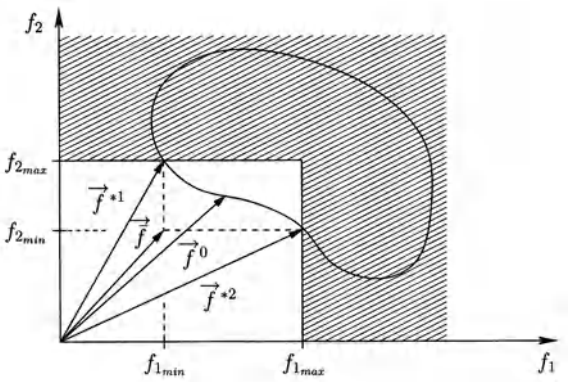


Fig. 3.3. The Fandel method (step 1).

The vector $\vec{\bar{f}}$ represents an "ideal" objective for the decision maker which we cannot reach in reality. This vector is the goal the decision maker wants to reach (the decision maker is ready to do everything necessary to reach this objective). Figure 3.3 illustrates the notation introduced above. In this figure, the gray area represents the part of the search domain which is not accessible owing to the constraints. The variables $f_{1_{min}}$, $f_{1_{max}}$, $f_{2_{min}}$ and $f_{2_{max}}$ bound a domain of variation for the objective functions f_1 and f_2 in the feasible area.
At step M, the user will obtain a solution which will be close to the ideal solution. Here, we compute a vector $\vec{f}^M$ which will serve to reduce the size of the search space. This vector is computed in the following way:

$$\vec{f}^{M} = \frac{1}{k} \cdot \sum_{i=1}^{k} \vec{f}^{*i} \tag{3.5}$$

- Now, we reduce the size of the search space by using the bounds of the constraints. We call our new space of constraints $\hat{Y}^M$:

$$\hat{Y}^M = \left\{ \vec{f}\,(\vec{x}) \in \mathbb{R}^k \mid \vec{g}\,(\vec{x}) \leq 0,\ \vec{h}\,(\vec{x}) = 0 \text{ and } \vec{f}\,(\vec{x}) \leq \vec{f}^{M} \right\} \tag{3.6}$$

- In this new constraint space, we minimize our objective functions:

$$\begin{array}{ll} \text{minimize} & f_j\,(\vec{x}),\ j = \{1, \cdots, k\} \\ \text{with} & \vec{g}\,(\vec{x}) \leq 0 \\ & \vec{h}\,(\vec{x}) = 0 \\ \text{and} & \vec{f}\,(\vec{x}) \leq \vec{f}^{M} \end{array}$$

  We denote by $\hat{f}^{*j}$ the solution of this problem and denote by $\vec{\hat{x}}^{*j}$ the corresponding decision vector.
- As before, we construct the matrix $\hat{B}$:

$$\hat{B} = \left(\vec{\hat{f}}^{*1}, \cdots, \vec{\hat{f}}^{*k} \right)^t \tag{3.7}$$

  where $\vec{\hat{f}}^{*k}$ corresponds to the vector $\vec{f}\left(\vec{\hat{x}}^{*k}\right)$. This matrix $\hat{B}$ defines a hyperplane (a line in the case of a problem with two objective functions), which has the following equation:

$$\hat{B} \cdot \vec{a} = c \cdot \vec{e} \tag{3.8}$$

  where $\vec{a} = (a_1, \cdots, a_k)^t$,
  $\vec{a} > 0, \sum_{i=1}^{k} a_j = 1$,
  c = constant,
  $\vec{e} = (1, \cdots, 1)^t$

The vector $\vec{a}$ and the constant c are the unknowns of the equation system. The goal is to find a c value such that the hyperplane defined by the preceding equation is tangential to the space of values of the objective function. The product $\hat{B} \cdot \vec{a}$ with the constraints given above corresponds to a convex combination of vectors $\vec{\hat{f}}^{*i}$. So, this product gives a hyperplane which passes through the extremities of the vectors $\vec{\hat{f}}^{*i}$.

The behavior of this step is illustrated in Fig. 3.4. In this figure, we can see both of the vectors $\vec{\hat{f}}^{*1}$ and $\vec{\hat{f}}^{*2}$, which correspond to vectors computed during the solution of the f^* problem. The vector $\vec{f}^{M}$ corresponds to the bounds of the feasible domain. These bounds are given by the decision maker.

Example 1

The following is a solution of the preceding problem:

$\vec{\hat{f}}^{*1} = (1, 0)$

$\vec{\hat{f}}^{*2} = (0, 2)$

$\hat{B} \cdot \vec{a} = \begin{pmatrix} 0 & 2 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ 1 - a_1 \end{pmatrix} = \begin{pmatrix} c \\ c \end{pmatrix}$

So we have:

$2 - 2 \cdot a_1 = c = a_1$

$a_1 = \frac{2}{3}$

Example 2:

The following is another solution of the preceding problem:

$\vec{\hat{f}}^{*1} = (2, 3)$

$\vec{\hat{f}}^{*2} = (4, 1)$

$\hat{B} \cdot \vec{a} = \begin{pmatrix} 2 & 4 \\ 3 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ 1 - a_1 \end{pmatrix} = \begin{pmatrix} c \\ c \end{pmatrix}$

So we have:

$2 \cdot a_1 + 4 - 4 \cdot a_1 = c = 3 \cdot a_1 + 1 - a_1$

$-4 \cdot a_1 + 3 = 0$

$a_1 = \frac{3}{4}$

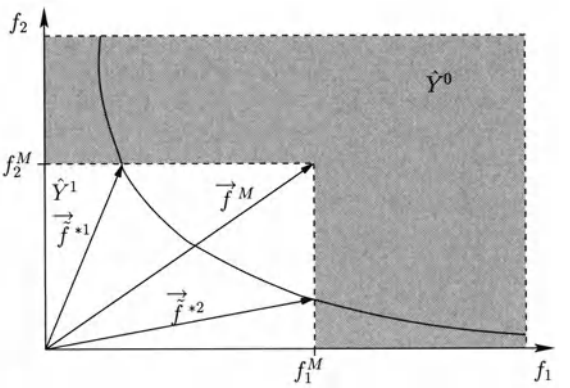


Fig. 3.4. The Fandel method (step 2).

To determine the weights of a desired tradeoff solution inside the subspace $\hat{Y}^{\mathcal{M}}$, the hyperplane (a line if we consider a problem in a space of dimension 2) is translated toward the solution optimal in the Pareto sense, while minimizing the constant c until the hyperplane is tangential to the tradeoff surface. This process is illustrated in Fig. 3.5. In this figure, after having computed the vectors $\vec{\hat{f}}^{*1}$ and $\vec{\hat{f}}^{*2}$, we determine the equation of the hyperplane (B) and then search for a parallel hyperplane which is tangential to the feasible domain. The tangency point then defines a new solution, denoted by $\vec{\tilde{f}}$.

This linear optimization problem gives us a vector $\vec{\tilde{f}}$, which corresponds to the vector of weights $\vec{w}$. The solution of our substitution problem can then be determined.

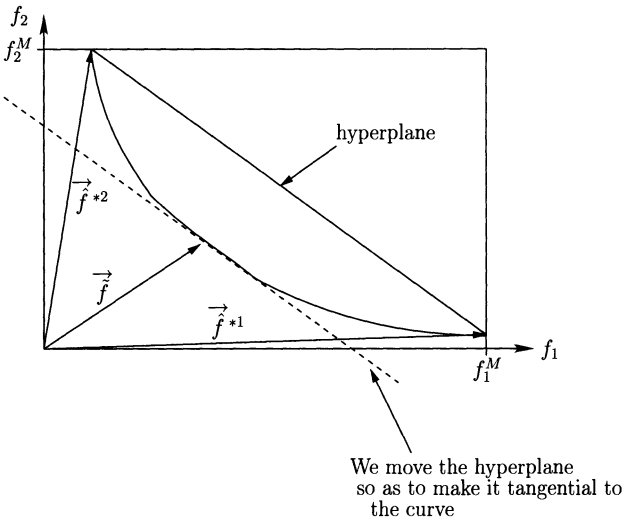


Fig. 3.5. The Fandel method (step 3).

Either the decision maker accepts this solution, or he/she starts a new iteration while fine-tuning the search domain. This can be done by fixing maximum bounds of some constraints on the objective functions (m objective functions for example). Let us denote by $\overline{Y}_i$ the higher bounds of these m selected objective functions so as to restrict the search space. Individual minima of our new subspace $\hat{Y}$ are determined by solving the following problem:

$$\begin{array}{|ll} \text{minimize} & f_j(\vec{x}),\ j = \{1, \cdots, k\} \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ & \vec{h}(\vec{x}) = 0 \\ \text{and} & f_i(\vec{x}) \leq \overline{Y}_i \end{array}$$

The indices i and j correspond to all the objective functions selected so as to restrict the search space.

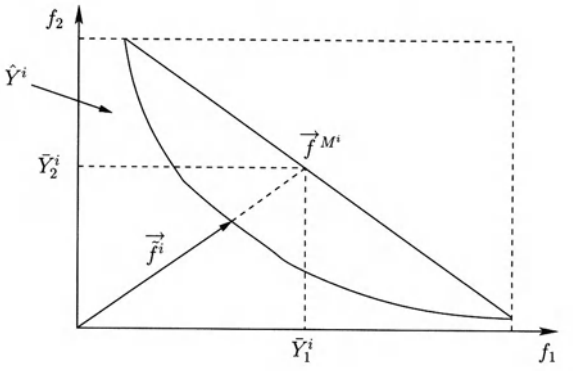
We have $\hat{Y}^{i+1} = \left\{ \vec{f}(\vec{x}) \in \mathbb{R}^k \mid \vec{g}(\vec{x}) \leq 0,\ \vec{h}(\vec{x}) = 0,\ f_j(\vec{x}) \leq \overline{Y}^j \right\}$.

The iteration begins with the computation of a new central point $\vec{f}^{M^{k+1}}$ (see Fig. 3.6).

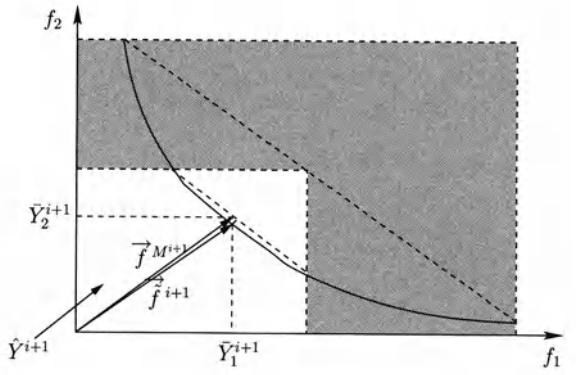
In Fig. 3.6a, we can see iteration k of the Fandel method. A search space $\hat{Y}^k$ has been determined. We construct the median vector $\overrightarrow{f^{Mk}}$ using the formula 3.5, and then we search for a solution $\vec{f}^{\,k}$ in the direction of the median vector.

In Fig. 3.6b, we can see iteration $k+1$ of the Fandel method. After having computed the median vector $\overrightarrow{f^{Mk}}$, we define a new search space by using the median vector. Next, we compute a new median vector $\overrightarrow{f^{Mk+1}}$, which defines a new search space $\hat{Y}^{k+1}$. After that, we perform a search for a solution in the direction of the median vector, which will give us the vector $\vec{f}^{\,k+1}$.

We see that, as we repeat the steps, we converge to a point which, normally, will satisfy the decision maker.



(a) Iteration k.



(b) Iteration $k+1$.

Fig. 3.6. The Fandel method (step 4).

3.3.3 Discussion

The main drawback of this method, which is common to many interactive methods, is that we can find only one solution and we cannot approximate a part of the tradeoff surface. We can also see that this method uses the hypothesis that the solution vector which will satisfy the decision maker is the one closest to the ideal solution. This is a strong hypothesis and does not fit the preferences of every user. Lastly, the method used to perform the interaction (reducing the search space by fine-tuning of the bounds of some constraints) requires the decision maker to have a deep a priori knowledge of the problem if he/she is to use the method efficiently. The decision maker must be able to find good bounds so as to restrict the search space of the problem.

3.4 STEP method

3.4.1 Principle

This method is similar to the Fandel method. Here, also, information about the preferences of the decision maker allows the search space to be restricted step by step. [Eschenauer et al. 90].

3.4.2 Presentation of the method

We start from the P problem. We transform this problem in the following way:

$$\begin{array}{ll} \text{minimize } \beta & \\ \text{with} & \left[\vec{f}\,(\vec{x}) - \vec{\vec{Y}}\right]^t \cdot \vec{w} < \beta \\ & \vec{g}\,(\vec{x}) \leq 0 \\ & \vec{h}\,(\vec{x}) = 0 \\ \text{and} & \vec{f}\,(\vec{x}) \leq \vec{\vec{Y}} \end{array}$$

where the vector $\vec{\vec{Y}}$ corresponds to a vector which contains the upper boundary as its coordinates and is used to restrict the search space. The vector $\vec{w}$ is defined below in this section.

As with the preceding approach, we build the matrix B.

The weights w_j of the various objective functions f_j are needed to solve the problem. We determine these weights. We give heavier weights to the objective functions f_j for which the excursion is large (f_j takes values inside a wide domain). The coefficients w_j have the following form:

$$w_j = \frac{v_j}{\sum_{i=1}^{k} v_i} \tag{3.9}$$

where

$$v_j = \frac{\max_i \left\{f_j^{*i}\right\} - f_j^{*j}}{\min_i \left\{f_j^{*i}\right\}} \cdot \frac{1}{f_j^{*j}} \tag{3.10}$$

with $i = 1, \cdots, k$; f_j^{*i} has the same meaning as in page 82. Hence the weights respect the conditions $\vec{w} > 0$ and $\sum_{j=1}^{k} w_j = 1$.

Let us analyze, now the behavior of the equations by which v_j and w_j are computed through an example.

Example

At iteration 10, we obtain the values of five objective functions as listed in Table 3.1.

As we can see from these results, the STEP method gives heavier weights to the objective functions which have the larger differences between the minimum of

Table 3.1. The values of the five objective functions at iteration 10 in our example.

f^{*1}	$\{1,4,3,6,2\}$
f^{*2}	$\{4,1,4,3,4\}$
f^{*3}	$\{3,2,1,2,3\}$
f^{*4}	$\{6,3,6,2,3\}$
f^{*5}	$\{7,5,3,2,1\}$

Table 3.2. Results corresponding to the values in Table 3.1.

v_1	$v_1 = \frac{7-1}{1} \cdot \frac{1}{1} = 6$	w_1	$\frac{6}{19} = 0.315$
v_2	$v_2 = \frac{5-1}{1} \cdot \frac{1}{1} = 4$	w_2	$\frac{4}{19} = 0.210$
v_3	$v_3 = \frac{6-1}{1} \cdot \frac{1}{1} = 5$	w_3	$\frac{5}{19} = 0.263$
v_4	$v_4 = \frac{6-2}{2} \cdot \frac{1}{2} = 1$	w_4	$\frac{1}{19} = 0.052$
v_5	$v_5 = \frac{4-1}{1} \cdot \frac{1}{1} = 3$	w_5	$\frac{3}{19} = 0.158$

the objective function f^{*i} and the maximum of that objective function f^{*i}. In this example, we have this behavior for the objective function f^{*1}.

We notice also that the way we compute the weights does not work with objective functions which can have null values.

We notice that, compared with the Fandel method, the weights have less importance, because interaction with the decision maker is done via the choice of the high bound $\overrightarrow{Y}$ and not through the choice of weights. Moreover, this method gives smaller weights to objective functions which have high values.

Once we have determined the weights, we compute a first solution $\overrightarrow{\tilde{f}}^0$ in the following way:

$$\begin{array}{ll} \text{minimize} & \overrightarrow{f}(\overrightarrow{x})^t \cdot \overrightarrow{w} \\ \text{with} & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\ \text{and} & \overrightarrow{h}(\overrightarrow{x}) = 0 \end{array}$$

The decision maker can now compare solution $\overrightarrow{\tilde{f}}^0$ with the ideal solution. If some of the coordinates of the vector of objective functions are insufficiently satisfactory, the decision maker can loosen some constraints in the following way:

$$\overline{Y_j} = \tilde{f}_j + \Delta f_j \tag{3.11}$$

with $j \in \{1, \cdots, k\}$. Otherwise, the preferred solution of the decision maker is given by $\overrightarrow{\tilde{f}}$.

Thus, we have obtained a new tradeoff solution. The weight w_j of the jth objective function has been set to zero. The problem now takes the following form:

$$\begin{array}{ll} \text{minimize} & \left[\overrightarrow{f}(\overrightarrow{x}) - \overrightarrow{\overline{Y}}\right]^t \cdot \overrightarrow{w} \\ \text{with} & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\ & \overrightarrow{h}(\overrightarrow{x}) = 0 \\ \text{and} & \overrightarrow{f}(\overrightarrow{x}) \leq \overrightarrow{\overline{Y}} \end{array}$$

with $w_j = 0$ and $j \in \{1, \cdots, k\}$.

So, the solution is restricted to the $\hat{Y}$ search subspace and has the least deviation with respect to the ideal vector of objective functions $\overrightarrow{f}$.

In the case of a problem with two objective functions, the extremity of the vector $\overrightarrow{\tilde{f}}$ is located at the intersection between the boundary of the domain $\hat{Y}$ and the line of slope $\frac{w_2}{w_1}$ which goes through the ideal point $\overrightarrow{f}$ (see in Fig. 3.7).

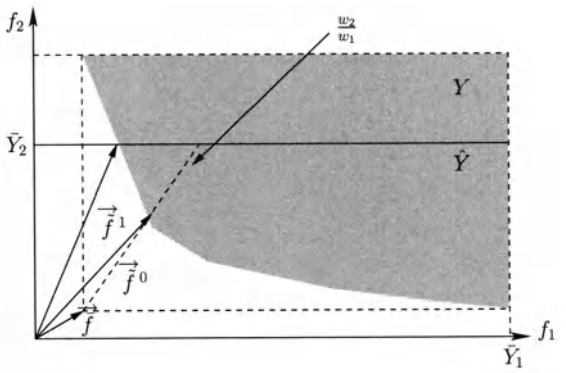


Fig. 3.7. The STEP method.

At the end of this step, if the decision maker accepts this solution, the algorithm stops there. Otherwise, the decision maker must start again.

3.4.3 Discussion

We can make the same remarks as in Sect. 3.3.3, about the Fandel method. Moreover, this method needs to work with objective functions which do not have null values. This last remark limits the field of application of this method.

3.5 Jahn method

3.5.1 Principle

This method is completely different from the last two others. We begin by choosing a starting point and then the problem is solved step by step through the search space toward the optimal solution in the Pareto sense [Eschenauer et al. 90]. This method is based on a cyclic minimization method (also called the "scalar optimization method toward a feasible direction" by Zoutendjik).

3.5.2 Presentation of the method

We start from the P problem. We transform this problem in the following way:

$$
\begin{array}{ll}
\text{minimize } \beta & \\
\text{with} & w_j \cdot \left[\nabla_{\vec{x}} f_j^u\right]^t \cdot \overrightarrow{\delta_f} \leq \beta \\
& v_i \cdot \left[\nabla_{\vec{x}} g_i^u\right]^t \cdot \overrightarrow{\delta_g} \leq \beta \\
\text{and} & t_l \cdot \left[\nabla_{\vec{x}} h_l^u\right]^t \cdot \overrightarrow{\delta_h} = \beta
\end{array}
$$

We also have $\vec{\delta_f} \in \mathbb{R}^k$, $\vec{\delta_g} \in \mathbb{R}^m$, $\vec{\delta_h} \in \mathbb{R}^p$, with

$i = \{1, \cdots, m\}$
$j = \{1, \cdots, k\}$
$l = \{1, \cdots, p\}$
$n = \{1, \cdots, k\}$

We have used the following notation:

β:	preference scalar function.
$\nabla_{\vec{x}}$:	nabla operator with respect to $\vec{x}$ ($\nabla_{\vec{x}} A = \sum_i \frac{\partial A}{\partial x_i} \cdot \vec{x}_i$).
$\vec{\delta}$:	direction of the minimization step.
w_j:	weight coefficient of the jth objective function.
v_i:	weight coefficient of the ith inequality constraint.
t_l:	weight coefficient of the lth equality constraint.

The solution of our new problem begins with the choice of a feasible starting vector $\vec{x}^0$ and by the computation of the corresponding vector of objective functions $\vec{f}(\vec{x}^0)$. The choice of the starting point influences the future possible alternatives, if we suppose the search space $\hat{Y}^0$ is restricted to

$$\hat{Y}^0 = \left\{ \vec{f}(\vec{x}) \mid \vec{f}(\vec{x}) \leq \vec{f}(\vec{x}^0) \right\} \tag{3.12}$$

We have given in Fig. 3.8 a sketch which explains this mechanism. In this figure, we can see that the choice of a feasible starting point $\vec{f}(\vec{x}^0)$ allows one to restrict the search space. So, we obtain the feasible domain $\hat{Y}^0$.

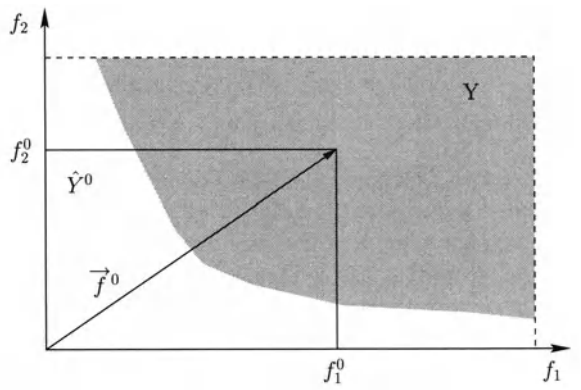


Fig. 3.8. Initialization of the Jahn method.

Depending on the objective functions vector $\vec{f}(\vec{x})$, the decision maker chooses one objective function f_i, which must be minimized as a priority. To do this, a step of direction $\vec{\delta}$ and of appropriate size λ must be determined, and it must satisfy the following relation:

$$\vec{x^{k+1}} = \vec{x^k} + \lambda^k \cdot \vec{\delta^k} \tag{3.13}$$

We have used the following notation:

$\overrightarrow{x^{k+1}}$: working variable at iteration $k+1$.
$\overrightarrow{x^k}$: working variable at iteration k.
λ^k: length of the minimization step.

$\overrightarrow{x^{k+1}}$ and $\overrightarrow{x^k}$ must satisfy the constraints of the problem ($\overrightarrow{g}$ and $\overrightarrow{h}$).

Moreover, these points must satisfy:

$$f_i^{k+1} < f_i^k \; \forall i \in I = \{1, \cdots, k\}$$
$$f_j^{k+1} \leq f_j^k \; \forall j \in J = \{I \mid j \neq i\}$$

These two relations show that the value of the computed objective function at iteration $k+1$ must dominate the value of the computed objective function at iteration k.

We obtain the direction of our step by solving our modified problem (the transformed problem presented at the beginning). When this is done, the weight coefficients w_j, v_i and t_l chosen by the decision maker influence the direction of the step. The decision maker can now choose the length of the step λ^k in an interval $[0, \lambda_{max}]$. We find λ_{max} by solving the following problem:

$$\begin{array}{ll}
\text{maximize } \lambda_{max} & \\
\text{with} & f_i\left(\overrightarrow{x^k} + \lambda_{max} \cdot \overrightarrow{\delta^k}\right) < f_i\left(\overrightarrow{x^k}\right) \; \forall i \in I = \{1, \cdots, k\} \\
& f_j\left(\overrightarrow{x^k} + \lambda_{max} \cdot \overrightarrow{\delta^k}\right) \leq f_j\left(\overrightarrow{x^k}\right) \; \forall j \in J = \{I \mid j \neq i\} \\
& \overrightarrow{g}\left(\overrightarrow{x^k} + \lambda_{max} \cdot \overrightarrow{\delta^k}\right) \leq 0 \\
\text{and} & \overrightarrow{h}\left(\overrightarrow{x^k} + \lambda_{max} \cdot \overrightarrow{\delta^k}\right) = 0
\end{array}$$

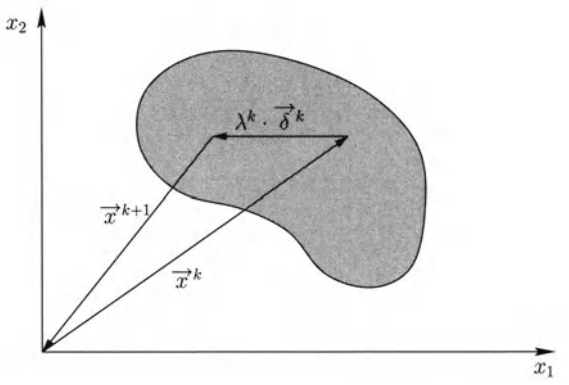
The new objective function vector $\overrightarrow{f^{k+1}}$, for a step of the given length and direction, is now computed ($\overrightarrow{f^{k+1}} = \overrightarrow{f}\left(\overrightarrow{x^k} + \lambda^k \cdot \overrightarrow{\delta^k}\right)$). If this objective functions vector is not accepted, we determine at the next iteration a new direction by solving our modified problem.

Variation of the direction of the step, which is done in accordance with the preferences of the decision maker, gives us minimization steps inside the search space Y, in the direction of the optimal solution in the Pareto sense $\overrightarrow{f}$.

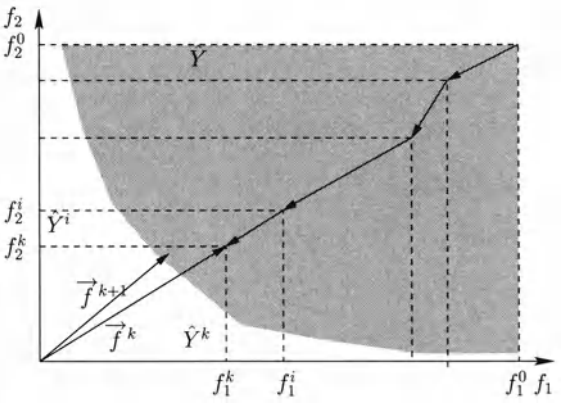
If the decision maker accepts the last objective function vector $\overrightarrow{f^k}$, the algorithm stops there. The vector $\overrightarrow{f^k}$ is not necessarily optimal in the Pareto sense, but it lies inside the search space $\hat{Y}^k$ (the search space at iteration k). So, an optimal point in the Pareto sense is obtained in the neighborhood of this solution by solving the following substitution scalar problem:

$$\begin{array}{ll}
\text{minimize } \sum_{i=1}^{k} w_i \cdot \overrightarrow{f}\left(\overrightarrow{x}\right) & \\
\text{with} & \overrightarrow{g}\left(\overrightarrow{x}\right) \leq 0 \\
& \overrightarrow{h}\left(\overrightarrow{x}\right) = 0 \\
\text{and} & \overrightarrow{f}\left(\overrightarrow{x}\right) \leq \overrightarrow{f}\left(\overrightarrow{x^k}\right)
\end{array}$$

A sketch of the iterative process inside X and Y is shown in Fig. 3.9. In this figure, we denote by $\overrightarrow{f}(\overrightarrow{x})$ the solution of the preceding substitution scalar problem. We can see that the vector $\overrightarrow{f^k}$ allows us to restrict the search space.



(a) Representation inside X.



(b) Representation inside Y.

Fig. 3.9. Representation of the Jahn method.

To illustrate the last step of the method, Fig. 3.10 represents the solution of the scalar substitution problem (by the weighted-sum-of-objective-functions method).

3.5.3 Discussion

We find again the drawback highlighted in Sect. 3.3.3, in relation to the Fandel method. Moreover, this method needs to be able to compute the derivatives of some

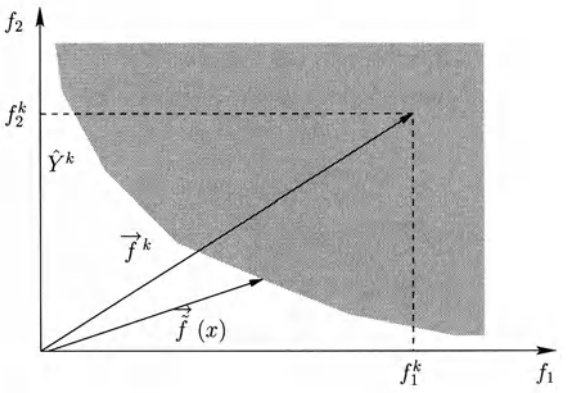


Fig. 3.10. Last step of the Jahn method.

relations linked to our optimization problem: derivatives of the objective functions vector and derivatives of the constraints vector. Consequently, if the problem does not admit derivatives at some points (as with combinatorial problems), this method cannot be applied to obtain a solution.

3.6 Geoffrion method

3.6.1 Principle

The approach used here is similar to the one used with the Jahn method. It is based on the Franck–Wolfe algorithm [Eschenauer et al. 90].

3.6.2 Presentation of the method

We start from the P problem. We transform it in the following way:

$$\left|\begin{array}{lll} \text{minimize} & \left(\nabla_{\vec{x}}\, p\left[\vec{f}\,(\vec{x})\right]\right)^t \cdot \vec{\delta} & \\ \text{with} & \vec{g}\,(\vec{x}) \leq 0 & \text{problem } P' \\ \text{and} & \vec{h}\,(\vec{x}) = 0 & \end{array}\right.$$

with $\vec{\delta} \in \mathbb{R}^k$.

The preference function $p\left[\vec{f}\,(\vec{x})\right]$ is not necessarily known by the decision maker. While running the algorithm, the decision maker has to provide some information about the tradeoff he/she would like to operate on the objective functions. The decision maker must also give a length for the step.

Two methods to determine the starting point $\vec{x^0}$ are possible. Either the decision maker chooses a vector using the Jahn method or the vector is computed by solving the following substitution problem:

$$\left|\begin{array}{lll} \text{minimize} & \sum_{i=1}^{k} w_i \cdot f_i\,(\vec{x}) & \\ \text{with} & \vec{g}\,(\vec{x}) \leq 0 & \\ \text{and} & \vec{h}\,(\vec{x}) = 0 & \end{array}\right.$$

We recognize here the weighted-sum-of-objective-functions method described in Sect. 2.1.2. By solving this problem, we obtain the point $\overrightarrow{x^0} = \overrightarrow{x^*}$.

Now we can solve the problem P'. The solution of this problem gives us a search direction δ. Information about the preference function is now required. We must transform the problem P' by using the following relation:

$$\nabla_{\overrightarrow{x}} p\left[\overrightarrow{f}(\overrightarrow{x})\right] = \sum_{i=1}^{k} \left[\frac{\partial p}{\partial f_i}\right]_{\overrightarrow{x}} \cdot \nabla_{\overrightarrow{x}} f_i(\overrightarrow{x}) \tag{3.14}$$

We obtain the following problem:

$$\begin{vmatrix} \text{minimize} & \sum_{i=1}^{k} w_i \cdot \left(\nabla_{\overrightarrow{x}} f_i\left(\overrightarrow{x}^k\right)\right)^t \cdot \overrightarrow{\delta} \\ \text{with} & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\ \text{and} & \overrightarrow{h}(\overrightarrow{x}) = 0 \end{vmatrix}$$

We obtain

$$w_i = \frac{\left[\frac{\partial p}{\partial f_i}\right]_{\overrightarrow{x}}}{\left[\frac{\partial p}{\partial f_1}\right]_{\overrightarrow{x}}} \tag{3.15}$$

with $i = \{1, \cdots, k\}$, and the direction of the step is given by $\overrightarrow{\delta^k} = \overrightarrow{x^{k+1}} - \overrightarrow{x^k}$.

The weight coefficients mirror the tradeoff performed by the decision maker between the objective function f_i and the reference objective function f_1.

There are two methods to determine the tradeoff ratio w_i without involving the decision maker. The first one consists in allowing a small variation Δf_i, which compensates a small variation Δf_1 of the objective function f_1 while the other objective functions are kept constant. Hence,

$$w_i = -\frac{\Delta f_1}{\Delta f_i} \tag{3.16}$$

The other method consists in "approximating" the weight coefficients by using the hypothesis that the derivative of $p\left[\overrightarrow{f}(\overrightarrow{x})\right]$ remains constant in the search step direction. Consequently, if all objective functions except the first and the ith, are kept constant, the unknown preference function $p\left[\overrightarrow{f}(\overrightarrow{x})\right]$ increases faster when the objective function i varies by b_i units than when the reference objective function 1 varies by b_1 units, the ratio being w_i:

$$w_i = \frac{b_i}{b_1} \tag{3.17}$$

After solution of the above substitution problem, the length of the step λ must be determined by the decision maker him/herself, who must consider the following goal:

$$\begin{vmatrix} \text{minimize} & p\left[f\left(\overrightarrow{x^k} + \lambda^k \cdot \overrightarrow{\delta^k}\right)\right] \\ \text{with} & \overrightarrow{g}\left(\overrightarrow{x^k} + \lambda^k \cdot \overrightarrow{\delta^k}\right) \leq 0 \\ \text{and} & \overrightarrow{h}\left(\overrightarrow{x^k} + \lambda^k \cdot \overrightarrow{\delta^k}\right) = 0 \end{vmatrix}$$

while applying the following hypothesis:

$$0 \leq \lambda^k \leq 1$$

Fig. 3.11 represents the behavior of this algorithm. In this figure, we have denoted by $\overrightarrow{f^{i}}$ the solution found at iteration i. The first point on this figure ($\overrightarrow{F^0}$) was obtained by solving the substitution problem using the weighted-sum-of-objective-functions method. We have $\overrightarrow{F^0} = \overrightarrow{f}\left(\overrightarrow{x^0}\right)$.

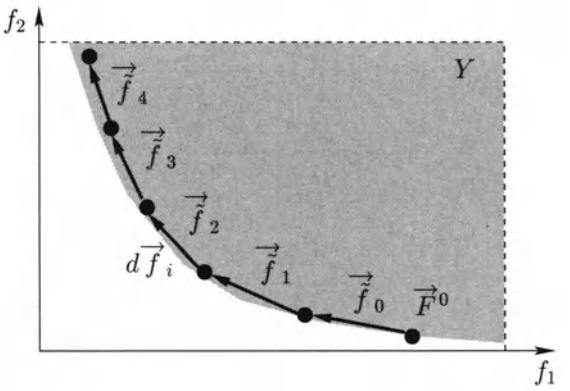


Fig. 3.11. Geoffrion method.

During the above minimization, the decision maker can be helped by a graphical representation of all the objective functions with respect to λ, or by a list of dependencies stored in a table.

Once the length of the step is determined, the new objective function vector can be computed:

$$\overrightarrow{f^{k+1}} = \overrightarrow{f}\left(\overrightarrow{x} + \lambda^k \cdot \overrightarrow{\delta^k}\right) \tag{3.18}$$

The decision maker then evaluates the result, and decides whether the process should continue or whether the solution can be accepted as a tradeoff solution. If the decision maker decides to continue, he/she can cover the tradeoff surface (see Fig. 3.11).

3.6.3 Discussion

We find again the usual drawback of interactive methods (see Sect. 3.3.3). Once again, we must underline the restriction of this method to problems for which we can compute derivatives.

3.7 Simplex method

3.7.1 Principle

This method has nothing in common with the simplex method used in linear programming. It is a sequential search method [Nelder et al. 65] for the optimum

of a problem. A software package [Multisimplex] computes this optimization for a multiobjective optimization problem in an interactive way.

This method uses $k + 1$ tries (where k represents the dimension of the decision variable $\vec{x}$) to define an improvement direction for the objective functions. The improvement of the objective functions is obtained using a fuzzy aggregation method (see Sect. 4.1).

3.7.2 Presentation of the method

We begin by choosing at random $k + 1$ values for the decision variable $\vec{x}$. The algorithm then evaluates the $k + 1$ points and removes the least "efficient" point. Then it creates a new point based on the removed point (see Fig. 3.12) and starts the computation again.

This algorithm includes some rules which allow us to choose points in such a way as to avoid rotating around a bad solution. The two main rules are the following:

Rule 1: reject the worst solutions.
A new position of the decision variable $\vec{x}$ is computed by reflection of the rejected position (see Fig. 3.12). After this transformation, we search for the new worst point. This method is repeated with this point removed, etc. At each step, we come closer to the area where the optimum that we are seeking is located.

Rule 2: never come back to a point which has just been rejected.
Without this rule, the algorithm could oscillate between two "bad" points.

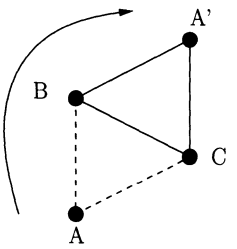


Fig. 3.12. Choice of a new point by symmetry.

In Fig. 3.13, we can see the behavior of the method in the search space, in the case of an example with two decision variables. In this figure are represented the level curves which represent the improvement "direction" of the objective function to be optimized. In this example, the goal of the optimization is to maximize the percentage which corresponds to a level curve.

Let us now present the whole simplex algorithm. We shall use the following notation:

W: the least favorable point or the point which has just been rejected.
B: the most favorable point.
N: the second most favorable point.

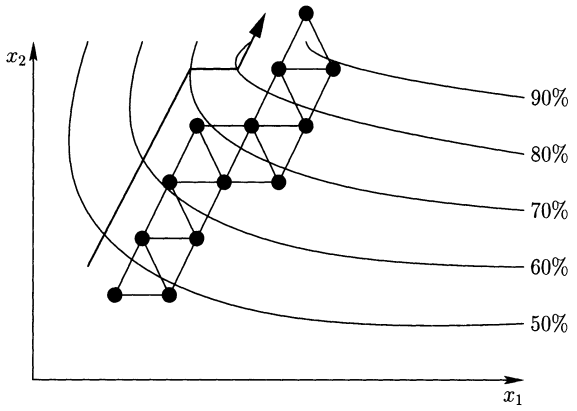


Fig. 3.13. Behavior of the simplex method.

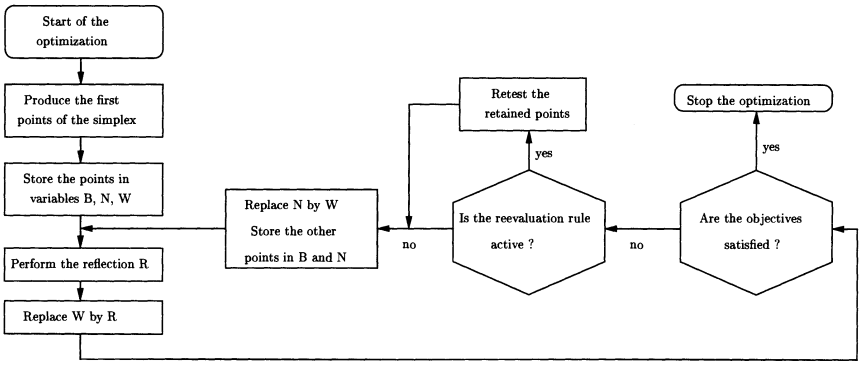


Fig. 3.14. The simplex algorithm.

The simplex algorithm is presented in Fig. 3.14.

There also exists a modified version of the simplex method. This method, called the modified simplex method, includes two more rules:

Rule 3: expand the displacement in the direction of the improvement of the objective function.

Rule 4: contract the displacement if this has been done in a direction which is not favorable to the improvement of the objective function.

For a problem with two decision variables, we obtain situation sketched in Fig. 3.15.

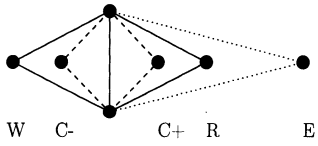


Fig. 3.15. The expansion and contraction processes of the modified simplex method.

Using the same notation as with the preceding algorithm, we obtain the representation in Fig. 3.16.

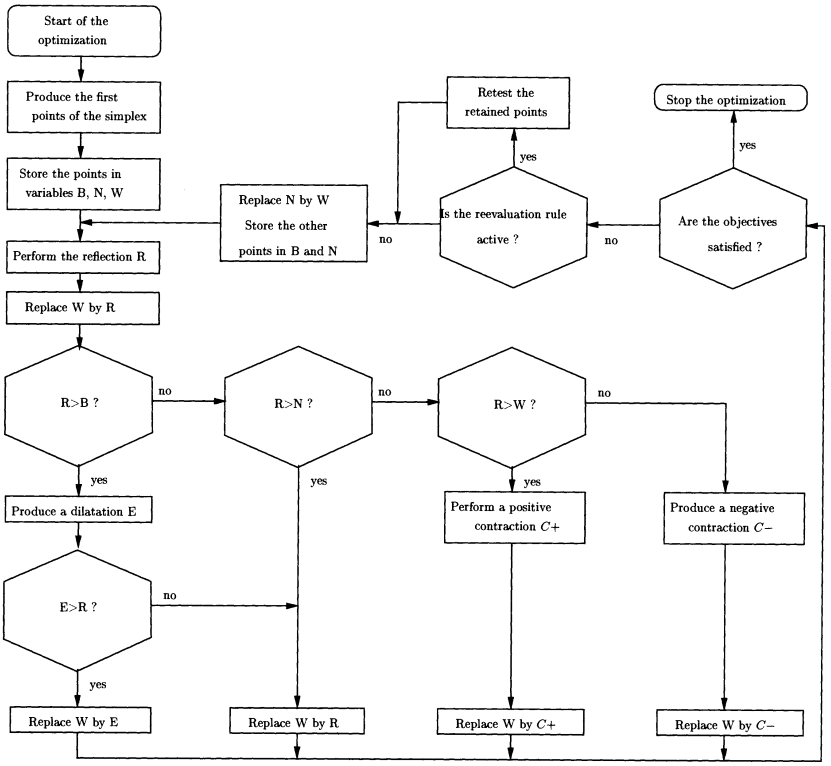


Fig. 3.16. The modified simplex algorithm.

We have the following relations for the various transformations:

reflection: $R = \bar{C} + \alpha \cdot (\bar{C} - W)$
expansion: $E = \bar{C} + \gamma \cdot (\bar{C} - W)$
positive contraction: $C_+ = \bar{C} + \beta^+ \cdot (\bar{C} - W)$
negative contraction: $C_- = \bar{C} - \beta^- \cdot (\bar{C} - W)$

where W is the rejected point, $\bar{C}$ is the center of mass of the remaining points, α is the reflection coefficient (default value $\alpha = 1$), γ is the expansion coefficient (default value $\gamma = 2$), β^+ is the positive contraction coefficient (default value $\beta^+ = 0.5$), and β^- is the negative contraction coefficient (default value $\beta^- = 0.5$).

3.7.3 Discussion

This method is really simple to put into effect. Moreover, a software package has been developed around this method: the Multisimplex software (see the URL [Multisimplex]).

3.8 Annotated bibliography

[Eschenauer et al. 90] This is a rare example of a book which talks about interactive methods. The presentation of these methods is very mathematical, and these methods are not illustrated through simple examples. On the other hand, we find in the last few chapters numerous examples of the application of multiobjective optimization to industrial problems, in particular a structural optimization problem of a parabolic antenna where one aims to minimize the vibrations of the structure.

[Miettinen 99] We also find a description of some interactive multiobjective optimization methods in this book.

4

Fuzzy methods

4.1 Introduction to fuzzy logic

In everyday life, everything cannot be described in a binary way. For example, the transition between day and night is progressive; the action of the clutch of a car is progressive too.

For a long time, the one and only tool to perform descriptions by logic was binary logic. Everything in logic was written in terms of TRUE and FALSE. The problem with this simplistic description is that it does not allow us to deal with the uncertainty and imprecision of human knowledge.

The automation scientist L. A. Zadeh has constructed a new logic based on fuzzy sets. It allows us to deal with imprecision and uncertainty in human knowledge, as well as with progressive transitions between states. The main difference between classical logic and fuzzy logic is the existence of a progressive transition between TRUE and FALSE [Zadeh 65, Remez 86, Vern 92].

4.1.1 Drawing a parallel with classical logic

To better understand the concept of fuzzy logic, we shall transpose the concepts of classical logic into fuzzy logic.

First of all, there are three main functions in classical logic. These functions work with binary parameters (0 or 1, TRUE or FALSE). Here are these foundation functions:

- The AND function: $C = A$ and B, or, in more mathematical language, $C = A \cdot B$. The truth table of this function is presented in Table 4.1.

Table 4.1. The AND function.

A \ B	0	1
0	0	0
1	0	1

Table 4.2. The OR function.

A \ B	0	1
0	0	1
1	1	1

Table 4.3. The NOT function.

A	C
0	1
1	0

- The OR function: $C = A$ or B or, in more mathematical language, $C = A + B$. The truth table of this function is presented in Table 4.2.

- The NOT function: C = not A or, in more mathematical language, $C = \overline{A}$. The truth table of this function is presented in Table 4.3.

All these functions of classical logic have an equivalent in fuzzy logic. The main difference comes from the fact that we work with a logical variable which varies continuously between 0 and 1.

The equivalents in fuzzy logic of the functions of classical logic are shown in Table 4.4.

Table 4.4. Equivalence between classical logic and fuzzy logic.

Classical logic	Fuzzy logic
$C = A$ and B	$C = \min(A, B)$
$C = A$ or B	$C = \max(A, B)$
$C =$ not A	$C = 1 - A$

4.1.2 Membership function

In general, in fuzzy logic, we always link the variable (A, B or C) to an exterior data item (a piece of information or a measurement for example). This linking is done through what we call a membership function. This function allows us to make a link between a piece of information (for example, the height of an individual which varies between 0 and 2 m) and a logical variable which varies continuously between 0 and 1. The following is an example of a membership function for height. This function defines a fuzzy logic variable "medium height". We consider that we have a medium height when it is between 1.70 m and 1.80 m. In Figs. 4.1 and 4.2 we illustrate the differences which exist between fuzzy logic and classical logic using this example.

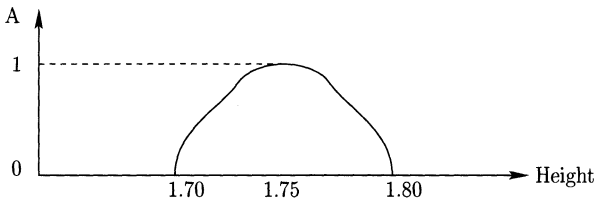


Fig. 4.1. "Medium height" membership function in fuzzy logic.

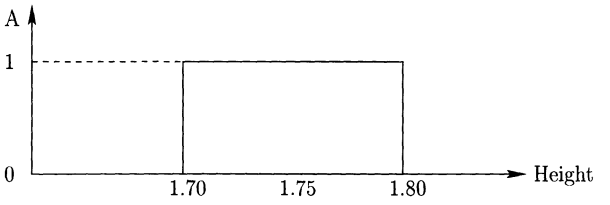


Fig. 4.2. "Medium height" function in classical logic.

The various functions of fuzzy logic that we have presented have the same properties as their counterparts in classical logic (distributivity, commutativity and the De Morgan rule). A transposition of these rules to fuzzy logic is presented in Table 4.5.

Table 4.5. Properties of classical and fuzzy logic.

Classical logic	Fuzzy logic
Distributivity	
$(A+B)\cdot C = A\cdot C + B\cdot C$	$\min(\max(A,B),C) = \max(\min(A,C),\min(B,C))$
Commutativity	
$(A+B)+C = A+(B+C)$	$\max(\max(A,B),C) = \max(A,\max(B,C))$
De Morgan rule	
$\overline{A\cdot B} = \overline{A}+\overline{B}$	$1-\min(A,B) = \min(1-A,1-B)$

We now apply these various fuzzy functions to the fuzzy variables "medium height" (between 1.70 and 1.80) and "tall height" (between 1.75 and 1.85):

- The AND function (min function in fuzzy logic; see Fig. 4.3),

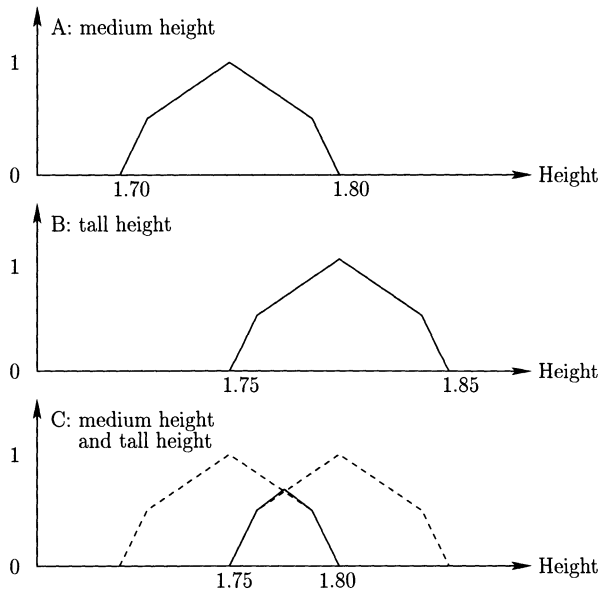


Fig. 4.3. The AND fuzzy function.

- The OR function (max in fuzzy logic; see Fig. 4.4),
- The NOT function (minus 1 in fuzzy logic; see Fig. 4.5).

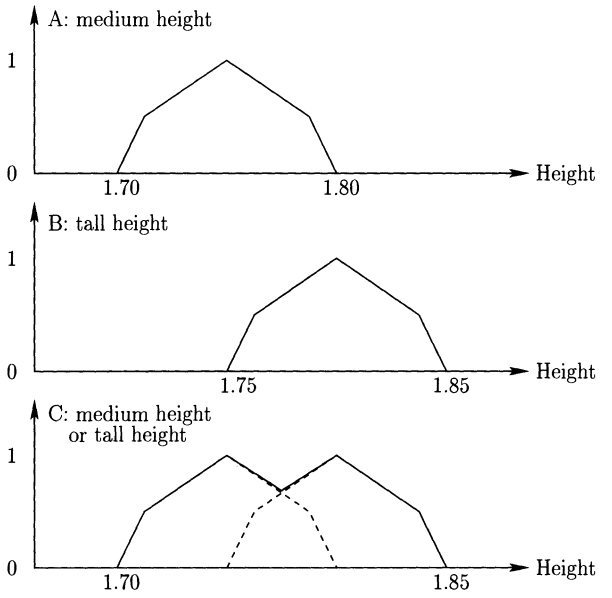


Fig. 4.4. The OR fuzzy function.

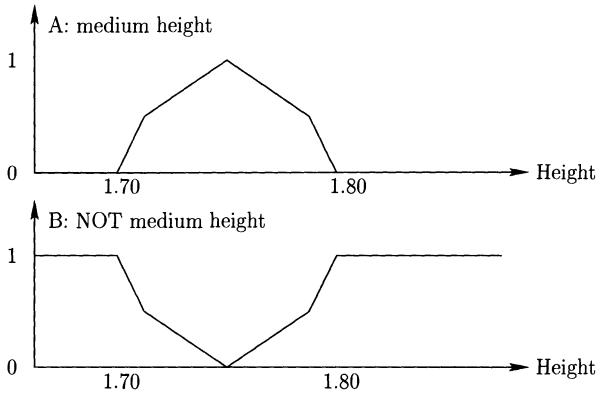


Fig. 4.5. The NOT fuzzy function.

4.2 Sakawa method

4.2.1 Principle

This method uses fuzzy logic at all levels (on the parameters of the problem as well as on the parameters of the constraints). The set of solutions that we find uses fuzzy logic too. These solutions have a membership level. This means that the solutions have a variable *correlation* with the initial objective ([Sakawa et al. 88] and [Sakawa 02]) (this correlation level with the initial objective is set by the decision maker).

4.2.2 Presentation of the method

This optimization method which uses fuzzy logic is presented through a linear optimization problem (also known as a linear programming problem). We start from the following problem:

$$\begin{vmatrix}\text{minimize } \overrightarrow{c}_1 \cdot \overrightarrow{x}^t, \cdots, \overrightarrow{c}_k \cdot \overrightarrow{x}^t \\ \text{with} \quad \sum_i a_{ij} \cdot x_i \leq b_j, j = \{1, \cdots, m\}\end{vmatrix}$$

We have $\overrightarrow{x} \in \mathbb{R}^n$, $\overrightarrow{c} \in \mathbb{R}^n$, $\overrightarrow{a} \in \mathbb{R}^n$ and $\overrightarrow{b} \in \mathbb{R}^m$. Here, n is the number of variables of the problem (the dimension of $\overrightarrow{x}$), k is the number of objective functions, and m is the number of inequality constraints.

We define X as the following set:

$$X = \left\{ \overrightarrow{x} \in \mathbb{R}^n \mid \sum_{i=1}^{n} a_{ij} \cdot x_i \leq b_j, j = \{1, \cdots, m\} \right\} \tag{4.1}$$

This is the set of values of $\overrightarrow{x}$ such that the constraints are respected.

For this problem, we expect that we shall have an uncertainty in the objective functions. So we consider that these parameters are fuzzy. We rewrite this problem the following way:

$$\begin{vmatrix}\text{minimize } \overrightarrow{\tilde{c}_1} \cdot \overrightarrow{x}^t, \cdots, \overrightarrow{\tilde{c}_k} \cdot \overrightarrow{x}^t \\ \text{with} \quad \overrightarrow{x} \in X\left(\tilde{a}, \tilde{b}\right)\end{vmatrix}$$

Here,

$$X\left(\tilde{a}, \tilde{b}\right) = \left\{ \overrightarrow{x} \in \mathbb{R}^n \mid \sum_{i=1}^{n} \tilde{a}_{ij} \cdot x_i \leq \tilde{b}_j, j = \{1, \cdots, m\} \right\} \tag{4.2}$$

For each parameter, we define a membership function:

$$\mu_{\tilde{c}_{j1}}(c_{j1}), \cdots, \mu_{\tilde{c}_{jn}}(c_{jn}); \mu_{\tilde{b}_1}(b_1), \cdots, \mu_{\tilde{b}_n}(b_n); \mu_{\tilde{a}_{i1}}(a_{i1}), \cdots, \mu_{\tilde{a}_{in}}(a_{in}) \tag{4.3}$$

Before we continue with the presentation of the method, we introduce two definitions.

Definition 22. ***Set of level*** α

The set of level α *of the fuzzy numbers* $\tilde{a}_{jr}$, $\tilde{b}_j$, $\tilde{c}_{jr}$ *is defined as the set* $L_\alpha\left(\tilde{a}, \tilde{b}, \tilde{c}\right)$ *for which the membership degree of the functions* $\tilde{a}$, $\tilde{b}$ *and* $\tilde{c}$ *is greater than or equal to* α*:*

$$L_\alpha\left(\tilde{a}, \tilde{b}, \tilde{c}\right) = \left\{ (a, b, c) \mid \mu_{\tilde{c}_{jr}}(c_{jr}) \geq \alpha, \mu_{\tilde{b}_j}(b_j) \geq \alpha \text{ and } \mu_{\tilde{a}_{ir}}(a_{ir}) \geq \alpha \right\} \tag{4.4}$$

In this definition, $i = \{1, \cdots, k\}$, $j = \{1, \cdots, m\}$ and $r = \{1, \cdots, n\}$. This set has the following property:

$$\alpha_1 \leq \alpha_2 \text{ if } L_{\alpha_1}\left(\tilde{a},\tilde{b},\tilde{c}\right) \supset L_{\alpha_2}\left(\tilde{a},\tilde{b},\tilde{c}\right) \tag{4.5}$$

Definition 23. *α-optimal solution in the Pareto sense*
$\vec{x}^* \in X\left(\tilde{a},\tilde{b}\right)$ is called an α-optimal solution in the Pareto sense of the multiobjective linear programming problem if and only if there does not exist an $\vec{x} \in X\left(\tilde{a},\tilde{b}\right)$ and an $(a,b,c) \in L_\alpha\left(\tilde{a},\tilde{b},\tilde{c}\right)$ such that $\vec{c}_i \cdot \vec{x} \leq \vec{c}_i^* \cdot \vec{x}$, $i = \{1,\cdots,k\}$ with a strict inequality for at least one i. The parameters (a^*,b^*,c^*) are called optimal parameters of level α.

We shall now determine the shape of the membership functions $\mu_i(\vec{c}_i \cdot \vec{x})$, $i = \{1,\cdots,k\}$. We begin by computing the extremum values of the objective functions under the constraints $\alpha = 0$ and $\alpha = 1$. We denote these values by $(\vec{c}_i \cdot \vec{x})^0$ and $(\vec{c}_i \cdot \vec{x})^1$:

- $(\vec{c}_i \cdot \vec{x})^0$ is an unacceptable value for $(\vec{c}_i \cdot \vec{x})$;
- $(\vec{c}_i \cdot \vec{x})^1$ is a very acceptable value for $(\vec{c}_i \cdot \vec{x})$.

Hence, we have:

$$\mu_i(\vec{c}_i \cdot \vec{x}) = \begin{cases} 1 \text{ or tends to } 1 \text{ if } (\vec{c}_i \cdot \vec{x})^1 \geq (\vec{c}_i \cdot \vec{x}) \\ d_i(\vec{c}_i \cdot \vec{x}) \quad \text{if } (\vec{c}_i \cdot \vec{x})^1 \leq (\vec{c}_i \cdot \vec{x}) \leq (\vec{c}_i \cdot \vec{x})^0 \\ 0 \text{ or tends to } 0 \text{ if } (\vec{c}_i \cdot \vec{x}) \geq (\vec{c}_i \cdot \vec{x})^0 \end{cases} \tag{4.6}$$

where $d_i(\vec{c}_i \cdot \vec{x})$ is a monotonic and strictly decreasing function, which may be or not linear.

Now that the membership functions of the objective function have been defined, we can rewrite our problem using a fuzzy formalism:

$$\begin{array}{ll} \text{maximize} & \mu_D(\mu_1(\vec{c}_1 \cdot \vec{x}),\cdots,\mu_k(\vec{c}_i \cdot \vec{x})) \\ \text{with} & \vec{x} \in X\left(\tilde{a},\tilde{b}\right) \\ \text{and} & (a,b,c) \in L_\alpha\left(\tilde{a},\tilde{b},\tilde{c}\right) \end{array}$$

where μ_D is an aggregation function. μ_D can be interpreted as the degree of satisfaction relative to the goal set by the decision maker. An example of such an aggregation function is

$$\min(\mu_1(\vec{c}_i \cdot \vec{x}),\cdots,\mu_k(\vec{c}_i \cdot \vec{x})) \tag{4.7}$$

So, finally, we obtain the following problem (if we use the above aggregation function):

$$\begin{array}{ll} \text{maximize} & \min(\mu_1(\vec{c}_i \cdot \vec{x}),\cdots,\mu_k(\vec{c}_i \cdot \vec{x})) \\ \text{with} & \vec{x} \in X\left(\tilde{a},\tilde{b}\right) \\ \text{and} & (a,b,c) \in L_\alpha\left(\tilde{a},\tilde{b},\tilde{c}\right) \end{array}$$

We try to maximize the minimal membership degree. [Munro et al. 86] describes a similar method, but applied to a general multiobjective optimization problem.

To sum up, we can divide this method into six steps as follows.

We start from the following problem:

$$\begin{vmatrix}\text{minimize } f_1(\vec{x}), \cdots, f_k(\vec{x}) \\ \text{with } \quad \vec{g}(\vec{x}) \leq 0\end{vmatrix}$$

Step 1: Under the constraint $\vec{g}(\vec{x}) \leq 0$, we minimize, and then we maximize, each objective function separately, so as to find the variation interval of the membership function of the objective functions.

Step 2: We define membership functions the following way:

$$\mu_i(\vec{x}) = \frac{f_{i1} - f_i(\vec{x})}{f_{i1} - f_{i0}} \tag{4.8}$$

where

- f_{i0} is the least interesting value of the membership function, and
- f_{i1} is the most interesting value of the membership function.

Step 3: We define the decision-making functions DM_i ("Decision Making") as follows:

$$DM_i(\vec{x}) = \begin{cases} 0 & \text{if } \mu_i(\vec{x}) \leq 0 \\ \mu_i(\vec{x}) & \text{if } 0 < \mu_i(\vec{x}) \leq 1 \\ 1 & \text{if } 1 \leq \mu_i(\vec{x}) \end{cases} \tag{4.9}$$

where

- $DM_i(\vec{x}) = 1$ when the objective function is reached, and
- $DM_i(\vec{x}) = 0$ when the objective is not reached.

Step 4: We maximize the functions DM_i.

Step 5: If there are no solutions, or if the solution does not satisfy the decision maker, then we must modify the membership functions and go back to step 3.

Step 6: Stop and print the results.

This description of the algorithm in six steps is taken from [Niimura et al.].

4.2.3 Discussion

This method is really interesting, because it allows preferences of the decision maker in relation to the solution proposed by the search method to be taken into account. In this way, the decision maker can obtain more solutions if he/she is less demanding about the quality of the solution.

This demandingness is symbolized by the membership level α. If the decision maker is really demanding, he/she will set a minimum level α close to 1.

Nevertheless, the Sakawa method seems cumbersome to deal with. It uses fuzzy sets and mathematical rules that are hard to apply. We prefer the approach used in the Reardon method, which we describe now.

4.3 Reardon method

4.3.1 Introduction

We present a simplified method that uses fuzzy logic to solve a multiobjective problem. Some examples which use this method can be found in [Reardon 97a] and [Reardon 97b].

4.3.2 Presentation of the method

We start from the P problem.

For each objective function, we define a membership function, which has the shape drawn in Fig. 4.6.

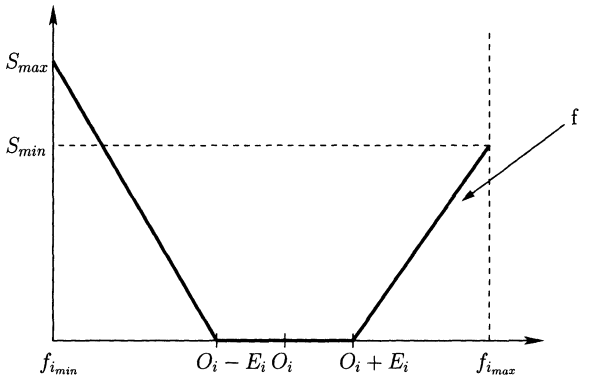


Fig. 4.6. The Reardon membership function.

This membership function allows one to "inform" the algorithm that the objective function has its value located inside the interval $[O_i - E_i, O_i + E_i]$ (the area where the membership value is 0). If the value of the objective function is located outside of this interval, we penalize the objective function more and more. In that case, the penalty varies between 0 and S_{min} or S_{max}.

The significance of these various parameters of the membership function is the following:

S_{min} and S_{max}: fuzzy scale factors.

$f_{i_{min}}$: minimum value of objective function number i.

$f_{i_{max}}$: maximum value of objective function number i.

O_i: experimental value of the objective function number i. We would like $f_i = O_i$.

E_i: acceptable error margin (the objective O_i has a "satisfaction interval" $2 \cdot E_i$ wide).

The equation of the membership function is the following:

- if $f_i \leq (O_i - E_i)$ then

$$f'(f_i) = \left[\frac{S_{max}}{f_{i_{min}} - (O_i - E_i)}\right] \cdot (f_i - (O_i - E_i))$$

- if $(O_i - E_i) \leq f_i \leq (O_i + E_i)$ then

$$f'(f_i) = 0$$

- if $f_i \geq (O_i + E_i)$ then

$$f'(f_i) = \left[\frac{S_{min}}{(O_i + E_i) - f_{i_{max}}}\right] \cdot (f_i - (O_i + E_i))$$

Here, when we compare this method with the "normal" fuzzy optimization method, the level is inverted. Instead of having 1 for the membership function when f_i belongs to the desired interval (between $O_i - E_i$ and $O_i + E_i$), the membership function value is equal to 0. This inversion of the membership level allows us to obtain a simpler global objective function, which merges or "aggregates" the objective functions:

$$F = \frac{1}{N} \cdot \sum_{i=1}^{N} f_i'(f_i) \tag{4.10}$$

As we can see in this formula, F is a simple mean of the membership functions of the corresponding objective functions.

These various membership functions have been normalized through the use of S_{min} and S_{max}. In this way, we can remove the influence of the various units of measurement and also minimize effects due to the variation intervals of the various objective functions. We have, in some sense, ranked the various objective functions.

Thanks to this trick, solving our problem is equivalent to finding a solution which minimizes F. So solving the problem is equivalent to finding a point such that the $f_i'(f_i)$ are minimum or such that the membership level is the greatest possible.

4.3.3 Discussion

This method is simple to use. For example, it has given good results on two test problems (Schaffer's F2 test problem [Reardon 97a] and the simplified Born–Mayer test problem [Reardon 97b]).

5

Multiobjective methods using metaheuristics

5.1 What is a metaheuristic ?

Metaheuristics, as stated in the preface, are general optimization methods dedicated to "hard optimization" problem [Sait et al. 99]. These methods are, in general, presented as concepts. As we shall see later, they are sometimes based on ideas that we can find in everyday life. The main metaheuristics are simulated annealing, tabu search and genetic algorithms.

5.2 Task decomposition in optimization

We note that, most of the time, multiobjective optimization methods and the associated metaheuristics can be totally decoupled. However, some methods, such as multiobjective genetic algorithms, cannot be uncoupled.

So, we can divide the set "multiobjective optimization method + metaheuristic + problem" in two different ways (see Figs. 5.1 and 5.2).

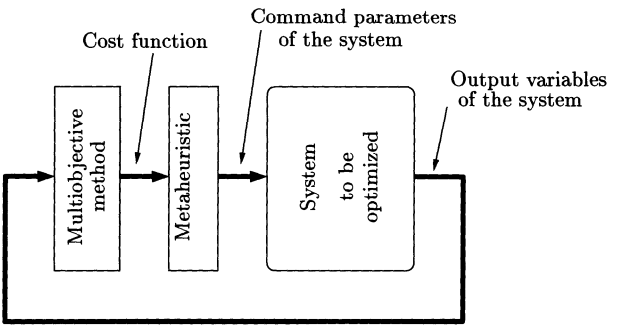


Fig. 5.1. Decomposition of the system when decoupling is possible.

The case illustrated in Fig. 5.1 justifies the study of multiobjective optimization methods as in the preceding chapters, without any reference to metaheuristics. The present chapter is dedicated to the metaheuristics which are linked to multiobjective methods. We shall mainly study the methods derived from genetic algorithms.

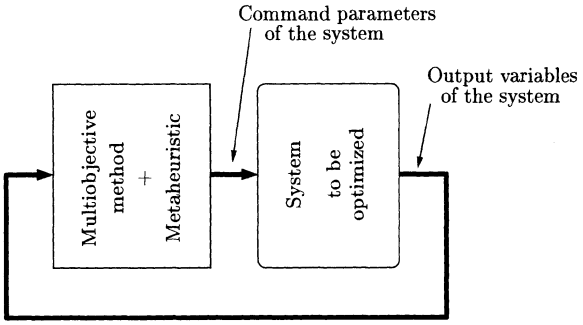


Fig. 5.2. Decomposition of the system when decoupling is not possible.

5.3 General concepts

If we consider the whole set of optimization methods, we notice that these methods are arranged around some general concepts. For example, we find a lot of methods (the Newton method, the quasi-Newton method, etc.) constructed around gradient descent. The general concept of this family is gradient descent, and this general concept is called a metaheuristic.

So, a metaheuristic is really a general method, which needs some transformation (minor transformation in general) before being applied to the solution of a particular problem.

If we consider the various metaheuristics, we notice that they are divided into three families:

- Deterministic methods to search for local optimum. These methods converge quickly but, most of the time, they do not find the global optimum. They just find a local optimum and are not based on a stochastic process to find the optimum (see Fig. 5.3).

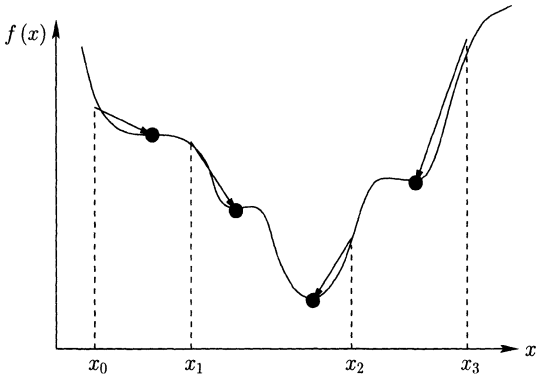


Fig. 5.3. Deterministic methods for local search using various starting solutions.

- Deterministic methods to search for a global optimum. These methods allow one to find quickly a global optimum and are not based on a stochastic process to find the optimum (see Fig. 5.4).

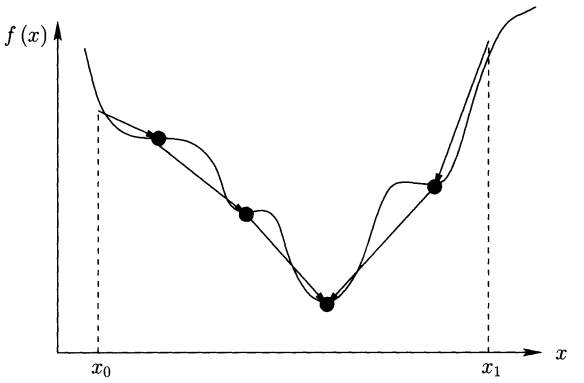


Fig. 5.4. Deterministic methods for global search.

- Stochastic methods to search for a global optimum. These methods are based on a stochastic process which is used to search for the optimum. They are less efficient (from the point of view of speed) than the deterministic methods, but they can, in principle, find a global optimum that is hard to reach (see Fig. 5.5).

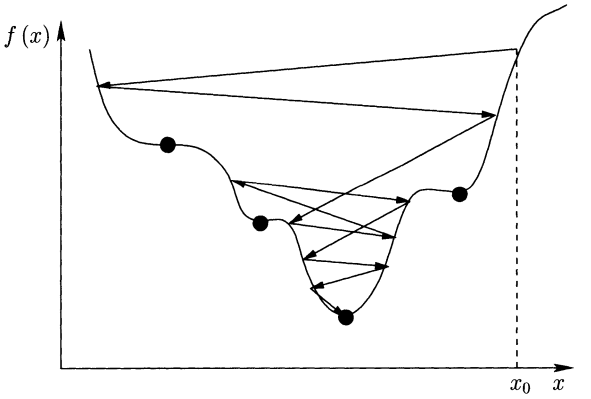


Fig. 5.5. Stochastic methods for global search.

We present in this chapter some search methods which belong to the last family.

5.4 Simulated annealing

This optimization method was first presented by researchers from IBM who were studying spin glasses. This method is based on a metallurgical process (annealing)

that finds a minimum. For a metal to reach a crystallographic structure close to a perfect crystal (the perfect crystal structure corresponds to a minimum of the energy of the arrangement of atoms of the metal), we need to heat it to a high temperature, and then slowly cool it so that the atoms have time to arrange themselves regularly (see [Bonomi 88], [Siarry 94] and [Siarry 99]).

This metallurgical process has been transposed into the domain of optimization and has given birth to an efficient and simple method. The simulated annealing algorithm is presented in algorithm 2.

Algorithm 2 Simulated annealing.

init T (initial temperature)
init x (starting point)
init ΔT (minimal temperature)

while(not(end))
  $y = \text{Neighbor}(x)$
  $\Delta C = C(y) - C(x)$
  if $\Delta C < 0$ then $y = x$
  else if rand $(0,1) < e^{-\frac{\Delta C}{T}}$ then $y = x$
  $T = \alpha(T)$
  if $T < \Delta T$ then end(while)
repeat(while)

The behavior of this algorithm is the following:

- We begin by choosing a starting point at random (x).
- We compute a neighbor of this point ($y = \text{Neighbor}(x)$).
- We evaluate this neighboring point and compute the gap with respect to the origin point ($\Delta C = C(y) - C(x)$).
- If this gap is negative, we take the point y as a new starting point. If the gap is positive, we can may the point y as a new starting point, but with a probability $e^{-\frac{\Delta C}{T}}$ (which varies in a way opposite to the temperature T).
- As the algorithm runs, we decrease the temperature T ($T = \alpha(T)$), often using levels.
- We repeat all these steps for as long as the system is not frozen (for example, the steps are repeated until the temperature reaches some minimum level).

We shall now describe multiobjective simulated annealing.

5.4.1 PASA (Pareto archived simulated annealing) method

5.4.1.1 Principle

This method, developed at EDF [Engrand et al. 98], uses an aggregation function of objective functions coupled with a system which archives nondominated solutions.

5.4.1.2 Presentation of the method

We suppose that the objective functions of the multiobjective problem are positive. This hypothesis allows us to use the following aggregation function, so as to transform our problem into a mono-objective minimization problem:

$$G(\vec{x}) = \sum_{i=1}^{n} \ln(f_i(\vec{x})) \tag{5.1}$$

When this is done, the expression

$$\Delta G = G(\vec{x}') - G(\vec{x}) = \sum_{i=1}^{n} \ln\left(\frac{f_i(\vec{x}')}{f_i(\vec{x})}\right) \tag{5.2}$$

represents the mean relative variation of the objective function between the current point ($\vec{x}$) and the point to test ($\vec{x}'$):

- if $\Delta G > 0$, the new solution $\vec{x}'$ deteriorates all the objective functions, with respect to the relative mean;
- if $\Delta G < 0$, the new solution $\vec{x}'$ improves all the objective functions, with respect to the relative mean.

Next, as with the original simulated annealing method, we define an acceptance probability p:

$$p = \exp\left(-\frac{\Delta G}{T}\right) \tag{5.3}$$

where T corresponds to the temperature of the annealing algorithm. This temperature has the same meaning as in mono-objective simulated annealing.

To archive the nondominated solutions, we use an archive of variable size (there are between 0 and N_{max} solutions in the archive). To manage the archive, we use the following rules:

- if solution $\vec{x}'$ is dominated by at least one solution in the archive, then solution $\vec{x}'$ is not archived;
- if solution $\vec{x}'$ dominates a solution $\vec{y}$ in the archive, then solution $\vec{x}'$ replaces solution $\vec{y}$ in the archive;
- if solution $\vec{x}'$ is not dominated by the solutions in the archive, then we put this solution in the archive and we remove from the archive all the solutions which are dominated by solution $\vec{x}'$.

We can add some other rules to improve the management of the solutions in the archive. For example, we can add a restrictive criterion when adding a solution to the archive:

- solution $\vec{x}'$ must dominate all the solutions in the archive and be located at some minimal distance from the neighboring solutions of the archive.

To allow the search to be performed on the whole tradeoff surface, it is necessary to rerun the search regularly using a starting point chosen at random, inside the archive. This step of the choice of a new starting point is called *return to base*.

The algorithm of the PASA method is described in algorithm 3.

Algorithm 3 The PASA method.

1. Choice of P_0, an initial population in the archive, eventually reduced to a single solution x_0, and initial choice for T_q.
2. $\overrightarrow{x_n'}$ a neighbor of $\overrightarrow{x_n}$, is given.
3. Compute $G(\overrightarrow{x_n'})$ and $\Delta G = G(\overrightarrow{x_n'}) - G(\overrightarrow{x_n})$.
4. If $\Delta G < 0$ then $\overrightarrow{x_{n+1}} = \overrightarrow{x_n'}$.
5. If $\Delta G > 0$ then $p = \exp\left(-\frac{\Delta G}{T_q}\right)$.
   - $\overrightarrow{x_{n+1}} = \overrightarrow{x_n'}$ with a probability p.
   - $\overrightarrow{x_{n+1}} = \overrightarrow{x_n}$ with a probability $1 - p$.
6. If we go out of the internal loop, then $T_{q+1} = \alpha \cdot T_q$ with $\alpha < 1$.
7. Nondomination principle applied to archive the solution $\overrightarrow{x_{n+1}}$:
   - if $\overrightarrow{x_{n+1}}$ dominates one of the archived solutions $\overrightarrow{y_p}$, $\overrightarrow{x_{n+1}}$ replaces $\overrightarrow{y_p}$ in the archived population;
   - if $\overrightarrow{x_{n+1}}$ is dominated by, at least, one of the archived solutions, then we do not archive it;
   - if $\overrightarrow{x_{n+1}}$ is not dominated by the archived population, then we add this solution to the archived population and we remove from the archive all the solutions dominated by $\overrightarrow{x_{n+1}}$.
8. *Return to base*: when it is necessary, $\overrightarrow{x_{n+1}} = \overrightarrow{y_i}$ (i chosen randomly in the archived population).
9. If (non-convergence) then return to step 2.

5.4.1.3 Discussion

In this algorithm, the initial population inside the archive can be computed by an initial first multiobjective optimization. The quality of the initial solutions in the archive will influence the quality of the result we obtain at the end of the current optimization.

Another advantage of this method is that we can add heuristic rules to manage the archive easily. This "flexibility" can allow us to add some "engineer knowledge", which may improve the quality of the solutions that we obtain at the end of the optimization, but especially the convergence speed toward this archived population.

A drawback of this method is that the form of the aggregation function of objective functions (Eq. 5.1) used does not allow objective functions with negative values.

5.4.2 MOSA (Multiple objective simulated annealing) method

5.4.2.1 Principle

This method has been presented in [Ulungu et al. 99]. It uses the simulated annealing to search for the tradeoff surface.

5.4.2.2 Presentation of the method

We begin by defining a series of functions which gives the acceptance probability of a bad solution, for each objective function:

$$\pi_k = \begin{cases} \exp\left(-\frac{\Delta f_k}{T_n}\right) & \text{if } \Delta f_k > 0 \\ 1 & \text{if } \Delta f_k \leq 0 \end{cases} \tag{5.4}$$

where T_n is the temperature of the simulated annealing at iteration n, f_k is the kth objective function, $\overrightarrow{x_n}$ is the solution we obtain at iteration n, $\overrightarrow{y}$ is the current point at iteration n, and $\Delta f_k = f_k(\overrightarrow{y}) - f_k(\overrightarrow{x}_n)$.

Next, once all these probabilities have been computed, we merge them. Many ways to do this exist; the following are two possibilities:

- We compute the product of all the probabilities:

$$t(\Pi, \lambda) = \prod_{k=1}^{N} (\pi_k)^{\lambda_k} \tag{5.5}$$

  where Π is the set of the π_k, $k = \{1, \cdots, N\}$, and λ is the set of the λ_k, $k = \{1, \cdots, N\}$.
- We take the smallest probability:

$$t(\Pi, \lambda) = \min_{k=1,\cdots,N} (\pi_k)^{\lambda_k} \tag{5.6}$$

Here, λ_k corresponds to a weight coefficient related to an objective function. This coefficient allows us to take into account some preferences between the various objective functions.

Lastly, there is the step of selection of a new solution. This step is performed using the following rule for acceptance of a solution:

First, we have

$$f_{eq}(\overrightarrow{x}) = \sum_{i=1}^{N} w_i \cdot f_i(\overrightarrow{x}) \tag{5.7}$$

and $\Delta f_{eq} = f_{eq}(\overrightarrow{x_{n+1}}) - f_{eq}(\overrightarrow{x_n})$.

- If $\Delta f_{eq} \leq 0$ then $\overrightarrow{x_{n+1}} = \overrightarrow{y}$.
- If $\Delta f_{eq} > 0$ then:
  $\overrightarrow{x_{n+1}} = \overrightarrow{y}$ with probability p.
  $\overrightarrow{x_{n+1}} = \overrightarrow{x_n}$ with probability $1 - p$.

This method for managing a multiobjective problem may seem not very different from the weighted-sum-of-objective-functions method. Nevertheless, a major difference exists: in the weighted-sum-of-objective-functions method, we begin by performing a weighted sum of objective functions (see Eq. 5.7).

When this weighted sum is "handled" by simulated annealing, it is transformed into an exponential expression:

$$p = \exp\left(-\frac{\Delta f_{eq}}{T_n}\right) \tag{5.8}$$

This acceptance probability can be decomposed in the following way:

$$p = \exp\left(-\frac{\sum_{i=1}^{N} w_i \cdot (f_i(\overrightarrow{x}) - f_i(\overrightarrow{x}^*_n))}{T_n}\right) \tag{5.9}$$

$$p = \prod_{i=1}^{N} \exp\left(-\frac{w_i \cdot (f_i(\overrightarrow{x}) - f_i(\overrightarrow{x}^*_n))}{T_n}\right) \tag{5.10}$$

$$p = \prod_{i=1}^{N} \left(\exp\left(-\frac{\Delta f_i}{T_n} \right) \right)^{w_i} \tag{5.11}$$

$$p = \prod_{i=1}^{N} \pi_i^{w_i} \tag{5.12}$$

The acceptance condition for a solution is the following:

- If $\Delta f_{eq} \leq 0$ then $\overrightarrow{x_{n+1}} = \overrightarrow{y}$.
- If $\Delta f_{eq} > 0$ then:
  $\overrightarrow{x_{n+1}} = \overrightarrow{y}$ with probability p.
  $\overrightarrow{x_{n+1}} = \overrightarrow{x_n}$ with probability $1 - p$.

The difference between the MOSA method and the weighted-sum-of-objective-functions method lies in the way we compute the acceptance probability. In the MOSA method, we begin by computing a probability for each objective function,

$$p_i = \exp\left(-\frac{\Delta f_i}{T_n} \right) \tag{5.13}$$

whereas with the weighted-sum-of-objective-functions method, the probability is computed for the equivalent objective function:

$$p = \exp\left(-\frac{\Delta f_{eq}}{T_n} \right) \tag{5.14}$$

The MOSA method considers a bigger part of the multiobjective optimization problem than does a weighted-sum-of-objective-functions method which uses simulated annealing.

5.4.2.3 Discussion

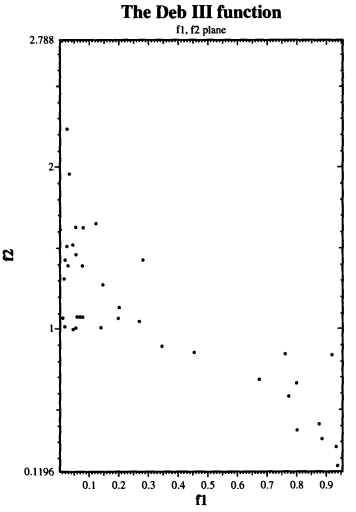
In [Ulungu et al. 99], the MOSA method is used the following way, for a biobjective combinatorial optimization problem.

1. Two weight coefficients are computed.
2. For each weight, a population of solutions is computed. Each pair of solutions will produce solutions which "fill" an area of the tradeoff surface.
3. The two populations are gathered together, and then the dominated solutions of the gathered population are removed.

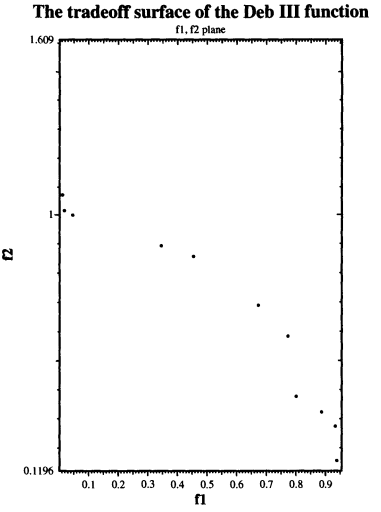
This method behaves well because, at a high temperature, the simulated annealing spreads the solutions over the whole tradeoff surface, not over a small part of it. At the end of the optimization, the current temperature must be "sufficiently high" to produce a "stochastic effect" which spreads the solutions over the tradeoff surface.

This effect can be demonstrated using the following example, which is related to a nonconvex biobjective test problem, the Deb test problem:

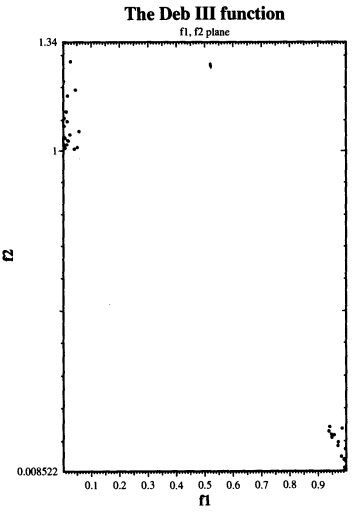
1. We computed 40 pairs of weight coefficients.
2. For each pair, a solution of the equivalent mono-objective problem was searched for using the simulated annealing method.



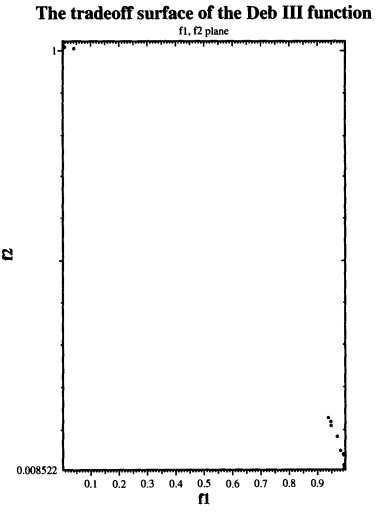
(a) The solution set obtained after 10^5 iterations.



(b) The tradeoff surface after 10^5 iterations.



(c) The solution set obtained after 10^6 iterations.



(d) The tradeoff surface after 10^6 iterations.

Fig. 5.6. Bias of the stochastic effect on the quality of the tradeoff surface.

3. This sequence was applied to the test problem for 10^5 iterations, and then 10^6 iterations.

As we can notice in Fig. 5.6, the search for the tradeoff surface with a weighted-sum-of-objective-functions method coupled to a simulated annealing method produces the whole tradeoff surface after 10^5 iterations, whereas the weighted-sum-of-objective-functions method is known to be inefficient on a nonconvex problem.

Next, after 10^6 iterations, we see that the solutions are spread around two points: this result shows the convex shape of the Deb non convex test problem.

So, as this figure shows us, the stochastic effect allows us to find solutions inside areas which are "unreachable" when the weighted-sum-of-objective-functions method is used.

5.5 Tabu search

This method, created by F. Glover [Glover 86, Glover 90, Glover et al. 97], is constructed so as to "avoid" local minima of the objective function. It is a combinatorial optimization technique which is presented here as an alternative to the simulated annealing.

Starting from any configuration, the tabu search method breeds a sequence of configurations which must reach the optimal configuration. At each iteration, the mechanism that changes a configuration (x_n) into the next (x_{n+1}), is the following:

- We construct the set of "neighbors" of x_n, that is to say, the set of configurations that are reachable in one "elementary" step from x_n (if this set is too large, we extract at random a fixed-size subset of it). This gives Neighborhood (x_n), the set (or subset) of points "close to" x_n.
- We evaluate the objective function f of our problem for each of the configurations which belong to Neighborhood (x_n). The configuration x_{n+1}, which follows the configuration x_n in the Markov chain constructed by the method, is the configuration of Neighborhood (x_n) on which f takes its minimal value.

Note that the configuration x_{n+1} is accepted even if $f(x_{n+1}) > f(x_n)$: this is the detail that allows tabu search to avoid local minima of f.

Nevertheless, such a process does not work in general, because there is a risk of returning to a configuration already evaluated during a preceding iteration, which leads to the appearance of a cycle. To avoid this phenomenon, we construct, at each iteration, a "tabu list" of forbidden movements; this list–which gave its name to the method–contains the reverse movements $(x_{n+1} \rightarrow x_n)$ of the last m movements $(x_n \rightarrow x_{n+1})$ that we have tested (usually $m = 7$). The search for a successor of the current configuration x_n is restricted to neighbors of x_n which can be reached without using a movement in the tabu list. The process can be stopped when we have performed a fixed number of iterations, without improving the best solution obtained so far.

The algorithm that we have described so far is called "simple tabu". According to [De Werra 90], tabu search should be more efficient than simulated annealing for the "graph coloring" problem. Nevertheless, the way we construct the tabu list–which, for reasons of economy of memory, contains forbidden movements, and not forbidden configurations–can forbid access to certain solutions, even though they have not been visited already. To avoid this drawback, we can use a more complex method called "generalized tabu", which allows us to cancel the tabu status of a movement when the

profit hoped for is "sufficient" (this circumstance is provided for using the notion of "aspiration level", which is detailed in [De Werra 90]).

The algorithm of this method is presented in algorithm 4.

Algorithm 4 Tabu search.

init $L = \emptyset$ (list of Tabu points)
init x (starting point)
init k_{fin} (max number of iterations)
init $k = 0$ and $l = 0$
while(not(end))
  $y =$ Neighborhood $(x) - L$ (we take a neighborhood without tabu points)
  $\Delta C = C(y) - C(x)$
  $x = y$ with $y \in$ Neighborhood $(x) - L$ such that $\min C(y) - C(x)$
  $L = L + \{x\}$
  $k = k + 1$
  Update L (we remove tabu movements from L using aspiration criteria)
  if $k = k_{fin}$ then end(while)
repeat

5.6 Genetic algorithms

Genetic algorithms are inspired by classical genetics (see [Goldberg 94]): we consider a "population" of points spread through the search space.

Before we explain in detail the behavior of a genetic algorithm, we shall present some words of the vocabulary of genetics. These words are often used to describe a genetic algorithm.

- Genotype or chromosome: this is another way to say "individual".
- Individual: corresponds to the coding, as a gene, of a potential solution of an optimization problem.
- Gene: a chromosome is composed of genes. With a binary coding, a gene has a value of either 0 or 1.
- Phenotype: each genotype represents a potential solution of an optimization problem. The value of this potential solution is called a phenotype.

This vocabulary is illustrated in Fig. 5.7.

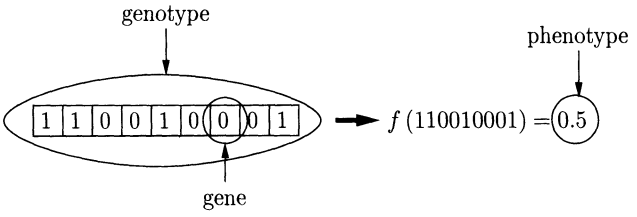
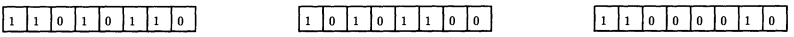


Fig. 5.7. The vocabulary of the genetic algorithms.

- Each individual is "coded" as a gene. For example, most often, we make a correspondence between a binary chain and an individual (this binary chain is an image of the position of the individual in the search space). For example, three individuals (of 8 bits each) are represented in Fig. 5.8.



(a) Individual 1. (b) Individual 2. (c) Individual 3.

Fig. 5.8. Three individuals.

- To each individual, we assign an efficiency (this value is also known as the "adaptation"). This efficiency corresponds to the performance of an individual in solving a given problem. For example, if we consider the problem of maximization of a function, the efficiency of the individual will increase with its ability to maximize this function.
  If we take the example of the maximization of a function $f(x)$, we may have the values shown in Table 5.1.

Table 5.1. An example of efficiency values.

$f(x)$	10	20	5
Efficiency	2	4	1
Individual	1	2	3

  As we can see in this example, individual 1 is less efficient than individual 2. That is, individual 1 maximizes $f(x)$ less than individual 2 does.
- After having determined the efficiency of the individuals, we perform a reproduction. We copy the individuals in proportion to their efficiency. For example, if individual 1 is reproduced two times, individual 2 will be reproduced four times and individual 3 will be reproduced one time. This behavior is similar to that in the real life. The better an individual is adapted to its environment (it protects itself better against predators, and it eats better), the more is its reproductive capacity.
- Once we have determined the efficiency function and performed the breeding step, we perform "crossovers" (defined later) between individuals. The more efficient an individual is in solving a problem, the more it shares parts of its chromosome with a large number of other individuals. To select an individual for this operation, we use a wheel on which each individual fills an area proportional to its efficiency (this process is called *roulette wheel selection*). This wheel, in the case of the example described above, is represented in Fig. 5.9.
  For this example, we have the results shown in Table 5.2.
- We now merge the starting individual with the individual that we have associated with it. This operation is called a crossover. To do so, we select randomly a position in the binary code of the individuals. At this position, we break the code and swap

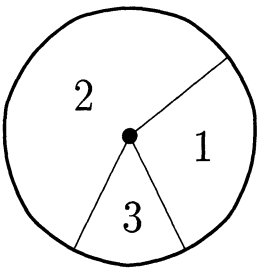


Fig. 5.9. Wheel used to select an individual for a crossover.

Table 5.2. Crossover, first step.

Selection for the crossover	
"Male"	"Female"
Selected individual with respect to its efficiency	Randomly selected individual
1	2
1	3
2	2
2	2
2	3
2	1
3	2

the right-hand parts between the two individuals. An example of a crossover is represented in Fig. 5.10.

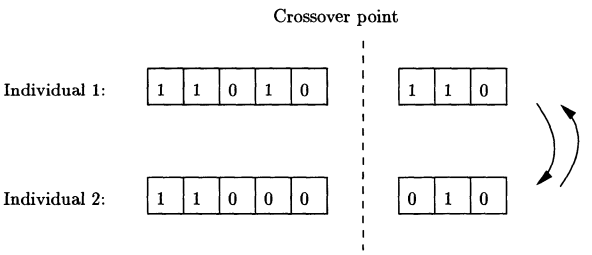


Fig. 5.10. Crossover, second step.

We repeat this operation for all the individuals in Table 5.2. We obtain the results shown in Table 5.3.

Here, we notice that we have two new elements in the result column for each pair of individuals in the first two columns. So, our population increases in size (it doubles its size). At this point, either we keep the whole population (but remove copies), or we remove the less efficient individuals, so as to keep a constant number of individuals in the population. Other methods of reduction of the size of the population can be used here instead (for example, we can select at random individuals to be removed).

Table 5.3. Crossover, final result.

"Male"	"Female"	Crossover position	Result
1 1 0 1 0 1 1 0	1 0 1 0 1 1 0 0	1	1 0 1 0 1 1 0 0
1	2		1 1 0 1 0 1 1 0
1 1 0 1 0 1 1 0	1 1 0 0 0 0 1 0	3	1 1 0 0 0 0 1 0
1	3		1 1 0 1 0 1 1 0
1 0 1 0 1 1 0 0	1 0 1 0 1 1 0 0	2	1 0 1 0 1 1 0 0
2	2		1 0 1 0 1 1 0 0
1 0 1 0 1 1 0 0	1 0 1 0 1 1 0 0	5	1 0 1 0 1 1 0 0
2	2		1 0 1 0 1 1 0 0
1 0 1 0 1 1 0 0	1 1 0 0 0 0 1 0	7	1 0 1 0 1 1 0 0
2	3		1 1 0 0 0 0 1 0
1 0 1 0 1 1 0 0	1 1 0 1 0 1 1 0	4	1 1 0 1 1 1 0 0
3	1		1 0 1 0 0 1 1 0
1 1 0 0 0 0 1 0	1 0 1 0 1 1 0 0	4	1 1 0 1 1 1 0 0
3	2		1 0 1 0 0 1 1 0

- Lastly, we can proceed to a mutation in a gene of an individual. To do so, we select at random some individuals (in general, the probability of selection for a mutation is small). Next, for each selected individual, we choose at random the position where the mutation will take place. Lastly, at this position, we change a 0 into 1 or vice versa.
- We start the process and repeat until we reach a stopping criterion (for example, some maximum number of iterations).

The pseudo-code of the genetic algorithm is presented in algorithm 5.

Algorithm 5 A genetic algorithm.

```
Population initialization
Objective function computation
Computation of the efficiency
For i = 1 to MaxIter
  Random selection,
  Selection proportional to the efficiency,
  Crossover,
  Mutation,
  Objective function computation,
  Computation of the efficiency,
End For
```

The advantage of this method is that, in some cases, it can produce a set of optimal solutions.

5.7 Multiobjective optimization and genetic algorithms

Genetic algorithms are well adapted to dealing with multiobjective optimization problems. The huge number of papers which have been published about this subject testify to this. Moreover, this domain is very dynamic and has not stopped growing.

In the preceding section, we have seen a description of a genetic algorithm. The way a genetic algorithm behaves when it is used to solve a mono-objective optimization problem is represented in Fig. 5.11.

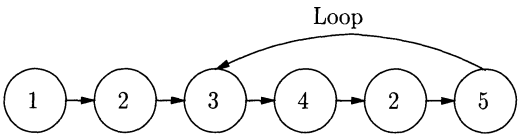


Fig. 5.11. The behavior of a mono-objective genetic algorithm.

The steps shown in this figure are as follows:

1 Population initialization.
2 Computation of the efficiency of the individuals in the population.
3 Crossover.
4 Mutation.
5 Selection.

The way a genetic algorithm behaves when it is used to solve a multiobjective optimization problem is represented in Fig. 5.12.

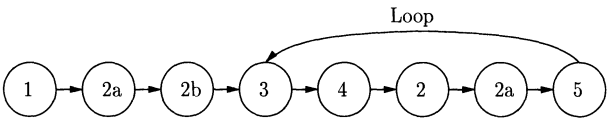


Fig. 5.12. The behavior of a multiobjective genetic algorithm.

The steps shown in this figure are as follows:

1 Population initialization.
2a Computation of the efficiency of the individuals in the population.
2b Vector/efficiency transformation.
3 Crossover.
4 Mutation.
5 Selection.

The one and only difference that we can notice is step 2b. In this step, we transform a vector (which contains the objective function values of each individual) into an efficiency. Various strategies, which will be described in the following subsections, can be adopted. All of the genetic algorithms which deal with multiobjective optimization problems can be described in the manner sketched above.

We shall now focus on various methods which have been used to deal with multiobjective optimization problems. These methods can be divided into two families:

- The first family is composed of "nonaggregative" methods. In these methods, there is no merging of the various objective functions to transform the original problem into a mono-objective one.

- The second family is composed of "aggregative" methods. The goal of these methods is to transform the original problem into a mono-objective one using an equivalent objective function.

Note: Most of the algorithms use a "distance" which characterizes the difference between two individuals. The distance between individual i and individual j is denoted $d(i,j)$. We compute this distance in the following way: each difference between the genes of the individuals counts as $+1$ (see Fig. 5.13). This distance is called the Hamming distance.

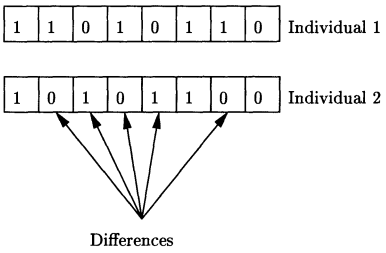


Fig. 5.13. The Hamming distance.

For the example shown in Fig. 5.13, we notice that there are five differences between individual 1 and individual 2. Consequently, the distance between individual 1 and individual 2 is equal to

$$d(1,2) = 5 \tag{5.15}$$

5.7.1 A "nonaggregative" method

5.7.1.1 *The vector evaluated genetic algorithm* (VEGA) method

Principle

This method allows us to deal with a multiobjective optimization problem without having to merge all the objective functions into one [Coello 98, Deb 99].

Presentation of the method

The VEGA algorithm uses a population of N individuals. These N individuals are divided into k groups (k corresponds to the number of objective functions of our problem) of $\frac{N}{k}$ individuals (where N is a multiple of k). At each group, we compute an objective function. This objective function allows us to determine the efficiency of an individual in the group. Then, individuals are mixed and we perform a crossover step with respect to the efficiency of each individual.

In Fig. 5.14, we have represented the various steps of the method. Here, we have represented the various kinds of population used during the running of the method (a set of either individuals or groups of individuals).

The following is the description of the various steps:

Step 1: Iteration i. Initialization of a population of size N.

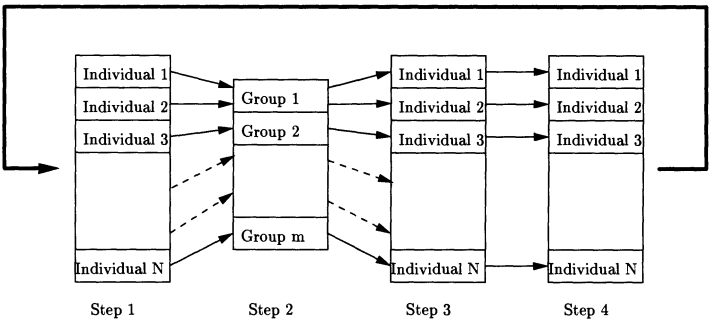


Fig. 5.14. Principle of the VEGA algorithm

Step 2: Creation of k groups (subpopulations).
Step 3: Computation of the efficiencies. Mixing of individuals.
Step 4: We apply the classical genetic algorithm (crossover–mutation–selection). Then we go on to the next iteration $(i+1)$.

Discussion

The danger with this method is that we may obtain, at the end of the optimization, a population composed of mean individuals with respect to all the objective functions. Such a population does not allow one to obtain a good approximation of the tradeoff surface. Instead, the population will be concentrated around a mean "point". Tricks have been found to overcome this effect (use of species inside a group, etc.).

Moreover, it has been shown that this method is equivalent to the weighted-sum-of-objective-functions method. Therefore, it does not allow us to find all of the solutions hidden in concavities.

5.7.2 The "aggregative" methods

5.7.2.1 *The multiple objective genetic algorithm* (MOGA) method

Principle

This method is presented in [Deb 99] and [Fonseca et al 93]. It uses a domination relation to compute the efficiency of an individual.

Presentation of the method

This method is based on the relation of domination in the Pareto sense. Here, the "rank" of an individual (an order number which allows one to rank an individual with respect to the others) is given by the number of individuals which dominate the considered individual.

For example, if we consider an individual x_i at generation t which is dominated by $p_i^{(t)}$ individuals, the rank of this considered individual is given by

$$rank\,(x_i, t) = 1 + p_i^{(t)} \tag{5.16}$$

To all the nondominated individuals, we assign the rank 1. So, we assign a high rank (and therefore a high penality) to the dominated individuals.

For the computation of the efficiency we can follow these steps:

1. Rank individuals with respect to their rank.
2. Assign an efficiency to each individual by interpolating from the best rank (rank 1) to the worst rank (rank n), using a function $f(rank)$. This function is often linear (see Fig. 5.15).

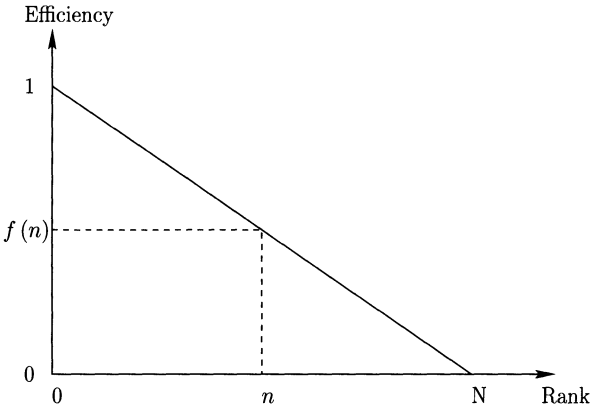


Fig. 5.15. An example of an efficiency function.

The algorithm of this method is presented in algorithm 6.

Algorithm 6 The MOGA algorithm.

```
Population initialization
Computation of the values of the objective functions
Assignment of a rank by use of domination
Assignment of an efficiency by use of the rank
For i = 1 to G
  Random selection in proportion to the efficiency
  Crossover
  Mutation
  Computation of the objective functions
  Assignment of a rank by use of domination
  Assignment of an efficiency by use of the rank
End For
```

Discussion

In some cases, the MOGA method does not allow us to obtain a good diversity of solutions for the approximation of the tradeoff surface. The NSGA and NPGA methods, which will be presented below, allow us to overcome this drawback.

For this kind of method, the number of comparisons performed during a generation is

$$N \cdot (N-1) \cdot n \tag{5.17}$$

where N corresponds to the number of points in the population and n to the number of objective functions.

5.7.2.2 The *nondominated sorting genetic algorithm* (NSGA) method

Principle

This method is based upon the MOGA method described in the preceding section. The main difference occurs during the computation of the efficiency of an individual [Srinivas et al. 93].

Presentation of the method

This method is based on a classification using many levels of individuals. In a first step, before proceeding to the selection, we assign a rank to each individual of the population (using the Pareto rank, see Sect. 1.6.2). All the nondominated individuals with the same rank are ranked into this category. To this category, we assign a dummy efficiency, which is inversely proportional to the Pareto rank of the considered category.

We can now say that the individuals of the considered category are uniformly spread inside it. So, we want a good approximation of the tradeoff surface (see Sect. 1.11), or, alternatively, a good diversity of solutions.

To maintain this diversity inside the population, we assign to these ranked individuals a new efficiency value. To do so, we use the following formula:

$$m_i = \sum_{j=1}^{K} Sh\left(d\left(i,j\right)\right) \tag{5.18}$$

$$Sh\left(d\left(i,j\right)\right) = \begin{cases} 1 - \left(\frac{d(i,j)}{\sigma_{share}}\right)^2 & \text{if } d\left(i,j\right) < \sigma_{share} \\ 0 & \text{otherwise} \end{cases} \tag{5.19}$$

Here, K corresponds to the number of individuals in the considered category, and $d\left(i,j\right)$ corresponds to a distance between individual i and individual j such as the distance introduced in Sect. 5.7. It is always possible to use a classical distance instead of the one presented. σ_{share} is the influence distance. As we can see from the above expression, all the individuals which are sufficiently close (for which the distance $d\left(i,j\right)$ is lower than σ_{share}) are taken into account in the computation of m_i. The others are ignored.

The efficiency value of individual i inside the considered category is

$$f_i = \frac{F}{m_i} \tag{5.20}$$

where F is the value of the efficiency assigned to the category to which the individual belongs. Figure 5.16 shows the various parameters which play a role during the computation of the efficiency of an individual.

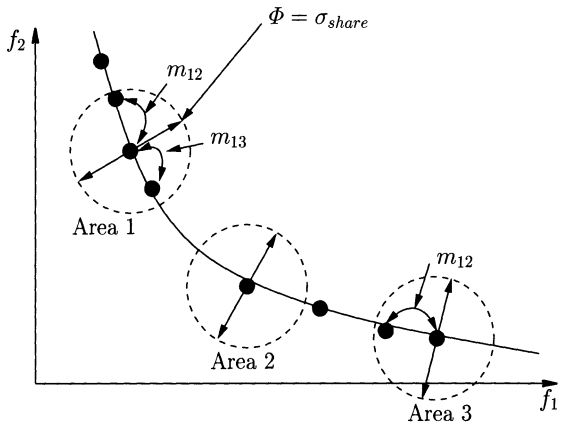


Fig. 5.16. The efficiency computation.

As we can see in Fig. 5.16, the parameter σ_{share} allows us to define an influence area for the computation of the efficiency of an individual.

After performing this computation for individual i, we ignore the individuals of this group. The process continues with the remaining individuals, where we operate a new classification, a new categorization and a new sharing operation. This process is repeated until all the individuals have been treated.

Because the individuals which have a Pareto rank equal to 1 have the best efficiency, they are reproduced more than the others. This property allows us to obtain a quicker convergence toward the tradeoff surface. The sharing allows us to maintain a uniform spread over the tradeoff surface.

The algorithm of this method is given in algorithm 7.

Algorithm 7 The NSGA algorithm.

```
Population initialization
Computation of the values of the objective functions
Assignment of a rank by use of the domination for
  each tradeoff surface
Computation of the enumeration of the neighboring points
Assignment of a shared efficiency
For i = 1 to G
  Random selection in proportion to the efficiency
  Crossover
  Mutation
  Computation of the values of the objective functions
  Assignment of a rank by use of the domination for
    each tradeoff surface
  Computation of the enumeration of the neighboring points
  Assignment of a shared efficiency
End For
```

We can distinguish two kinds of niches:

- Genotypic niches:

$$Sh_{gen}(d(i,j)) = \begin{cases} 1 - \left(\frac{d_{hamming}(i,j)}{\sigma_{gen}}\right)^2 & \text{if } d_{hamming}(i,j) < \sigma_{gen} \\ 0 & \text{otherwise} \end{cases} \quad (5.21)$$

Here, we compute the enumeration of niches (of radius σ_{gen}) inside the chromosome space; the distance taken into account is the Hamming distance ($d_{ham}(i,j)$).

- Phenotypic niches:

$$Sh_{phen}(d(i,j)) = \begin{cases} 1 - \left(\frac{d_{euclid}(i,j)}{\sigma_{phen}}\right)^2 & \text{if } d_{euclid}(i,j) < \sigma_{phen} \\ 0 & \text{otherwise} \end{cases} \quad (5.22)$$

Here, we compute the enumeration of niches (of radius σ_{phen}) inside the phenotypic space; the distance taken into account is the Euclidean distance (if we consider a continuous optimization problem).

In addition, we can consider combined niches:

- Combined niches:

$$Sh_{comb}(d(i,j)) = \begin{cases} 1 - \left(\frac{d_{hamming}(i,j)}{\sigma_{gen}} \cdot \frac{d_{euclid}(i,j)}{\sigma_{phen}}\right)^2 & \text{if } d_{euclid}(i,j) < \sigma_{phen} \\ & \text{and } d_{hamming}(i,j) < \sigma_{gen} \\ 1 - \left(\frac{d_{euclid}(i,j)}{\sigma_{phen}}\right)^2 & \text{if } d_{euclid}(i,j) < \sigma_{phen} \\ & \text{and } d_{hamming}(i,j) \geq \sigma_{gen} \\ 1 - \left(\frac{d_{hamming}(i,j)}{\sigma_{gen}}\right)^2 & \text{if } d_{euclid}(i,j) \geq \sigma_{phen} \\ & \text{and } d_{hamming}(i,j) < \sigma_{gen} \\ 0 & \text{otherwise} \end{cases}$$

Here, we compute the enumeration of niches (of radius σ_{gen} and σ_{phen}) inside the genotypic and phenotypic spaces.

A phenotypic niche allows us to "tune" the diversity of the solutions in the objective function space, whereas a genotypic niche allows us to tune the diversity of the solutions inside the parameter spaces.

The combined niche allows us to obtain solutions which are diversified both in the objective function space and in the parameter spaces.

Discussion

The efficiency of this method lies in the fact that all objective functions are reduced to a dummy efficiency value obtained using the ranking with respect to the Pareto rank. This method has the drawback that it is sensitive to the value of σ_{share}.

For this method, the number of comparisons performed on a population for one generation is the same as with the MOGA method. Nevertheless, an excess cost due to the sharing appears. This excess cost is proportional to $N \cdot (N-1)$.

5.7.2.3 *The niched pareto genetic algorithm* (NPGA) method

Principle

This method is based upon the NSGA method described in the preceding section. The main difference occurs during the selection process [Horn et al. 93].

Presentation of the method

In a classical genetic algorithm, the method of selection between two individuals uses a selection wheel such as the one described in Sect. 5.6. In the present method, the way in which we select individuals changes. Instead of comparing two individuals, we use a group of I individuals (typically ten). If the individuals are either dominated or nondominated, we use a sharing of the efficiency value as described in Sect. 5.7.2.2.

The algorithm of this selection function, for a multiobjective optimization problem where all objective functions have to be maximized, is represented in algorithm 8. Its implementation inside a genetic algorithm is represented in algorithm 9.

Algorithm 8 Details of the selection function.

```
function selection
 Mixes(random_pop_index)
 Candidate_1 = random_pop_index[1]
 Candidate_2 = random_pop_index[2]
 Candidate_1_dominated = false
 Candidate_2_dominated = false
 for Set_index_comp = 3 to t_dom + 3 do
 /* select t_dom individuals at random inside S */
 begin
  Indiv_comp = random_pop_index[Set_index_comp]
  if S[Indiv_comp] dominate S[Candidate_1]
    then Candidate_1_dominated = true
  if S[Indiv_comp] dominate S[Candidate_2]
    then Candidate_2_dominated = true
 end
if (Candidate_1_dominated AND NOT Candidate_2_dominated)
  then return Candidate_2
  else if (NOT Candidate_1_dominated AND Candidate_2_dominated)
         then return Candidate_1
  else
    do Sharing
end
```

The selection function chooses an individual from the population S. The values of t_{dom} and σ_{share} must be given by the user.

Discussion

As this method does not use the Pareto selection on the whole population, but just on a part at each run, it is very fast. Nevertheless, it requires from the user the parameters t_{dom} and σ_{share}, on which the performance of the algorithm depends. In [Deb 01], we can find techniques to "automatically" tune these parameters.

5.7.2.4 The *weighted-average-ranking genetic algorithm* (WARGA) method

Principle

This method is based upon the MOGA method. The main difference lies in the way we establish the domination relation between two individuals.

Algorithm 9 The NPGA algorithm.

Population initialization
Computation of the values of the objective functions
For $i = 1$ to G
  Tournament selection between two individuals
   (use of t_{dom})
    Only candidate 1 is dominated: Select candidate 2
    Only candidate 2 is dominated: Select candidate 1
    Both candidates are dominated or are not dominated:
      Perform an efficiency sharing
      Select the candidate with the smaller number of
       neighbors (use of σ_{share})
  Crossover
  Mutation
  Computation of the values of the objective functions
End For

Presentation of the method

In this method, the domination rank is computed as described in algorithm 10, which computes the weighted average ranking (WAR).

Algorithm 10 The WAR relation.

A solution to compare (vector of dimension n).
S set of solutions (S does not contain A).

repeat
  $i = 1$
  Be given A_i the variable i of the point A.
  N_i = number of points from S better than A_i with respect to the variable i.
  $i = i + 1$
until $i > n$

The rank assigned to A worth $N = \sum_{i=1}^{n} N_i$

The efficiency of an individual is then computed in the same way as with the MOGA method.

Let us take an example. In Fig. 5.17 (where f_1 and f_2 are to be minimized), we have:

- With respect to f_1, A is dominated by 4 points.
  With respect to f_2, A is dominated by 0 point.
  So, $N(A) = 4 + 0 = 4$
- With respect to f_1, B is dominated by 1 point.
  With respect to f_2, B is dominated by 3 points.
  So, $N(B) = 1 + 3 = 4$

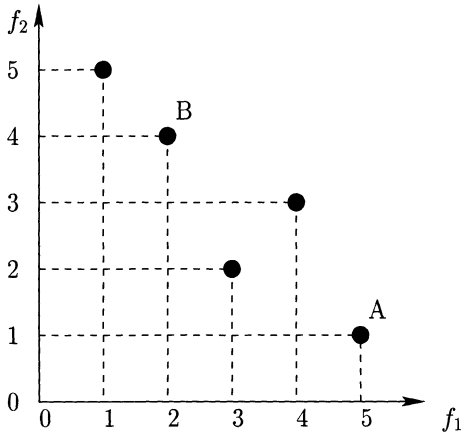


Fig. 5.17. The WAR relation.

Discussion

For this method, the number of comparisons required to perform the computation of the domination ranks for a population of N individuals, considering one generation, is

$$N \cdot (N-1) \cdot n \tag{5.23}$$

Nevertheless, the algorithm which implements the WAR domination is much simpler than of the classical domination relation.

5.7.2.5 Elitism in genetic algorithms

The SPEA (*strength Pareto evolutionary algorithm*) method presented in [Zitzler et al. 98] ranks the nondominated solutions in a population separated from the current population. Individuals are then selected from inside the current population for a crossover step. Individuals with which they will reproduce are selected from the nondominated separate population.

In [Gandibleux 00], a similar method has been used, with the following difference: the elite population is initialized with two extreme individuals, as explained below. The starting problem is a biobjective one. We start from the following problem:

$$\begin{vmatrix} \text{minimize} & f_1(\vec{x}) \\ \text{minimize} & f_2(\vec{x}) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{vmatrix}$$

The first initial individual of the elite population ($\vec{x_1}$) is obtained by solving

$$\begin{vmatrix} \text{minimize} & f_1(\vec{x}) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{vmatrix}$$

The second initial individual of the elite population ($\vec{x_2}$) is obtained by solving

$$\begin{array}{|ll} \text{minimize} & f_2(\overrightarrow{x}) \\ \text{with} & \overrightarrow{g}(\overrightarrow{x}) \leq 0 \\ \text{and} & \overrightarrow{h}(\overrightarrow{x}) = 0 \end{array}$$

This "precaution" allows us to "guarantee" that the tradeoff surface will spread between these two extreme solutions. Indeed, the objective function vectors $(f_1(\overrightarrow{x_1}), f_2(\overrightarrow{x_1}))$ and $(f_1(\overrightarrow{x_2}), f_2(\overrightarrow{x_2}))$ are nondominated, from generation to generation. They will transmit all their characteristics to other individuals. At the end of the optimization, all the nondominated individuals will be spread over the whole tradeoff surface.

In [Parks et al. 98], an archive of nondominated individuals is used. During the first steps of the crossover step, those authors add some individuals which belong to the archive of nondominated individuals in the current population. [Parks et al. 98] conclude that the performance is better when the proportion of injected individuals is higher.

A definition of elitism is given in [Laumanns et al. 00]:

Definition 24. ***Elitism***
Let P^t, a population obtained from a given genetic algorithm after t iterations (generations), be given. Let $P^t(x)$, the probability that an individual $x \in P^t$ is selected for the crossover and/or mutation step during generation t, be given. Then the genetic algorithm is said to be "elitist" if and only if, for a given preference relation $\prec$ for a given decision problem, the following condition is satisfied:

$\forall t \in \mathbb{N}, \exists \overrightarrow{x} \in P^{*t}$ *such that* $P^{t+1}(\overrightarrow{x}) > 0$

with

$$P^{*t} = \left\{ \overrightarrow{x} \in P^t \mid \nexists \overrightarrow{x'} \in P^t \mid \overrightarrow{x'} \prec \overrightarrow{x} \right\}$$

$$P^t = \bigcup_{r \leq t} P^r$$

The operation $\bigcup_{r \leq t} P^r$ corresponds to the merging of all the sets P^r with $r \leq t$. So P^{*t} corresponds to the nondominated individuals of P^t.

The main idea which we wish to emphasize in this discussion of elitism is that it is important not to forget to include the best solutions that we obtain, one generation after another.

Lastly, in mono-objective optimization, it is common to save the best solution obtained during the optimization process. In multiobjective optimization, a similar process consists in putting into an archive all the nondominated individuals obtained during the optimization. This archive represents the best approximation of the tradeoff surface we have obtained so far. Moreover, as this archive is used just to save nondominated individuals, it does not change the behavior of the genetic algorithm.

5.8 Annotated bibliography

[Sait et al. 99] This book provides descriptions of some metaheuristics: simulated annealing, tabu search, genetic algorithms, etc. The methods are clearly

described, and theoretical elements related to convergence of these methods are presented. There are also some industrial examples to illustrate each of these methods.

[Goldberg 94] A reference introductory book about genetic algorithms. Everything related to genetic algorithms is presented clearly.

[Davis 91] A book which presents a whole set of applications of genetic algorithms.

[Glover et al. 97] This is a reference book about tabu search. Unlike the [Goldberg 94] genetic algorithms book, this book is more complex. The presentation is sometimes hard to follow. Nevertheless, we advise you to read this book before using tabu search.

[Ausiello et al. 99] A book about the complexity of algorithms and approximation methods. This book talks about algorithm "performances". The technical level of this book is high but the presentation of the various concepts is clear. It also contains an encyclopedic chapter about various optimization problems, with the main results about each problem.

[Deb 01] The reference book about how to use genetic algorithms in a multiobjective context. All the methods which use a genetic algorithm are described. The author illustrates each method with a simple "handwritten" example. These various examples allow the reader to understand the behavior of the methods in detail.

6

Decision aid methods

6.1 Introduction

The methods we have presented so far have been based on the domination relation. This relation (which we can define in various ways: Pareto domination or lexicographic domination, for example) allows us to "filter" elements of a set, and to just keep all the elements we can compare themselves. However, there exists another way to obtain a set of solutions, which is based on the setting up of an order relation between these various elements. With the order relation so defined, we can obtain a set of solutions (with a partial order relation) or one and only one solution (with a complete order relation). The other major difference, with respect to the "classical" multiobjective optimization methods, comes from the fact that the decision aid methods work only on discrete sets of points ("classical" multiobjective optimization methods can work on continuous sets). Moreover, decision aid methods allow us to answer several problems, listed in Table 6.1.

Table 6.1. The various problems answered by decision aid methods.

Problem	Result	Procedure
Choice of a subset of the "best" actions or, by default, the most "satisfactory".	Choice	Selection
Sorting by assigning actions to predefined categories.	Sorting	Assignment
Arranging of equivalence classes composed of actions, these classes are completely or partially sorted.	Arranging	Classification

The development of decision aids was based on the following facts: in some cases, when we have to compare three actions together, we can encounter a "looping" between these actions. This phenomenon is called *intransitivity* (this term will be defined mathematically in the next section) of a preference and of indifference, or the *Condorcet paradox*. It can be summarized the following way: given three actions A, B and C, we can have $A \geq B$, $B \geq C$ and $C \geq A$ (here, the symbol $\geq$ corresponds to the preference relation). This paradox is represented in Fig. 6.1.

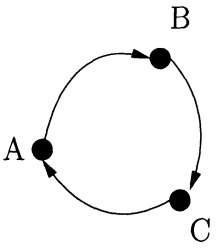


Fig. 6.1. The Condorcet paradox.

Another example, which shows the intransitivity of indifference, is the example of the sandwiches [Scharlig 85]. Somebody offers you two sandwiches, the first one is made of bread and ham and the second one is made of a little more bread and a little less ham. So, let us say you take the sandwich with a little more bread (because you prefer having more bread than ham). Now, somebody offers you another sandwich with less ham than yours and more bread than yours. Here again, you choose the sandwich with more bread (again because you prefer having more bread than ham). This process is repeated until you have in your hands a piece of bread without ham.

During this whole process, you always have preferred a solution with more bread. Now, however, when you consider the whole problem, it is clear that you have a preference for the sandwich with ham over the sandwich without any ham.

So, as with the preference relation, the indifference relation is also intransitive.

It is this set of properties which has provoked the development of decision aids. Indeed, classical multiobjective optimization does not take into account these properties and, in some cases, leads to incoherent solutions in the solution of "multicriteria" optimization problems (this term is often preferred to the term "multiobjective" in the decision aid context).

Decision aid methods aim to take into account this intransitivity property of order relations, and they also provide families of methods intended to solve problems (selection, sorting and arrangement) different from the ones handled by classical multiobjective optimization.

When we look at the domain of decision aids, we notice some redundant terms:

- action,
- classification,
- complete or partial order.

An "action" corresponds to an object, a decision, a candidate or something else. The selection (or the sorting, i.e. the classification) operates on this entity.

So, a decision aid method can allow us to choose the best action, or to classify the actions with respect to one or more criteria.

Depending on the kind of classification we wish to perform, we can use various kinds of classification rules or choice rules. These rules define a complete order (if all the actions are classified) or a partial order (if after the classification, some actions are still not comparable, so they are not classified).

6.2 Definitions

6.2.1 Order and equivalence relations

To begin with, we remind the reader of the definitions of order relations and equivalence relations, which allow us to classify actions with respect to some criteria.

Definition 25. ***Equivalence relation***
A binary relation R (a relation between two elements) defined on the set A is an equivalence relation if it is:

- ***reflexive:*** $\forall x \in A,\ x\,R\,x$
- ***symmetric:*** $\forall\ (x, y) \in A \times A,\ x\,R\,y \Rightarrow y\,R\,x$
- ***transitive:*** $\forall\ (x, y, z) \in A \times A \times A,\ x\,R\,y$ *and* $y\,R\,z \Rightarrow x\,R\,z$

The simplest example of an equivalence relation is the "=" relation on the set $\mathbb{N}$ of natural integers.

The definition of an order relation is as follows:

Definition 26. ***Order relation***
A binary relation R defined on the set A is an order relation if this relation is:

- ***reflexive***
- ***antisymmetric:*** $\forall\,(x, y) \in A \times A,\ x\,R\,y$ *and* $y\,R\,x \Rightarrow x = y$
- ***transitive***

The simplest example of an order relation is the "$\leq$" relation on the set $\mathbb{N}$ of natural integers.

These various relations allow us to construct comparison relations (we shall call these relations preference relations). These relations will allow us to sort the elements of a set on which we perform these comparisons.

There exist several kinds of orders. Their definitions are as follows:

Definition 27. ***Preorder***
A preorder is a reflexive and transitive relation.

Definition 28. ***Order***
An order is an antisymmetric preorder (so, it is an order relation).

In a preorder, equivalent solutions are allowed while in an order, equivalent solutions are not allowed. The following are some more definitions related to preorder.

Definition 29. ***Total preorder***
A total preorder is a preorder relation for which all the elements of a set can be in the relation together: incomparability between two elements is not allowed.

An example of a total preorder is the "$\leq$" relation in the set $\mathbb{R}$. Indeed, given two elements A and B of $\mathbb{R}$, we have either $A \leq B$ or $B \leq A$.

Definition 30. ***Partial preorder***
A partial preorder is a preorder relation for which some elements of a set can be in the relation together: incomparability between two elements is allowed.

For example, the domination relation is a partial order relation. Indeed, if we denote by $A \prec B$ the relation "A dominates B", there exist actions for which we have neither $A \prec B$ nor $B \prec A$. These actions belong to the tradeoff surface.

These various definitions allow us to "build" comparison relations between actions. Depending on the problem that we are working on, we can build either an order relation (to perform a classification of actions, or to define an order on this set of solutions) or an equivalence relation (to look for the best action amongst a set of actions or to define a preorder on this set of solutions).

6.2.2 Preference relations

The various definitions presented above allow us to define the preference relation, the indifference relation and the incomparability relation between two actions:

- The preference relation between two actions: this relation, which we denote by P, corresponds to a preference for one action with respect to the other. We have $a\,P\,b$ if a is preferred to b.
- The indifference relation between two actions: this relation, which we denote by I, corresponds to an indifference between two actions. We have $a\,I\,b$ if there is indifference between a and b.
- The incomparability relation between two actions: this relation, which we denote by R, means that two actions are incomparable. We have $a\,R\,b$ if there is an incomparability between a and b (this means that we have neither $a\,P\,b$, nor $a\,I\,b$, nor $b\,P\,a$).

These three relations $\{P, I, R\}$ define what we call a "preference structure".

A preference structure can be completely characterized by the definition of a preference relation "in the broad sense":

$a\,S\,b$ if $a\,P\,b$ or if $a\,I\,b$.

The preference relation in the broad sense is the combination of a preference relation "in the strict sense" (the P relation) and an indifference relation (the I relation). The preference relation is also called the "outranking relation".

6.2.3 Definition of a criterion

Definition 31. ***A criterion*** *[Vincke 89]*
A criterion is a function g, defined on a set of actions A, which takes its values in a totally sorted set, and which represents the preferences of a decision maker with respect to a point of view.

For example, the mean mark of a student is a criterion which may be used to determine the skill of this student.

According to the method, the definition of the value scale of a criterion (or the way we give a value to a criterion) may change. Nevertheless, the definition of a criterion remains the same.

6.2.4 Analysis

Usually, at the end of the running of a decision aid method, we proceed to a sensitivity analysis phase and a robustness analysis phase. Both analyses allow us to verify the "reliability" of the result produced by the method, or the sensitivity of the results with respect to the parameters the decision maker has chosen.

Definition 32. ***Sensitivity analysis*** *[Maystre et al. 94]*
"This is an analysis which consists in the repeating of the original decision aid method while we vary the values associated with the various parameters of the method, values which are often chosen arbitrarily. Its main goal is to define the parameters which closely condition the chosen solution, that is to say, the parameters for which a small variation induces a large change in the proposed solution."

Definition 33. ***Robustness analysis*** *[Maystre et al. 94]*
"This is an analysis which tries to determine the variation domain of some parameters in which the sorting of solutions or the choice of a solution remains stable. It is used to give the decision maker a robust solution, which informs the decision maker about the ability of the solution to resist variations between reality and the model used."

6.3 Various methods

We shall present various decision aid methods here. We shall begin with the ELECTRE methods (I, IS, II, III, IV and TRI ("tri" in French can be translated as "sort"); then we shall end with two methods close to the ELECTRE methods: the PROMETHEE methods (I and II).

Both sets of methods deal with a precise problem. The important elements which differentiate these methods are the definition of the relation used to sort the actions (each method uses a different preference relation) and the method used to exploit the results of sorting.

6.3.1 Introduction

Before beginning the presentation of these methods, we shall describe some notation, and a method to represent the results.

6.3.1.1 Notation

The decision aid methods allow us to choose or sort actions with respect to a set of criteria. We denote the actions by a_i, $i = 1, \cdots, k$ and the criteria by g_j, $j = 1, \cdots, n$. The assessment of the a_i with respect to the criteria g_j is denoted by $g_j(a_i)$.

For example, if the criterion g_1 is the mean mark in course 1, the expression $g_1(a)$ is the mean mark of student a in course 1. This criterion takes a student as a variable and produces the mean mark of the considered student.

Here too, each decision aid method will produce its own definition of the value of a criterion.

6.3.1.2 The representation

The decision aid methods use a directed graph (a set of nodes and vertices which link some of the nodes; each vertex is directed using an arrow) to represent the relations between various actions.

In this graph, we begin by adding the nodes, which are an image of the actions considered, then we add the vertices between the nodes. We draw a vertex directed from node a_i to node a_j if action a_i "outranks" action a_j. If there is no relation between these two actions, we do not draw any vertex between these actions.

We shall present a simple example. In Table 6.2 are represented seven actions, and the relations which link them. We read this table starting from a row and proceeding to a column (for example, from the row a_2 and the column a_1, we have $a_2 \, S \, a_1$).

Table 6.2. Examples of relations between actions.

	a_1	a_2	a_3	a_4	a_5	a_6	a_7
a_1							
a_2	S						
a_3	S						
a_4	S	S	S				
a_5	S	S					S
a_6							
a_7							

When we represent these various relations on a graph, we have the graph in Fig. 6.2. We notice that action a_6 is not linked to another action. So we say it is incomparable.

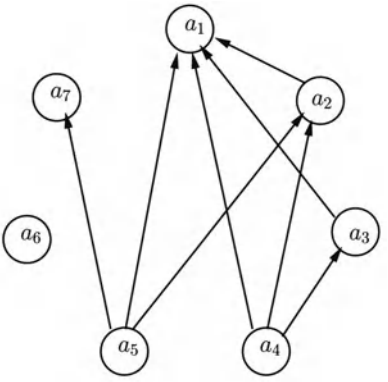


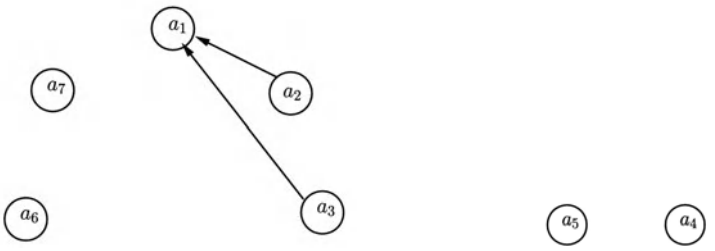
Fig. 6.2. The graph of the relations between actions.

From this graph, we can extract some important information. For example, we can isolate the kernel N of the graph. Once this kernel is isolated, we denote by A/N the remaining actions (which do not belong to the kernel).

Definition 34. ***The kernel of a graph*** *[Maystre et al. 94]*
The kernel of a graph is composed of a set of nodes such that:

- *all the nodes which do not belong to the kernel are outranked by at least one node from the kernel;*
- *the nodes of the kernel do not outrank other nodes of the kernel.*

If we apply this definition to the example, we obtain the graphs in Fig. 6.3. If the outranking is replaced by a Pareto domination, the definition of the kernel corresponds to the definition of the tradeoff surface.



(a) The graph of the set A/N.

(b) The graph of the set N.

Fig. 6.3. The graphs of the sets A/N and N.

This representation allows us, next, to compute the concordance index and the discordance index, which allow us to sort or choose actions with respect to some criterion.

Definition 35. ***Concordance index*** *[Maystre et al. 94]*
"The criterion j is said to tally with the hypothesis "action a_i outranks action a_k" if action a_i is at least as good as action a_k with respect to the criterion j; this can be translated as: $g_j(a_i) \geq g_j(a_k)$."

Definition 36. ***Nondiscordance*** *[Maystre et al. 94]*
"The nondiscordance condition allows the decision maker to refuse the outranking of one action over another obtained after applying the concordance condition, when a very high opposition exists on at least one criterion."

With these definitions, we shall now present the decision aid methods.

6.3.2 The ELECTRE I method

6.3.2.1 Principle

This method allows us to choose the best action with respect to a group of criteria. It was defined by B. Roy in 1968 [Vincke 89, Maystre et al. 94].

This first method will be applied to a concrete example: the selection of the best student in a class.

6.3.2.2 Presentation of the method

Our problem is the following:

- We have N actions a_i, $i = 1, \cdots, N$.
- We have K criteria g_j, $j = 1, \cdots, K$ (we define $F = \{j \mid j = 1, \cdots, K\}$).

We begin by associating a weight P_j with each criterion g_j (according to the preferences of the decision maker).

To establish the relations between actions

We perform comparisons between pairs of actions (a_i and a_k). These comparisons can be decomposed with respect to the preference relations. We have the following comparison sets:

- $J^+(a_i, a_k) = \{j \in F \mid g_j(a_i) > g_j(a_k)\}$: this is the set of criteria for which action a_i is preferred to action a_k.
- $J^=(a_i, a_k) = \{j \in F \mid g_j(a_i) = g_j(a_k)\}$
- $J^-(a_i, a_k) = \{j \in F \mid g_j(a_i) < g_j(a_k)\}$

These various comparisons are performed on pairs of actions (a_i, a_k) with respect to the criterion g_j. The result that we obtain at the end of this step is a set of indices which represent the criteria that satisfy the relations defined above for any given pairs of actions.

To convert the relations between actions into numerical values

The goal is now to convert the various sets that we have obtained after performing the comparisons on the pairs of actions (a_i, a_k) into numerical values.

For each set, we determine the sum of the weights of the criteria which belong to each family:

- $P^+(a_i, a_k) = \sum_j P_j$ with $j \in J^+(a_i, a_k)$: this is the sum of the weights of the criteria which belong to the set $J^+(a_i, a_k)$.
- $P^=(a_i, a_k) = \sum_j P_j$ with $j \in J^=(a_i, a_k)$
- $P^-(a_i, a_k) = \sum_j P_j$ with $j \in J^-(a_i, a_k)$

We define $P(a_i, a_k) = P^+(a_i, a_k) + P^=(a_i, a_k) + P^-(a_i, a_k)$. We can represent all this information in a table of dimension $N \times N$.

To merge the various numerical values

We now transform all the numerical values associated with each pair of actions and with each criterion (which gives us K numerical values for each pair of actions) into two numerical values: the concordance index and the nondiscordance index.

The above definitions allow us to compute the following:

- The concordance index:

$$C_{ik} = \frac{P^+(a_i, a_k) + P^=(a_i, a_k)}{P(a_i, a_k)} \tag{6.1}$$

  We have $0 \leq C_{ik} \leq 1$. This index shows how the starting hypothesis "a_i outranks a_k" tallies with the reality represented by the estimations of the actions.
- The concordance set:

$$J(a_i, a_k) = J^+(a_i, a_k) \cup J^=(a_i, a_k) \tag{6.2}$$

- The nondiscordance index, denoted by D_{ik}. It is defined the following way:

$$D_{ik} = \begin{cases} 0 & \text{if } J^-(a_i, a_k) = \emptyset \\ \frac{1}{\delta_j} \cdot \max\left(g_j(a_k) - g_j(a_i)\right) \text{ with } j \in J^-(a_i, a_k), & \text{otherwise} \end{cases} \tag{6.3}$$

  and δ_j is the amplitude of the scale associated with the criterion j. Here, we have $0 \leq D_{ik} \leq 1$.
- The nondiscordance set: this is the set $J^-(a_i, a_k)$.

To filter the actions

We have now all information needed to extract the best action or the best actions. To do so, we must judge the outranking relations. For an outranking relation to be judged as reliable, we must have

$$\left.\begin{array}{l} C_{ik} \geq c \\ D_{ik} \leq d \end{array}\right\} \Leftrightarrow a_i \, S \, a_k \tag{6.4}$$

We have the following quantities in the above equation:

c: the concordance threshold, usually $\simeq 0.7$. This is the threshold beyond which the starting hypothesis "a_i outranks a_k" will be considered as valid.

d: the nondiscordance threshold, usually $\simeq 0.3$. This is the threshold below which the starting hypothesis "a_i outranks a_k" will not be valid.

This definition tells us that for an outranking relation to be valid, it must tally sufficiently with the starting hypothesis "a_i outranks a_k" and be sufficiently nondiscordant with it.

This last step allows us to extract the best solutions from the set of actions that we have before we run the method (those which respect the relation 6.4).

6.3.2.3 Example

We shall now illustrate the behavior of the ELECTRE I method with an example.

Presentation of the data

We consider the marks of five students in five matters. These data are listed in Table 6.3.

Table 6.3. Marks of five students $(E_1, \cdots, E_5)$ in five subjects $(M_1, \cdots, M_5)$.

	M_1	M_2	M_3	M_4	M_5
E_1	7	13	8	12	11
E_2	8	11	11	12	11
E_3	20	2	10	3	15
E_4	16	14	16	14	13
E_5	12	12	8	8	10

We try to determine who is the best student using three criteria:

C_1: the mean mark in all of the subjects must be the highest possible.

C_2: the minimal mark must be strictly higher than 8.

C_3: the variations of the marks around the mean mark must be the smallest possible.

The mathematical definitions of the various criteria are as follows:

$$C_1(E) = \frac{\sum_{\text{subject}} \text{Mark}(E, \text{subject})}{5}$$

$$C_2(E) = \begin{cases} 10 \text{ if } \min_{\text{subject}} (\text{Mark}(E, \text{subject})) > 8 \\ 0 \quad \text{otherwise} \end{cases}$$

$$C_3(E) = \frac{\sqrt{\sum_{\text{subject}} (\text{Mark}(E, \text{subject}) - C_1(E))^2}}{5}$$

In the above, we have the following symbols:

E: one student out of the five.

subject: an index which runs over all the subjects for the student considered (five subjects altogether).

Mark $(E, \text{subject})$: corresponds to the mark of the student E in the subject considered.

The actions are the following:

a_1: E_1 is the best student.
a_2: E_2 is the best student.
a_3: E_3 is the best student.
a_4: E_4 is the best student.
a_5: E_5 is the best student.

We have now to choose the weights for all the criteria. Using this choice, we define an order of importance for these criteria. For example, we may want the mean mark to be the main criterion and the other two criteria to have the same weight as each other. So, we choose $P_{C_1} = 0.5$, $P_{C_2} = P_{C_3} = 0.25$.

In the first step, we compute the value of each criterion for each student. These values are listed in Table 6.4.

Table 6.4. The values of the criteria for each student.

	C_1	C_2	C_3
E_1	10.2	0	1.035
E_2	10.6	10	0.606
E_3	10	0	3.08
E_4	14.6	10	0.5366
E_5	10	10	0.8

Table 6.5. The calibrated value of the criteria for each student.

	C_1'	C_2'	C_3'
E_1	0.87	0	16.08
E_2	2.61	20	19.45
E_3	0	0	0
E_4	20	20	20
E_5	0	20	17.9

The scales of the various criteria are not suitable for our purpose, so we calibrate the scales as follows:

- For the criterion C_1, we associate 0 with 10 and 20 with 14.6.
- For the criterion C_2, we associate 0 with 0 and 20 with 10.
- For the criterion C_3, we associate 20 with 0.5366 and 0 with 3.08.

For intermediate values of each criterion, we perform a linear interpolation between the points given above. We obtain the values listed in Table 6.5. Here, the amplitudes of the scales (the variables δ_j) are all equal to 20 (the values of each criterion are spread over the interval $[0, 20]$).

To establish the relations between actions

We now perform a comparison action by action for each criterion. These comparisons are listed in a table for each criterion (see Tables 6.6, 6.7 and 6.8).

The presence of the symbol J^- at the intersection of row a_1 and column a_2 of the table related to criterion C_1' means that the index 1 belongs to the set $J^-(a_1, a_2)$. If we gather together the information in all the three tables related to this pair of actions, we obtain

$$J^{-}(a_1, a_2) = \{1, 2, 3\}$$

Indeed, for the three criteria C_1', C_2' and C_3', action a_2 is prefered to action a_1.

Table 6.6. Comparisons between actions for the criterion C_1'.

	a_1	a_2	a_3	a_4	a_5
a_1		J^-	J^+	J^-	J^+
a_2	J^+		J^+	J^-	J^+
a_3	J^-	J^-		J^-	$J^=$
a_4	J^+	J^+	J^+		J^+
a_5	J^-	J^-	$J^=$	J^-	

Table 6.7. Comparisons between actions for the criterion C_2'.

	a_1	a_2	a_3	a_4	a_5
a_1		J^-	$J^=$	J^-	J^-
a_2	J^+		J^+	$J^=$	$J^=$
a_3	$J^=$	J^-		J^-	J^-
a_4	J^+	$J^=$	J^+		$J^=$
a_5	J^+	$J^=$	J^+	$J^=$	

Table 6.8. Comparisons between actions for the criterion C_3'.

	a_1	a_2	a_3	a_4	a_5
a_1		J^-	J^+	J^-	J^-
a_2	J^+		J^+	J^-	J^+
a_3	J^-	J^-		J^-	J^-
a_4	J^+	J^+	J^+		J^+
a_5	J^+	J^-	J^+	J^-	

We consider now all the information contained in the preceding tables, and compute the concordance indices C_{ik}, the nondiscordance indices D_{ik}, and Tables 6.9–6.11 which list the information related to J^+, $J^=$ and J^-. These tables gather together the information listed in the preceding tables, with respect to all of the criteria. For example, in table 6.9, at the intersection of column a_1 and row a_2, we find the set $\{1, 2, 3\}$. This mean that action a_2 is preferred to action a_1 when criteria 1, 2 and 3 are considered.

Table 6.9. Summary for the set J^+.

	a_1	a_2	a_3	a_4	a_5
a_1		$\emptyset$	$\{1,3\}$	$\emptyset$	$\{1\}$
a_2	$\{1,2,3\}$		$\{1,2,3\}$	$\emptyset$	$\{1,3\}$
a_3	$\emptyset$	$\emptyset$		$\emptyset$	$\emptyset$
a_4	$\{1,2,3\}$	$\{1,3\}$	$\{1,2,3\}$		$\{1,3\}$
a_5	$\{2,3\}$	$\emptyset$	$\{2,3\}$	$\emptyset$	

The indices allow us to verify the coherency of the information in these tables: when we merge the three tables into one ($J = J^+ \cup J^= \cup J^-$), we must find the set $\{1, 2, 3\}$ in each cell of the table.

To convert the relations between actions into numerical values

We now compute the table of the concordance sets ($J = J^+ \cup J^=$), the table of the concordance coefficient (C_{ik}) and the table of the nondiscordance coefficient (D_{ik}).

Table 6.10. Summary for the set $J^{=}$.

	a_1	a_2	a_3	a_4	a_5
a_1		$\emptyset$	$\{2\}$	$\emptyset$	$\emptyset$
a_2	$\emptyset$		$\emptyset$	$\{2\}$	$\{2\}$
a_3	$\{2\}$	$\emptyset$		$\emptyset$	$\{1\}$
a_4	$\emptyset$	$\{2\}$	$\emptyset$		$\{2\}$
a_5	$\emptyset$	$\{2\}$	$\{1\}$	$\{2\}$	

Table 6.11. Summary for the set J^{-}.

	a_1	a_2	a_3	a_4	a_5
a_1		$\{1,2,3\}$	$\emptyset$	$\{1,2,3\}$	$\{2,3\}$
a_2	$\emptyset$		$\emptyset$	$\{1,3\}$	$\emptyset$
a_3	$\{1,3\}$	$\{1,2,3\}$		$\{1,2,3\}$	$\{2,3\}$
a_4	$\emptyset$	$\emptyset$	$\emptyset$		$\emptyset$
a_5	$\{1\}$	$\{1,3\}$	$\emptyset$	$\{1,3\}$	

Table 6.12. Summary of the concordance sets $J = J^{+} \bigcup J^{=}$.

	a_1	a_2	a_3	a_4	a_5
a_1		$\emptyset$	$\{1,2,3\}$	$\emptyset$	$\{1\}$
a_2	$\{1,2,3\}$		$\{1,2,3\}$	$\{2\}$	$\{1,2,3\}$
a_3	$\{2\}$	$\emptyset$		$\emptyset$	$\{1\}$
a_4	$\{1,2,3\}$	$\{1,2,3\}$	$\{1,2,3\}$		$\{1,2,3\}$
a_5	$\{1,2,3\}$	$\{2\}$	$\{1,2,3\}$	$\{2\}$	

Table 6.12 contains the concordance sets.

Before computing the concordance coefficients, we compute the coefficients P_{ik}^{+} and $P_{ik}^{=}$ (Tables 6.13 and 6.14).

Table 6.13. Summary of the coefficients P_{ik}^{+}.

	a_1	a_2	a_3	a_4	a_5
a_1		0	0.75	0	0.5
a_2	1		1	0	0.75
a_3	0	0		0	0
a_4	1	0.75	1		0.75
a_5	0.5	0	0.5	0	

Table 6.14. Summary of the coefficients $P_{ik}^{=}$.

	a_1	a_2	a_3	a_4	a_5
a_1		0	0.25	0	0
a_2	0		0	0.25	0.25
a_3	0.25	0		0	0.5
a_4	0	0.25	0		0.25
a_5	0	0.25	0.5	0.25	

To merge the numerical values

We now compute the table of the concordance coefficients (Table 6.15) using the following formula:

$$C_{ik} = \frac{P_{ik}^{+} + P_{ik}^{=}}{P_{ik}}$$

where $P_{ik} = 1, \forall i$ and $\forall k$.

Table 6.15. Summary of the concordance coefficients C_{ik}.

	a_1	a_2	a_3	a_4	a_5
a_1		0	1	0	0.5
a_2	1		1	0.25	1
a_3	0.25	0		0	0.5
a_4	1	0.75	1		1
a_5	0.5	0.25	1	0.25	

Table 6.16 contains the nondiscordance coefficients. We notice that δ_j is equal to 20 for all the criteria.

Table 6.16. Summary of the nondiscordance coefficients D_{ik}.

	a_1	a_2	a_3	a_4	a_5
a_1		0.087	0	0.196	0.196
a_2	0		0	0.13	0
a_3	0.0435	0.1305		1	0.895
a_4	0	0	0		0
a_5	0.0435	0.0775	0	0.105	

To filter the actions

We have now all the information needed to perform the concordance test and the nondiscordance test. We set the threshold c for the concordance test to 0.75. This test is satisfied if $C_{ik} \geq 0.75$. We set the threshold d for the nondiscordance test to 0.25. This test is satisfied if $D_{ik} \leq 0.25$.

The C_{ik} which satisfy the concordance test are C_{13}, C_{21}, C_{23}, C_{25}, C_{41}, C_{42}, C_{43}, C_{45} and C_{53}.

Only D_{34} does not satisfy the nondiscordance test.

So, we can say that:

- Action a_1 outranks action a_3, because the concordance relation C_{13} is satisfied and the nondiscordance relation D_{13} is not.
- Action a_2 outranks actions a_1, a_3 and a_5 because the concordance relations C_{21}, C_{23}, C_{25} are satisfied and the nondiscordance relations D_{21}, D_{23}, D_{25} are not.
- Action a_4 outranks all the others, because the concordance relations C_{4i}, $i = 1, 2, 3, 5$, are satisfied and the non discordance relations D_{4i}, $i = 1, 2, 3, 5$, are not.
- Action a_5 outranks action a_3, because the concordance relation C_{53} is satisfied and the nondiscordance relation D_{53} isn't.

We represent these results in the graph shown in Fig. 6.4.

To conclude this example, if we consider the numbers of outranked students, we can say that student E_4 is better than all the others, E_2 is ranked second, E_1 and E_5 are equivalent and E_3 is the last.

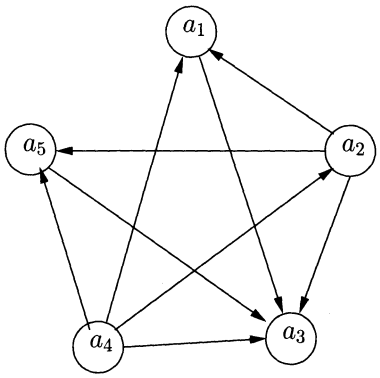


Fig. 6.4. Graph of the relations between actions.

6.3.2.4 Discussion

Despite the significant number of steps we had to perform to obtain the result in the above example, we notice that the choice we make at the end of the analysis is coherent. It corresponds to the sorting that we could have done in a "natural" way.

So, this method mimics "human" reasoning. We can apply this choice method to a very large data set and obtain relations between actions which may be hard to obtain "by hand".

6.3.3 The ELECTRE IS method

6.3.3.1 Principle

This method corresponds to the ELECTRE I method with fuzzy logic added [Maystre et al. 94].

6.3.3.2 Presentation of the method

We shall now define the concordance index by criteria, the global concordance index, the nondiscordance index by criteria, the global nondiscordance index and the outranking relation.

Concordance index and non discordance index

- The concordance index by criteria:

$$\begin{cases} c_j(a_i, a_k) = 0 & \text{if } p_j < g_j(a_k) - g_j(a_i) \\ 0 < c_j(a_i, a_k) < 1 & \text{if } q_j < g_j(a_k) - g_j(a_i) \leq p_j \\ c_j(a_i, a_k) = 1 & \text{if } g_j(a_k) - g_j(a_i) \leq q_j \end{cases}$$

This index is similar to the one defined in the ELECTRE III method (see Sect. 6.3.5).

- The global concordance index:

$$C_{ik} = \frac{\sum_{j=1}^{m} p_j \cdot c_j(a_i, a_k)}{\sum_{j=1}^{m} p_j}$$

This index is similar to the one defined in the ELECTRE III method.

- The nondiscordance index by criteria; this index is a binary one:

$$d_j(a_i, a_k) = \begin{cases} 0 \text{ if } g_j(a_k) - g_j(a_i) < v_i(a_i, a_k) - q_j(a_i, a_k) \cdot \frac{1-C_{ik}}{1-c} \\ 1 \text{ otherwise} \end{cases}$$

where c is the global concordance threshold.

- The global nondiscordance index; this index is also a binary one:

$$D_{ik} = \begin{cases} 0 \text{ if } d_j(a_i, a_k) = 0, \forall j = 1, \cdots, m \\ 1 \text{ otherwise} \end{cases}$$

- The outranking relation; this relation is also binary:

$$S(a_i, a_k) = \begin{cases} 1 \text{ if } C_{ik} \geq c \text{ and } D_{ik} = 0 \\ 0 \text{ otherwise} \end{cases}$$

6.3.3.3 Discussion

As with the ELECTRE I method, we can draw the graph of the actions and look for the kernel of this graph. We must not forget to replace the maximal circuit (a circuit which is not included in another circuit) by a fictitious action. The analysis of the results is the same as with the ELECTRE I method.

The advantage of adding the aspect of fuzzy logic to the ELECTRE I method is that we reduce the sensitivity of the method to the parameters chosen by the decision maker. Indeed, the fuzzy method allows one to choose a parameter as an "interval" instead of giving this parameter a fixed value.

6.3.4 The ELECTRE II method

6.3.4.1 Principle

This method is quite similar to the ELECTRE I method. The difference lies in the definition of two outranking relations:

- the strong outranking;
- the weak outranking.

6.3.4.2 Presentation of the method

Definition of the outranking relations

The actions are compared pair by pair, and preference relations are established:

- The strong outranking relations: action a_i strongly outranks action a_k if

$$\begin{cases} C_{ik} \geq c^{+} \\ \text{and} \\ g_j(a_k) - g_j(a_i) \leq D_1(j), \forall j \\ \text{and} \\ \frac{P_{ik}^{+}}{P_{ik}^{-}} \geq 1 \end{cases} \tag{6.5}$$

and/or

$$\begin{cases} C_{ik} \geq c^{0} \\ \text{and} \\ g_j(a_k) - g_j(a_i) \leq D_2(j), \forall j \\ \text{and} \\ \frac{P_{ik}^{+}}{P_{ik}^{-}} \geq 1 \end{cases} \tag{6.6}$$

with $c^{+} \geq c^{0}$ and $D_2(j) \leq D_1(j)$. This relation allows us to consider that an action a_i strongly outranks an action a_k if:
  - the outranking relation is strongly concordant and fairly nondiscordant (see Eq. 6.5),
  - the outranking relation is fairly concordant and weakly nondiscordant (see Eq. 6.6).
- The weak outranking relation: action a_i weakly outranks action a_k if

$$\begin{cases} C_{ik} \geq c^{-} \\ \text{and} \\ g_j(a_k) - g_j(a_i) \leq D_1(j), \forall j \\ \text{and} \\ \frac{P_{ik}^{+}}{P_{ik}^{-}} \geq 1 \end{cases} \tag{6.7}$$

with $c^{0} \geq c^{-}$. This relation allows us to consider that an action a_i weakly outranks an action a_k if the outranking relation is weakly concordant and fairly nondiscordant (see Eq. 6.7).

The variables $D_1(j)$ and $D_2(j)$ are thresholds of nondiscordance. These values are defined for each criterion g_j. The thresholds are chosen so as to have $D_2(j) \leq D_1(j)$. Consequently,

- if $g_j(a_k) - g_j(a_i) \leq D_2(j)$, the nondiscordance is weak and the criterion j does not show a major opposition to the hypothesis "action a_i outranks action a_k";
- if $D_2(j) < g_j(a_k) - g_j(a_i) \leq D_1(j)$, the nondiscordance is mean and the criterion j does not show a major opposition to the hypothesis "action a_i outranks action a_k".

Definition of the concordance coefficients

The concordance coefficients are computed in the same way as in the ELECTRE I method:

$$C_{ik} = \frac{P_{ik}^{+} + P_{ik}^{=}}{P_{ik}} \tag{6.8}$$

The difference lies in the fact that we will have three concordance thresholds c^+, c^0 and c^- which correspond to a concordance with strong, mean and weak presumption, respectively.

The concordance test is accepted if

$$\left.\begin{array}{l} C_{ik} \geq c^{+} \\ \text{or} \\ C_{ik} \geq c^{0} \\ \text{or} \\ C_{ik} \geq c^{-} \end{array}\right\} \text{ and } \frac{P_{ik}^{+}}{P_{ik}^{-}} \geq 1$$

The classification process

Once we have determined the various elements, we must perform:

- a direct classification,
- a reverse classification.

We start by representing the relations between actions in two graphs:

- a graph called the "strong outranking graph", denoted by G_F, in which we link two nodes together if there exists a strong outranking relation between the two actions represented by these two nodes;
- a graph called the "weak outranking graph", denoted by G_f, in which we link two nodes together if there exists a weak outranking relation between these two actions represented by these two nodes.

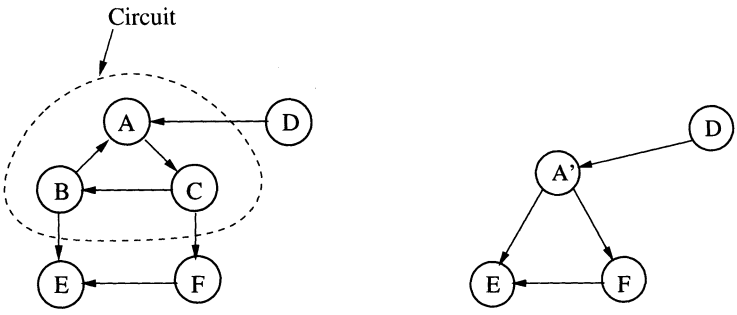
We must, in a first step, modify both graphs so as to remove the circuits.

Definition 37. ***A circuit***
A circuit is a set of actions in which we have, for example, action A dominates action B, action B dominates action C and action C dominates action A.

An example of how to remove a circuit is represented in Fig. 6.5. In this figure, we can see how easy it is to deal with a circuit. A circuit corresponds to relations between actions which are similar to a Condorcet paradox. It is not possible to decide which action is the best. So, to remove this ambiguity, we replace actions which draw a circuit by a fictitious action. This action will symbolize a group of equivalent actions to the decision maker.

The modifications of the graphs are performed as follows:

- we denote by G'_F the graph which corresponds to the strong-outranking original graph G_F in which we have replaced all circuits by fictitious actions;
- we denote by G'_f the graph which corresponds to the weak-outranking original graph G_f in which we have replaced all circuits by fictitious actions.



(a) The graph of actions before the transformation of the circuit.

(b) The graph of the actions after the transformation of the circuit.

Fig. 6.5. How to replace a circuit by a fictitious action.

The direct classification is performed in six steps:

Step 1: At each new step l, actions which have already been classified are removed for the strong-outranking graph; the remaining actions constitute the set A_l (which is a subset of A), and their relations are given by the graph Y_l (which is a subgraph of the modified strong-outranking graph G'_F).

Step 2: In the graph Y_l, all the nodes which are not outranked are counted: they form the set D.

Step 3: Elements from D which are not linked in the weak-outranking graph G'_f form the set U.

Step 4: The set B contains all the nodes from U which are not outranked by any other nodes from U. The equivalence class of actions classified during step l, denoted by A_l, is defined by the merging of the sets $D - U$ and B. The sets B and $D - U$ represent all the nodes which:

1. have not yet been classified;
2. are not outranked by any nodes from the graph Y_l, which is nothing other than the strong-outranking graph G'_F from which we have removed the nodes and the corresponding edges already classified;
3. for the set $D - U$, have no weak outranking relations between them; and
   for the set B, outrank, considering weak outranking relations, other nodes which satisfy conditions 1 and 2.

Step 5: With each of the actions classified during step l (and which form, consequently, the equivalence class A_l) is associated the rank $l + 1$; this way, to each potential action there corresponds a rank obtained using direct classification, and if $\text{rank}(a_i) < \text{rank}(a_k)$, this means that action a_i is "better" than action a_k. The nodes classified during step l are removed from the strong-outranking graph, which gives us the new subgraph Y_{l+1}.

Step 6: Lastly, if Y_{l+1} does not include any nodes, the classification is completed; otherwise, the process continues to step $l + 1$.

To establish the order relation

Once the direct classification is completed, we construct another classification: the reverse classification. To do so, we perform a direct classification but with the following modifications:

- reverse the directions on the edges of the graphs G'_F (strong-outranking graphs) and G'_f (weak-outranking graphs);
- once the rank has been obtained in the same way as in the preceding procedure ($\text{rank}(a_i) = l + 1$), we adjust it the following way:

$$\text{rank}_2(a_i) = 1 + \text{rank}_1(a_i)_{max} - \text{rank}_1(a_i)$$

We now gather together the information that we have obtained during the direct classification and the reverse classification (this is, in fact, an intersection between the two classifications). We obtain a partial preorder. We must follow these steps:

- if a_i is preferred to a_k in both complete preorders (the direct classification and the reverse classification), then the same must be true in the partial preorder;
- if a_i is equivalent to a_k in one complete preorder but is preferred in the other, then a_i must be preferred to a_k in the final classification;
- if, in the first preorder, a_i is preferred to a_k and if, in the second preorder, a_k is preferred to a_i, then both actions must be incomparable in the final preorder.

6.3.4.3 Example

We use again the example of Sect. 6.3.2.3.

First, we determine the coefficients $\frac{P^+_{ik}}{P^-_{ik}}$ (see table 6.17).

Table 6.17. Values of the coefficients $\frac{P^+_{ik}}{P^-_{ik}}$.

	a_1	a_2	a_3	a_4	a_5
a_1		0	$+\infty$	0	1
a_2	$+\infty$		$+\infty$	0	$+\infty$
a_3	0	0		0	0
a_4	$+\infty$	$+\infty$	$+\infty$		$+\infty$
a_5	1	0	$+\infty$	0	

We now compute the following coefficients:

- $g_1(a_k) - g_1(a_i)$, which we denote by G^1_{ik} (see Table 6.18).
- $g_2(a_k) - g_2(a_i)$, which we denote by G^2_{ik} (see Table 6.19).
- $g_3(a_k) - g_3(a_i)$, which we denote by G^3_{ik} (see Table 6.20).

We compute the matrix of the outranking relations, using the following thresholds: $c^+ = 0.75$, $c^0 = 0.7$, $c^- = 0.65$, $D_1 = 20$ and $D_2 = 16$ (see Table 6.21). In this table, the symbol S_F corresponds to the strong outranking relation, the symbol S_f corresponds to the weak outranking relation and $\times$ corresponds to the absence of an outranking relation between two elements.

We now plot the strong-outranking graph and the weak-outranking graph. These graphs are represented in Fig. 6.6.

We then perform the following classifications:

Table 6.18. Values of the coefficients G^1_{ik}.

	a_1	a_2	a_3	a_4	a_5
a_1		1.74	−0.87	19.13	−0.87
a_2	−1.74		−2.61	17.39	−2.61
a_3	0.87	2.61		20	0
a_4	−19.13	−17.39	−20		−20
a_5	0.87	2.61	0	20	

Table 6.19. Values of the coefficients G^2_{ik}.

	a_1	a_2	a_3	a_4	a_5
a_1		20	0	20	20
a_2	−20		−20	0	0
a_3	0	20		20	20
a_4	−20	0	−20		0
a_5	−20	0	−20	0	

Table 6.20. Values of the coefficients G^3_{ik}.

	a_1	a_2	a_3	a_4	a_5
a_1		3.37	−16.08	3.92	1.82
a_2	−3.37		−19.45	0.55	−1.55
a_3	16.08	19.45		20	17.9
a_4	−3.92	−0.55	−20		−2.1
a_5	−1.82	1.55	−17.9	2.1	

Table 6.21. Matrix of outranking relations.

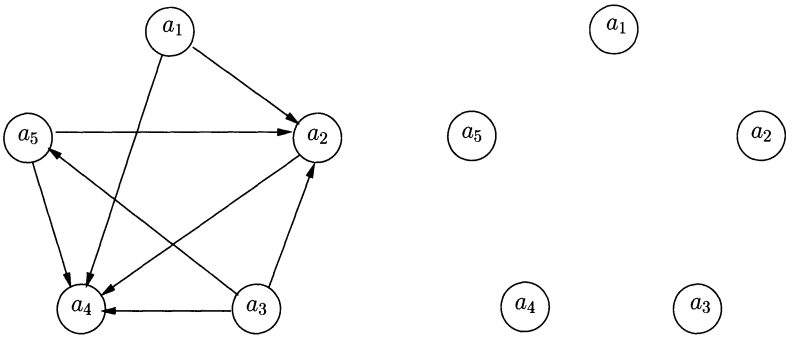
	a_1	a_2	a_3	a_4	a_5
a_1		×	×	×	×
a_2	S_F		S_F	×	S_F
a_3	×	×		×	×
a_4	S_F	S_F	S_F		S_F
a_5	×	×	S_F	×	

- direct (see Table 6.22);
- reverse (see Table 6.23).

Table 6.22. The direct classification.

Step	Y_l	D	U	B	A_l	r_{l+1}
0	$\{1,2,3,4,5\}$	$\{4\}$	$\emptyset$	$\{4\}$	$\{4\}$	1
1	$\{1,2,3,5\}$	$\{2\}$	$\emptyset$	$\{2\}$	$\{2\}$	2
2	$\{1,3,5\}$	$\{1,2\}$	$\emptyset$	$\{1,5\}$	$\{1,5\}$	3
3	$\{3\}$	$\{3\}$	$\emptyset$	$\{3\}$	$\{3\}$	4

Lastly, we represent the information that we have obtained using the direct and reverse classifications on a graph (see Fig. 6.7). On the abscissa, we have the rank obtained from the reverse classification and, on the ordinate, we have the rank obtained from the direct classification.



(a) The strong-outranking graph. (b) The weak-outranking graph.

Fig. 6.6. The outranking graphs.

Table 6.23. The reverse classification.

Step	Y_l	D	U	B	A_l	r_{l+1}
0	$\{1,2,3,4,5\}$	$\{1,3\}$	$\emptyset$	$\{1,3\}$	$\{1,3\}$	4
1	$\{2,4,5\}$	$\{5\}$	$\emptyset$	$\{5\}$	$\{5\}$	3
2	$\{2,4\}$	$\{4\}$	$\emptyset$	$\{4\}$	$\{4\}$	2
3	$\{4\}$	$\{2\}$	$\emptyset$	$\{2\}$	$\{2\}$	1

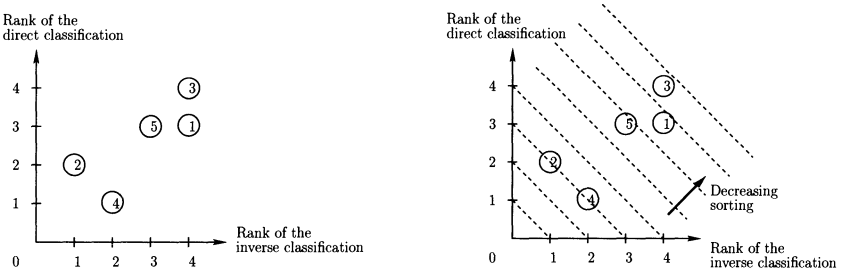


Fig. 6.7. Classifications of actions.

6.3.4.4 Discussion

This method allows us to classify actions. In our example, we have performed two classifications:

- a direct classification (or from the best action to the worst one): 2, 4, 5, $\{3,1\}$.
- a reverse classification (or from the worst action to the best one): 4, 2, $\{5,1\}$, 3. We have presented the actions from the best one to the worst one in both cases so that the differences between the two classifications can be clearly seen.

A global classification, such as the one given in Fig. 6.7, gives $\{2,4\},5,1,3$. The classification was obtained by sliding the line

Rank of the direct classification $= -$Rank of the reverse classification

from the right to the left. The best actions are the ones we encounter first. This method solves the sorting problem efficiently.

Nevertheless, this method is complex to apply (the example demonstrates this fact). The method uses a significant number of sets (Y_l, D, B, U and A_l). Moreover, this method uses numerous parameters (the concordance thresholds c^+, c^0 and c^-, and the nondiscordance thresholds by criteria $D_1(j)$ and $D_2(j)$).

More information about this method can be found in [Vincke 89] and [Maystre et al. 94].

6.3.5 The ELECTRE III method

6.3.5.1 Principle

This method uses the same principles as the ELECTRE II method (see [Vincke 89] and [Maystre et al. 94]). The main difference lies in the introduction of pseudo-criteria instead of classical criteria. For these criteria, two thresholds are defined:

- an indifference threshold,
- a strict preference threshold.

ELECTRE III introduces a credibility degree (also called a reliability degree) of the outranking. The final decision is not anymore a binary choice between the acceptance or rejection of an action.

6.3.5.2 Presentation of the method

This ELECTRE method is very much inspired by fuzzy logic.

First, let us define a pseudo-criterion.

Definition 38. ***Pseudo-criterion***

A pseudo-criterion is defined using two bands which correspond to two different areas of preference:

- *an indifference band, in which the difference between a_i and a_k does not lead to a particular preference for one or the other of the two actions;*
- *a strict preference band, in which we can say that we prefer one or the other of the two actions.*

Between these two bands, we can say that we have a weak preference for an action.

A pseudo-criterion is a function g for which the discriminating power is characterized by two thresholds $q(g)$ and $p(g)$ in the following way:

$\forall a_i, a_k \in A$

The indifference relation is defined by

$$a_i \, I \, a_k \Leftrightarrow -q(g(a_i)) \leq g(a_i) - g(a_k) \leq q(g(a_k))$$

The weak preference is defined by

$a_i \, Q \, a_k \Leftrightarrow q\,(g\,(a_k)) < g\,(a_i) - g\,(a_k) \leq p\,(g\,(a_k))$

The strict preference is defined by

$a_i \, P \, a_k \Leftrightarrow p\,(g\,(a_k)) < g\,(a_i) - g\,(a_k)$

The thresholds p and q must respect the following three relations:

$$\frac{q\,(g\,(a_k) - g\,(a_i))}{g\,(a_k) - g\,(a_i)} \geq -1$$

$$\frac{p\,(g\,(a_k) - g\,(a_i))}{g\,(a_k) - g\,(a_i)} \geq -1$$

$$q_j \leq p_j$$

The parameter p_j (the threshold function p associated with the criterion j) is called the "strict preference threshold", and the parameter q_j (the threshold function q associated with the criterion j) is called the "indifference threshold". For example, the threshold functions p and q can be defined as:

- a constant,
- a function with respect to the action considered, for example

$$p\,(g\,(a_i)) = \alpha + \beta \cdot g\,(a_i) \tag{6.9}$$

The behavior of the various relations I, Q, and P is represented in Fig. 6.8.

The concordance index

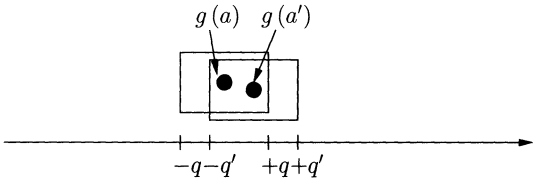
For the concordance index, we use the following formula, illustrated in Fig. 6.9:

$$\begin{cases} c_j\,(a_i, a_k) = \frac{g_j(a_i) + p_j - g_j(a_k)}{p_j - q_j} & \text{if } q_j < g_j\,(a_k) - g_j\,(a_i) \leq p_j \\ c_j\,(a_i, a_k) = 1 & \text{if } g_j\,(a_k) - g_j\,(a_i) \leq q_j \\ c_j\,(a_i, a_k) = 0 & \text{if } p_j < g_j\,(a_k) - g_j\,(a_i) \end{cases}$$

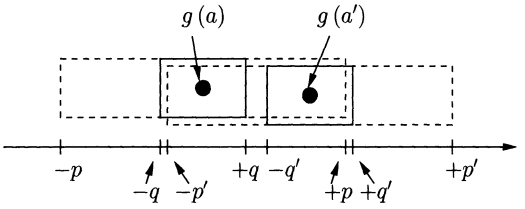
The global concordance index is defined by

$$C_{ik} = \frac{\sum\limits_j p_j \cdot c_j\,(a_i, a_k)}{\sum\limits_j p_j} \tag{6.10}$$

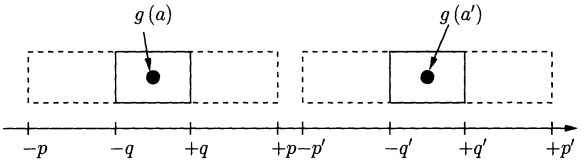
On the abscissa of Fig. 6.9, we have represented the difference between two actions for a given criterion $g_1\,(a_1) - g_1\,(a_2)$.



(a) An indifference relation.



(b) A weak preference relation.



(c) A strict preference relation.

Fig. 6.8. Graphical representation of the various preference relations.

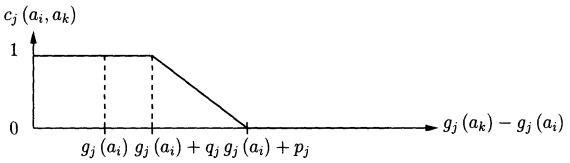


Fig. 6.9. Representation of the concordance index.

The non discordance index

We now define the veto threshold v_j. This is the value of $g_j(a_k) - g_j(a_i)$ for which a careful decision maker gives no credibility to the outranking of action a_k with respect to action a_i.

The nondiscordance index is defined by the following formula, illustrated in Fig. 6.10:

$$\begin{cases} d_j(a_i, a_k) = 1 & \text{if } v_j < g_j(a_k) - g_j(a_i) \\ d_j(a_i, a_k) = \frac{g_j(a_k) - g_j(a_i) - p_j}{v_j - p_j} & \text{if } p_j \leq g_j(a_k) - g_j(a_i) \leq v_j \\ d_j(a_i, a_k) = 0 & \text{if } g_j(a_k) - g_j(a_i) < p_j \end{cases}$$

We have $q_j < p_j < v_j$.

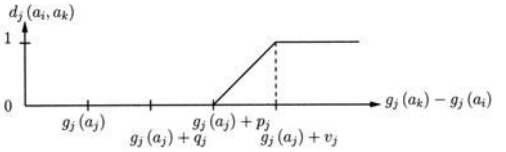


Fig. 6.10. Representation of the nondiscordance index.

Definition 39. ***Outranking credibility degree***
We define the outranking credibility degree as

$$\delta_{ik} = C_{ik} \cdot \prod_{j \in \bar{F}} \frac{1 - d_j(a_i, a_k)}{1 - C_{ik}} \tag{6.11}$$

In [Roy et al. 93], we find the rules that the function δ_{ik} (which gives the outranking credibility degree) must respect.

If F is the set of criteria, then we have

$$\bar{F} = \{j \mid j \in F, d_j(a_i, a_k) > C_{ik}\} \tag{6.12}$$

This is the set of criteria for which the nondiscordance index is greater than the global concordance index.

Exploitation

To use the fuzzy outranking relation defined later, we define the discrimination threshold here. This is a threshold which will allow us to distinguish the outranking relations to be taken into account during the classification [Maystre et al. 94].

- Discrimination threshold: $S(\lambda)$ is a function defined for all values $\lambda \in [0, 1]$. We compute this threshold by considering the pairs of actions (a_e, a_m) and (a_i, a_j):
  1. Computation of η:

$$\begin{cases} \lambda = \delta_{ij} \\ \lambda - \eta = \delta_{em} \end{cases}$$
$$\eta = \delta_{ij} - \delta_{em}$$

  2. Apply the discrimination: if $\eta > S(\lambda)$, then the outranking of a_j by a_i is strictly more credible than the outranking of a_m by a_e.

- Example of how to compute the threshold function: We present here an example presented in [Roy et al. 93]. In the first step, we compare high values of the outranking credibility. We consider that an outranking of credibility 0.99 is more credible than an outranking of credibility 0.80. On the other hand, we consider that an outranking of credibility 0.99 is not always more credible than an outranking of credibility 0.85.
  We now compare two classifications of small credibility degree. We consider that an outranking of credibility 0.30 is more credible than an outranking of credibility 0. On the other hand, we consider that an outranking of credibility 0.25 is not always more credible than an outranking of credibility 0. This can be translated using the following equations:

  $0.99 - 0.80 > S(0.99 - 0.80)$
  $0.99 - 0.85 < S(0.99 - 0.85)$
  $0.25 - 0 < S(0.25 - 0)$
  $0.30 - 0 > S(0.30 - 0)$

  We now make the hypothesis that $S(\lambda)$ has the following form: $S(\lambda) = A - B \cdot \lambda$. The parameters $A = 0.3$ and $B = 0.15$ allow us to satisfy all the preceding inequalities.
  If there were more equations, we might have to choose another form for $S(\lambda)$, a second-degree equation, for example. We only have to satisfy the condition that the function $S(\lambda)$ is monotonic and nondecreasing.
- Example of application: We now apply another threshold function to the verification of the credibility of one outranking against another:

  $S(\lambda) = 0.2 - 0.15 \cdot \lambda$

  Let two groups of actions, (a_1, a_2) and (a_2, a_1) be given. The outranking credibility indices are $\delta_{12} = 0.12$ and $\delta_{21} = 0.33$.

  $\eta = 0.12 - 0.33 = -0.21$
  $S(\lambda) = S(0.12) = 0.2 - 0.15 \cdot 0.12 = 0.182$

  $\eta < S(\lambda)$, so the outranking of a_2 by a_1 is not more credible than the outranking of a_1 by a_2.

Terminology related to the ELECTRE III method

Before presenting the classification method, we introduce some definitions:

Definition 40. ***Potential of an action***
This is the number of actions to which it is strictly preferred. We denote by $p(a_i)$ the potential of action a_i.

Definition 41. ***Weakness of an action***
This is the number of actions which are preferred to this action. We denote by $f(a_i)$ the weakness of action a_i.

Definition 42. *Qualification of an action*
We define $q(a_i) = p(a_i) - f(a_i)$.

Definition 43. *Fuzzy outranking relation*

$$a_i S_A^{\lambda_1} a_k \Leftrightarrow \begin{cases} \delta_{ik} > \lambda_1 \\ \delta_{ik} > \delta_{ki} + S(\delta_{ik}) \end{cases} \quad \text{and} \tag{6.13}$$

Definition 44. λ_l*-potential*

$$p_A^{\lambda_l}(a_i) = card\left(\left\{a_k \in A \,|\, a_i S_A^{\lambda_l} a_k\right\}\right) \tag{6.14}$$

Definition 45. λ_l*-weakness*

$$f_A^{\lambda_l}(a_i) = card\left(\left\{a_k \in A \,|\, a_k S_A^{\lambda_l} a_i\right\}\right) \tag{6.15}$$

Definition 46. λ_l*-qualification of action* a_i *related to the set* A

$$q_A^{\lambda_l}(a_i) = p_A^{\lambda_l}(a_i) - f_A^{\lambda_l}(a_i) \tag{6.16}$$

Here, card (A) means the cardinality of the set A.

The classification process

Here now is the description of the classification method. We call this method the "descending distillation chain", because it is an iterative process with the effect of extracting solutions which have a maximal λ-qualification.

Step 1: We write $A_n = A$ and $n = 0$.
Step 2: We write $D_l = A_n$ and $l = 0$.
Step 3: $\lambda_l = \max \delta_{ik}$, where $a_i, a_k \in A$ and $a_i \neq a_k$.
$S(\lambda_l) = \alpha + \beta \cdot \lambda_l$.
Step 4: $\lambda_{l+1} = \max \delta_{ik}$, where $\delta_{ik} < \lambda_l - S(\lambda_l)$ and $a_i, a_k \in D_l$.
Step 5: We use the fuzzy outranking relation $a_i S_A^{\lambda_{l+1}} a_k$.
We compute the λ_{l+1}-potential, the λ_{l+1}-weakness and the λ_{l+1}-qualification for each action $a_i \in D_l$.
We compute $\bar{q} = \max q^{\lambda_{l+1}}(a_i), a_i \in D_l$.
We compute $D_{l+1} = \left\{a_i \in D_l \,|\, q^{\lambda_{l+1}}(a_i) = \bar{q}\right\}$.

Step 6: $\text{card}(D_{l+1}) = 1$, where $\lambda_{l+1} = 0$?
If the answer is yes, we add 1 to l and go back to step 4.
If the answer is no, we go to step 7.

Step 7: $C_{n+1} = D_{l+1}$ and $A_{n+1} = A_n / C_{n+1}$ (A_{n+1} is equal to the set A_n from which we have removed the elements of C_{n+1}).

Step 8: $A_{n+1} = \emptyset$?
If the answer is yes, then the classification is completed.
If the answer is no, we add 1 to n and go back to step 2.

For the ascending distillation method, we use the same steps, but we change the following elements:

- At step 5, we replace "we compute $\bar{q} = \max q^{\lambda_{l+1}}(a_i), a_i \in D_l$" by "we compute $\underline{q} = \min q^{\lambda_{l+1}}(a_i), a_i \in D_l$ ".
- Also at step 5, we replace "we compute $D_{l+1} = \left\{a_i \in D_l \mid q^{\lambda_{l+1}}(a_i) = \bar{q}\right\}$" by "we compute $D_{l+1} = \left\{a_i \in D_l \mid q^{\lambda_{l+1}}(a_i) = \underline{q}\right\}$".

Representation of the results

At the end of the distillation processes, we obtain two lists of actions, associated with ranks which correspond to the iteration at which the action was extracted. To obtain an idea of the relations which exist between these different actions, we simply have to represent these actions in a graph of the ascending distillation rank versus the descending distillation rank.

For example, if after the distillation processes we obtain the lists of actions represented in Table 6.24, the representation of these actions will give the graph shown in Fig. 6.11.

Table 6.24. Results of the distillation processes.

Action	Rank computed during the descending distillation	Rank computed during the ascending distillation
a_1	1	3
a_2	2	2
a_3	4	4
a_4	3	1

6.3.5.3 Discussion

This method is highly configurable. This is also one of its main drawbacks. It is hard, even for an expert, to find the good coefficients and to justify the choices.

Nevertheless, ELECTRE III offers the decision maker a tool to draw the solutions in a 2D space; in general, these solutions have a dimension greater than 2 (the dimension of an action is equal to the number of criteria).

As we may notice, actions which have a low qualification (recall that the qualification is the difference between the number of criteria which "vote" for an action and the number of criteria which "vote" against that action) will obtain a classification after the descending distillation (they will be at the end of the sorting) which will

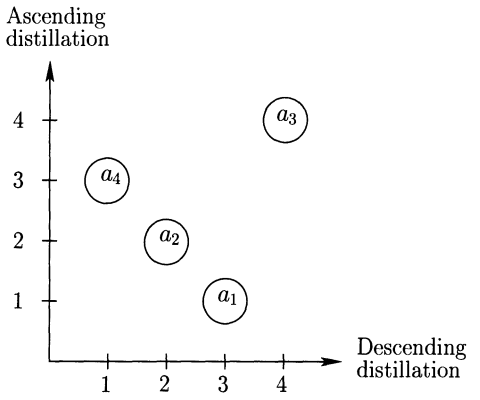


Fig. 6.11. Representation of the actions after the distillation processes.

be totally different from the classification obtained after the ascending distillation (they will be at the beginning of the sorting). We describe these actions as "floating". These are actions which we have difficulties performing choices about. Because the qualification of these actions is low, they have almost as many criteria which vote for the outranking as they have criteria which vote against.

6.3.6 The ELECTRE IV method

6.3.6.1 Principle

This is also a method dedicated to the classification of actions. It is directly inspired by the ELECTRE III method. The main difference arises from the fact that we do not need to give a weight to each criterion [Vincke 89, Maystre et al. 94].

6.3.6.2 Presentation of the method

We begin by comparing actions pair by pair. We define the following notation:

- $m_p(a_i, a_k)$ is the number of criteria for which action a_i is strictly preferred to action a_k.
- $m_q(a_i, a_k)$ is the number of criteria for which action a_i is preferred to action a_k.
- $m_{in}(a_i, a_k)$ is the number of criteria for which actions a_i and a_k are considered as indifferent, although action a_i has a better valuation than action a_k.
- $m_o(a_i, a_k) = m_o(a_k, a_i)$ is the number of criteria for which actions a_i and a_k have the same valuation.

If m is the total number of criteria, we must have

$$m = m_p(a_i, a_k) + m_q(a_i, a_k) + m_{in}(a_i, a_k) + m_o(a_i, a_k) + m_{in}(a_k, a_i) + m_q(a_k, a_i) + m_p(a_k, a_i)$$

We may notice a certain correspondence between this notation and the notation introduced for the ELECTRE III method. This correspondence is illustrated in Fig. 6.12:

Area 1: action a_i is strictly preferred to action a_k ($a_i P a_k$).

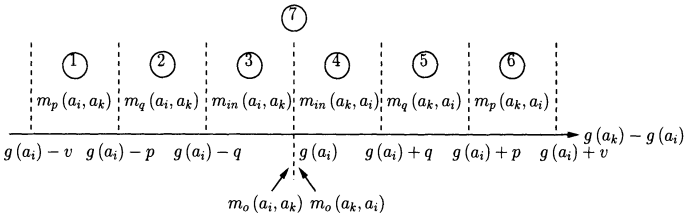


Fig. 6.12. Correspondence between the notations of the ELECTRE III and ELECTRE IV methods.

Area 2: action a_i is weakly preferred to action a_k ($a_i\, Q\, a_k$).
Area 3: action a_i is hardly preferred to action a_k ($a_i\, I\, a_k$).
Area 4: action a_k is hardly preferred to action a_i ($a_k\, I\, a_i$).
Area 5: action a_k is weakly preferred to action a_i ($a_k\, Q\, a_i$).
Area 6: action a_k is strictly preferred to action a_i ($a_k\, P\, a_i$).
Area 7: actions a_i and a_k are indifferent.

Definitions related to the ELECTRE IV method

Next, we define four levels of domination in the outranking relation:

Definition 47. ***Quasi-domination***
We denote this relation by S_q: a_i outranks a_k with quasi-domination if

$$a_i\, S_q\, a_k \Leftrightarrow \begin{cases} m_p(a_k, a_i) + m_q(a_k, a_i) = 0 & \text{and} \\ m_{in}(a_k, a_i) \leq 1 + m_{in}(a_i, a_k) + m_p(a_i, a_k) + m_q(a_i, a_k) \end{cases}$$

Definition 48. ***Canonical domination***
We denote this relation by S_c: a_i outranks a_k with canonical domination if

$$a_i\, S_c\, a_k \Leftrightarrow \begin{cases} m_p(a_k, a_i) = 0 & \text{and} \\ m_q(a_k, a_i) \leq m_p(a_i, a_k) & \text{and} \\ m_q(a_k, a_i) + m_{in}(a_i, a_k) \leq 1 + m_{in}(a_i, a_k) \\ \qquad + m_q(a_i, a_k) + m_p(a_i, a_k) \end{cases}$$

Definition 49. ***Pseudo-domination***
We denote this relation by S_p: a_i outranks a_k with pseudo-domination if

$$a_i\, S_p\, a_k \Leftrightarrow \begin{cases} m_p(a_k, a_i) = 0 & \text{and} \\ m_q(a_k, a_i) \leq m_q(a_i, a_k) + m_p(a_i, a_k) \end{cases}$$

Definition 50. *Veto-domination*
We denote this relation by S_v: a_i outranks a_k with veto-domination if

$$a_i \, S_v \, a_k \Leftrightarrow \begin{cases} m_p(a_k, a_i) = 0 & or \\ m_q(a_k, a_i) = 1 & and \\ not \; a_k \, P_{v_j} \, a_i, \; \forall j & and \\ m_p(a_i, a_k) \geq \frac{m}{2} \end{cases}$$

Exploitation of the method

Once the outranking relations have been determined, we proceed to the fuzzy outranking relation, by associating a credibility degree with S_q, S_c, S_p and S_v. Lastly, we apply the same classification method (ascending distillation and descending distillation) as with the ELECTRE III method.

6.3.6.3 Discussion

The main advantage of this method is that it removes the weights associated with the various criteria. At the end of the analysis, however, we find a similar, though less pronounced, drawback, arising through the choice of the credibility degree associated with relations S_q, S_c, S_p and S_v.

The advantage lies in the fact that there are just four coefficients to choose, when we can have a lot more criteria (and many more weights to determine in the ELECTRE III method). Nevertheless, we find again the complexity of the ELECTRE III method.

6.3.7 The ELECTRE TRI method

6.3.7.1 Principle

This method allows us to classify the actions into various categories [Maystre et al. 94]. The categories are separated by some "reference action". As the comparison is done between actions and reference actions, this method allows us to deal with many more actions. For example, if we have 100 actions and 5 reference actions, we will have to perform 100×5 comparisons in the ELECTRE TRI method. If we use another method, we may have to perform 100×99 comparisons (each action is compared with all the others).

6.3.7.2 Presentation of the method

The choice of reference actions

We choose actions which are perfectly comparable between them: each of these actions outranks or is outranked by the others. We speak then of "simple multiobjective segmentation".

Once the reference actions have been chosen, we must close the classification categories. To do so, we proceed the following way:

- we take an action which is a lower bound of the lower category,
- we take an action which is an upper bound of the upper category.

Computation of the parameters

We compute:

- the concordance indices by criterion,
- the global concordance indices,
- the nondiscordance indices by criterion.

All these parameters are computed in the same way as in the ELECTRE III method.

The credibility degrees are computed as in the ELECTRE III method, but with respect to the reference actions. We denote by b_k the kth reference action k. The credibility degree is given by

$$\sigma_s(a_i, b_k) = C(a_i, b_k) \cdot \prod_{j \in \bar{F}} \frac{1 - d_j(a_i, b_k)}{1 - C(a_i, b_k)} \tag{6.17}$$

where $C(a_i, b_k)$ is the global concordance index, $\bar{F}$ is defined by

$$\bar{F} = \{j \mid j \in F,\ d_j(a_i, b_k) > C(a_i, b_k)\} \tag{6.18}$$

and $C(a_i, b_k) \equiv C_{ik}$.

We now establish the outranking relations. The outranking relation between action a and the reference action b is established using the credibility degrees and a "cut" threshold λ, which is constant. The process we have to follow to establish the relation between two actions a and b is represented in Fig. 6.13.

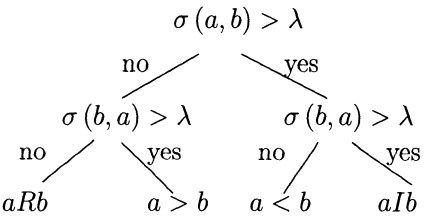


Fig. 6.13. The outranking relation of the ELECTRE TRI method.

Before performing the assignment of the actions to the various categories, we must satisfy seven requirements:

1. No action can be indifferent to more than one reference action.
2. Each action must be assigned to one and only one category (uniqueness).
3. The assignment of an action doesn't depend of the affectation of the other actions (independence).
4. Affectation of actions to categories must be conform to the definition of the reference actions (conformity).
5. When two actions compare identically with the reference actions, they must be assigned to the same category (homogeneity).
6. If action a' dominates action a (that is to say $\forall j,\ g_j(a') > g_j(a)$) then action a' must be assigned to a category greater than or equal to the category of the action a (monotonicity).

7. The gathering together of two neighboring categories must not modify the assignment of the actions to categories which are not of concern (stability).

We can now proceed to the assignment of actions to the categories that we have defined above.

There exist two types of assignment process: optimistic assignment and pessimistic assignment. These are defined in Table 6.25.

Table 6.25. The various assignment processes.

Assignment process	Pessimistic	Optimistic
Objective	Sort the actions into the lowest possible categories.	Sort the actions into the highest possible categories.
Process	Put an action in a category so that this action outranks the lower reference action of this category; $a\,S\,b^h \Rightarrow a \in C^{h+1}$.	Put an action in a category so that the higher reference action is preferred to this action; $b^h > a \Rightarrow a \in C^h$.
Direction	From top to bottom.	From bottom to top.

Next, we can represent the result of this classification using a figure similar to Fig. 6.14. In this figure, we have classified the actions in three categories: C_1, C_2 and C_3.

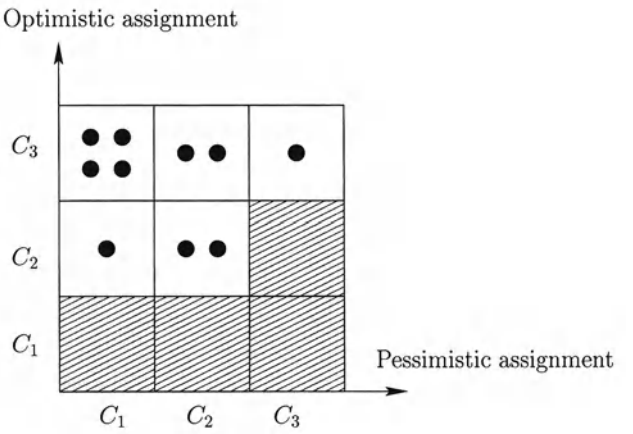


Fig. 6.14. Result of both affectation processes.

6.3.7.3 Discussion

The main difficulty of this method lies in the fact that we must be able to compare each action with the actions which bound the various categories. This comparison is not always possible. It is also difficult to define the categories a priori, because the definition of the categories is very much linked to the choice of the reference actions.

An industrial application of the ELECTRE TRI method is presented in Chap. 12.

6.3.8 The PROMETHEE I method

6.3.8.1 Principle

The PROMETHEE method (*preference ranking organization method for enrichment evaluations*) allows one to construct a partial preorder (equivalent solutions may exist). To do so, it uses a valuated preference relation, which gives birth to a valuated preference graph. The method has been defined by Brans, Vincke and Mareschal [Brans et al. 86] and belongs to the family of outranking methods, as defined by B. Roy.

6.3.8.2 Presentation of the method

The valuated preference relation is defined the following way:

- $P(a,b) = 0$ means indifference between action a and action b or no preference for action a against action b.
- $P(a,b) \simeq 0$ means a weak preference for action a against b.
- $P(a,b) \simeq 1$ means a strong preference for action a against b.
- $P(a,b) = 1$ means a strict preference for action a against b.

The link between the preference relation and any criterion g_j is the following:

$$P_j(a,b) = G(g_j(a) - g_j(b)) \tag{6.19}$$

The function $G(x)$ must be nondecreasing for $x > 0$, and equal to 0 for $x \leq 0$. An example of such a function $G(x)$ is plotted in Fig. 6.15.

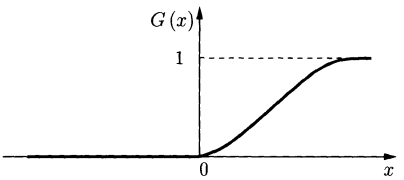


Fig. 6.15. Example of function $G(x)$.

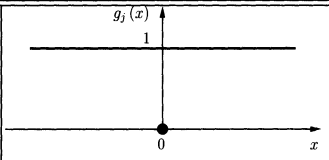
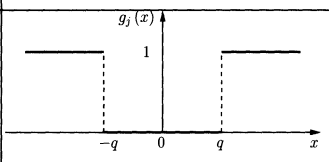
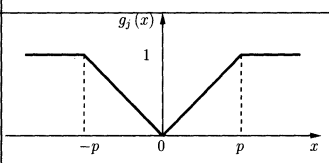
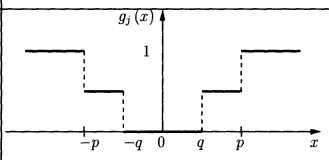
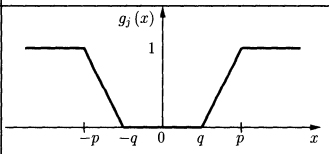
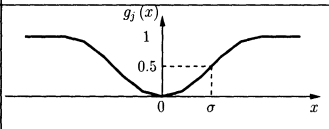
The function $g_j(x)$ is called a "generalized criterion". It can take several forms. These various forms are listed in Table 6.26.

The various generalized criteria have the following equations:

- Usual criterion:

$$g_j(x) = \begin{cases} 0 \text{ if } x = 0 \\ 1 \text{ if } x \neq 0 \end{cases} \tag{6.20}$$

Table 6.26. Various generalized criteria.

Form of the generalized criterion	
Usual criterion	
Quasi-criterion	
Linear preference criterion	
Levels criterion	
Criterion with a linear preference and an area of indifference	
Gaussian criterion	

- Quasi-criterion:

$$g_j(x) = \begin{cases} 0 \text{ if } -q \leq x \leq q \\ 1 \text{ if } x < -q \text{ or } x > q \end{cases} \tag{6.21}$$

- Linear preference criterion:

$$g_j(x) = \begin{cases} \frac{|x|}{p} \text{ if } -p \leq x \leq p \\ 1 \quad \text{if } x < -p \text{ or } x > p \end{cases} \tag{6.22}$$

- Levels criterion:

$$g_j(x) = \begin{cases} 0 \text{ if } |x| \leq q \\ \frac{1}{2} \text{ if } q < |x| \leq p \\ 1 \text{ if } p < |x| \end{cases} \tag{6.23}$$

- Criterion with a linear preference and an area of indifference:

$$g_j(x) = \begin{cases} 0 & \text{if } |x| \leq q \\ \frac{(|x|-q)}{(p-q)} & \text{if } q < |x| \leq p \\ 1 & \text{if } p < |x| \end{cases} \tag{6.24}$$

- Gaussian criterion:

$$g_j(x) = 1 - \exp\left(-\frac{x^2}{2 \cdot \sigma^2}\right) \tag{6.25}$$

Let us now define the multiobjective outranking index of action a against action b (denoted by $P(a,b)$):

$$P(a,b) = \frac{\sum_{i=1}^{k} \pi_i \cdot P_i(a,b)}{\sum_{i=1}^{k} \pi_i} \tag{6.26}$$

where π_i is the weight of the criterion i, and P_i is the preference function related to criterion i. This index varies between 0 and 1:

- $P(a,b) \simeq 0$ means a weak preference for action a against action b for the whole set of criteria;
- $P(a,b) \simeq 1$ means a strong preference for action a against action b for the whole set of criteria.

We shall now present some definitions related to the representation of the outranking relations as a graph. The representation as a graph is similar to the one used in the ELECTRE methods. The single difference lies in the fact that we write the numerical values of the preferences on the edges. An example is given in Fig. 6.16.

On a graph of this type, we define:

- the flux coming into the node a:

$$\phi^-(a) = \sum_{b \in A} P(b,a) \tag{6.27}$$

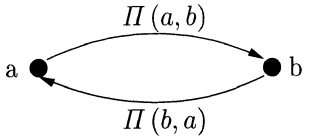


Fig. 6.16. Graph representing the outranking relation between a and b.

- the flux coming out of the node a:

$$\phi^{+}(a) = \sum_{b \in A} P(a,b) \tag{6.28}$$

- the net flux of node a:

$$\phi(a) = \phi^{+}(a) - \phi^{-}(a) \tag{6.29}$$

We can now define the following two preorders:

$$\begin{cases} a \, \mathrm{P}^{+} \, b \text{ if } \phi^{+}(a) > \phi^{+}(b) \\ a \, I^{+} \, b \text{ if } \phi^{+}(a) = \phi^{+}(b) \end{cases} \tag{6.30}$$

$$\begin{cases} a \, P^{-} \, b \text{ if } \phi^{-}(a) < \phi^{-}(b) \\ a \, I^{-} \, b \text{ if } \phi^{-}(a) = \phi^{-}(b) \end{cases} \tag{6.31}$$

Lastly, we can define the partial preorders of the PROMETHEE I method:

- action a outranks action b (denoted by $a \, P_I \, b$):

$$\begin{cases} \text{if } a \, P^{+} \, b \text{ and } a \, P^{-} \, b \\ \text{or } a \, P^{+} \, b \text{ and } a \, I^{-} \, b \\ \text{or } a \, I^{+} \, b \text{ and } a \, P^{-} \, b \end{cases} \tag{6.32}$$

- actions a and b are indifferent (denoted by $a \, I_I \, b$) if $a \, I^{+} \, b$ and $a \, I^{-} \, b$;
- otherwise, actions a and b are incomparable (denoted by $a \, R \, b$).

6.3.8.3 Discussion

The PROMETHEE I method gives the decision maker a classification of the various actions. The problem is that this method does not allow us to classify all the actions. Some actions can remain incomparable.

The PROMETHEE II overcomes this drawback.

6.3.9 The PROMETHEE II method

6.3.9.1 Principle

This method allows us to classify all the actions and does not leave any actions incomparable with respect to the others [Brans et al. 86]. So, it realizes a complete preorder.

6.3.9.2 Presentation of the method

The only difference from the PROMETHEE I method lies in the definition of the outranking relation which is the following:

- action a outranks action b (denoted by $a\, P_{II}\, b$) if $\phi(a) > \phi(b)$,
- actions a and b are indifferent if $\phi(a) = \phi(b)$.

6.3.9.3 Discussion

It is simpler for the decision maker to give an answer to a decision problem using a complete preorder. Nevertheless, a partial preorder can contain more informations. Indeed, incomparability of an action can be very useful when one is making a decision.

6.4 Annotated bibliography

[Maystre et al. 94] In this book, we find a description of all the ELECTRE decision aid methods. We find, in addition to a detailed description of each method, applications of these methods to examples where each step of the method is clearly set out. This book is a reference source and all the notation we have used in this chapter comes from this book. In French.

[Scharlig 85] An important book for any reader who wants to be introduced to the ideas which belong to the field of decision aids. This book was written by a former scientific journalist involved in decision aids. The style of this book is appealing. It is clear, and the author succeeds in presenting all of the ideas with the minimum number of equations. A really good book. In French.

[Scharlig 96] This book is a sequel to [Scharlig 85]. It describes the ELECTRE and PROMETHEE methods with the same economy of equations.

[Roy et al. 93] A book by the creator of the field of multicriteria decision aids. This book contains all the elements related to multicriteria decision aids and more. It has the advantage that it can be read at several levels. It is the book that people using decision aids must have.

Part II

Evaluation of methods, and criteria for choice of method

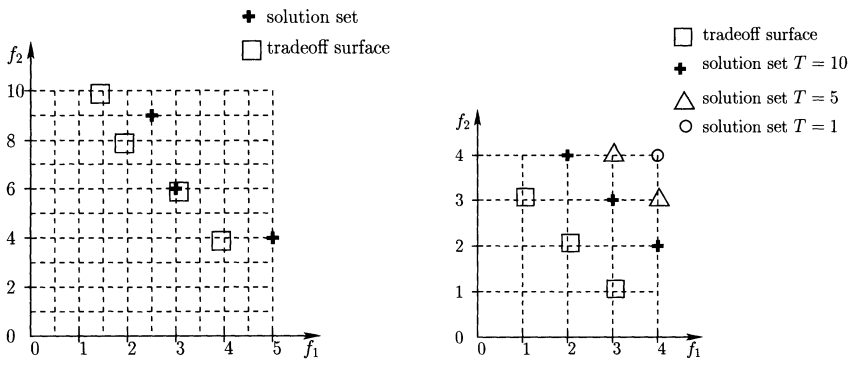
7

Performance measurement

7.1 Introduction

To measure the performance of a multiobjective optimization method, we must have some "measurement instruments". We have mainly used the performance indices presented in [Van Veldhuizen 99].

To illustrate the formulas presented later, we present two examples (see in Fig. 7.1). The first one is extracted from [Van Veldhuizen 99]. The second one illustrates the progress of a biobjective optimization method in the solution of a problem (we want to minimize the objective functions f_1 and f_2); we have presented the tradeoff surface and the evolution of the solution set over three generations.



(a) Example 1.

(b) Example 2.

Fig. 7.1. Two examples which illustrate the performance indices.

In Fig. 7.1b, the parameter T corresponds to the iteration at which the solution set has been obtained.

In this chapter, we shall use the following notation to define the various solution sets:

- The optimal tradeoff surface is denoted by TPF (theoretical Pareto frontier).
- The solution set obtained at iteration T is denoted by $PF_{\text{current}}(T)$ (Pareto front current).
- The result of gathering together the nondominated solutions of each of the sets $PF_{\text{current}}(i)$, $i = 1, \cdots, T$, is denoted by $PPF(T)$ (practical Pareto frontier). The mathematical definition of this set is the following:

$$PPF(T) = PF_{\text{current}}(T) \bigcup PPF(T-1)$$

and

$$PPF(1) = PF_{\text{current}}(1)$$

after all the nondominated solutions of $PPF(T)$ have been removed.

In this chapter, we shall present several measurements that will allow us to "visualize" characteristics of the solution set. We call these measurements "metrics".

7.2 Error ratio

7.2.1 Definition

This ratio allows us to measure the nonconvergence of a multiobjective optimization method toward the tradeoff surface. The definition of the error ratio is the following:

$$E = \frac{\sum_{i=1}^{n} e_i}{n} \tag{7.1}$$

where n is the number of elements in the solution set, and

$$e_i \quad \begin{cases} = 0 \text{ if the solution } i \in \text{tradeoff surface} \\ = 1 \text{ otherwise} \end{cases}$$

The closer this metric is to 1, the less the solution set has converged toward the tradeoff surface.

7.2.2 Examples

Example 1

In this example, we have:

- For the point of coordinates $(2.5, 9)$, $e_1 = 1$ because this point does not belong to the tradeoff surface.
- For the point of coordinates $(3, 6)$, $e_2 = 0$ because this point belongs to the tradeoff surface.
- For the point of coordinates $(5, 4)$, $e_3 = 1$ because this point does not belong to the tradeoff surface.

The index E is therefore equal to $\frac{2}{3}$.

Example 2

In this example, we compute the error ratio index for the solution set obtained at iteration $T = 10$. In this case, because none of these points belong to the tradeoff surface, we have $e_i = 1$, for $i = 1, 2, 3$.

The index E is therefore equal to 1.

7.2.3 Discussion

As we can see from these examples, the closer the metric E is to 1 the less the solution set has converged toward the tradeoff surface.

To know whether or not an element belongs to the tradeoff surface, it is enough to gather together both sets (tradeoff surface TPF + solutions set PPF), and then to sort the elements of this new set using the Pareto ranking. If the rank of the considered point is equal to 1, then it belongs to the tradeoff surface.

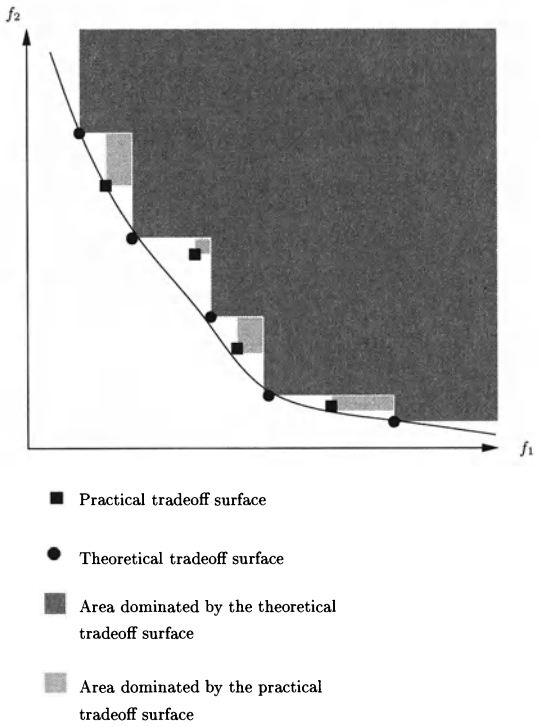


Fig. 7.2. Example for which the error ratio is equal to 1, but the tradeoff surface has not converged toward the theoretical Pareto surface.

As we can see in Fig. 7.2, in some cases the error ratio may be equal to 1 without having a tradeoff surface which has converged toward the theoretical Pareto surface. It is enough to have the solutions of the practical Pareto surface, which lie in the nondomination area defined by the points of the theoretical Pareto surface. This

"problem" is due to the fact that, to be able to compute the error ratio, we have to discretize the theoretical Pareto surface and, as a result, areas of nondomination appear.

7.3 Generational distance

7.3.1 Definition

The generational distance allows us to measure the distance between the tradeoff surface and the solution set. The definition of this metric is the following:

$$G = \frac{\left(\sum_{i=1}^{n} d_i^p\right)^{\frac{1}{p}}}{n} \tag{7.2}$$

In general, we choose $p = 2$. In the above definition, d_i is the distance between solution i and the closest solution which belongs to the tradeoff surface TPF, and n is the number of elements in the solution set PPF.

7.3.2 Examples

Example 1

Let us apply this definition to the elements of the solutions set:

$$G = \frac{\left[\left[(2.5-2)^2+(9-8)^2\right]+\left[(3-3)^2+(6-6)^2\right]+\left[(5-4)^2+(4-4)^2\right]\right]^{\frac{1}{2}}}{3} = 0.5$$

Example 2

In this example, we compute this metric for three solution sets ($T = 1$, $T = 5$, $T = 10$):

- For $T = 1$, we have

$$G_1 = \frac{\left[(4-2)^2+(4-2)^2\right]^{\frac{1}{2}}}{1} = 2.82$$

- For $T = 5$, we have

$$G_2 = \frac{\left[\left[(4-2)^2+(3-2)^2\right]+\left[(3-2)^2+(4-2)^2\right]\right]^{\frac{1}{2}}}{2} = 1.58$$

- For $T = 10$, we have

$$G_3 = \frac{\left[\left[(3-4)^2+(1-2)^2\right]+\left[(2-3)^2+(2-3)^2\right]+\left[(1-2)^2+(3-4)^2\right]\right]^{\frac{1}{2}}}{3} = 0.81$$

7.3.3 Discussion

As example 2 shows, the farther from the tradeoff surface the solutions set is, the greater is the generational distance.

The metric could be used as a stopping criterion for the optimization method, provided that we know how to obtain the tradeoff surface TPF (which is only the case when we use a multiobjective test problem). We can also use the derivative of this metric as a stopping criterion. If the variation of G is small, iteration after iteration, this will mean that the solution set PPF is not coming closer to the optimal tradeoff surface. This circumstance arises when the solution set has converged toward a "ghost" Pareto surface which is different from the optimal Pareto surface.

There exists a modified version of the generational distance:

$$GD = \frac{\sum_{i=1}^{n} d_i}{n} \tag{7.3}$$

GD corresponds to a mean distance with respect to the tradeoff surface. It is used with the metric presented in the following section.

7.4 The STDGD metric

7.4.1 Definition

This metric ("standard deviation from the generational distance") allows us to measure the distortion of the computed tradeoff surface relative to the theoretical tradeoff surface. The definition is the following:

$$STDGD = \frac{\sum_{i=1}^{n} (d_i - GD)^2}{n} \tag{7.4}$$

The principle of this metric is illustrated in Fig. 7.3 where PPF corresponds to the computed tradeoff surface, TPF corresponds to the theoretical tradeoff surface, and GD refers to the generational distance. The gray area corresponds to the surface which is computed by the STDGD metric.

7.4.2 Discussion

This metric computes the error as the least mean squared error between the theoretical tradeoff surface, translated by the GD distance, and the computed tradeoff surface. This measure relates to the distortion between the computed tradeoff surface and the theoretical tradeoff surface.

This metric was developed with the aim of distinguishing methods which compute a distorted tradeoff surface from methods which compute a tradeoff surface which looks like the theoretical tradeoff surface.

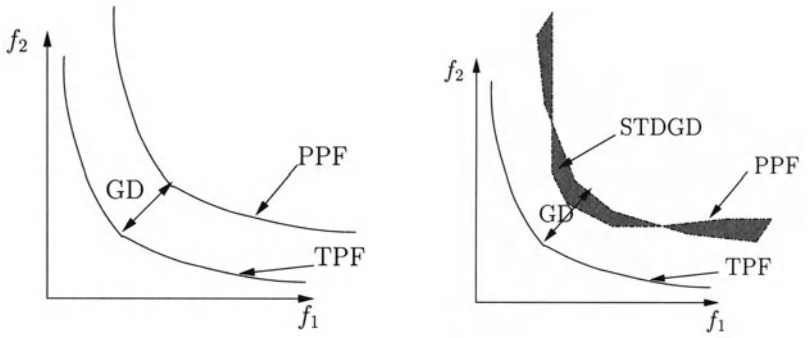


Fig. 7.3. The definition of the STDGD metric.

7.5 Maximal error with respect to the tradeoff surface

7.5.1 Definition

This metric (the maximum Pareto front error) allows us to measure the distance between the tradeoff surface and the solution set. Its definition is the following (for a biobjective optimization problem):

$$ME = \max_j \left(\min_i \left(\left| f_1^i(\vec{x}) - f_1^j(\vec{x}) \right|^p + \left| f_2^i(\vec{x}) - f_2^j(\vec{x}) \right|^p \right)^{\frac{1}{p}} \right) \quad (7.5)$$

It is, in fact, the greatest minimal distance between elements of the solutions set and their closest neighbors which belong to the tradeoff surface.

7.5.2 Examples

Example 1

To find the greatest minimal distance, we consider each point of the solution set and compute the various distances with respect to the points of the tradeoff surface (see Table 7.1).

- For the point of the solution set of coordinates $(2.5, 9)$, the minimal distance is 1.12;
- For the point of the solution set of coordinates $(3, 6)$, the minimal distance is 0;
- For the point of the solution set of coordinates $(5, 4)$, the minimal distance is 1.

The greatest minimal distance is 1.12, so $ME = 1.12$.

Example 2

We shall follow the same steps as with example 1. Here, the greatest minimal distances are:

Table 7.1. Example 1: computation of the maximal error with respect to the tradeoff surface.

Point of the solution set	Point of the tradeoff surface	Distance
(2.5, 9)	(1.5, 10)	1.41
(2.5, 9)	(2, 8)	1.12
(2.5, 9)	(3, 6)	3.04
(2.5, 9)	(4, 4)	5.22
(3, 6)	(1.5, 10)	4.27
(3, 6)	(2, 8)	2.23
(3, 6)	(3, 6)	0
(3, 6)	(4, 4)	2.23
(5, 4)	(1.5, 10)	6.94
(5, 4)	(2, 8)	5
(5, 4)	(3, 6)	2.82
(5, 4)	(4, 4)	1

Table 7.2. Example 2 (for $T = 10$): computation of the maximal error with respect to the tradeoff surface.

Point of the solution set	Point of the tradeoff surface	Distance
(4, 2)	(3, 1)	1.41
(4, 2)	(2, 2)	2
(4, 2)	(1, 3)	3.16
(3, 3)	(3, 1)	2
(3, 3)	(2, 2)	1.41
(3, 3)	(1, 3)	2
(2, 4)	(3, 1)	3.16
(2, 4)	(2, 2)	2
(2, 4)	(1, 3)	1.41

Table 7.3. Example 2 (for $T = 5$): computation of the maximal error with respect to the tradeoff surface.

Point of the solution set	Point of the tradeoff surface	Distance
(4, 3)	(3, 1)	2.23
(4, 3)	(2, 2)	2.23
(4, 3)	(1, 3)	3
(3, 4)	(3, 1)	3
(3, 4)	(2, 2)	2.23
(3, 4)	(1, 3)	2.23

Table 7.4. Example 2 (for $T = 1$): computation of the maximal error with respect to the tradeoff surface.

Point of the solution set	Point of the tradeoff surface	Distance
(4, 4)	(3, 1)	3.16
(4, 4)	(2, 2)	2.82
(4, 4)	(1, 3)	3.16

- for the solution set with $T = 10$ (see Table 7.2), $ME = 1.41$;
- for the solution set with $T = 5$ (see Table 7.3), $ME = 2.23$;
- for the solution set with $T = 1$ (see Table 7.4), $ME = 2.82$.

7.5.3 Discussion

As example 2 shows, the more distant the solution set is from the tradeoff surface TPF, the greater the metric ME is. So, this metric measures the distance between the tradeoff surface TPF and the solution set PPF.

7.6 Hyperarea and hyperarea ratio

7.6.1 Definition

This metric allows us to measure the area occupied by the solution set PPF. Its definition is the following:

$$H = \left\{ \bigcup_i a_i \,|\, v_i \in PPF \right\} \tag{7.6}$$

where a_i is the area occupied by the solution v_i (see in Fig. 7.4 for an illustration of this area).

A way to compute this metric when considering a solution set PPF with two dimensions is the following:

Step 1: Extract the tradeoff surface from the solutions set PPF. We call this tradeoff surface S_c.

Step 2: Sort the elements of S_c by increasing order with respect to the objective function f_1.

Step 3: Apply the following algorithm:

1. $i = 0$, $H = f_1(x_0) \cdot f_2(x_0)$
2. $i = i + 1$
3. if $i \neq N$, $H = H + [f_1(x_i) - f_1(x_{i-1})] \cdot f_2(x_i)$ then go to step 2
4. if $i = N$ then end

where N is the number of elements in the solution set S_c, and x_i is one element of the solution set S_c.

We can also define the hyperarea ratio:

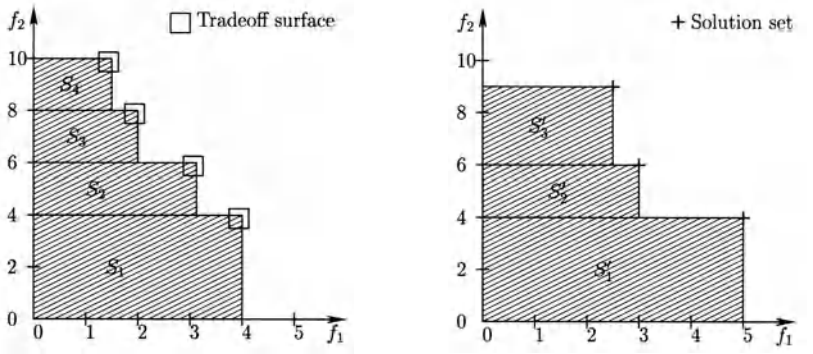
$$HR = \frac{H_{es}}{H_{sc}} \tag{7.7}$$

where H_{es} is the area occupied by the solution set PPF, and H_{sc} is the area occupied by the tradeoff surface TPF.

7.6.2 Examples

Example 1

In this example, we compute both areas.



(a) The area of the tradeoff surface.

(b) The area of the solutions set.

Fig. 7.4. The computation of the areas.

For the solution set, we have:

$H_{es} = S_1 + S_2 + S_4$
$H_{es} = 5 \cdot 4 + 3 \cdot (6 - 4) + 2.5 \cdot (9 - 6)$

For the tradeoff surface, we have:

$H_{sc} = S_1 + S_2 + S_3 + S_4$
$H_{sc} = 4 \cdot 4 + 3 \cdot (6 - 4) + 2 \cdot (8 - 6) + 1.5 \cdot (10 - 8)$

Hence:

$H_{es} = 33.5$
$H_{sc} = 29$

We compute the hyperarea ratio as

$HR = \frac{H_{es}}{H_{sc}} = \frac{33.5}{29} = 1.155$

Example 2

We compute the areas for the three solution sets and for the tradeoff surface.

- For the tradeoff surface:

$H_{sc} = 3 \cdot 1 + 2 \cdot (2 - 1) + 1 \cdot (3 - 2)$
$H_{sc} = 6$

- For the solutions set such that $T = 10$:
  $H_{es_1} = 4 \cdot 2 + 3 \cdot (3-2) + 2 \cdot (4-3)$
  $H_{es_1} = 13$
  So, we have the following hyperarea ratio:
  $HR_1 = \frac{H_{es_1}}{H_{sc}} = \frac{13}{6} = 2.17$
- For the solutions set such that $T = 5$:
  $H_{es_2} = 4 \cdot 3 + 3 \cdot (4-3)$
  $H_{es_2} = 15$
  So, we have the following hyperarea ratio:
  $HR_2 = \frac{H_{es_2}}{H_{sc}} = \frac{15}{6} = 2.5$
- For the solutions set such that $T = 1$:
  $H_{es_3} = 4 \cdot 4$
  $H_{es_3} = 16$
  So, we have the following hyperarea ratio:
  $HR_3 = \frac{H_{es_3}}{H_{sc}} = \frac{16}{6} = 2.67$

7.6.3 Discussion

As with the "maximal error with respect to the tradeoff surface" and "generational distance" metrics, the hyperarea ratio allows us to measure the convergence of the solution set PPF toward the tradeoff surface TPF. The closer the value of this metric is to 1, the better is the tradeoff surface.

- If the value of the hyperarea ratio is smaller than 1, the area occupied by the practical tradeoff surface is smaller than the area occupied by the theoretical tradeoff surface. We can say that the points of the practical tradeoff surface are not spread over the whole tradeoff surface.
- If the value of the hyperarea ratio is greater than 1, the practical tradeoff surface is distant from the theoretical tradeoff surface. We have not sufficiently converged, so this metric gives us information about the spread of the points of the theoretical tradeoff surface.

7.7 Spacing metric

7.7.1 Definition

The definition of the spacing metric is the following:

$$S = \left[\frac{1}{n-1} \cdot \sum_{i=1}^{n} (\bar{d} - d_i)^2 \right]^{\frac{1}{2}} \tag{7.8}$$

where

$$d_i = \min_j \left(\left| f_1^i(\vec{x}) - f_1^j(\vec{x}) \right| + \left| f_2^i(\vec{x}) - f_2^j(\vec{x}) \right| \right) \tag{7.9}$$

$i, j = 1, \cdots, n$ and $i \neq j$, $\bar{d}$ is the mean value of all the d_i, and n is the number of elements in the solution set.

This metric allows us to measure the uniformity of the spread of the points of the solution set in the (f_1, f_2) plane.

7.7.2 Examples

Example 1

Point i	Point j	Distance
$(2.5, 9)$	$(3, 6)$	3.5
$(2.5, 9)$	$(5, 4)$	7.5
$(3, 6)$	$(2.5, 9)$	3.5
$(3, 6)$	$(5, 4)$	4
$(5, 4)$	$(2.5, 9)$	7.5
$(5, 4)$	$(3, 6)$	4

For the point of coordinates $(2.5, 9)$, $d_1 = 3.5$. For the point of coordinates $(3, 6)$, $d_2 = 3.5$.

For the point of coordinates $(5, 4)$, $d_3 = 4$. So, we have

$\bar{d} = \frac{3.5+3.5+4}{3} = 3.67$

Lastly, we compute the index S:

$S = \left[\frac{1}{2} \cdot \left((3.67 - 3.5)^2 + (3.67 - 3.5)^2 + (3.67 - 4)^2\right)\right]^{\frac{1}{2}}$

$S = 0.288$

Example 2

We follow the same steps as in example 1. For the solution set where $T = 10$, we have:

Point i	Point j	Distance
$(4, 2)$	$(3, 3)$	2
$(4, 2)$	$(2, 4)$	4
$(3, 3)$	$(4, 2)$	2
$(3, 3)$	$(2, 4)$	2
$(2, 4)$	$(4, 2)$	4
$(2, 4)$	$(3, 3)$	2

For the point of coordinates $(4, 2)$, $d_1 = 2$. For the point of coordinates $(3, 3)$, $d_2 = 2$. For the point of coordinates $(2, 4)$, $d_3 = 2$. So, we have $\bar{d} = 2$ and $S = 0$.

It is not interesting to compute this index for the solution set where $T = 5$, because there are only two points, they will evidently be well spaced. For the solution set where $T = 1$, it is nonsense to compute this index.

7.7.3 Discussion

As these two examples show, the spacing metric allows us to measure the uniformity of the spreading of solutions in a set (PPF or TPF). The closer this metric is to 0, the better the solutions are spread.

In some cases, this metric may be hard to interpret. When we are working on a discontinuous test problem, the tradeoff surface of this problem has some holes (discontinuities introduced by the shape of the tradeoff surface). These holes will increase the value computed using this metric. So, it is better not to use this metric for a discontinuous test problem.

7.8 The HRS metric

7.8.1 Definition

A spacing metric which computes a mean error with respect to an ideal spacing has a tendency to "hide" some important gaps. Because of this we have developed the HRS (hole relative size) metric. This allows us to measure the size of the biggest hole in the spacing of the points on the tradeoff surface. The definition of the HRS metric is the following:

$$HRS = \frac{\max_i d_i}{\bar{d}} \tag{7.10}$$

where $\bar{d}$ and d_i have the same meaning as in the spacing metric.

7.9 Overall nondominated vector generation metric

7.9.1 Definition

First, we create a set which contains the best nondominated solutions obtained after each generation (the PPF set). Next, we define a metric which is the cardinality of this set:

$$ONVG = \text{card}\,(PPF) \tag{7.11}$$

Then, we can compare the cardinality of the solution set PPF with the cardinality of the tradeoff surface TPF, by computing the ratio

$$ONVGR = \frac{\text{card}\,(PPF)}{\text{card}\,(TPF)} \tag{7.12}$$

7.9.2 Examples

Example 1

In this example, the tradeoff surface TPF is composed of four nondominated solutions and the solution set PPF is composed of three nondominated solutions. So we have

$ONVG = 3$
$ONVGR = \frac{3}{4} = 0.75$

Example 2

- For the solution set where $T = 10$, we have

  $ONVG_1 = 3$
  $ONVGR_1 = \frac{3}{3} = 1$

- For the solution set where $T = 5$, we have

  $ONVG_2 = 2$
  $ONVGR_2 = \frac{2}{3} = 0.66$

- For the solution set where $T = 1$, we have

  $ONVG_1 = 1$
  $ONVGR = \frac{1}{3} = 0.33$

7.9.3 Discussion

The overall nondominated vector generation ratio tells us the number of nondominated solutions in the solution set PPF as a proportion of the number in the tradeoff surface TPF.

7.10 Progress measure

7.10.1 Definition

We measure the improvement of the objective function f_i by using the following metric:

$$P = \ln \sqrt{\frac{f_{i_{max}}(0)}{f_{i_{max}}(k)}} \tag{7.13}$$

where $f_{i_{max}}(k)$ refers to the best value of the objective function i at iteration k. To take into account the possibility of the existence of multiple solutions, Van Veldhuizen modifies this expression the following way [Van Veldhuizen 99]:

$$RP = \ln \sqrt{\frac{G_0}{G_k}} \tag{7.14}$$

where G_k is the generational distance at iteration k.

7.10.2 Example

Example 2

- For the solution set PPF where $T = 1$, we have

  $f_{2_{max}}(1) = 4$ and $f_{1_{max}}(1) = 4$.

  These values will be used as a starting point for the computation of the progress measure.

- For the solution set PPF where $T = 5$, we have

  $f_{2_{max}}(5) = 3$ and $f_{1_{max}}(5) = 3$.

  So, we have:

  $$P_{11} = \ln \sqrt{\frac{f_{1_{max}}(1)}{f_{1_{max}}(5)}} = \ln \sqrt{\frac{4}{3}} = 0.14$$

  $$P_{21} = \ln \sqrt{\frac{f_{2_{max}}(1)}{f_{2_{max}}(5)}} = \ln \sqrt{\frac{4}{3}} = P_{11}$$

- For the solution set PPF where $T = 10$, we have

  $f_{2_{max}}(10) = 2$ and $f_{1_{max}}(10) = 2$.

  So, we have:

  $$P_{12} = \ln \sqrt{\frac{f_{1_{max}}(1)}{f_{1_{max}}(10)}} = \ln \sqrt{\frac{4}{2}} = 0.34$$

  $$P_{22} = \ln \sqrt{\frac{f_{2_{max}}(1)}{f_{2_{max}}(10)}} = \ln \sqrt{\frac{4}{2}} = P_{12}$$

7.10.3 Discussion

This metric allows us to measure the progress of a multiobjective optimization method toward the ideal solution.

7.11 Generational nondominated vector generation metric

This metric allows us to measure how many nondominated solutions are generated at each iteration.

$$GNVG = \text{card}\left(PF_{\text{current}}(i)\right) \tag{7.15}$$

where $PF_{\text{current}}(i)$ is the solution set at iteration i.

7.12 Nondominated vector addition

7.12.1 Definition

This metric allows us to measure how many new nondominated solutions have been added between two iterations. The definition of this metric is the following:

$$NVA = \text{card}\left(PPF(i)\right) - \text{card}\left(PPF(i-1)\right) \tag{7.16}$$

7.12.2 Example

Example 2

- For the solution set where $T = 1$, we have $\text{card}(PPF(1)) = 1$.
- For the solution set where $T = 5$, we have $\text{card}(PPF(5)) = 2$. Hence, $NVA_1 = \text{card}(PPF(5)) - \text{card}(PPF(1)) = 1$. There is an addition of a new nondominated solution between iteration 1 and iteration 5.
- For the solution set where $T = 10$, we have $\text{card}(PPF(10)) = 3$. Hence, $NVA_2 = \text{card}(PPF(10)) - \text{card}(PPF(5)) = 1$. There is an addition of a new non dominated solution between iteration 5 and iteration 10.

7.12.3 Discussion

The NVA metric can, in some cases, be negative. Indeed, the optimization method may lose nondominated solutions during the progress of the optimization. This metric can be used to stop an optimization method. If the metric is always around 0 after some number of iterations, this certainly means that this method is not able to find new nondominated solutions. Nevertheless, we must not forget that a method may find no new solutions but still be able to move these solutions toward the optimal tradeoff surface TPF.

This metric can be seen as a production "rate" (it does not produce values between 0 and 1) of nondominated solutions.

7.13 Waves metric

This metric corresponds to the maximal rank of the Pareto ranking. This rank can be computed by using algorithm 11.

Algorithm 11 MaxRankPareto.

S contains the solution set
$d = 0$
while $\text{card}(S) \neq 0$
  $S = S - \text{Pareto}(S)$
  $d = d + 1$
end while
MaxRankPareto $= d$

$$W = \text{MaxRankPareto}(PPF) \tag{7.17}$$

In algorithm 11, the Pareto function determines a set which contains the nondominated solutions of a set which is given to this function as a parameter.

If the value of this metric is close to 1, this means that the individuals tend to converge toward the tradeoff surface. In general, this metric is used during an optimization to visualize the convergence of the solution set toward the tradeoff surface. The shape of the curve that we obtain looks like waves (see in Fig. 7.5).

This metric measures the thickness of the solution set PPF in the form of the number of tradeoff surfaces that we can extract from this set.

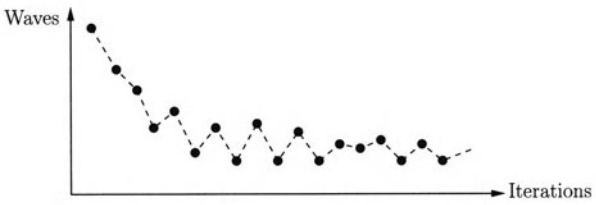


Fig. 7.5. The W metric during an optimization.

7.14 Zitzler metrics

In [Zitzler et al. 99], several performance metrics are presented. These metrics can be divided into two families:

- the relative metrics: these allow us to perform a comparison between two tradeoff surfaces;
- the absolute metrics: these allow us to measure the "quality" of a tradeoff surface.

The metrics that we have presented up to now have been absolute metrics, because they do not allow us to perform a comparison between two tradeoff surfaces.

7.14.1 The relative metrics

In [Zitzler et al. 99], only one relative metric is presented. The definition of this metric is given below.

Let two tradeoff surfaces F' and F'' be given. The following function C transforms the two tradeoff surfaces F' and F'' into a real value included in the interval $[0, 1]$:

$$C(F', F'') = \frac{\text{card}\left(\left\{\overrightarrow{x''} \in F'' ; \exists \overrightarrow{x'} \in F' \mid \overrightarrow{x'} \preceq \overrightarrow{x''}\right\}\right)}{\text{card}(F'')} \tag{7.18}$$

where $\text{card}(F)$ corresponds to the number of elements inside the set F, and $\overrightarrow{x'} \preceq \overrightarrow{x''}$ means that the vector $\overrightarrow{x'}$ dominates (or is equal to) the vector $\overrightarrow{x''}$.

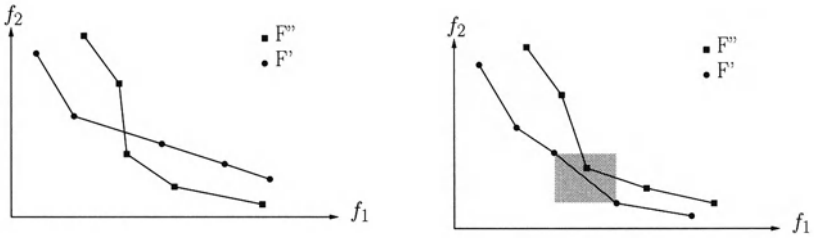
When $C(F', F'') = 1$, this means that all the solutions of the set F' are dominated by (or equal to) solutions of the set F''. When $C(F', F'') = 0$, this means that all the solutions of the set F' dominate (in the strong sense) the solutions of the set F''. We must always compute both $C(F', F'')$ and $C(F'', F')$, because $C(F', F'')$ is not necessarily equal to $1 - C(F'', F')$.

So, this metric allows us to compute what portion of the tradeoff surface F' dominates the tradeoff surface F''.

Example 1

In Fig. 7.6a, we can see two sets F' and F'', which we would like to compare using the relative Zitzler metric.

We start by computing $C(F', F'')$. $\text{card}(F'')$ corresponds to the number of solutions in the set F'': $\text{card}(F'') = 5$.



(a) First example. (b) Second example.

Fig. 7.6. Example of application of the relative Zitzler metric.

The numerator of the expression (7.18) gives us the number of solutions in the set F'' which are dominated by solutions in the set F'. In our example, we have

$$\text{card}\left(\left\{\overrightarrow{x''} \in F'' ; \exists \overrightarrow{x'} \in F' \mid \overrightarrow{x'} \preceq \overrightarrow{x''}\right\}\right) = 2$$

because two solutions in the set F'' are dominated by solutions in the set F'. So, $C(F', F'') = \frac{2}{5} = 0.4$.

Let us now compute $C(F'', F')$:

$\text{card}(F') = 5$.
$\text{card}\left(\left\{\overrightarrow{x'} \in F' ; \exists \overrightarrow{x''} \in F'' \mid \overrightarrow{x''} \preceq \overrightarrow{x'}\right\}\right) = 3$.

So, $C(F'', F') = \frac{3}{5} = 0.6$.

We obtain the following matrix:

C (column, row)	F'	F''
F'	×	0.4
F''	0.6	×

Example 2

We now use the example shown in Fig. 7.6b. We start by computing $C(F', F'')$:

$\text{card}(F'') = 5$.
$\text{card}\left(\left\{\overrightarrow{x''} \in F'' ; \exists \overrightarrow{x'} \in F' \mid \overrightarrow{x'} \preceq \overrightarrow{x''}\right\}\right) = 4$.

So, $C(F', F'') = \frac{4}{5} = 0.8$.

Let us now compute $C(F'', F')$:

$\text{card}(F') = 5$.
$\text{card}\left(\left\{\overrightarrow{x'} \in F' ; \exists \overrightarrow{x''} \in F'' \mid \overrightarrow{x''} \preceq \overrightarrow{x'}\right\}\right) = 0$.

So, $C(F'', F') = \frac{0}{5} = 0$.

We obtain the following matrix:

C (column, row)	F'	F''
F'	×	0.8
F''	0	×

Discussion

As we may notice in example 2, we do not necessarily have $C(F', F'') + C(F'', F') = 1$. The expression $1 - C(F', F'') - C(F'', F')$ measures the number of solutions in the sets F' and F'' which are dominated by other solutions in those sets. The result computed by the relative Zitzler metric is an image of the proportion of the tradeoff surface of the set F' which is dominated by the tradeoff surface of the set F''.

7.14.2 The absolute metrics

In [Zitzler et al. 99], three absolute metrics are presented. The first is strictly equivalent to the generational distance presented in Sect. 7.3. It has the following definition:

$$M_1(F') = \frac{1}{\text{card}(F')} \cdot \sum_{\vec{x'} \in F'} \min\left(\left\|\vec{x'} - \vec{x''}\right\|; \vec{x''} \in F''\right) \tag{7.19}$$

where F' is the practical tradeoff surface, F'' is the theoretical tradeoff surface, $\text{card}(F)$ is the number of elements of the set F, and

$$\left\|\vec{x'} - \vec{x''}\right\|$$

is the distance (Euclidean for example) between the vector $\vec{x'}$ and the vector $\vec{x''}$. This metric measures the mean distance between a practical tradeoff surface (F' or the tradeoff surface we have just computed) and a theoretical tradeoff surface (F'').

The second metric presented in [Zitzler et al. 99] allows us to compute an image index of the number of niches of a certain size that we can find on a tradeoff surface. The definition of this metric is the following:

$$M_2(F', F'') = \frac{1}{\text{card}(F') - 1} \cdot \sum_{\vec{x'} \in F'} \text{card}\left(\left\{\vec{y'} \in F' \mid \left\|\vec{x'} - \vec{y'}\right\| > \sigma\right\}\right) \tag{7.20}$$

This metric has a value included in the interval $[0, \text{card}(F')]$.

When $M_2(F', F'') = \text{card}(F')$, this means that no vector from F' has a neighbor at a distance less than σ. So, this metric looks like a spacing metric. The difference between this metric M_2 and the spacing metric described in Sect. 7.7 is that M_2 gives us a "normalized" value (we just have to divide the result of the metric M_2 by $\text{card}(F')$ to obtain a measurement included between 0 and 1).

Lastly, a third metric is presented, which measures the spread of the tradeoff surface. Its definition is the following:

$$M_3(F') = \sqrt{\sum_{i=1}^{n} \max\left(\|x'_i - y'_i\|, \vec{x'}, \vec{y'} \in F'\right)} \tag{7.21}$$

where n corresponds to the dimension of the space F'.

So, this metric measures the hypervolume which contains the tradeoff surface (the sum of the greatest distance for each component i).

7.15 Laumanns metric

A possible measurement of the size of the dominated space of objective functions is the following:

Definition 51. ***Laumanns metric***

Let an arbitrary vector of the objective function $\vec{f}$ be given.
Let $\overline{\vec{f}} = (\overline{f_1}, \overline{f_2}, \cdots, \overline{f_N})$, the vector of minimal components of $\vec{f}$, be given.
Let $\underline{\vec{f}} = (\underline{f_1}, \underline{f_2}, \cdots, \underline{f_N})$, the vector of maximal components of $\vec{f}$, be given.
We define:

$$\nu\left(\vec{f}\right) = \left\{\vec{f} \in \mathbb{R}^N \mid \vec{f} = \underline{\vec{f}} + \sum_{i=1}^{N} a_i \cdot \left((\underline{f_1}, \cdots, \overline{f_i}, \cdots, \underline{f_N}) - \underline{\vec{f}}\right) \quad a_i \in [0,1]\right\} \tag{7.22}$$

and

$$D\left(A^*, \vec{f}\right) = \bigcup_{\vec{f} = F(\vec{x}) \in A^*} \left\{\vec{f'} = F\left(\vec{x'}\right) \in \nu \mid \vec{x} \prec \vec{x'}\right\} \tag{7.23}$$

The measurement of the size of the dominated space is

$$s(A^*, F) = \frac{\lambda(D)}{\lambda(\nu)} \tag{7.24}$$

where $\lambda(A)$ is a Lebesgue measurement of the closed set A.

The Lebesgue measurement allows us to measure the size of the sets, even if these sets are composed of an infinite number of elements. Intuitively, this measurement corresponds to the hypervolume of the set (the volume in a space of m dimensions, where m corresponds to the number of objective functions of our optimization problem, for example).

This metric is presented in [Laumanns et al. 00] and is close to the third Zitzler metric. The value obtained is included between 0 and 1.

7.16 Annotated bibliography

[Van Veldhuizen 99] This thesis provides an inventory of some metrics used for the performance measurement of multiobjective optimization methods.

[Zitzler et al. 99] In this paper, a performance metric is presented which has, since that paper, often been used to perform comparisons between optimization methods.

[Cooper et al. 00] This is a book about data envelopment analysis. This field is related to performance analysis of economic systems using the DEA method. It is a fairly new field, but the method used there can be easily transposed to multiobjective optimization. This book also describes the difficulties we can face when we try to measure the performance of a system.

[Hooker 95] This paper criticizes the tests actually performed to evaluate the performance of optimization methods. It proposes a new way to evaluate the performance of optimization methods.

8

Test functions for multiobjective optimization methods

8.1 Introduction

We have seen that the field of multiobjective optimization is plentiful. The methods we can use are numerous.

Nevertheless, some methods are efficient only on problems which obey some constraints. Now, in some cases, it is very difficult, and maybe impossible, to tell if a multiobjective optimization problem satisfies a hypothesis or not.

It is to be able to classify the various methods of treatment of a multiobjective optimization problem that we have introduced test problems. Each of these problems allows us to test the method against a very specific difficulty (a discontinuous tradeoff surface, nonconvexity, etc.).

8.2 The Deb test problems

Before presenting the multiobjective-optimization test problems, we must remember that a representation of a tradeoff surface is satisfactory if the spread of points on this tradeoff surface is uniform.

In this case, if we are not satisfied by a solution, we can choose another solution in its neighborhood. Owing to the uniformity of the representation by solutions, the variation of the values of the objective functions will not be sudden when we go from one solution to a neighboring one (see Fig. 1.24).

A test function is a function which is aimed at making a multiobjective optimization problem hard to solve when one is trying to obtain a uniform spread of solutions over the tradeoff surface.

We can discern two families of difficulties:

- difficulties which prevent the method from converging toward the tradeoff surface;
- difficulties which prevent the method from obtaining a uniform spread of solutions over the tradeoff surface.

In the first family, we find a multifrontality of the tradeoff surface. In the second family, we find the following difficulties:

- nonconvexity,
- discontinuity,
- nonuniformity of the tradeoff surface.

Deb [Deb 98a] has proposed a series of test functions based on the following generic biobjective test problem:

$$\begin{array}{|ll} \text{minimize} & f_1(\vec{x}) = f(x_1, \cdots, x_m) \\ \text{minimize} & f_2(\vec{x}) = g(x_{m+1}, \cdots, x_N) \cdot h(f(x_1, \cdots, x_m), g(x_{m+1}, \cdots, x_N)) \\ \text{with} & \vec{g}(\vec{x}) \leq 0 \\ \text{and} & \vec{h}(\vec{x}) = 0 \end{array}$$

In this generic problem, each function plays a specific role:

f: controls the uniformity of the spread of solutions over the tradeoff surface.
g: controls the "multifrontality" of the tradeoff surface. It also allows one to create an isolated optimum.
h: allows one to control the shape of the tradeoff surface (convex, nonconvex, etc.).

8.2.1 The nonconvex Deb function

As we have seen with the weighted-sum-of-objective-functions method, some methods react badly when faced with a nonconvexity. Most of the time, these methods do not succeed in covering a nonconvex part of the tradeoff surface with solutions.

The following Deb nonconvex function allows us to test this aspect:

$$f(x_1) = 4 \cdot x_1$$

$$g(x_2) = \begin{cases} 4 - 3 \cdot \exp\left(-\left(\frac{x_2 - 0.2}{0.02}\right)^2\right) & \text{if } 0 \leq x_2 \leq 0.4 \\ 4 - 2 \cdot \exp\left(-\left(\frac{x_2 - 0.7}{0.2}\right)^2\right) & \text{if } 0.4 < x_2 \leq 1 \end{cases}$$

$$h(f, g) = \begin{cases} 1 - \left(\frac{f}{\beta \cdot g}\right)^{\alpha} & \text{if } f \leq \beta \cdot g \\ 0 & \text{otherwise} \end{cases}$$

$$\alpha = 0.25 + 3.75 \cdot (g(x_2) - 1) \text{ and } \beta = 1.$$

So, the problem is the following:

$$\begin{array}{|ll} \text{minimize} & f_1(\vec{x}) = f(x_1) \\ \text{minimize} & f_2(\vec{x}) = g(x_2) \cdot h(f, g) \\ \text{with} & x_1, x_2 \in [0, 1] \end{array}$$

The set of feasible values and the tradeoff surface are represented in Fig. 8.1.

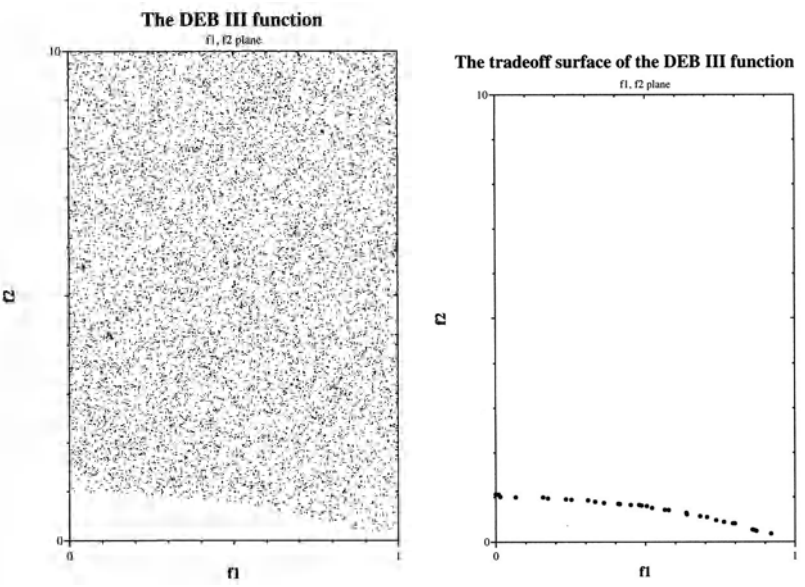
To obtain this kind of representation of the solution set for a test problem, we have applied algorithm 12.

One advantage of this kind of problem is that we can obtain an analytical expression for the tradeoff surface. The tradeoff surface is located at $(x_1, 0)$, with $x_1 \in [0, 1]$.

For this problem, the analytical expression of the tradeoff surface is:

$$\begin{array}{|l} f(x_1) = 4 \cdot x_1 \\ g(x_1) = 4 - 3 \cdot \exp(-100) \\ h(x_1) = 1 - \left(\frac{4 \cdot x_1}{4 - 3 \cdot \exp(-100)}\right)^{\alpha} \text{ if } f(x_1) \leq g(x_1) \\ \text{with } x_1 \in [0, 1] \text{ and } \alpha = 0.25 + 3.75 \cdot (3 \cdot (1 - \exp(-100))) \end{array}$$

So,



(a) The set of feasible values.

(b) The tradeoff surface.

Fig. 8.1. The shape of the Deb nonconvex function.

Algorithm 12 Random computation of the shape of a solution set.

x_1, x_2 are the variables of the problem.
$x1_{max}$, $x1_{min}$, $x2_{max}$, $x2_{min}$ are the bounds of the search space
Const $\left(\overrightarrow{f}(x_1, x_2)\right)$ is a function which is equal to 1 if the constraints of the problem are satisfied and 0 otherwise
N is the number of points to be computed for the representation of the solutions set.
$I = 0$ is an index.
rand (A, B) is a function which returns a random number included between A and B.

while$(I < N)$
  $x_1 = \text{rand}(x1_{min}, x1_{max})$
  $x_2 = \text{rand}(x2_{min}, x2_{max})$
  if Const $\left(\overrightarrow{f}(x_1, x_2)\right) = 1$ print x_1, x_2, $\overrightarrow{f}(x_1, x_2)$ and $I = I + 1$
end while

$$\begin{vmatrix} f_1(x_1) = 4 \cdot x_1 \\ f_2(x_1) = \begin{cases} (4 - 3 \cdot \exp(-100)) \cdot \left(1 - \left(\frac{4 \cdot x_1}{4 - 3 \cdot \exp(-100)}\right)^{\alpha}\right) & \text{if } f(x_1) \leq g(x_1) \\ 0 & \text{otherwise} \end{cases} \\ \text{with } x_1 \in [0, 1] \text{ and } \alpha = 0.25 + 3.75 \cdot (3 \cdot (1 - \exp(-100))) \end{vmatrix}$$

If we make the approximation $\exp(-100) \simeq 0$, then

$$\begin{vmatrix} f_1(x_1) = 4 \cdot x_1 \\ f_2(x_1) = \begin{cases} 4 \cdot \left(1 - x_1^{11.5}\right) & \text{if } f(x_1) \leq g(x_1) \\ 0 & \text{otherwise} \end{cases} \\ \text{with } x_1 \in [0,1] \end{vmatrix}$$

8.2.2 The discontinuous Deb function

The great majority of multiobjective optimization methods do not succeed in guaranteeing an uniform spread of solutions over the tradeoff surface when they face a discontinuity.

The discontinuous Deb function is defined as follows:

$$\begin{aligned} f(x_1) &= x_1 \\ g(x_2) &= 1 + 10 \cdot x_2 \\ h(f,g) &= 1 - \left(\frac{f}{g}\right)^{\alpha} - \left(\frac{f}{g}\right) \cdot \sin(2\pi \cdot q \cdot f) \end{aligned}$$

The parameter q allows us to define the number of discontinuous areas in the interval $[0,1]$. The parameter α if often chosen to be equal to 2.

The problem is defined as follows:

$$\begin{vmatrix} \text{minimize } f_1(\vec{x}) = f(x_1) \\ \text{minimize } f_2(\vec{x}) = g(x_2) \cdot h(f,g) \\ \text{with} \qquad x_1, x_2 \in [0,1] \end{vmatrix}$$

The set of feasible values and the tradeoff surface are represented in Fig. 8.2.

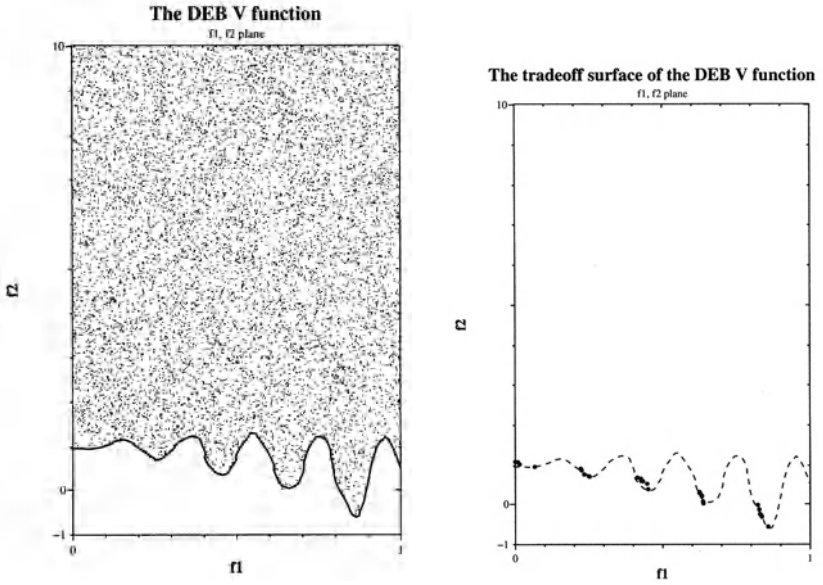
The analytical expression for the tradeoff surface is:

$$\begin{vmatrix} f_1(x_1) = x_1 \\ f_2(x_1) = 1 - x_1^2 - x_1 \cdot \sin(2\pi \cdot q \cdot x_1) \\ \text{with } x_1 \in [0,1] \end{vmatrix}$$

So, we have

$$f_2 = 1 - f_1^2 - f_1 \cdot \sin(2\pi \cdot q \cdot f_1) \tag{8.1}$$

We must not forget to remove the points of the envelop which do not belong to the tradeoff surface.



(a) The set of feasible values.

(b) The tradeoff surface.

Fig. 8.2. The shape of the discontinuous Deb function.

8.2.3 The multifrontal Deb function

Another major difficulty that we can encounter is multifrontality. This difficulty arises when a multiobjective optimization problem has one or more "ghost" tradeoff surfaces.

A ghost tradeoff surface is often due to the existence of an isolated global optimum and a local optimum. During the optimization process, the solutions converge to the local optimum and so form a tradeoff surface which is different from the "optimal" tradeoff surface.

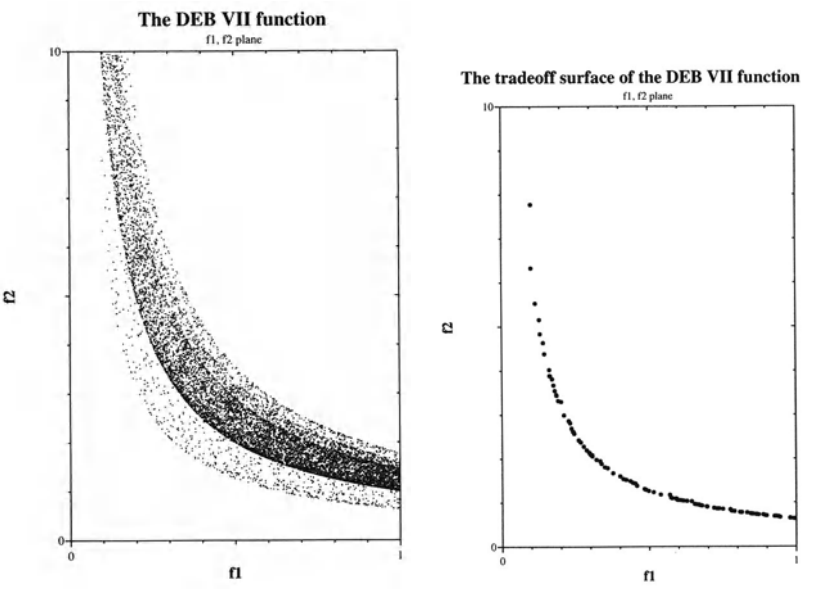
The multifrontal Deb function is defined as follows:

$$
\begin{aligned}
&f(x_1) = x_1 \\
&g(x_2) = 2 - \exp\left(-\left(\tfrac{x_2 - 0.2}{0.04}\right)^2\right) - 0.8 \cdot \exp\left(-\left(\tfrac{x_2 - 0.6}{0.4}\right)^2\right) \\
&h(f,g) = \tfrac{1}{f}
\end{aligned}
$$

The problem is defined as follows:

$$
\begin{cases}
\text{minimize } f_1(\vec{x}) = f(x_1) \\
\text{minimize } f_2(\vec{x}) = g(x_2) \cdot h(f,g) \\
\text{with} \qquad x_1, x_2 \in [0,1]
\end{cases}
$$

The set of feasible values and the tradeoff surface are represented in Fig. 8.3. The analytical expression for the tradeoff surface is the following:



(a) The set of feasible values.

(b) The tradeoff surface.

Fig. 8.3. The shape of the multifrontal Deb function.

$$
\begin{vmatrix}
f_1(x_1) = x_1 \\
f_2(x_1) = \left(2 - \exp(-25) - 0.8 \cdot \exp\left(\frac{3}{2}\right)\right) \cdot \frac{1}{x_1} \\
\text{with } x_1 \in [0,1]
\end{vmatrix}
$$

So, we have

$$f_2 = \frac{1.9156806}{f_1} \tag{8.2}$$

8.2.4 The nonuniform Deb function

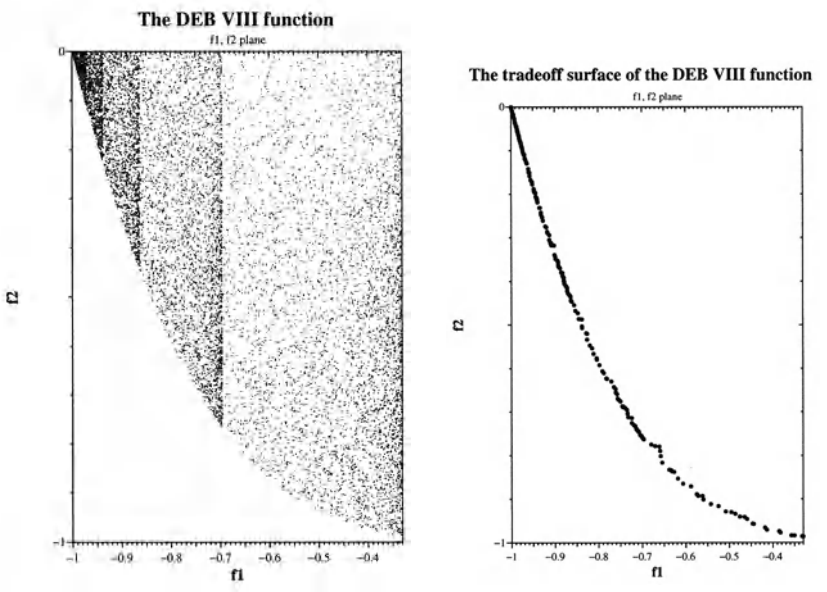
In some cases, owing to the structure of the objective function to be optimized, the solutions will be concentrated around some points. In this case, we have a problem of nonuniformity of the representation by solutions on the tradeoff surface.

The non uniform Deb function is defined as follows:

$$
\begin{aligned}
&f(x_1) = 1 - \exp(-4 \cdot x_1) \cdot \sin^4(5\pi \cdot x_1) \\
&g(x_2) = 1 - 10 \cdot x_2 \\
&h(f,g) = \sqrt{1 - f}
\end{aligned}
$$

$$
\begin{vmatrix}
\text{minimize } f_1(\vec{x}) = -f(x_1) \\
\text{minimize } f_2(\vec{x}) = -g(x_2) \cdot h(f,g) \\
\text{with} \qquad x_1, x_2 \in [0,1]
\end{vmatrix}
$$

The set of feasible values and the tradeoff surface are represented in Fig. 8.4.



(a) The set of feasible values.

(b) The tradeoff surface.

Fig. 8.4. The shape of the nonuniform Deb function.

The analytical expression for the tradeoff surface is the following:

$$f_2 = -\sqrt{1 - f_1} \tag{8.3}$$

with $f_1 \in [-1, 0]$.

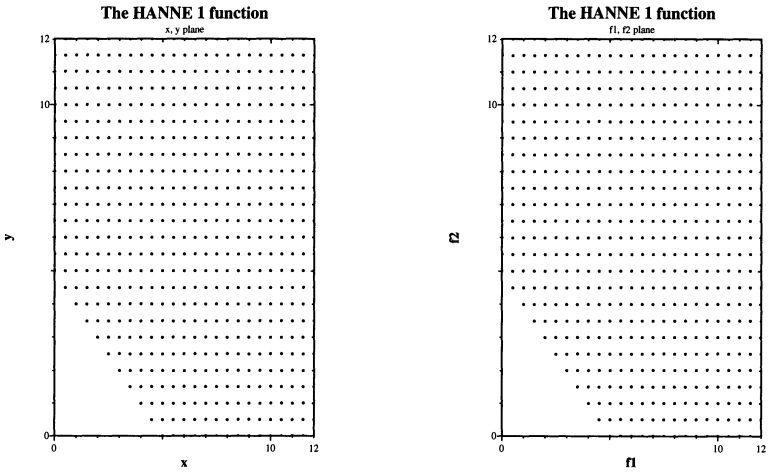
8.3 The Hanne test problems

The main difference between these and the Deb test problems lies in the introduction of a constraint. Otherwise, the Hanne test problems cover nearly the same field as the Deb test problems.

8.3.1 The linear Hanne function

This test problem is the following:

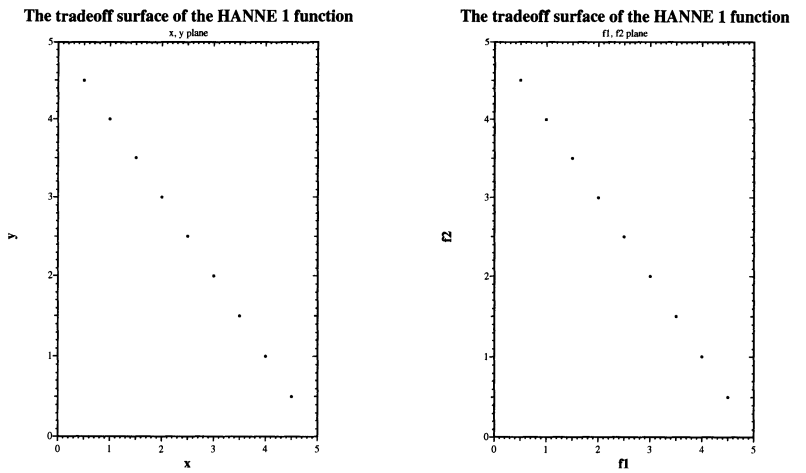
$$\begin{array}{|ll} \text{minimize} & f_1(\vec{x}) = x_1 \\ \text{minimize} & f_2(\vec{x}) = x_2 \\ \text{with} & x_1 \geq 0 \\ & x_2 \geq 0 \\ \text{and} & x_1 + x_2 \geq 5 \end{array}$$



(a) The set of x values.

(b) The solution set.

Fig. 8.5. The set of the linear Hanne function (solution set).



(a) The set of x values associated with the tradeoff surface.

(b) The tradeoff surface.

Fig. 8.6. The shape of the linear Hanne function (tradeoff surface).

The set of feasible values and the tradeoff surface are represented in Figs. 8.5 and 8.6.

In Fig. 8.5a, we have represented the solution set in the space of the parameter $\vec{x}$. In Fig. 8.5b, we have represented the solution set in the space of the objective functions. We have obtained these two sets by applying algorithm 13.

In Fig. 8.6a, we have represented the tradeoff surface of the test problem in the space of the parameters. In Fig. 8.6b we have represented the tradeoff surface in the space of the objective functions. This kind of representation (parameter space and objective function space) allows us to visualize the possible difficulties in the parameter space (the solution set can be divided into pieces in the parameter space, for example).

Algorithm 13 Sequential computation of the shape of the solution set.

x_1, x_2 parameters of the test problem
$x_{1_{max}}$, $x_{1_{min}}$, $x_{2_{max}}$, $x_{2_{min}}$ boundaries of the parameters
Δx_1, Δx_2 variation step of the parameters
Const $\left(\vec{f}\,(x_1, x_2)\right)$ function which is equal to 1 if the constraints of the considered problem are satisfied and 0 otherwise.

$x_1 = x_{1_{min}}$
$x_2 = x_{2_{min}}$
do
  do
    if Const $\left(\vec{f}\,(x_1, x_2)\right) = 1$ print x_1, x_2, $\vec{f}\,(x_1, x_2)$
    $x_1 = x_1 + \Delta x_1$
  while $(x_1 < x_{1_{max}})$
  $x_2 = x_2 + \Delta x_2$
while $(x_2 < x_{2_{max}})$

8.3.2 The convex Hanne function

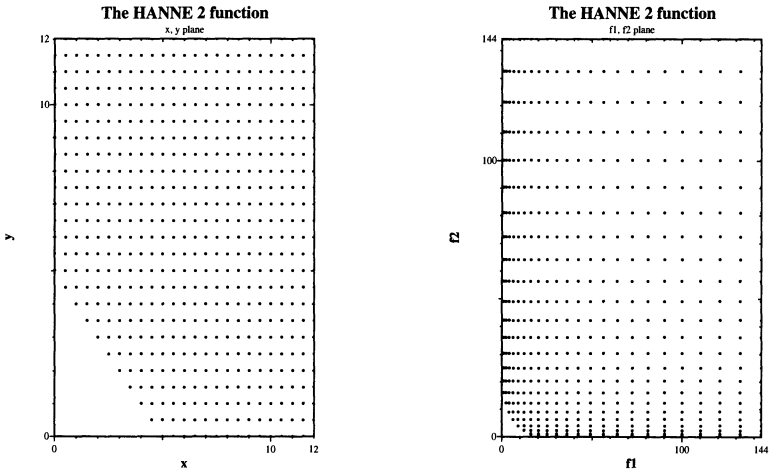
This test problem is the following:

$$\begin{array}{ll} \text{minimize } f_1(\vec{x}) = x_1^2 & \\ \text{minimize } f_2(\vec{x}) = x_2^2 & \\ \text{with} & x_1 \geq 0 \\ & x_2 \geq 0 \\ \text{and} & x_1 + x_2 \geq 5 \end{array}$$

The set of feasible values and the tradeoff surface are represented in Figs. 8.7 and 8.8.

In Figs. 8.7a and 8.7b, we have represented the shape of the solution set in the parameter space and in the objective function space, respectively. Here, unlike the preceding problem (the linear Hanne problem), the two shapes are not similar because the solution set in the objective function space is not a simple copy of the solution set in the parameter space anymore.

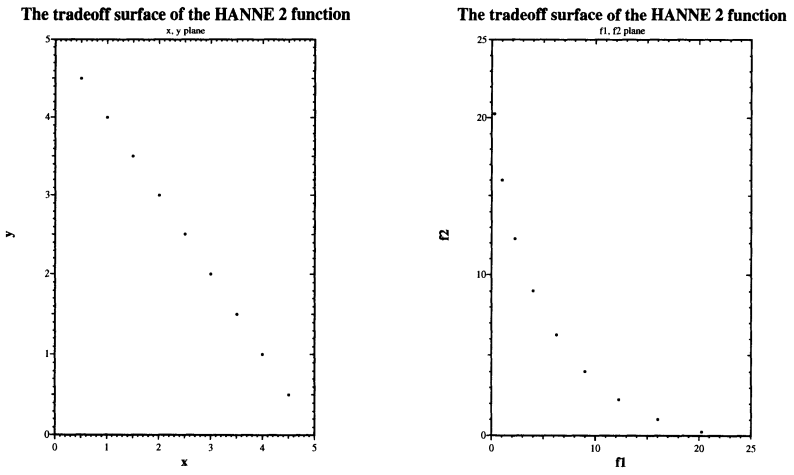
One advantage of the sequential computation of the shape of the solution set is, as we can see when comparing the parameter space and the objective function space,



(a) The set of feasible values for x.

(b) The solution set.

Fig. 8.7. The shape of the convex Hanne function (solution set).



(a) The set of x values associated with the tradeoff surface.

(b) The tradeoff surface.

Fig. 8.8. The shape of the convex Hanne function (tradeoff surface).

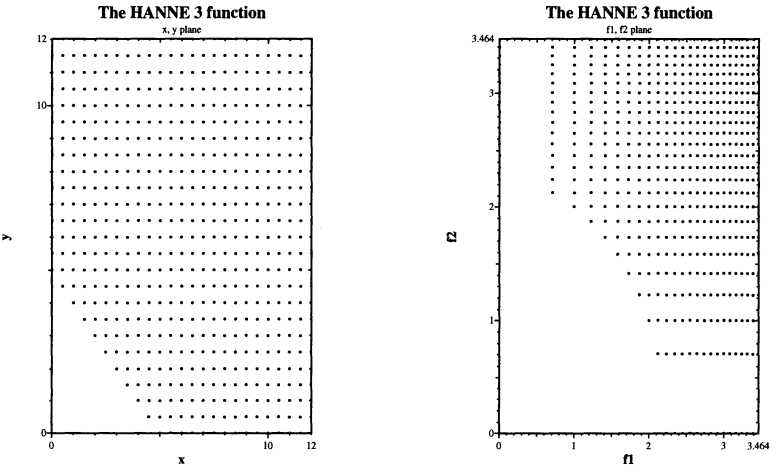
that we can visualize the effect of the objective functions on the distortion of the solution set.

8.3.3 The nonconvex Hanne function

This test problem is the following:

$$
\begin{array}{|ll}
\text{minimize } f_1(\vec{x}) = \sqrt{x_1} \\
\text{minimize } f_2(\vec{x}) = \sqrt{x_2} \\
\text{with} \quad x_1 \geq 0 \\
\qquad\quad x_2 \geq 0 \\
\text{and} \quad x_1 + x_2 \geq 5
\end{array}
$$

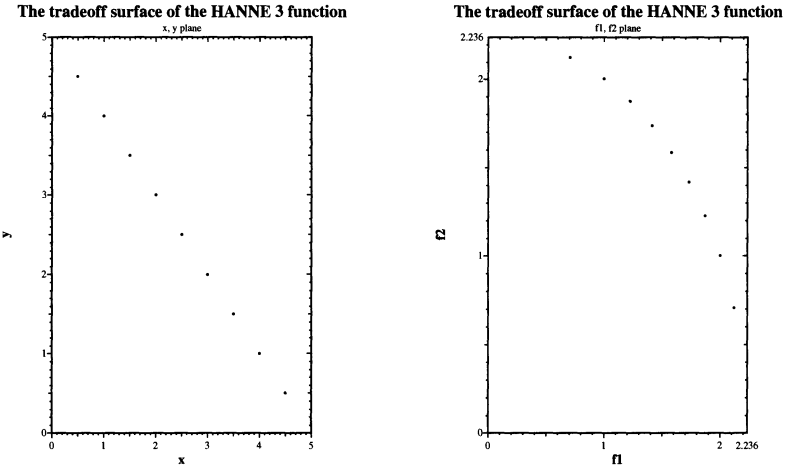
The set of feasible values and the tradeoff surface are represented in Figs. 8.9 and 8.10.



(a) The set of feasible values of x.

(b) The solution set.

Fig. 8.9. The shape of the nonconvex Hanne function (solution set).



(a) The set of x values associated with the tradeoff surface.

(b) The tradeoff surface.

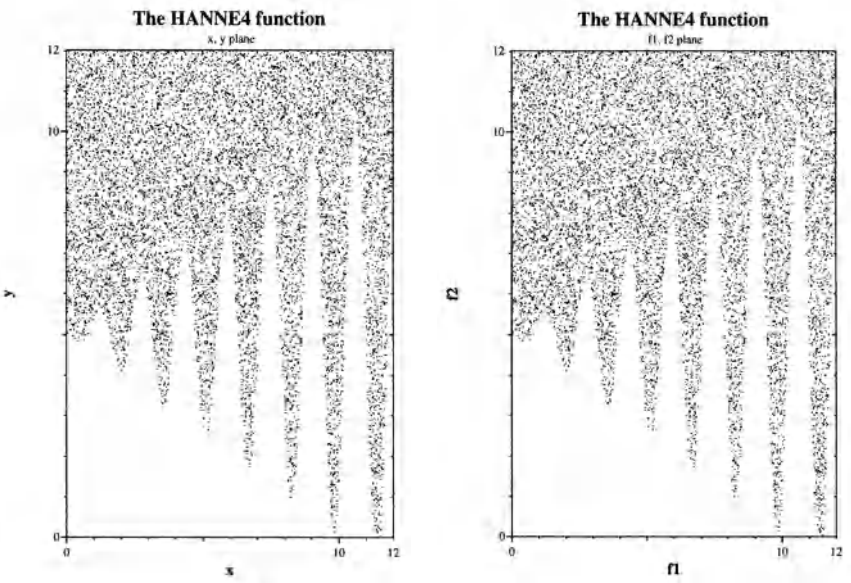
Fig. 8.10. The shape of the nonconvex Hanne function (tradeoff surface).

8.3.4 The discontinuous Hanne function

This test problem is the following:

$$
\begin{array}{|ll}
\text{minimize } f_1(\overrightarrow{x}) = x_1 \\
\text{minimize } f_2(\overrightarrow{x}) = x_2 \\
\text{with} \quad x_1 \geq 0 \\
\qquad\quad x_2 \geq 0 \\
\text{and} \quad x_2 - 5 + 0.5 \cdot x_1 \cdot \sin(4 \cdot x_1) \geq 0
\end{array}
$$

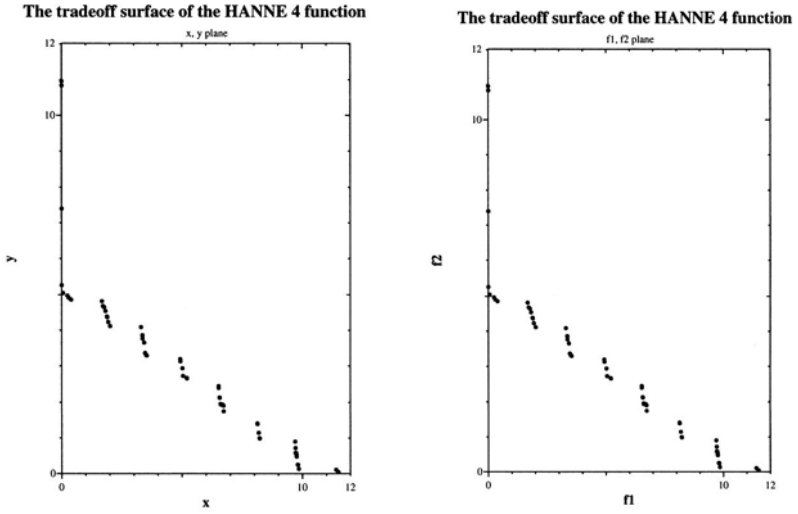
The set of feasible values and the tradeoff surface are represented in Figs. 8.11 and 8.12.



(a) The set of feasible values of x.

(b) The solution set.

Fig. 8.11. The shape of the discontinuous Hanne function (solution set).



(a) The set of x values associated with the tradeoff surface.

(b) The tradeoff surface.

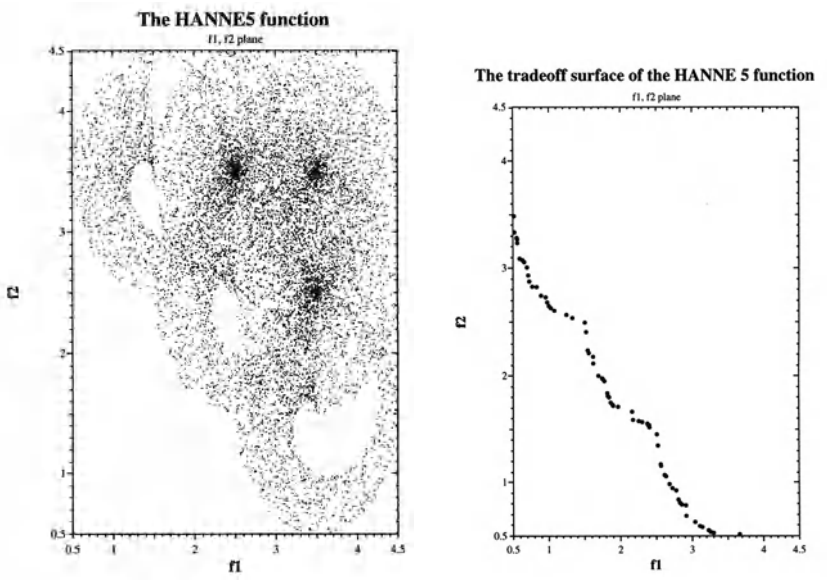
Fig. 8.12. The shape of the discontinuous Hanne function (tradeoff surface).

8.3.5 The Hanne function with several efficiency areas

This test problem is the following:

$$\begin{vmatrix} \text{minimize } f_1(\vec{x}) = \text{int}(x_1) + 0.5 + (x_1 - \text{int}(x_1)) \cdot \sin(2\pi \cdot (x_2 - \text{int}(x_2))) \\ \text{minimize } f_2(\vec{x}) = \text{int}(x_2) + 0.5 + (x_1 - \text{int}(x_1)) \cdot \cos(2\pi \cdot (x_2 - \text{int}(x_2))) \\ \text{with} \quad x_1 \geq 0 \\ \qquad\quad x_2 \geq 0 \\ \text{and} \quad x_1 + x_2 \geq 5 \end{vmatrix}$$

where $\text{int}(x)$ corresponds to the integer part of x.



(a) The set of feasible values.

(b) The tradeoff surface.

Fig. 8.13. The shape of the Hanne function with several efficiency areas.

The set of feasible values and the tradeoff surface are represented in Fig. 8.13.

8.4 Annotated bibliography

[Deb 98a] This paper presents a method for constructing test problems. The generic expression allows one to construct all kinds of test problems: convex, non-convex, with local optima, etc. These test problems are often used for comparisons between methods.

[Whitley et al. 96] This paper deals with mono-objective test problems. The authors underline some incoherencies in some of the actual test problems used. They then propose some rules that all families of test problems should respect.

9

An attempt to classify multiobjective optimization methods

9.1 Introduction

As we have seen through this book, there exists a huge number of multiobjective optimization methods. This situation is a problem when one has to choose an optimization method to solve a concrete problem. At present, few scientific papers deal with the classification of optimization methods.

Nevertheless, we can find in [Miettinen 99] an organization chart that allows us to choose a multiobjective optimization method with respect to the problem we would like to solve. This organization chart is very complicated and, as we shall see, the approach proposed there is maybe not the best one for this kind of situation. Alternatively, we can find in [Talbi 98] a technique which allows us to classify hybrid metaheuristics.

In this chapter, we shall present two approaches to such a classification:

- a mathematical approach, which tries to describe an optimization method as briefly as possible;
- a graphical approach, where we try to find a hierarchical decomposition of optimization methods.

Throughout this chapter, we make a distinction between the following expressions:

- multiobjective treatment method;
- optimization method;
- multiobjective optimization method.

We mean by "multiobjective treatment method" a method used to transform the original multiobjective problem into a multiobjective problem adapted to a solution method. This method could be, for example, the weighted-sum-of-objective-functions method.

Next, we mean by "optimization method" a search method used to solve a multiobjective problem. The search method does not include the transformation method used for the multiobjective problem.

Lastly, we mean by "multiobjective optimization method" the combination of a multiobjective treatment method and an optimization method.

We shall now study the mathematical approach to the classification, and then we shall study the graphical approach. We shall show how we can obtain a hierarchical

decomposition of the structure of optimization methods, but, above all, we shall discover that a classification of multiobjective optimization methods has to be seen as a tool to aid the user to ask him/herself questions so as to choose a method, rather than simply as a tool to select a method.

We shall illustrate each kind of classification of multiobjective optimization methods (mathematical and graphical) with some examples.

9.2 "Mathematical" classification of optimization methods

9.2.1 Introduction

This approach takes into account only the way in which a multiobjective optimization problem is transformed into another multiobjective optimization problem (see [Ehrgott 00]). It does not try to describe the optimization method itself.

9.2.2 The Ehrgott classification formula

Let us consider the following multiobjective optimization problem:

$$\min_{\vec{x} \in X} \vec{f}(\vec{x})$$

Here, the word *"min"* should really be written in quotes because, for multiobjective optimization problems, until an order relation has been specified, the meaning of the word is not sufficiently precise for the optimization to be performed.

If we now specify that the order relation to be used in this problem is that of domination in the Pareto sense, then the meaning of *"min"* has been made precise: we are looking for the minimum, in the Pareto sense, on the set of decision variables X, of the objective function vector $\vec{f}(\vec{x})$.

The elements needed for a good specification of a multiobjective optimization problem are the following:

- the space of the decision variables (X in our example);
- the space of the objective functions ($\mathbb{R}^N$ in our example);
- a sorted space ($\mathbb{R}^P$ with the domination relation as a relation order, which will be denoted by $\preceq$);
- a transformation model (we shall denote this model by θ in our example). We shall make the meaning of this expression precise below.

Ehrgott [Ehrgott 00] has proposed that an optimization method can be represented using the following expression:

$$\left(X, \vec{f}, \mathbb{R}^N\right) / \theta / \left(\mathbb{R}^P, \preceq\right)$$

This expression illustrates well the process of transformation of a multiobjective optimization problem into another multiobjective optimization problem which is adapted to solution via a search method. This transformation is represented in Fig. 9.1.

As we can see, we are making a distinction here between the transformation method applied to a multiobjective optimization problem (the weighted-sum-of-objective-functions method is a transformation method) and the search method (the simulated annealing method is a search method).

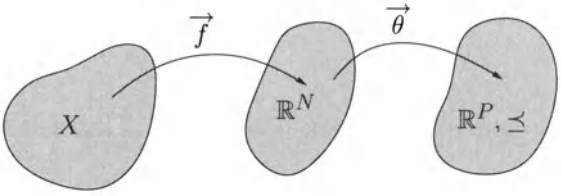


Fig. 9.1. The transformation of a multiobjective optimization problem.

To better understand the classification expression given above, let us present some examples.

Example 1: To reduce the problem to a minimization problem

Not all optimization problems are described as minimization problems. For example, in the following problem, we have two maximizations and a minimization, which lead to the following transformation:

$$\left|\begin{array}{l}\text{minimize } f_1(\overrightarrow{x}) \\ \text{maximize } f_2(\overrightarrow{x}) \\ \text{maximize } f_3(\overrightarrow{x}) \\ \text{with } \quad \overrightarrow{x} \in X \subset \mathbb{R}^4\end{array}\right. \rightarrow \left|\begin{array}{l}\text{minimize } f_1(\overrightarrow{x}) \\ \text{minimize } -f_2(\overrightarrow{x}) \\ \text{minimize } -f_3(\overrightarrow{x}) \\ \text{with } \quad \overrightarrow{x} \in X \subset \mathbb{R}^4\end{array}\right.$$

For this transformation of a multiobjective optimization problem, the transformation model is the following:

$$\theta(z_1, z_2, z_3) = (z_1, -z_2, -z_3)$$

The space of decision variables is X. The space of objective functions is $\mathbb{R}^3$. The space of objective functions after transformation is $\mathbb{R}^3$. The order that we have chosen on this space is the domination relation on the space $\mathbb{R}^3$, which we denote by $\preceq$. Consequently, the classification expression for the method is the following:

$$\left(X, (f_1, f_2, f_3), \mathbb{R}^3\right) / (z_1, -z_2, -z_3) / \left(\mathbb{R}^3, \preceq\right)$$

Example 2: The Tchebychev method

The transformation performed on the multiobjective optimization problem is the following:

$$\left|\begin{array}{l}\text{minimize } f_1(\overrightarrow{x}) \\ \text{maximize } f_2(\overrightarrow{x}) \\ \text{maximize } f_3(\overrightarrow{x}) \\ \text{with } \quad \overrightarrow{x} \in X\end{array}\right. \rightarrow \left|\begin{array}{l}\text{minimize } \max\left(f_1(\overrightarrow{x}), -f_2(\overrightarrow{x}), -f_3(\overrightarrow{x})\right) \\ \text{with } \quad \overrightarrow{x} \in X\end{array}\right.$$

For this transformation of a multiobjective optimization problem, the transformation model is the following:

$$\theta(z_1, z_2, z_3) = \max(z_1, -z_2, -z_3)$$

The space of decision variables is X. The space of objective functions is $\mathbb{R}^3$. The space of objective functions after transformation is $\mathbb{R}$. The order that we haven chosen on this space is the classical order relation $\leq$. Consequently, the classification expression for the method is the following:

$$\left(X, (f_1, f_2, f_3), \mathbb{R}^3\right) / \max / (\mathbb{R}, \leq)$$

Example 3: The weighted-sum-of-objective-functions method

The transformation performed on the multiobjective optimization problem is the following:

$$\left|\begin{array}{l} \text{minimize } f_1(\vec{x}) \\ \text{maximize } f_2(\vec{x}) \\ \text{maximize } f_3(\vec{x}) \\ \text{with } \quad \vec{x} \in X \end{array}\right| \rightarrow \left|\begin{array}{ll} \text{minimize} & \omega_1 \cdot f_1(\vec{x}) - \omega_2 \cdot f_2(\vec{x}) \\ & -\omega_3 \cdot f_3(\vec{x}) \\ \text{with} & \vec{x} \in X \\ \text{and} & \omega_i \geq 0 \end{array}\right|$$

For this transformation of a multiobjective optimization problem, the transformation model is the following:

$$\theta(z_1, z_2, z_3) = \omega_1 \cdot z_1 - \omega_2 \cdot z_2 - \omega_3 \cdot z_3$$

The space of decision variables is X. The space of objective functions is $\mathbb{R}^3$. The space of objective functions after transformation is $\mathbb{R}$. The order that we haven chosen on this space is the classical order relation $\leq$. Consequently, the classification expression for the method is the following:

$$\left(X, (f_1, f_2, f_3), \mathbb{R}^3\right) / \omega_1 \cdot z_1 - \omega_2 \cdot z_2 - \omega_3 \cdot z_3 / (\mathbb{R}, \leq)$$

9.2.3 Conclusion

This classification has the advantage that it is usable both on optimization problems and on optimization methods (see [Ehrgott et al. 00]).

This common property of the description allows the user to compare the description of the problem with the description of the solution method and so to select a solution method which has points in common with the description of the problem; such a method will be better adapted to the solution of the problem.

9.3 Hierarchical classification of multiobjective optimization methods

9.3.1 Introduction

This approach can present some drawbacks such as an explosion of the number of hierarchical levels due to the large number of distinct elements which describe an optimization method. Nevertheless, it is possible to limit this drawback by choosing carefully the elements which describe an optimization method and by limiting the extent to which the classification process is carried out.

9.3.2 A hierarchical graph

9.3.2.1 A hierarchy for the treatment of a multiobjective problem

We have already encountered some of the elements of the description of multiobjective optimization methods during the presentation of the various methods.

For example, we can distinguish three different approaches with respect to the knowledge we have about the problem to be solved. We can solve this problem in three ways:

- *a priori* if we have sufficient knowledge about the problem and are able to give, with high precision, our preferences related to the objective functions;
- *a posteriori,* if we do not have much knowledge about the problem: we try to approach the tradeoff surface by covering it with a finite number of solutions;
- *progressively,* by having a dialogue with the optimization method so that we can make our preferences precise.

This first step in the description of optimization methods gives us three distinct branches (see Fig. 9.3).

The branch of progressive methods is fully disconnected from the other two (because of the interaction phase which differentiates the progressive methods from the a priori and a posteriori methods).

The branches of a priori and a posteriori methods are not independent, because an a priori method can be used as an a posteriori method (we perform another run of the a priori optimization method with another set of preferences given by the decision maker). On the other hand, the converse is not necessarily true: it is not interesting to use an a posteriori method as an a priori method (constructing an approximation to the tradeoff surface and saving only the solution which corresponds to the compromise closest to the compromise formulated by the decision maker).

Another level that we can add to this hierarchy comes from the way we deal with objective functions:

- We can try to transform them into a mono-objective function: this is the "aggregative" approach. This approach can then be divided into three further levels:
  - aggregation using constraints: the ε constraint method is an example of such an aggregation;
  - aggregation without constraints: the weighted-sum-of-objective-functions is an example of such an aggregation;
  - hybrid aggregation: the Corley method is an example of such an aggregation.
- We can treat a multiobjective optimization problem in a vectorial way; this means without performing any aggregation of the objective functions. The MOGA method is an example of such a treatment.

The last level that we can consider, which is specific to multiobjective optimization, deals with the "overlapping" of the multiobjective treatment with the optimization method:

- We may need to use a multiobjective treatment method totally decoupled from the optimization method. In that case, we call the former method a "decoupled multiobjective treatment". For example, the weighted-sum-of-objective-functions method is a decoupled multiobjective treatment optimization method, because it does not need a dedicated optimization method to search for solutions.

- We may also need to use multiobjective treatment methods which do not allow any choice in the optimization method to be used. For example, the MOGA method is a multiobjective optimization method in which the optimization method is heavily coupled to the multiobjective treatment method: we cannot use this method with anything else than a genetic algorithm.

So, we obtain the hierarchies shown in Figs. 9.2 and 9.3.

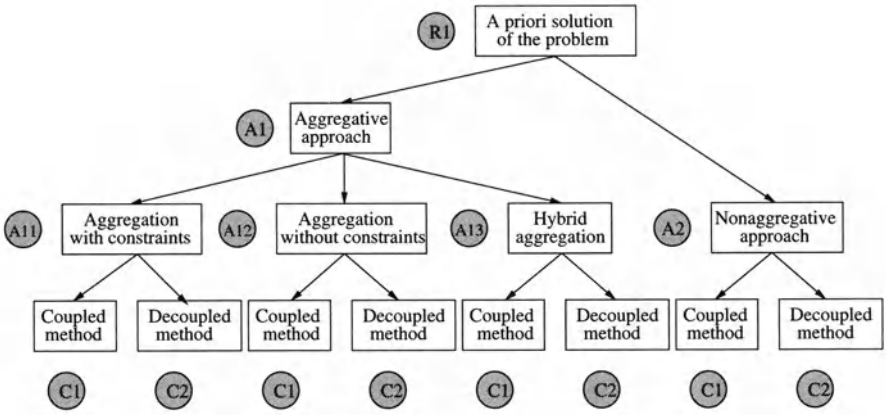


Fig. 9.2. The hierarchy of the a priori methods.

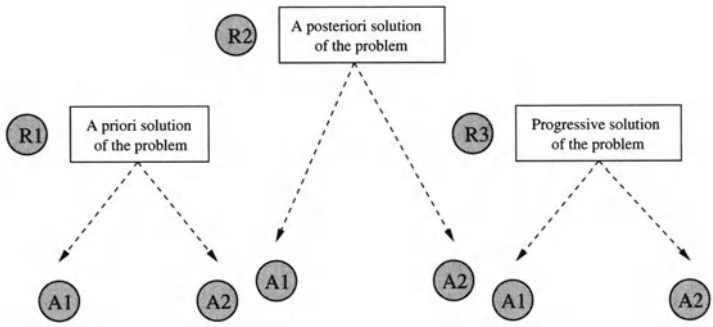


Fig. 9.3. The higher hierarchical level of the transformation methods of a multiobjective problem.

The first feature that we can establish is the presence of a nonaggregative branch in the trees with roots R1 and R3. This is not surprising for the root R3 (progressive solution of a multiobjective optimization problem), because some methods (not presented in this book) require the user to select those individuals which he/she prefers. However, for the root R1 (a priori solution of a multiobjective optimization problem), the nonaggregative branch illustrates the existence of methods such as the ELECTRE methods (decision aid methods), in which we construct an order relation on the basis of the preferences of the decision maker. The order relation constructed in this way

can have an aggregative or a nonaggregative form. In the latter case, the method will order individuals so as to reflect the preferences of the decision maker. So, these methods do not produce one and only one solution but a complete family of solutions.

Let us present some examples.

Example 1: The weighted-sum-of-objective-functions method

This method is mostly dedicated to the a priori resolution of a multiobjective problem. It transforms the multiobjective problem so that we can deal with it as with a mono-objective function; so it is an aggregative approach without constraints. Moreover, this method is highly uncoupled from the optimization method, in the sense that we can use any optimization method to find a solution to the problem.

The path in the hierarchy is:

$$R1/A11/C2$$

The weighted sum-of-objective-functions method can also be used in the a posteriori or progressive solution of a problem. In that case, the method is used inside a system and not in its basic form.

Example 2: The MOGA method

This is a method to solve a problem a posteriori. It constructs a family of solutions for a given multiobjective problem.

The multiobjective treatment method is nonaggregative. We do not ask the decision maker to express his/her preferences. This method uses the domination relation directly to solve the problem. The multiobjective treatment method is highly coupled to the optimization method, as we have emphasized above.

The path in the hierarchy is:

$$R2/A2/C1$$

9.3.2.2 A hierarchy for interactions

In this hierarchy, some elements which could allow us to describe the progressive methods are missing. For these methods, the interaction mode is a discriminating element.

When we analyze the working modes of interactive methods, we can divide them into four categories:

- interactions via preferences; we denote this category by $I1$;
- interactions via constraints, we denote this category by $I2$;
- hybrid interactions via preferences and constraints; we denote this category by $I3$;
- interactions of another kind (for example, the selection of a subpopulation of interesting solutions); we denote this category by $I4$.

We must therefore add four extra branches to our hierarchy (see Fig. 9.4).

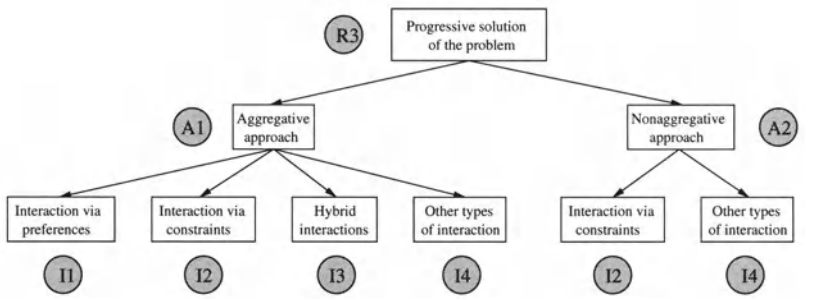


Fig. 9.4. The hierarchy of modes of interaction.

9.3.2.3 A hierarchy for optimization methods

The hierarchy that we have presented in relation to the multiobjective treatment of the multiobjective problem is not sufficiently discriminating to allow a good classification of optimization methods. Indeed, there exist a lot of methods in the category of coupled methods. It is useful to go deeper in the hierarchy for this category.

When we look at the methods used in this domain, we notice that they belong to three families:

- Neighborhood improvement methods: these methods use a starting point and improve it via an "analysis" of all or part of the neighborhood of the current point. In this family, we find the simulated annealing and the local search methods, for example.
- Evolutionary methods: these methods transform individuals of a population so that they come closer to an optimal area or an optimal point.
- Other methods: these methods are used less in the domain of multiobjective optimization. We find, for example:
  - Methods of solution via construction of solutions: these methods assign a definitive value to one or several variables at a given iteration. They never change a decision. We can cite as an example of these methods the "greedy method".
  - Methods of solution via decomposition: these methods can, for example, decompose the solution set of a problem into a tree and walk through it to look for the best solution. An example of such a method is the branch and bound method.

However, it is not interesting to follow this decomposition further, because it makes the hierarchy cumbersome and, as we have emphasized above, this thought process is not fruitful in relation to the methods used in this field.

The last hierarchical level that we shall add deals with evolutionary methods. With these methods, we can distinguish two main families:

- Elitist evolutionary methods: these methods have an archive, in which the best individuals encountered during a run of the optimization are saved. The individuals in this archive can be transmitted intact to the next iteration or they can be used by the optimization method to improve the method. The PASA method is an example of such a method.
- Nonelitist evolutionary methods: these methods do not have any archive, and are generally simpler to put into effect. The MOGA method is an example of such a method.

The hierarchy which is specifically concerned with optimization methods is represented in Fig. 9.5.

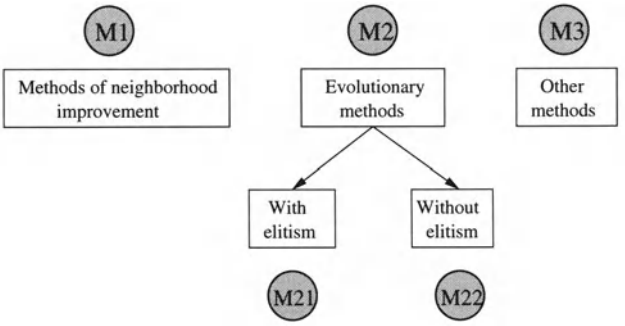


Fig. 9.5. An extra hierarchy dedicated to optimization methods.

9.3.2.4 How to use this hierarchy

Now, we face a problem. Without some minimal knowledge about the problem to be solved and the methods available, it is very hard to choose a multiobjective optimization method.

When we are choosing a multiobjective optimization method, several factors have an influence:

- Requirements related to the quality of the solution: some problems have an optimum that is very hard to locate. These problems are probably best solved using a metaheuristic such as simulated annealing. For these problems, we may have to spend several hours or even several days to find an approximate solution. This is the case for strategic problems, such as the problem of optimization of a structure, where a structure with less material, but which still meets the specifications, will allow one to save an important part of the total cost of a building. For other problems with less of a strategic element, we can accept a solution which is an improvement on the initial solution. So we can choose a method such as a local search, or a simulated annealing with a rapid decrease of the temperature.
- The specific structure of the search space: the presence of local optima will influence the choice of the optimization method. We must admit that, for some really difficult problems, it is impossible to know in advance the number of local optima and so to judge the a priori difficulty of the problem. Also, the convex or nonconvex nature of the shape of the tradeoff surface will influence the choice of the multiobjective treatment method. Here, too, it is very difficult to make a judgment a priori about the shape of the tradeoff surface, and so to choose a multiobjective treatment method suited to the problem. For a convex problem, we should choose a weighted-sum-of-objective-functions method, whereas, for a nonconvex problem, we should choose a Tchebychev method.
- The ability to express precisely our preferences: if the preferences are well defined, we choose a solution method of the a priori kind. We must recall that trying to approximate the tradeoff surface is a task which may be long and difficult. If the preferences are not well defined, we have two possible solutions:

- We help the decision maker to make his/her preferences precise through a progressive solution of the problem. This method of solution is not the favorite method of the users, because the decision maker must be present during the whole search. Moreover, the time between two interaction steps can be very long.
- We approximate the tradeoff surface by covering it with a finite number of solutions well spaced over it: This is the a posteriori approach. This approach to the solution of a multiobjective problem is especially convenient when the decision process is long and/or not well identified. It gives the decision maker the opportunity to judge the proposed solutions using successive comparisons, or to extract a solution using a decision aid method.

It is clear that if the decision maker has all these answers available, the selection of a method will be easier. So, we must see this hierarchy more as a tool to help ask questions about the selection of a method than as a tool to help the actual selection.

9.3.3 Classification of some methods

We shall now classify the methods presented in this book using the hierarchy that we have set out above.

The scalar methods

Method	Path in the hierarchy
Weighted sum of objective functions	$R1/A12/C2$
Keeney–Raiffa	$R1/A12/C2$
Distance to an objective	$R1/A12/C2$
Compromise	$R1/A11/C2$
Hybrid	$R1/A13/C2$
Goal attainment	$R1/A11/C2$
Goal programming	$R1/A11/C2$
Lexicographic ordering	$R1/A11/C2$
Proper equality constraints	$R2/A11/C2$
Proper inequality constraints	$R2/A11/C2$
Lin–Tabak	$R2/A11/C2$
Lin–Giesy	$R2/A11/C2$

The interactive methods

Method	Path in the hierarchy
Surrogate worth tradeoff	$R3/I2/C2$
STEP	$R3/I2/C2$
Gandel	$R3/I2/C2$
Jahn	$R3/I4/C2$
Geoffrion	$R3/I4/C2$
Simplex	$R3/I1/C2$

For the Jahn and Geoffrion methods, we notice that the interactions are performed via the choice of a step length. This interaction mode does not allow the decision maker to model his/her preference in a simple way.

The fuzzy methods

Method	Path in the hierarchy
Sakawa	$R1/A12/C2$
Reardon	$R1/A12/C2$

The decision aid methods

Method	Path in the hierarchy
ELECTRE I	$R1/A12/C2$
ELECTRE IS	$R1/A12/C2$
ELECTRE II	$R1/A12/C2$
ELECTRE III	$R1/A12/C2$
ELECTRE IV	$R1/A12/C2$
ELECTRE TRI	$R1/A12/C2$
PROMETHEE 1	$R1/A12/C2$
PROMETHEE 2	$R1/A12/C2$

The methods based on metaheuristics

Method	Path in the hierarchy
PASA	$R\{1,2\}/A12/C1/M1/E1$
MOSA	$R\{1,2\}/A12/C1/M1/E2$
MOGA	$R2/A2/C1/M2/E2$
NSGA	$R2/A2/C1/M2/E2$
NPGA	$R2/A2/C1/M2/E2$
WARGA	$R2/A2/C1/M2/E2$

We have classified the first two of these methods in the categories $R\{1,2\}$ because, in their daily use, these methods are better classed as $R2$ but, in theory, they are better classed as $R1$. The reason is that they exploit the stochastic effect of simulated annealing.

9.3.4 How to choose a method

We shall now illustrate the choice of a method through two examples:

- The sizing of a beam (see Sect. 1.6.3). This problem is well defined. We have two objective functions: minimize the section of the beam and minimize the distortion of the beam. We know, a priori, that this problem is not difficult. It has not any local optimum, and the structure of the tradeoff surface is convex.
  Such a beam may form part of a more complex structure (a bridge, for example). In this case we may want to test several solutions to simulate the global behavior of this complex structure with respect to the solution we have chosen for the shape of the beam.
- The sizing of a telecommunication network (see Chap. 11). For this problem, we have several objective functions: the global cost due to the positioning of the infrastructure of the network, and the availability of the network. This problem is, a priori, a difficult combinatorial optimization problem. It is possible to formulate a tradeoff between the two objective functions.

Example 1: the sizing of a beam

First level in the hierarchy: what mode of solution to choose?

We want to obtain an approximation to the tradeoff surface, because we want to have several solutions in our hands, to test them on a more complex structure. So, we have a solution mode of the a posteriori kind.

Second level in the hierarchy: what multiobjective treatment mode to choose?

Here, we have a choice. We choose a nonaggregative multiobjective treatment mode, because we have in mind the following property of this mode of treatment: there are no preferences to be determined.

Third level in the hierarchy: what kind of coupling with respect to the optimization method to choose?

We choose a method from the genetic algorithms family, because we have already programmed a genetic algorithm. So we choose a coupled method.

Fourth level in the hierarchy: what kind of method?

We have already chosen an evolutionary method.

Fifth level in the hierarchy: what kind of elitism?

Here again, we have a choice. Our problem is "simple" to solve. We choose a method without elitism because we do not want to lose time in programming an elitism module and including it in our genetic algorithm program for such a simple problem.

Path in the hierarchy

The path in the hierarchy is the following:

$$R2/A2/C1/M2/E2$$

Final choice

We have a choice between MOGA, NSGA, NPGA and WARGA. The final choice will be the NSGA method, because it allows us to obtain a good spread of solutions on the tradeoff surface, without excessively complex tuning to perform.

Example 2: the sizing of a telecommunication network

First level in the hierarchy: what mode of solution to choose?

For this problem, we have the possibility to define preferences with respect to objective functions. So, we look for a specific solution. The solution mode that we choose is the a priori solution mode.

Second level in the hierarchy: what multiobjective treatment mode to choose?

We do not know a priori the shape of the tradeoff surface. So, we have to choose a method which is efficient on problems where the tradeoff surface is nonconvex (because that is the most unfavorable case).

Because we have chosen an a priori solution mode, we choose an aggregative multiobjective treatment mode.

Third level in the hierarchy: what kind of coupling with respect to the optimization method to choose?

Here, we have a choice. We choose, as a first step, an uncoupled method. If the solution of the problem is not satisfactory, we can start again, but with a coupled method.

Fourth level in the hierarchy: what kind of method?

The scientific literature tells us that the best results have been obtained with methods of the "neighborhood improvement" kind (because these methods are easier to implement than evolutionary methods). Moreover, in an a priori solution method, it is not interesting to use an evolutionary method, because we are looking for just one solution.

Fifth level in the hierarchy: what kind of elitism?

Owing to the a priori difficulty of the search, it is more interesting to keep track of all good solutions encountered during the optimization. We therefore choose a method with elitism.

Path in the hierarchy

The path in the hierarchy is the following:

$$R1/A12/C2/M1/E2$$

Final choice

In the list of methods that we have examined, there are no methods with such a path in the hierarchy. Nevertheless, if we remove the elitism part of the path, we can choose a multiobjective treatment method such as the Tchebychev distance, coupled to an optimization method. To satisfy our specification in terms of method choice, we have to implement an archive module to keep track of the best solutions encountered during the optimization, or else continue our search for a method via a bibliographic analysis.

Part III

Case studies

10

Case study 1: qualification of scientific software[1]

10.1 Introduction

When we have to program a scientific software package (e.g. [Ascend]) dedicated to the simulation of an industrial process (such as chemical distillation chain or energy production in a nuclear reactor), we cannot take into account all the parameters of the process. Therefore, the engineer must reduce the complexity of the process model so that it can be simulated. The various approximations that have to be made often induce a gap, which may or may not be important, between the results obtained using the simulation and the results obtained via measurements. So we use an "artifice" which allows us to fill the gap: we bias the software so that its results correspond to the measurements. This bias is applied via modification of the parameters of the model.

10.2 Description of the problem

We shall describe the problem using an example. The physical domain involved is that of two-phase fluid flow.

The experimental data are measurements of the "void ratio" at ten points uniformly spread over the section of a channel, one side of which is heated. Inside this channel, there is a flow of fluid in the form of a mixture of vapor and liquid. An example of the kind of curve obtained is represented in Fig. 10.1. This void ratio can be computed using a software package in which the model used to simulate the flow depends on two parameters, C_1 and C_∞. The shape of the curve predicted by the simulation can be modified by changing these parameters. The problem that we have to solve is to find one or more pairs of values of (C_1, C_∞) such that the computed curve is as close as possible to the experimental curve.

The problem that then arises is: what does "as close as possible" mean here?

- Gaps of zero size between the computed and experimental curves at all measurement points?
- A gap of zero size at the point with the highest void ratio (point 10 in Fig. 10.1), known as the "hot spot"?
- Gaps with an uniform distribution?

[1] This chapter was written in collaboration with M. Dumas of CEA (michel.dumas@cea.fr).

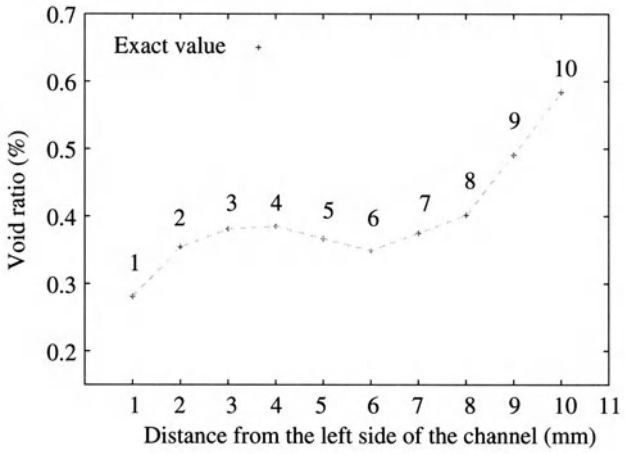


Fig. 10.1. A measurement of the vacuum rate at ten points of measurement.

- The minimum mean square gap?
- ...?

As we can see, the choice of the definition of "as close as possible" is important and has a major influence on the solution chosen.

Another approach is to consider the absolute gaps between computation and experiment at each measurement point as objective functions. In this case, our difficulty in choosing a definition of "as close as possible" can be seen in a new light. The definitions listed above can be seen as particular solutions of a multiobjective problem. These solutions are in a sense equivalent, because they are pieces of the tradeoff surface and are nondominated.

Describing the choice of the parameters (C_1, C_∞) as a multiobjective optimization problem allows us to clarify the problem: owing to the large number of objective functions to be considered, a unique solution to this problem does not exist; in fact, there are an infinite number of solutions. We can pick out the following types of solutions:

- Values of (C_1, C_∞) that allow us to obtain a very small mean error.
- Values of (C_1, C_∞) that allow us to simulate the hot spot with high accuracy. Such values could be useful for the sizing of the channel.

There is an important difference between this approach and the classical approach, which consists in defining a priori a formula to aggregate the objective functions and then solving the problem using that formula. In the best case, the multiobjective approach produces all possible solutions and then leaves the user to choose from them. The user can favor some behaviors of the simulation software over others in order to satisfy a particular need (high accuracy at some measurement points, for example). The user must perform a tradeoff to choose a solution, and will be conscious that this choice will have a major influence on the results of a simulation.

10.3 To represent the tradeoff surface

The problem that we face now is the problem of the representation of the solutions. Ideally, we should represent the whole tradeoff surface. We recall that the definition of the tradeoff surface is the following:

> The tradeoff surface is the set of points such that if we improve one or more criterion, we obtain a deterioration of the other criterion. (see Sect. 1.6)

The difficulty in the present example is that the tradeoff surface must be represented in a ten–dimensional space. So we have to find another representation.

This other representation can be in the form of the Pareto area, which is the image of the tradeoff surface in the space of the parameters (see in Fig. 10.2). The advantage of this approach is that the parameter space is of dimension 2, so is "easily" drawable. The problem that we now face is that, to obtain a good description of the Pareto

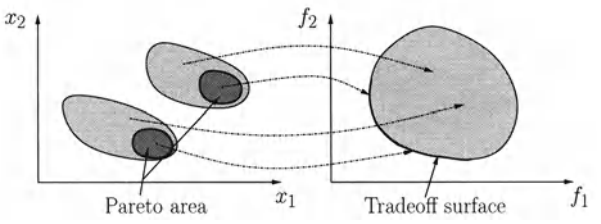


Fig. 10.2. The correspondence between the tradeoff surface and the Pareto area.

area, we must perform a huge number of computations of the objective functions and, owing to the time needed to compute the value of an objective function, these computations are not practicable.

The solution is to change our simulator. We have used a neural approximator (see [Dreyfus et al. 01]), which can perform computations nearly equivalent to those of the original simulator, within some error margin, of course.

To "train" the neural approximator, we proceeded in the following way:

Step 1: The parameters C_1 and C_∞ were varied in a predetermined domain:

$$0 \leq C_1 \leq 2 \text{ and } 0.5 \leq C_\infty \leq 2$$

We sampled 1500 distinct points in this parameter space.

Step 2: We performed a simulation for each of these points.

Step 3: We used the values of C_1 and C_∞, and the void ratio measured at ten points in the channel to train our neural approximator (see [Dreyfus et al. 01] for more details about how to train a neural approximator).

In this way, we determined ten neural approximators: one for each measurement point. In Figs. 10.3, 10.4 and 10.5, we can see examples of the kind of results obtained.

These figures represent the response of the neural network when we use as parameters values of C_1 and C_∞ which cover all of the parameter space. The three approximators chosen correspond to measurement points 1, 7 and 10. The behavior of these approximators is representative of the behavior for all ten measurement

points and the absolute values of their gaps from the experimental values were used to define the three objective functions of the problem.

As we can see when we look at these three curves, when we move a point through the parameter space, some objective functions are antagonistic. Some measurements increase while others decrease. This is an important characteristic of a multiobjective problem.

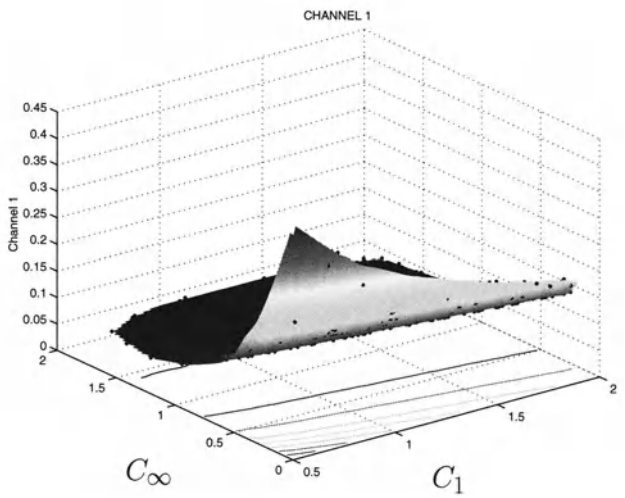


Fig. 10.3. Response of the neural approximator for measurement point 1.

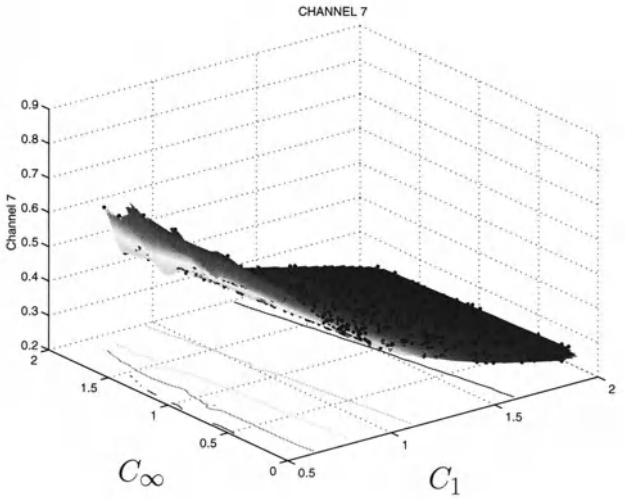


Fig. 10.4. Response of the neural approximator for measurement point 7.

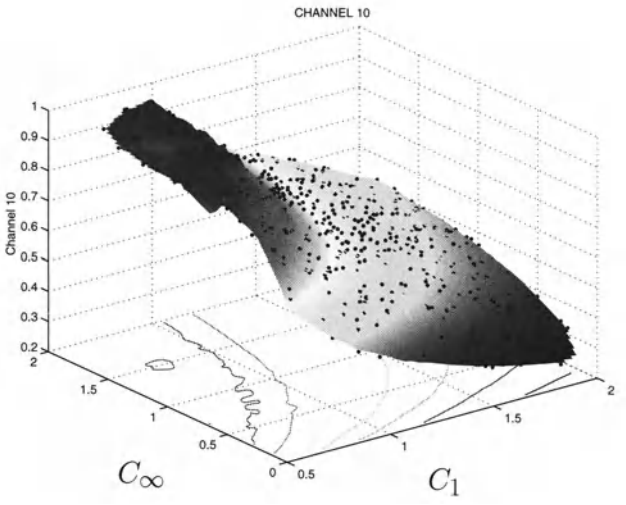


Fig. 10.5. Response of the neural approximator for measurement point 10.

We have to locate one or more areas of the parameter space which contain the Pareto optimal solutions: the Pareto area. To do so, we can plot these curves in a plane, where we represent only the level curves whose values correspond to the experimental values. In Fig. 10.6, we have represented the level curves for measurement points 1, 7 and 10. Each curve defines a set of pairs of values (C_1, C_∞) which allows us to obtain, for the corresponding measurement point, a gap of zero relative to the computation. These three curves define, in the parameter space, a domain whose image is the tradeoff surface. In this space, if we move a point x toward the minimum–level curve for point 10, we increase the values of the level curves for points 7 and/or 1. Similarly, if we move this point toward the minimum level curve for point 1, we increase the values of the level curves for points 7 and/or 10.

This behavior is typical of a point which belongs to the tradeoff surface.

In Fig. 10.7, we have represented the Pareto area. Inside this area, we have chosen three points (A, B and C) which correspond to Pareto optimal solutions (because they have been chosen inside the Pareto area). In Fig. 10.8, we have represented the shapes of the curves which correspond to these three points. These curves all have similar gaps with respect to the experimental curve. These results illustrate the fact that some points in the Pareto area give results that are accurate at the edges of the channel but not at the center, whereas others give a shape similar to the experimental shape and have a small maximal gap.

10.4 Conclusion

The procedure described above can be generalized by use of a multiobjective genetic algorithm. This kind of method makes it very easy and efficient to construct an approximation to the Pareto area via the use of neural approximators, which provide a rapid simulation.

We can use aa NSGA algorithm with management of the diversity of the solutions in the parameter space, or a genetic algorithm with a real code (see [Perrin et al. 97]).

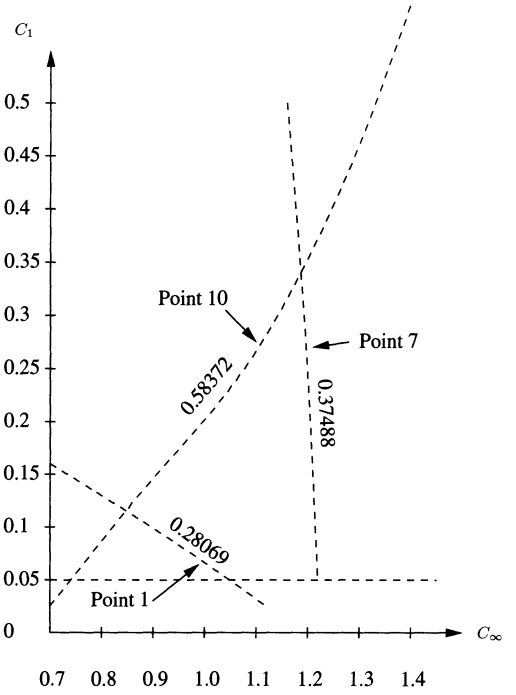


Fig. 10.6. The curves of minimum levels for measurement points 1, 7 and 10.

In this way, with a population of relatively small size, we can hope to obtain an approximation to the Pareto area quickly.

The main freedom that this method allows is the possibility of dynamical calibration with respect to the needs of the users.

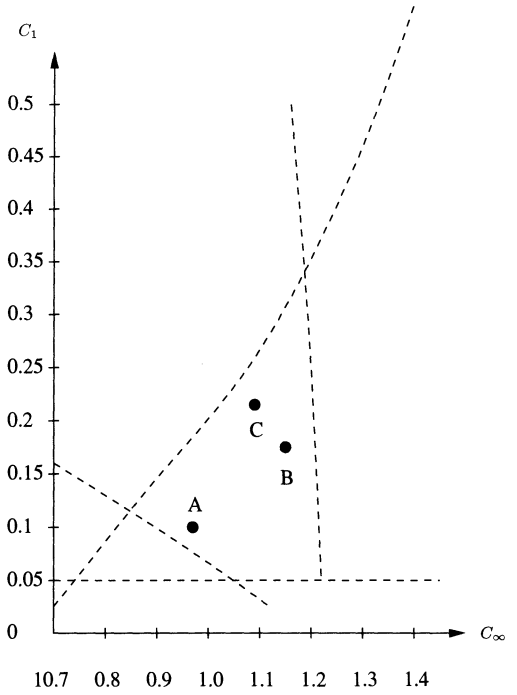


Fig. 10.7. Three points in the Pareto area.

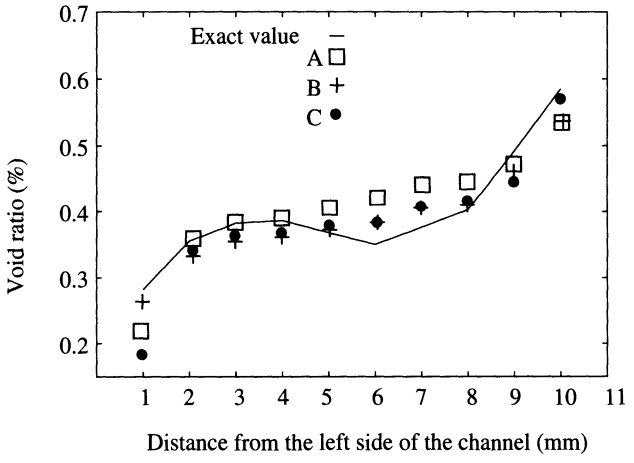


Fig. 10.8. The curves corresponding to the points in the Pareto area shown in Fig. 10.7.

11

Case study 2: study of the extension of a telecommunication network[1]

A service network is continuously changing its shape. When it first become operational, it is just at its start of its life. The network will evolve in response to the increase in the demand that it must route from one point to another. Its evolution will be driven by the obsolescence of the equipment which makes up the network, and, even more so, by the strategic decisions of the operator that manages the network.

These strategic decisions are responsible for most of the investment in time involved. Therefore, these decisions must be taken judiciously. However, there are dependencies between elements, sometimes contradictory, sometimes extremely correlated, which make this evolution very complex.

Consider an existing network, shown in Fig. 11.1 in bold, and assume that an extension (shown by fine lines) of this network is envisaged at the south of the network.

Before such a decision is reached, numerous studies will have been performed. These studies will characterize the geographical area that the operator would like to develop (the boundaries of the area, the potential customers inside the area, existing infrastructure inside the area, the economic balance of the area, etc.), determine the strategic points (with respect to concentration of customers, existing infrastructure, etc.), determine what links are most useful or easier to open between the nodes, and characterize the suppliers present on the market (the kind of equipment they propose, the appropriateness of this equipment to the existing network, delivery time, costs, etc.). So, this problem is extremely complex, and numerous economic studies must be performed to solve it.

The multicriteria problem presented here concerns a part of the global problem. This study deals with how to size the proposed extension of the network: what are the most profitable investments, with respect to the availability of the network, which allow the network to route all the demands of a target demand matrix?

11.1 Network

A network is modeled by a nonoriented graph G, for which the elements x are the nodes ($x \in N$) and the edges ($x \in A$). Each of these elements is characterized by its capacity c_x and its installation cost $C(x, c_x)$.

[1] This chapter was written by C. Decouchon-Brochet of France Télécom - Research and Development (camille.decouchonbrochet@francetelecom.com)

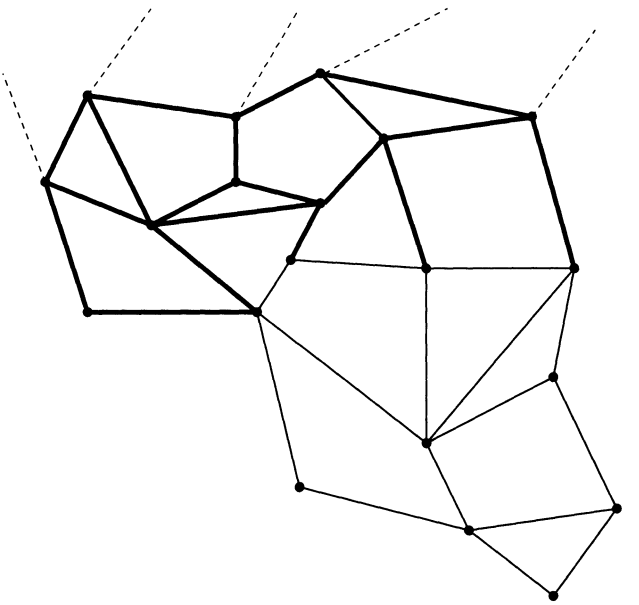


Fig. 11.1. An existing network and its envisaged extension.

The function of a network is to route information, so the characteristics of this information must be specified. These characteristics are specified as a set of demands (the demand matrix), each of which corresponds to some *quantity* which has to be routed from an *origin* node to a *destination* node (the routing).

We shall assume that these demands are not concatenated, that is to say, they are divisible and so can be routed from the origin node to the destination node using several routes. On the other hand, the edges are bidirectional, that is to say they can route information both ways.

11.2 Choice criteria

11.2.1 Parameter used to model the availability

The availability of a network is characterized by its ability to react to changes in the demands to be routed. In other words, if part of the demand is not routed, the availability of the network is deteriorated. For example:

- unreliability of all or part of the network decreases the availability because the failures that occur have not been envisaged.
- the absence of oversizing of the network decreases its availability, because it will not be able to deal with an increase in the demand.

In the study presented here, the reliability of the network, or, more precisely, of its response to the demands, is achieved by guaranteeing two separate routes (with respect to both edges and nodes) of sufficient capacity on the network. Thus, the network is always able to deal with possible failures.

The parameter that have been used to model the availability is the level of demands that can be met, T. The closer this parameter is to 1, the more satisfactory is the availability of the network.

11.2.2 Link between availability and cost

Of course, the extent to which measurements to guarantee a good availability of the network are taken is limited by the cost of putting those changes into effect. If two separate routes are guaranteed for each demand, this implies an oversizing of the network, which will necessarily be costly. On the other hand, reducing the availability offered by the network will allow one to have a lower infrastructure cost.

By and large, the availability problem consists in finding a tradeoff between the global cost of investment and the resulting availability of the network, and availability to pay a high price in investment to guarantee an extremely high level of availability or to pay penalties in the case of an event which saturates the network at points where availability is normally guaranteed.

In the following section, we describe the costs which are considered in this study.

11.2.3 Costs

The costs which appear in a study of the extension of a network can be divided into three parts.

11.2.3.1 Investment costs

These are costs related to work on the network infrastructure. This work can be of various kinds, such as:

- addition of an edge;
- addition of a node;
- modification of an edge (extension of the capacity of the edge, etc.);
- modification of a node (replacement of the equipment at this node by equipment of a new generation, etc.).

For all these possible modifications, there exist several kinds of costs:

- costs related to preliminary studies;
- costs related to the purchase of equipment;
- costs related to installation work (i.e. costs related to human activity).

11.2.3.2 Maintenance/management costs

These costs are costs which allow the network to function. They are:

- costs related to the conversion of the network (which allow one, for example, to update obsolete routes in the course of the evolution of the network);
- costs related to preventive maintenance;
- costs related to routine maintenance:
  - diagnosis;
  - repair.

Here too, these costs are composed of the costs of equipment purchase and costs related to human activity.

11.2.3.3 Costs related to economic activity of the network (profitability and penalties)

These costs gather together various costs related to the economic activity of the network.

The extension of the network necessitates investment costs, but it will allow the network to route new demands (demands with a new origin and/or destination) and so to serve new customers. Profits can be included at this level. These profits can, however, be decreased by the costs of penalties due to a poor quality of service at certain times (guaranteed availability not reached, etc.).

11.3 Study of a network extension

11.3.1 Problem

As described at the beginning of this chapter, an operator wishes to extend a network that it is managing into a neighboring area. The subnetwork which corresponds to this extension is represented in Fig. 11.2. The nodes of the subnetwork to be created are known. The links are proposed. The goal is to determine whether we must bring these links into service, and, in the case of a positive answer, with what capacity. Of course, bringing each link into service has a cost which grows with its capacity.

Hence, we define a matrix which contains the cost of bringing into service an edge a of capacity c_a: $C(c_a)$. The costs related to the activity of the network are evaluated and added to the investment cost, and this cost is scaled to each edge.

11.3.2 Modeling

11.3.2.1 Variables

The variables to be considered are:

- the value of the capacity of each edge a, c_a;
- the cost of bringing into service an edge a of capacity c_a, $C(c_a)$;
- the parameter which characterizes the availability of the network, T.

We denote by C_T the total cost of investment, equal to the sum of the costs of installed edges:

$$C_T = \sum_{i=1}^{N_a} C(c_i) \tag{11.1}$$

where i corresponds to the edges, and N_a is the number of edges of the network extension.

11.3.2.2 Criteria

The problem consists in determining the optimal combination of capacities (the variables c_i, $i \in [1, N_a]$) to be put on the edges so as to optimize in the best way possible two parameters of this subnetwork:

- the total cost that arises from bringing its infrastructure into service;
- a parameter which characterizes the availability of the network.

This problem can be expressed as

$$\begin{cases} \min_{c_i} C_T \\ \max_{c_i} T \end{cases}$$

11.3.2.3 Constraints

The determination of the links to be brought into service must be done while satisfying some constraints:

- the total investment cost of the network must be less than a maximum threshold:

$$C_T < C_{\text{threshold}}$$

- the capacities of the links for routing are limited;
- at least single routing of all of the demands must be guaranteed;
- a minimal value of the parameter which characterizes the availability of the network is imposed: 0.5.

11.3.3 Necessary data

11.3.3.1 Fixed infrastructure of the network

The hard infrastructure of the extension network must be known. In the example that we are considering:

- the number of nodes is 10;
- the number of potential edges is 15;
- the characteristics of these elements are as described in Fig. 11.2. The nodes are identified using letters and the edges are numbered (from 0 to 14).

11.3.3.2 Variable infrastructure of the network

Each of the edges can have one of these four values of its capacity, in units of 155 Mbit/s:

- 0 if the edge is not used;
- 16;
- 32;
- 48.

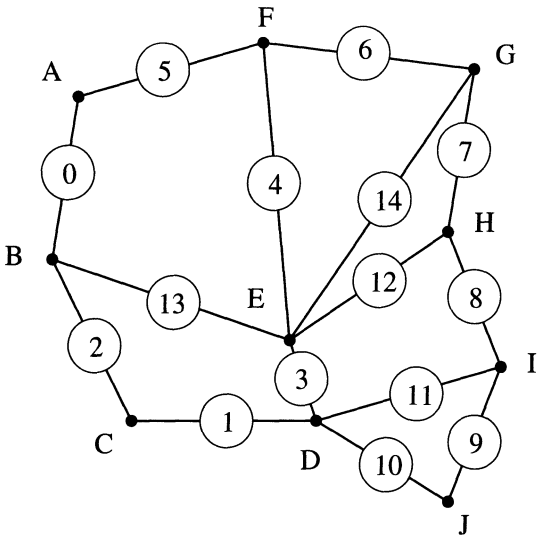


Fig. 11.2. The subnetwork to be created.

Table 11.1. Matrix of the demands to be routed.

Code of demand	Origin	Destination	Magnitude (in units of 155 Mbit/s)
0	G	J	4
1	G	C	8
2	A	I	12
3	B	J	16
4	B	H	4
5	B	I	4
6	F	D	4
7	B	G	8
8	G	A	4
9	A	C	16
10	J	A	12
11	E	H	4
12	E	I	8
13	E	F	4
14	E	C	12

11.3.3.3 Demand matrix

The matrix of the demands to be routed on this new subnetwork also needs to be known. We consider 15 demands, represented in Table 11.1.

11.4 Method of solution

The type of problem presented here is a multicriteria optimization problem, because several criteria must be optimized simultaneously. The main difficulty in such a problem comes from the contradictions which exist between the criteria. Indeed, in

concrete cases, it is rare to find a solution which optimizes all the criteria simultaneously. So, we must find a tradeoff between the various criteria, knowing that there does not exist only one tradeoff for a given problem, but several tradeoffs, and that the number of tradeoffs grows with the number of criteria. This complexity is due to:

- the important combinatorial features of the network sizing problems;
- the fact that there exist many quasioptimal solutions with respect to each individual criterion, which implies a significant number of tradeoffs.

Hence, the objective when we solve such a problem is to find a set of nondominated solutions, which can then be proposed to the decision maker, so that he/she can make a choice.

Our problem can be solved using an evolutionary algorithm, which is described in Fig. 11.3. This algorithm allows one to extract the solutions which correspond to the most interesting tradeoffs. It is then the task of the decision maker to choose, from the solutions presented, the one which seems best to his/her mind.

11.4.1 Definition of an admissible solution

An admissible solution is a combination of capacities to be put on the edges $\{c_i \mid i \in [1, N_a]\}$ which allows the infrastructure so defined to satisfy the constraints.

11.4.2 Algorithm

We recall in Fig 11.3. the general organization chart of a genetic algorithm.

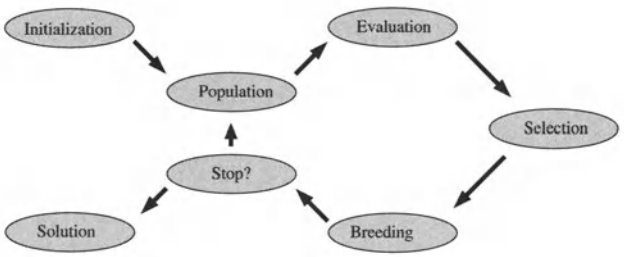


Fig. 11.3. Organization chart of a genetic algorithm.

11.4.2.1 Initialization 1: encoding

The initialization phase uses a technique to code the solutions. The encoding defined here is very simple because it is based on the indexation of the set of potential modifications of the network.

One variable for each new element is created; the value of that variable specifies what capacity is put on the corresponding element. We have seen that 15 edges can be added to the existing network to make up the subnetwork we are trying to create. So, the encoding is composed of 15 variables (in the order of the numbering of the edges). Each of these variables can be:

- zero if the associated edge is not created;
- equal to 1 if the associated edge has a capacity of 16 (in units of 155 Mbit/s);
- equal to 2 if the associated edge has a capacity of 32 (in units of 155 Mbit/s);
- equal to 3 if the associated edge has a capacity of 48 (in units of 155 Mbit/s).

So, there exist $4^{15} = 1\,073\,741\,824$ possible combinations of links to be considered, for such a small number of edges to be added!

11.4.2.2 Initialization 2: construction of an initial solution

The first phase of the initialization is followed by the creation of an initial population. The only constraint which concerns this initial population is that it must belong to the domain of admissible solutions.

11.4.2.3 Evaluation 1: computation of the current solution

The evaluation of the solution consists in determining the current solution that respects the constraints, and in finding the values of the criteria we are trying to optimize.

To determine whether the current solution respects the constraints, we try to route the given demands, on the network composed of the edges and capacities of the current solution. This implies that we find the shortest path for all the demands at each evaluation step of the algorithm.

This route computation has a nonnegligible impact on the efficiency of the algorithm because, for example, for demand number 2 between A and I, if all the edges are installed, there exist 6 routes which have the minimal number of links (4 links). These routes are represented in Fig. 11.4.

Note: this routing can be a bicriteria routing which tries to privilege the shortest path of highest quality.

Regarding the evaluation of the criteria, we have to compute:

- the total cost of investment related to the current solution,
- the value of the parameter which represents the availability of the network.

This second evaluation implies that, when determining the shortest–path routes of the kind described above for all the demands, we have to determine precisely, for each demand, the existence of a "two-routing" on separate paths (in terms of nodes and edges), so as to verify the availability of the network for the current solution.

If we consider again demand number 2 between A and I, the 2 separate paths could be AB+BC+CD+DI and AF+FG+GH+HI. These paths are represented in Fig. 11.5.

Note: If only the eight edges of the above two-routing constitute the current solution, then this solution corresponds to the addition of eight edges of capacity 16, the others being unused. The encoding will be 1 1 1 0 0 1 1 1 1 0 0 1 0 0 0.

So as to avoid an exhaustive enumeration of all the two-routings possible for each demand, a heuristic for finding two-routings on separate paths has been created. This heuristic needed to be made sufficiently efficient, since the global algorithm uses the results of the heuristic.

However, to avoid the need to search systematically for these two-routings, those two-routings which had already been computed for a given individual were saved continuously in a dictionary.

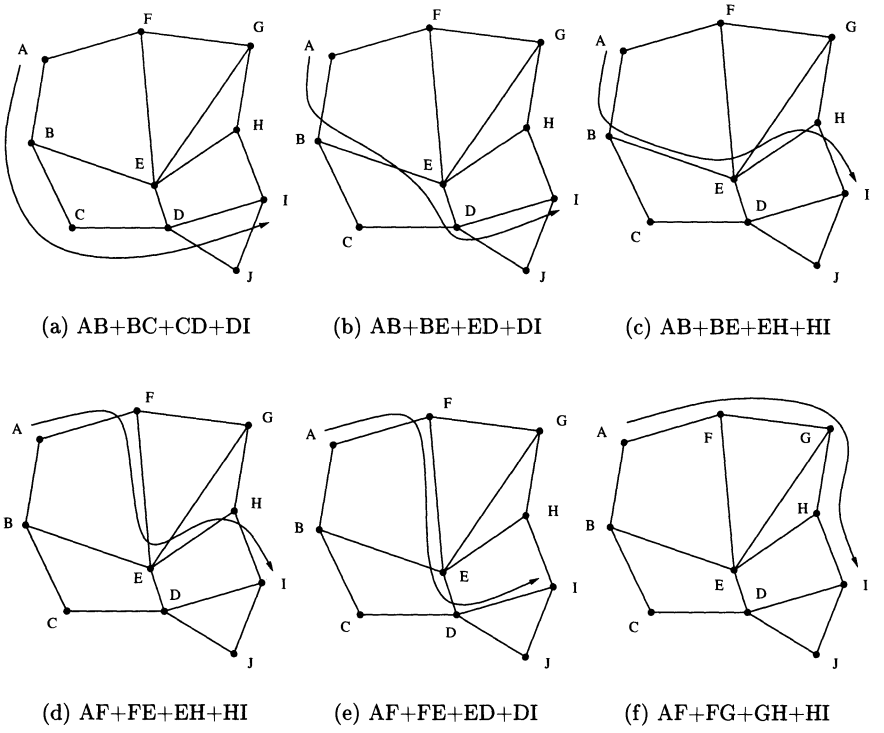


Fig. 11.4. Routes with the shortest path (in number of links) for a demand between A and I.

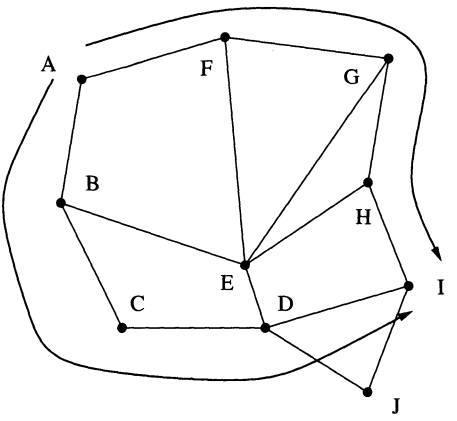


Fig. 11.5. Example of a two-routing between A and I.

11.4.2.4 Evaluation 2: localization of the solution with respect to the current Pareto frontier

In a multicriteria problem, an individual is more efficient if the number of alternatives which dominate this individual is small. In practice, we evaluate this number on a set of specific alternatives, since it is not possible to know its value exactly. Frequently the current population of the algorithm is used. In our case, we have used the evaluation function proposed in [Fleming 93], which is more economic in computation time (total sorting time $O\left(n^2\right)$). The evaluation of each individual was performed using the following steps:

- comparison of all the individuals with all others, while saving the number of individuals which dominate each individual;
- returning these values; the maximum value is associated with the nondominated individuals;
- scaling so as to obtain the final "fitness" of the individuals.

Because this evaluation used the current population, the values of the performances of the individuals varied during the running of the algorithm. This is a general drawback of this kind of method that we cannot overcome.

11.4.2.5 Selection

This phase determines the way in which the individuals which will produce the next generation are selected. In general, it is preferable to favor the most efficient individuals over the others. Several recent studies [Deb et al. 00c] have shown the importance of defining an elitist selection process for this purpose. We have used this principle by creating an archive population which contains the nondominated solutions. This population is updated at each generation and is used to perform a random selection.

11.4.2.6 Breeding

After the selection process, we can perform the breeding step. We have used two types of elementary operators:

- an operator which involves one individual, *mutation*;
- an operator which involves two individuals (a binary operator), *crossover*.

The algorithm must perform a sufficiently large and regular sampling of the Pareto frontier. To do this, the elementary operators listed above were associated with an *optimization* operation, with the goal of improving the value of an individual for a given criterion, and a *diversification* operation (see in Fig. 11.6):

- Optimization operation: this first operation is similar to a random local optimization. Only mutations are involved during this operation, and a mutation was defined for each criterion:
  - The first mutation improves the value of the first criterion (the total cost produced by bringing the network infrastructure into service) by reducing the gene modification indices, since the smaller the indices, the smaller the capacity of the associated edges, and so the smaller the infrastructure cost.

- The second mutation tends to improve the availability of the network by adding routing paths and by increasing the capacity of the saturated edges.

Like the criteria that we are trying to optimize, these mutations have contradictory effects.

- Diversification operation: the objective of this operation is to enlarge and to sample as regularly as possible the Pareto frontier. To do this, mutations as defined above are necessary. However, to avoid premature convergence, a random perturbation operator and a crossover were defined.

These operators and operations were used in the following way:

- selection of the parents;
- random choice of whether or not to use the crossover operator;
- random choice of whether or not to use a mutation: if yes, random selection of the mutation type;
- if no operators were used, one of the two individuals was chosen randomly.

Note: the random choices of whether or not to use crossover operator and whether or not to use a mutation were performed in accordance with a predefined crossover probability and a predefined mutation probability, respectively.

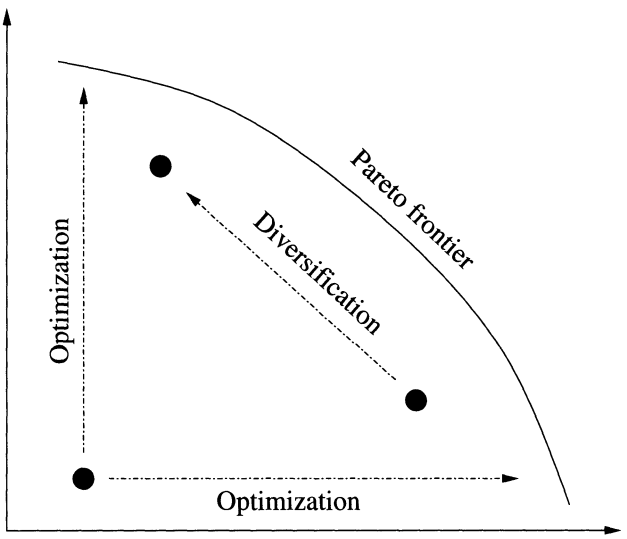


Fig. 11.6. Various breeding operators.

11.4.2.7 Stopping criterion

Our stopping criterion was composed of two criteria:

- a first criterion related to the size of the domain explored;
- a second criterion based on the "quality" of the current solution.

11.4.2.8 Global algorithm

The global algorithm that was used is shown in Fig. 11.7. In this figure, the "dominator" term corresponds to a nondominated solution.

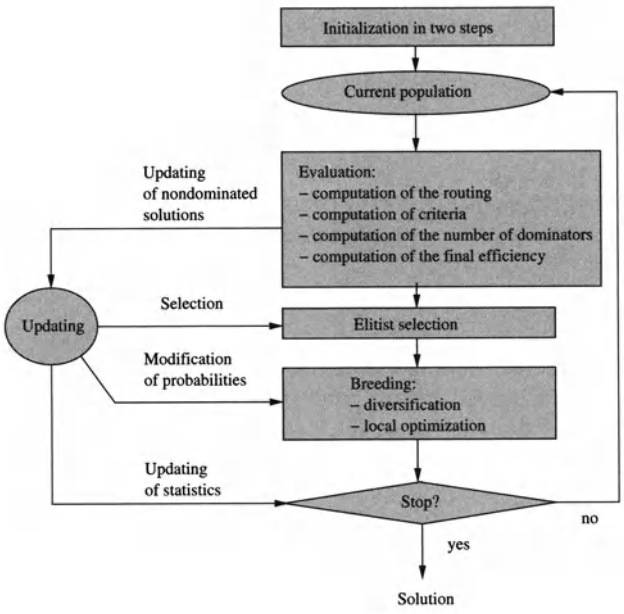


Fig. 11.7. Organization chart of the genetic algorithm used.

11.5 Results

So, what are the edges and the capacities of the edges which will allow the network to offer a good availability at a low cost? We found 21 Pareto optimal solutions for this problem, described in Table 11.2. We notice that as the availability increases, so does the cost, as we would expect.

The first eight solutions do no obey the constraints, since the availability of the network is less than 0.5. The decision maker therefore has to consider 13 Pareto optimal solutions. We may notice also that some edges are overused, such as edges 1, 3, 5, 8 and 14. Some other edges, such as edges 9, 11, 12 and 13, are less used.

The results are presented in Fig. 11.8.

The Pareto frontier is clearly visible on this graph. The part of the graph shaded gray covers the solutions which do not obey the constraints, since the availability of the network here is less than 0.5.

11.6 Conclusion

The algorithm that we have used produces, in a reasonable time, a Pareto frontier which is sufficiently diversified and continuous. This algorithm can also deal with a

Table 11.2. The efficient solutions obtained.

Encoding	Cost	Availability
2 3 2 2 1 1 1 1 1 1 2 1 1 1 2	48944	0
2 3 2 2 1 2 1 1 1 1 2 1 1 1 2	51072	0.1
2 3 2 2 1 2 1 2 2 1 1 1 1 1 2	53056	0.133
2 3 2 2 1 2 1 1 2 1 2 1 1 1 2	53184	0.17
2 3 2 2 1 2 1 1 1 1 2 1 1 1 3	53776	0.23
2 2 2 2 2 2 0 1 3 1 2 1 1 2 2	54336	0.29
2 3 2 2 1 2 1 2 2 1 2 1 1 1 2	55104	0.31
2 3 2 2 1 2 2 1 2 1 2 1 1 1 2	55264	0.37
2 3 2 2 1 2 2 2 2 1 2 1 1 1 2	57184	0.57
2 3 2 3 1 2 2 2 2 1 2 1 1 1 2	59872	0.74
2 3 2 2 1 2 2 2 2 1 2 1 1 1 3	59888	0.77
2 3 2 2 2 2 2 2 2 1 2 1 1 1 3	61680	0.81
2 3 2 2 3 2 2 2 2 1 2 1 1 1 3	63472	0.83
2 3 2 2 2 3 2 2 2 1 2 1 1 1 3	63808	0.84
2 3 2 3 2 2 2 2 2 1 2 1 1 1 3	64368	0.86
2 3 2 3 2 2 2 2 2 2 2 1 1 1 3	65488	0.88
2 3 2 2 2 3 2 2 3 2 2 1 1 1 3	67040	0.89
2 3 2 3 2 3 2 2 2 2 2 1 1 1 3	67616	0.9
2 3 2 2 3 3 2 2 3 1 2 1 1 1 3	67712	0.92
2 3 2 2 3 3 2 2 3 2 2 1 1 1 3	68832	0.93
2 3 2 2 2 3 2 2 3 2 2 1 3 1 3	71872	0.96

third criterion, corresponding to the profitability of this extension of the network. This third criterion could be a cost that behaves differently from our first criterion, which is the investment cost.

Of course, the operational problem is more complex than the model presented here. Indeed, a huge number of criteria are involved in the considerations of the decision maker, for example the delivery time of bringing the network into service. Some criteria can, unlike the ones presented here, be nonquantifiable but instead qualitative. Such criteria include, for example, strategic criteria related to competition, and political criteria.

Nevertheless, a frontal, global approach is not realistic. Network designers must use their expertise to define subproblems which are algorithmically tractable and also sufficiently significant, and which will be helpful to the decision maker. The approach presented here is a possible facet of the support that can be provided to decision makers.

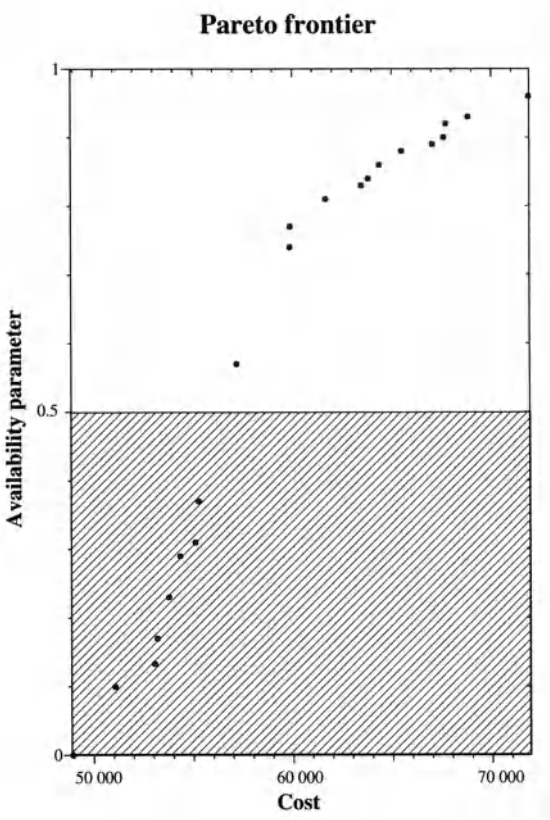


Fig. 11.8. The set of efficient solutions.

12

Case study 3: multicriteria decision tools to deal with bids[1]

12.1 Introduction

EADS (EADS LV), which is a wholly owned subsidiary of the EADS (European Aeronautic Defence and Space Company) group, which was itself formed from a merger of Daimler Chrysler Aerospace AG (DASA), Construcciones Aeronauticas (CASA) and Aérospatiale Matra on July 10, 2000, has used throughout its life a multicriteria decision model based on the ELECTRE TRI method to decide whether or not to respond to calls for bids. This model had also been used by one of the predecessor organizations of EADS LV since January 1, 1996. The commercial results of the model, which is also used to decide whether or not to make spontaneous commercial offers, are saved in a database. EADS LV makes several hundred commercial offers every year. These offers either are unsolicited or are made in response to calls for bids or via official consultation.

Since the creation of a first–generation model using nine criteria in 1996, frequent modifications have been made to the model. A major modification to the model was made in 2001; this was possible only because of the experience of the initial period and, especially, because of the acquisition of data. So as to take account of the variety of commercial offers of products or services, a second–generation model with five criteria was developped into a family of models made to measure.

The models of the first and second generation are briefly described in the following sections. For obvious confidentiality reasons, the numerical values and the graphs have been deliberately modified.

12.2 First–generation model

12.2.1 Goal of the model

The first model dedicated to aiding decisions was based on the following problem: "Should we respond to this call for bids ?" It was quickly generalized to the question: "Should we make an offer for this new deal ?"

To accept a new deal means authorizing a team which has issued a "new deal card" (NDC) to spend time on preparing a detailed commercial offer. This obviously

[1] This chapter was written by J.-M. Contant, J.-L. Bussière and G. Piffaretti of EADS Launch Vehicles.

involves a cost, which will be recovered in the contract in the case of success and included in the profits and losses in the case of failure. To refuse the deal means neither to start any work nor to issue any commercial offer for this deal. A budget allocation is generally sought for each NDC. It is granted entirely or partially. This situation illustrates well the dilemma of the manager, who stands to win or lose a lot of money on the basis of a large or small amount of advice (for an annual budget of about 2 or 3 M€, decisions considered here have an influence of about 0.5 to 1 M€). For the team inside the enterprise that has issued the NDC, the problem is quite different. The team members are often the judge of the deal but are also trying to obtain, by all the means available, the authorization to issue the offer and to collect the related budget.

The decision aid model that we developed uses the sorting problem[2]. The answer is constructed by assigning to each new deal one of the four following categories:

- C1: no;
- C2: doubtful no;
- C3: doubtful yes;
- C4: yes.

The new deals are not judged one against the other, but are evaluated independently. The association between a category and a deal gives an indication of how much interest we should show in this new deal. The sorting problem is very suitable here, since commercial offers can arrive in any order, and independently of one another. The classification and selection problems would not have given an useful answer, because comparison of the offers would be impossible and useless.

12.2.2 The criteria of the first–generation model

For many years, decisions about whether to make commercial offers were taken locally in an empirical and intuitive way. It was very difficult to save data and results. In the case of success, the team was focused on the new tasks and, in the case of failure, was not really motivated to discuss the subject. A "decision making" committee for the new deals was then created, but controlled only 50% of the offers; the remainder was decided locally without referring to the committee. One of the consequences was that the exact budget volume of this activity was ignored for a long time.

The validation of the first model was done in 1995, from a sample of 11 typical commercial offers which characterized the four categories of yes, doubtful yes, doubtful no and no. The model, brought into effect in 1996, was based on nine criteria, divided into three families, which aimed to quantify the following aspects:

- probability of winning the deal;
- strategic value of the deal;
- economic value of the deal.

[2] The sorting problem is that of assigning actions to categories by looking at the intrinsic value of each action. The categories are determined independently of the actions to be assigned to them. We recall the definition of an overranking relation [Roy 85]:

An overranking relation is a binary relation S defined on a set of actions A such that, aSb if, be given the quality of the evaluation of the action s and the nature of the problem, there is sufficient number of arguments to admit that action a is at least as good as action b, without any important reason to refuse this assertion.

Family 1: Probability to win the call for bids

Criterion 1: nature of relations with the customer

This corresponds to taking account of the relations with the customers that issue the calls for bids. The scale related to this criterion was the following:

Bad relations	Unknown	Badly known	Known	Good relations	Excellent
0	2	3.5	6	7	8.5

Criterion 2: competences of EADS LV and its partners

This is the combination of the basic technical capacity to produce the product and the capacity to apply that technical capacity in the appropriate way. The scale related to this criterion was the following:

Insufficient	Fragile	Recent	Correct	Quite good	Strong	Leader
0	2.5	5	7.5	11	12	16

Criterion 3: competitiveness of the EADS LV price against the market price

To have a chance to win the call for bids, EADS LV must sell at a competitive price. The scale used was continuous and varied from 0 to 2.

Family 2: Strategic value

Criterion 4: credibility of the call for bids

This criterion must allow one to detect a doubtful call for bids, one that EADS LV has no chance of winning. The scale related to this criterion was the following:

Very much not credible	Not credible	Doubtful	Without opinion	Credible	Very credible
0	2	3.5	5.5	7.5	9.5

Criterion 5: strategic will

This corresponds to the strategic will for a given deal, which arises out of the strategic program of the enterprise. This will is also in phase with the R&D investment. The scale related to this criterion was the following:

Excluded	Very weak	Weak	Mean	Strong	Very strong	Obligatory
0	6	7	9.5	10.5	11	14

Criterion 6: availability of the teams

This criterion takes into account the planning of the technical and commercial activity of EADS LV so as to avoid crisis situations due to momentary overload. The scale related to this criterion was the following:

Not available	Overloaded	Heavily loaded	Loaded	Available	Underloaded
0	3.5	6.5	10	12	14

Family 3: Economic value

Criterion 7: self–financed research effort and investment effort

This criterion takes into account the possible cofinancing needs required by the customer. The scale related to this criterion was continuous and varied from 0 to 1.5 M€.

Criterion 8: EADS LV effort

This is the ratio between the profit required and the expected sales figure before taxes. The purpose of this criterion is to help us to decide how much money is to be spent on the preparation of the offer and to put a limit on this amount, in relation to the expected profit in the case of success. The scale related to this criterion was continuous and varied from 0% to 40%.

Criterion 9: expected expenses on studies and in production

This criterion, which represents the potential expenses that would arise in the case of success, is intended to favor the full use of the production tools. Consequently, it favors big deals. The scale related to this criterion varied from 0 to 530 000 hours.

12.2.3 Evolution of the first–generation model

Over a period of two years, during which the decision–making committee did not take all calls for bids into account, the committee dealt with:

- in 1996, 104 commercial offers;
- in 1997, 96 commercial offers.

EADS LV can face two situations in a call for bids, either in competition or by mutual consent:

- In the case of a call for bids in competition, EADS responds to a call open to other organizations.
- In the case of a call for bids by mutual consent, EADS is the only bidder, which does not mean that the deal is automatically won. Indeed, in this case, the enterprise must fight to fit into the budget of the client or to match the market price.

After two years of running, a performance study was conducted to correlate the predictions with the results (successes or failures):

- the error rate on the prediction "yes" was 65 %;
- the error rate on the prediction "no" was 15 %.

At this stage, the best criteria to be used in the decision were not necessarily imaginable and others were hard to tune, because we needed to have some knowledge to tune them. For example, there was a carefully thought–out veto based on the criterion "self–financed research effort" that implied refusal of the actions which needed a cofinancing greater than 500 k€, whereas the committee, in contradiction with its own decision, proceeded with and won bids which needed a cofinancing greater than 1 M€.

To clarify the situation, a study on the sensitivity of the cumulative data from 1996–1997 was conducted, by varying the shapes of the reference actions. Five scenarios, including the optimal scenario, were considered. We found an error rate of about 15% on the prediction "yes" (the percentage of cases where the prediction was "yes" but the outcome was unfavorable); this corresponds to a potential economy of about 0.5 M€. From January 1998, a new tuning of the model was operational.

Since 1999, the decision–making committee has seen its power greatly increased. It has become the "committee for new deals". It has the same weekly meetings and defines a new "new deal card" on an Excel spreadsheet, to better characterize the candidate actions. This NDC serves also to follow the preparation of the offer and allows one to correlate the retained hypotheses when the final parameters of the offer sent to the customer are chosen. The new committee removed a dysfunction by strictly controlling the endowment budget (1.5 M€). It examined 90% of the current offers in 1999, and then 100% in 2000. The very small and very large deals are not treated by this committee.

In 2000, a study of the performance over a time interval of seven months was conducted. A huge amount of work was performed on the data to yield a set of actions with reliable indicators. Obtaining information about the successes and failures was laborious. Nevertheless, it allowed us to calculate some valuable ratios related to the commercial performance of the enterprise, which were presented to the management of the enterprise.

The methods and the decision aid tools now exist and could easily be brought into service. But, after more than five years of experience, the questioning of the choice of criteria with respect to their relevance or selectivity and the detailed modeling of these criteria, appear to be essential points in the use of such a decision aid. In the final analysis, without historical quantitative data, the selection of the criteria is normative, and highly dependent on the culture of the contributor. So, this constitutes a model of the first generation, which can only be a Cartesian intellectual exercise, one of trying to find how to divide the problem into consistent components or into criteria.

Nevertheless, the work with the first–generation model gradually brought some rigor into the evaluation of offers and strengthened the selectivity. The gain in selectivity between 1997 and 1998 (30%) generated an economy of about 500 k€ in the endowment budget. The goal in the long term is to save more money.

12.3 Understanding the insufficiency of the first–generation model

Various criteria which were supposed to be important were, in fact, nondiscriminating for the following reasons:

- the unverifiable criteria allowed us to give some deals marks that "spoke for the defense" and which were more like a socio–cultural investigation than the reality of the competition;
- the persons involved, by their nature, did not want to lose a deal;
- the routine induced repetitive decisions which were less thoughtful.

During this period, the environment and the nature of the enterprise evolved considerably:

- the aerospace establishment at Cannes (satellites) was sold to Alcatel in 1999;
- the enterprise was privatized in mid–1999 and was then subject to market forces;
- Aérospatiale Matra merged mid-1999;
- Aérospatiale Matra, DASA (Germany) and CASA (Spain) merged in July 2001.

These changes caused some important disruptions in the deal database and led us to select the deals of 1996–1997 for the samples dedicated to the tuning of the model.

To better understand the insufficiencies of the first–generation model, a study, described below, was carried out. This study revealed several kinds of dysfunction:

- Criteria which were nondiscriminative and so were never used in the decision.
- Criteria that were ineffective owing to the marking method and so were without effect on the decision. They needed to be corrected or removed.
- Criteria which were, in fact, constraints and so were always satisfied.

12.3.1 Examples of nondiscriminative criteria

Nature of relations with the customer (criterion 1)

As far as the marking of the deals since 1996 is concerned, the graph in Fig. 12.1 illustrates the bad distribution of the recorded marks.

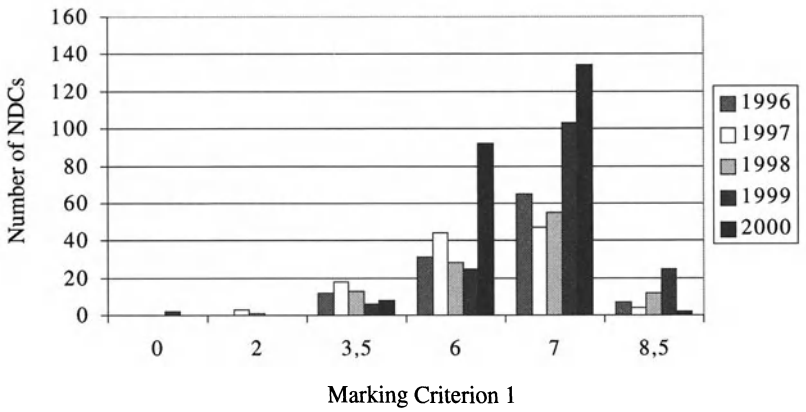


Fig. 12.1. Nature of relations with the customer.

The relations with the customer seem often to be good; the mark assigned most often is "7". However, this is not a sufficient reason, in a time of crisis, for a customer to grant privileged treatment to commercial offers from EADS LV (see Fig. 12.2).

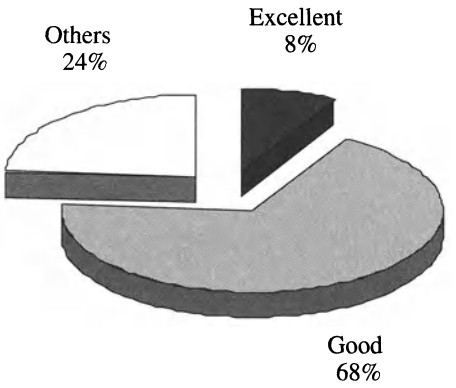


Fig. 12.2. Number of deals lost as a function of relations with the customer.

Credibility of the call for bids (criterion 4)

The situation for the criterion 4 "credibility of the call for bids" is similar to the situation for the preceding criterion. The enterprise, whose customer portfolio is very stable, has never faced a doubtful call for bids in six years.

Availability of the teams (criterion 6)

The projected load plan of the technical and commercial teams gives, in theory, the information necessary for the marking. However, the distribution of the marking since 1996 for this criterion, represented in Fig. 12.3, shows the difficulty of making this evaluation.

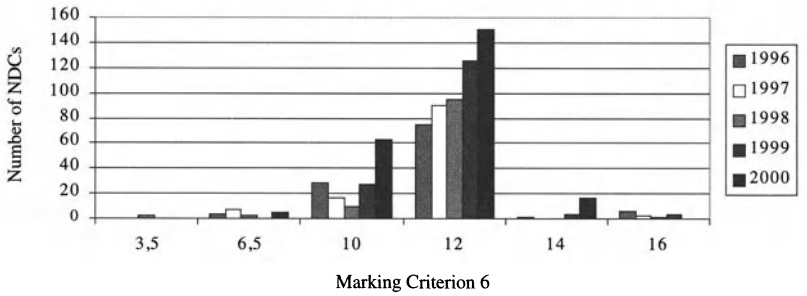


Fig. 12.3. Frequencies of the marks for criterion 6.

This criterion is implicated for two reasons. On the one hand, it is difficult to evaluate long before the time the load plan of the technical and commercial teams, and, on the other hand, it is very unlikely that a deal will be refused because of a work overload of the teams. Further, if there is a question about the survival of an enterprise in crisis, it is always possible to obtain a deal by making a team available, either by increasing the working hours or by working during weekends and public holidays.

Expected expenses on studies and in production (criterion 9)

The analysis of the marking of the deals since 1996 on this criterion is represented in Fig. 12.4 and shows that the majority of the deals generated small expenses (less than 50 000 hours).

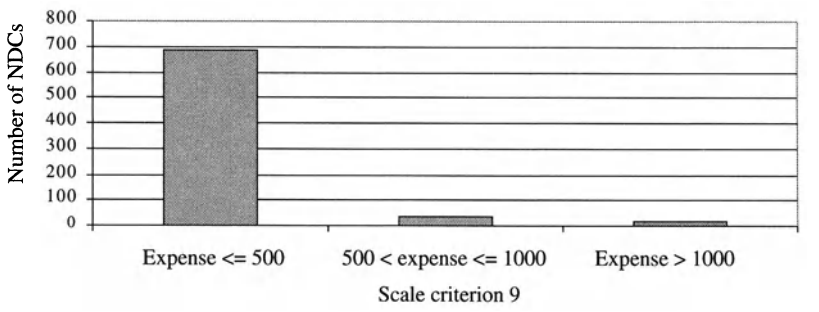


Fig. 12.4. Frequencies of the marks for criterion 9.

Favoring the "big" deals remains an element in the decision process, but it has no place in the model, since it is a pious hope that cannot be transformed into reality. After all, it is difficult to bias the size of an ideal deal a priori since, in a time of crisis, all deals are good.

12.3.2 Ineffective criteria due to the marking method

Competences of EADS LV and its partners (criterion 2)

In the process described above, the people involved evaluate their competences themselves, but the competition is the only objective measurement, which raises the following questions:

- Who assigns marks in accordance with the criterion?
- Who is able, internally, to mark objectively?
- Is it in our interest to mark ourselves with the help of an entity external to the enterprise?

It is illusory to measure such a criterion; our attempt to do so arises from a naive and academic view of the problem, which leads necessarily to major marking mistakes. If we are to keep such a criterion, we must necessarily solve the marking problem.

Competitiveness of the EADS LV price against the market price (criterion 3)

Owing to the complexity and the lack of transparency of this subject, both inside and outside the enterprise, it is extremely difficult to mark in accordance with this criterion with accuracy; the market price is often difficult to determine. Without any possibility of finding a marking ratio which could serve as a real computation tool, this criterion has been abandoned in favor of a broader but measurable technico-economic criterion.

Strategic will (criterion 5)

The strategic plan of EADS LV does not contain, unlike the case for many other companies, detailed qualitative elements related to the priorities of the enterprise. Moreover, the technical and commercial teams are always persuaded of the high priority of their activity (see Fig. 12.5), and so we certainly must not ask them to assess such a criterion. We may notice that the strategic will of the enterprise has nothing in common, to a first approximation, with the probability of winning a deal. However, even in the worst case, strategic will creates a favorable environment and produces investments, which have an influence in the medium to long term.

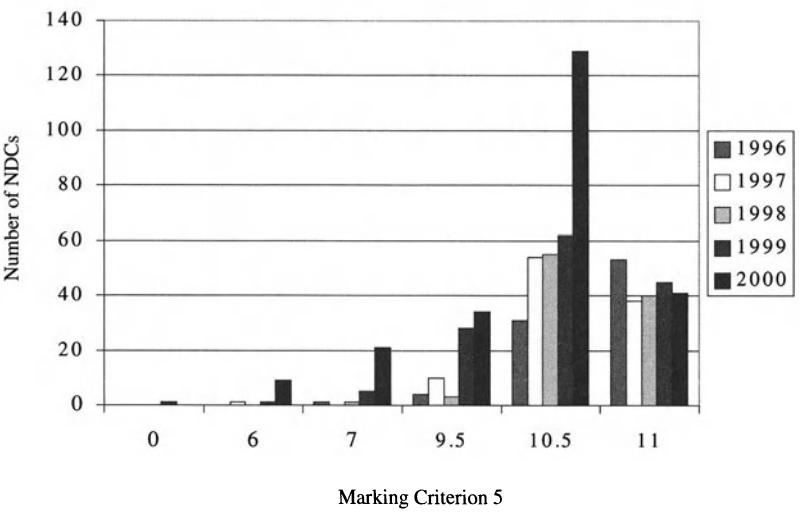


Fig. 12.5. Strategic will, taking R&D into account.

12.3.3 Criteria which are in fact constraints

Self–financed research effort and investment effort (criterion 7)

About 20 deals each year were affected by this criterion; the self–financed research effort and investment effort was in the range between 0 and 1.5 M€. This criterion is represented in Fig. 12.6.

This criterion is not discriminating since, for real strategic deals, an enterprise can always find a means to meet the needs of a customer, notably through R&D investment. It is, in fact, a constraint that we have to take into account, and not a criterion.

EADS LV effort (criterion 8)

This criterion is not discriminating, and can be transformed into a constraint which keeps the financial effort, deal after deal, within a reasonable interval in relation to the expected sales figure. As the graph in Fig. 12.7 shows, this rule is known by the actors in the process, so it became a constraint satisfied by all deals.

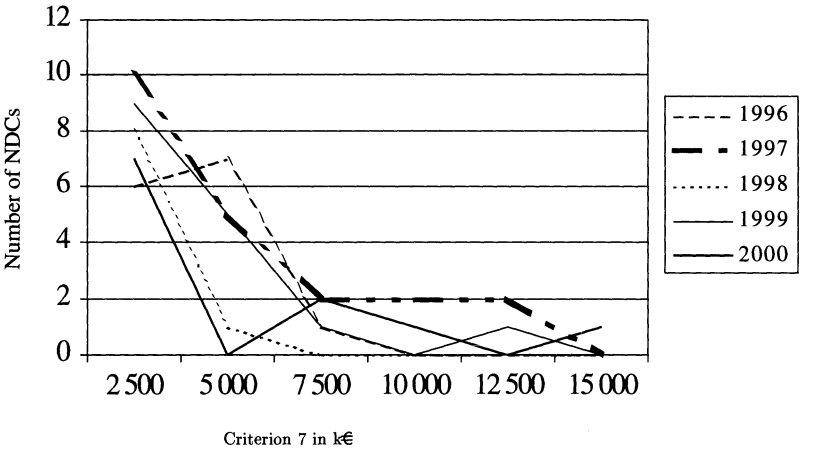


Fig. 12.6. Self-financed research effort.

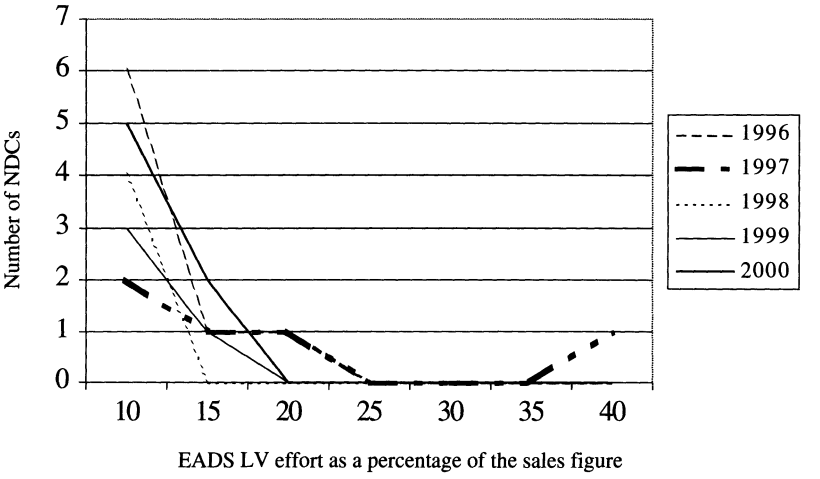


Fig. 12.7. Frequencies of the marks for criterion 8.

12.4 Second-generation model

The most fundamental evolution of the model was the modification of the question which the EADS LV decision aid model tried to answer. The new model does not try to answer the question: "Should we respond to a call for bids?" but the question "Will we win this call for bids?" This essential modification was justified by the fact that the enterprise always judges the performance of the model with respect to an established result (calls for bids won or lost).

The new problem is more restrictive than the previous one and the strategic and economic interests are not represented anymore. These interests are, of course, taken into account by the members of the committee for new deals, but the model does not pretend to give an answer to all of the questions at the same time. The answer to the question of whether or not we should respond is the responsibility of the committee,

which may or may not take into account the predictions of the model at its weekly meeting.

12.4.1 Principle of the rebuilding of the model

The new model was built using the following principles:

- Do not use experts who are involved in the marking and are also judges.
- For the marking, use criteria based on accumulated historical factual information, the accuracy of which increases with the volume of deals treated, and so with time.
- Replace qualitative ideas and avoid, as far as possible, purely qualitative criteria.
- Use the experience gathered since the creation of the model to construct new scales. All the characteristics of the deals have been saved in a database since 1996. The knowledge about the distribution of the marks in past years must allow us to define better levels, either with respect to their absolute position or with respect to their range of variation.
- Provide easy access to the information needed for the marking. This will allow more efficient marking and accelerate the marking process.

As far as the scales and their levels are concerned, the principles applied were:

- Fewer levels, but with a larger spread, and it should be easy to distinguish between them.
- When we have recourse to qualitative criteria in spite of everything, it is essential to use a two–dimensional matrix, to avoid the danger of direct marking. This allows us to use two elements (one for each dimension), and thus marking gives an answer to two exact questions, instead of requiring an intuitive, hazardous estimation.

12.4.2 Abandonment of the principle of the uniqueness of the model

The new deals are very disparate; deals exist in specific environments (determined by the commercial situation, the competition, and the institutional or industrial customer), and concern various areas (strategic missiles, satellites, rockets, equipment, etc.) and various activities (supply of equipment, services, studies, etc.).

It became evident during the years of use of the model that an approach which used a unique, omnipotent criterion was too primitive and was limited in the possibilities it offered for improvements. It was decided to create families of separate models, each of which has a specific tuning and/or special criteria chosen to fit various activities of the enterprise. By use of the commercial plan, a segmentation of the activities of the enterprise was successfully performed and a by–segment model was created for the products and services. The qualitative criteria were replaced by quantitative criteria based on accessible factual data and measured by scores recorded by the enterprise, segment by segment.

This approach was used to build, first of all, a model with four criteria. However, this model did not explain the totality of the vast sample used. A deeper study, based on rules related to coherent families of criteria developed by Bernard Roy, allowed us to identify a fifth criterion. The samples used for the tuning of the model consist of all available deals for a given segment, and there is no limit on the expected accuracy of such a model, which could be improved by an increase in the number of criteria or by refining the measurement ratios used in the criteria.

12.4.3 Architecture of the families of models

So as to simplify the use of the decision aid tools, a limited number of models were used:

- the ELECTRE TRI 2001 Products model; the various bids are applied to NDCs dedicated to product supply;
- the ELECTRE TRI 2001 Services model, the various bids are applied to NDCs dedicated to services (studies, tests, etc.).

The five criteria for the ELECTRE TRI 2001 Products model are:
- criterion 1: technico-economic position of the product;
- criterion 2: technological control and maturity of the product;
- criterion 3: customer position;
- criterion 4: financial credibility of the deal with respect to the customer;
- criterion 5: position with respect to the competition.

The positions are evaluated by plotting the number of deals lost on the abscissa and the number of deals won on the ordinate. This information, gathered since 1996, allows us to determine the positions of customers, products and segments. A position is related to a given set of deals, which is chosen with respect to:

- customers: representation of the commercial performance for various customers;
- segments: representation of the commercial performance of a segment;
- products: representation of the commercial performance of a product or of a family of products.

The mark is computed using the position of the customer, segment or product. These positions allow us to build a quantitative evaluation on objective facts, which is one of the constraints imposed on the new model.

12.4.4 Criteria

Technico-economic position of the product

This criterion was built on factual data, and allows us to model the technico-economic competitiveness of the product and the history of the relationship with the customer. It is based on the new deals whose definitive status (lost or won) is known (deals inventoried since 1996).

For each product, family of products or customer, a graph of the kind illustrated in Fig. 12.8 is plotted. Here, the number of new deals lost is plotted on the abscissa and the number of new deals won is plotted on the ordinate. The area of the disk associated with a product or customer is proportional to the sales figure earned from that product or customer.

Technological control and maturity of the product

This criterion allows us to evaluate the level of development of the product and its technological advance, and to deduce its technical competitiveness. A nomenclature for products has been established and prescribed inside the enterprise to characterize each product, which is positioned on a life curve. Each position on this curve corresponds to a step in the development of the product or in its commercial life. In the

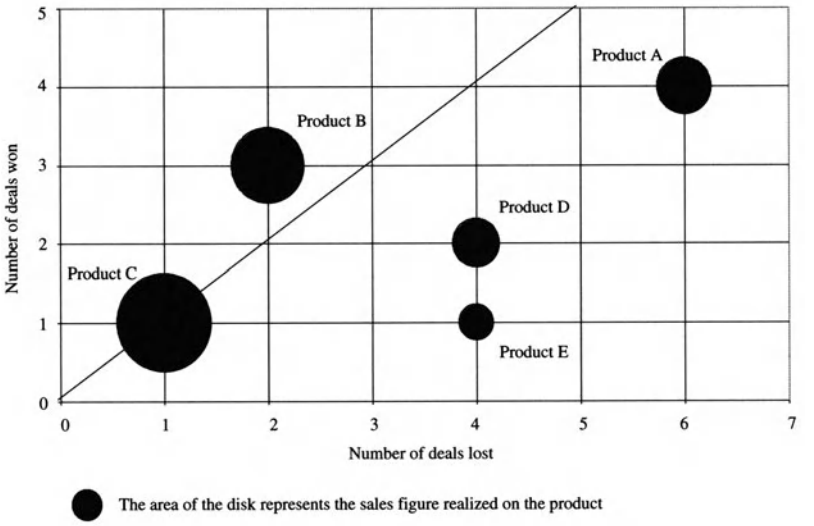


Fig. 12.8. Product position (illustration of the shape of the graph).

field of space products, production volumes are small and products are often made by hand, and thus flight qualification is a very important step for these products. A mark is established with respect to the position of the product on the curve, as shown in Fig. 12.9.

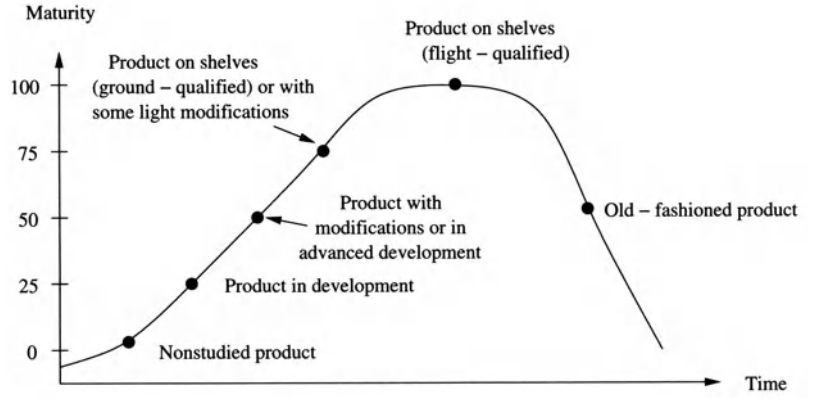


Fig. 12.9. Life curve of a product (illustration of the shape of the graph).

Customer position

This criterion is constructed in a similar way to the criterion of the technico-economic position of the product.

Financial credibility of the deal with respect to the customer

This criterion takes into account, as far as possible, the cancellation risk of a project. Because EADS LV has no influence, in general, on the cancellation risk, the chosen criterion is the financial credibility of the customer in relation to the deal, which takes into account the global credibility of the project.

Thus, the credibility of the project depends on two financial factors, which are used as the two dimensions of the marking matrix.

It was noted during studies of deals related to high-pressure tanks that half of the lost deals were deals which were given up because of customers that had not completed their project. This was especially the case for a satellite constellation, an ambitious project which was more or less abandoned. So it is important to have a criterion which allows us to evaluate the credibility of the project. The reasons for cancellation are numerous and the evaluation of the situation is difficult.

We distinguish the following factors:

- The credibility of the customer: the "given up" deals are already taken into account by the customer position criterion.
- The probability of the intervention of an external event: a political decision or a budget reallocation may call a deal into question. These events are, by their nature, extremely difficult to predict, but factors that point towards projects liable to be affected by such events can be identified.
- Insurmountable technical difficulties: an ambitious project can be cancelled because it is unrealizable from a technical point of view within the budget allocation.
- Subcontracting: EADS LV can be subcontracted by a project manager enterprise. When the NDC is issued, the main contract has not been won. If the project manager loses the deal, EADS LV, as a subcontractor, also loses the deal.
- A deal made very quickly: our experience has shown that the cancellation probability does not evolve significantly with time. A project has as much chance of being cancelled at the start of the project as at the end of the project.
- The context of the deal can give an indication about the credibility of the deal. What to do if the context is new?

Position with respect to the competition

This criterion is an adjustment criterion, which allows us to modify the severity of the model, and so to shape the predictions of the model to tally with the commercial reality in two types of situations: calls for bids in competition and by mutual consent which have very different success rates. The principle that we have followed for the marking is a matrix principle tuned with recent measured quantitative values, which are used in the "instrument panel" that measures the commercial performance of the enterprise.

12.4.5 Tuning of the model

The method proved to be valuable and well suited to its purpose.

The models of the first and second generation use the same ELECTRE TRI method, developed by LAMSADE at Paris-Dauphine University, together with its associated software, to deal with the marks computed using the criteria. The ELECTRE TRI method is described elsewhere in this book. The sorting problem, which

consists in assigning actions to categories by examining the intrinsic value of the action, allows us to deal with one action or one call for bids at a time.

Optimization methods often use a utility function (or a unique synthetic criterion) which allows one to sort the actions into an order from best to worst. These methods are based on a strong hypothesis, and we can be satisfied with a less rich result obtained by a method that avoids a strong mathematical hypothesis and an obligation to ask hard questions of the decision maker. To obtain such a result, the concept of overranking (a set of overlapping relations composed of concordance and nondiscordance) proposed by Roy was adapted to the sorting problem. The transformation of the valued overranking relation into a perfect overranking relation S was performed by using a cut level 1 of 0.75. A pessimistic procedure, which consists in assigning the action a to the highest category Ci such that the action a overranks the lowest reference action bi (the profiles) of the category concerned, was used. The whole set of possibilities of the ELECTRE TRI method has not been used: the preference thresholds p and the concordance thresholds q are equal, and the veto threshold has not been used.

12.4.6 Performance of the model

The assessment of the performance in comparison with the reality of deals won and lost is complex, since we need to reason and work segment by segment and to consider the four types of predictions, that is to say, the four categories of yes, doubtful yes, doubtful no and no.

In simple terms, the accuracy is noticeable; with the help of the ELECTRE TRI 2001 model with five criteria, the won–lost scores can be represented with an accuracy from 96% to 100% for the categories C1 (no) and C4 (yes).

The score obtained for the fluids segment, which is presented as an example in Fig. 12.10, shows an accuracy of 100% for the C1 (no) and C2 (doubtful no) categories, an error rate of 12.5% for the C3 (doubtful yes) category and an accuracy of 100% for the C4 (yes) category.

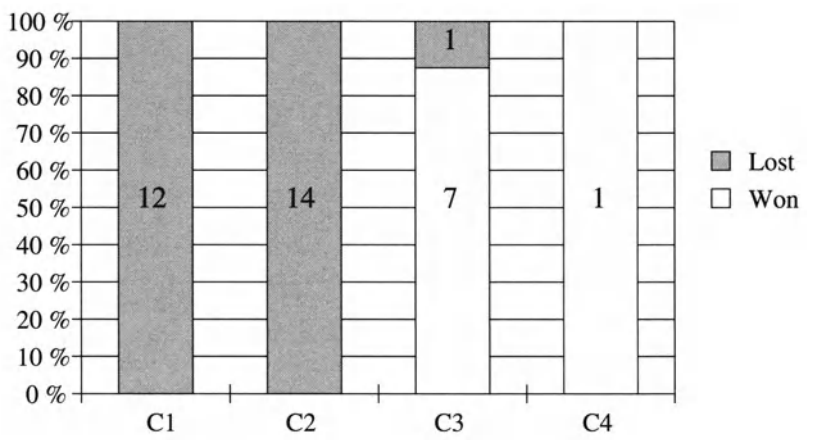


Fig. 12.10. Fluids segment: results for a sample.

12.5 Conclusion

The control of the commercial data needed to bring this model into service has allowed us to produce a commercial "instrument panel" monthly, thus contributing to a better management of the enterprise.

This "instrument panel" is composed of:

- consolidated and by–segment statistics about deals won and lost;
- 17 positions of products or customers;
- a number of relevant ratios.

An example of these data is represented in Fig. 12.11.

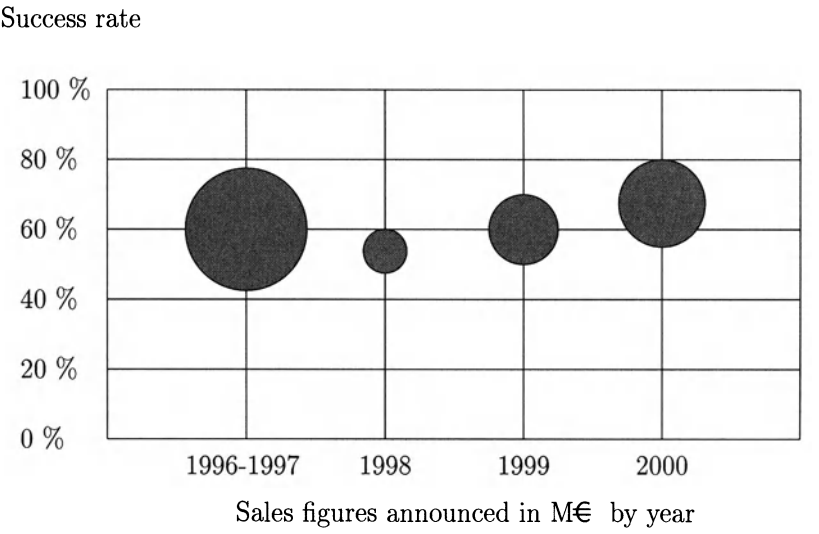


Fig. 12.11. An example of ratio, the commercial success rate, with announced sales figures in M€ (represented by the areas of the circles).

The model has had a great impact on the development of the strategic plan of the enterprise, by introducing a list of priorities for all products and services. It has also contributed to prescribing a segmentation of the activities, which has become the skeleton of the commercial plan, and has prescribed a restructuring of the product nomenclature.

The future evolution of the model is expected to be directed towards homogeneity of the accuracy in all segments, a possible modification of the equivalence families, division of the problems using a richer modeling and decentralization of the decision aid tools within the enterprise. As with expert systems, most of the ratios used in the marking of the criteria reflect accumulated experience. Hence, each new situation, after a phase of prediction error, is automatically integrated into subsegment decisions. For recurring products, it is possible to reconstruct the past, and to predict the future with high accuracy in the case of a linear evolution.

As a consequence of role of these decision aid tools in bringing information together and in planning, which leads the enterprise to build quantitative databases favorable to

the management of the enterprise, generalization of these tools to all budget allocation problems is predictible in the medium time.

Decision aids for budget allocation in the context of commercial offers have a promising future, because the potential economies are considerable and are not unique to the field of space. The latter is especially true because, as a consequence of abuses in recent years, the tendency is clearly towards the generalization of the use of calls for bids for placing contracts, particularly within the European Union.

13

Conclusion

As we have seen throughout this book, multiobjective optimization is not an easy task.

The task begin with the definition of a multiobjective optimum. A general well-established definition is the Pareto definition of a multiobjective optimum. But this definition is too general, and modifications have been introduced: lexicographic optimality, maximal optimality, etc.

Then comes the solution mode. Depending on our knowledge of the problem, we can:

- seek an unique solution from the start if the preferences of the decision maker are well determined and if he/she does not wish to choose from a set of solutions;
- clarify the preferences of the decision maker during the running of the optimization;
- approximate the tradeoff surface using a set of solutions.

We also have many possibilities in the choice of the treatment of the multiobjective problem. We can reduce the problem to a problem with one objective function or deal with the problem vectorially.

Lastly, we have to choose an optimization method to solve the multiobjective problem. This last task is difficult even in mono-objective optimization, so we can understand why, in multiobjective optimization, the problem is harder.

Nevertheless, the game is worth playing. Multiobjective optimization offers the user more degrees of freedom to model the problem. It also emphasizes some difficulties with the modeling that we often avoid in mono-objective optimization. Before solving a mono-objective problem, we often perform a tradeoff between several objective functions, without the knowledge of the builder of the model. Multiobjective optimization makes this tradeoff visible, and this is also the difficulty of multiobjective optimization: a difficulty produced by a tradeoff that is hard to solve. We must remember that multiobjective optimization will not solve this problem with the wave of a magic wand. It is always down to the decision maker to perform the tradeoff and to choose one solution in preference to another. Multiobjective optimization offers tools to help the decision maker to make this choice, however.

Several areas within the field of multiobjective optimization are receiving particular attention at present:

- **Control of the diversity of the solutions on the tradeoff surface.**

In the a posteriori resolution methods, we try to obtain an approximation to the tradeoff surface. A good approximation corresponds to a set of points uniformly spread over the tradeoff surface. Several methods (some of them still under developments) are aimed at reaching this goal. We can also try to obtain a good diversity of solutions in the parameter space.

- **Development of multiobjective optimization methods that allow one to obtain quickly an approximation to the tradeoff surface.**
  There are many degrees of freedom which we can make use of to improve the speed of convergence of a multiobjective optimization method. For example, we can:
  - use a different metaheuristic (simulated annealing, genetic algorithms, etc.);
  - use a different multiobjective treatment method;
  - use a more suitable neighborhood for the problem we are trying to solve.

- **Finalization of methods which integrate the preferences of a decision maker to perform an *a priori* solution.**
  For some years, we have seen papers being published that deal with the a priori solution of multiobjective problems, using tools from preference theory or decision aids. We expect that multiobjective optimization and decision aids will join together soon.

- **Study of data visualization methods which allow one to understand the results of a multiobjective optimization better.**
  Not many visualization methods dedicated to the representation of the results of a multiobjective optimization exist. However, several methods from the field of statistics are available, for example:
  - principal-components analysis;
  - nonlinear projection.

- **Creation of new metrics which allow one to assess, during the running of the optimization, the quality of the intermediate results.**
  Some metrics, although very useful, have a restricted domain of application. Some others need information which is impossible to obtain in the case of a "real" problem, which restricts their use to test problems. Some techniques from economics are applicable to multiobjective optimization, for example, data envelopment analysis.

Multiobjective optimization will develop in the near future as is testified by the increasing number of publications and new conferences dedicated to multiobjective optimization. This development will be accompanied by the arrival of multiobjective optimization in enterprises: enterprises are being increasingly confronted with the requirement to square the circle in the form of a tradeoff between cost, security and quality.

References

[Alexandrov et al. 94] N. Alexandrov, J. E. Dennis, *Algorithms for Bilevel Optimization*, Technical report TR-94-77, September 1994, `http://www.icase.edu/Dienst/UI/2.0/Describe/ncstrl.icase/TR-94-77`.

[Altenberg 97] L. Altenberg, *Fitness distance correlation analysis: an instructive counterexample*, Proceedings of the Seventh International Conference in Genetic Algorithms (ICGA97), San Francisco, 1997, `http://pueco.mhpcc.edu/~altenberg`.

[Altschuler et al. 94] E. L. Altschuler, T. J. Williams, E. R. Ratner, F. Dowla, F. Wooter, *Method of constrained global optimization*, Physical Review Letters, 1994.

[Andreatta et al. 97] A. A. Andreatta, S. E. R. Carvalho, C. C. Ribeiro, *A Framework for the Development of Local Search Heuristics for Combinatorial Optimization Problems*, PhD thesis, Catholic University of Rio de Janeiro, 1997, `http://citeseer.nj.nec.com/andreatta97framework.html`.

[Andreatta et al. 98] A. A. Andreatta, S. E. R. Carvalho, C. C. Ribeiro, *An Object Oriented Framework for Local Search Heuristics*, Technical report, University of Rio de Janeiro, 1998, `http://citeseer.nj.nec.com/151568.html`.

[Argaud 92] J. P. Argaud, *Une méthode d'optimisation des plans de rechargement pour la gestion du combustible nucléaire*, Internal report, EDF-DER, number HI-072/96/013/0, January 1992.

[Ascend] Ascend software package, `http://www-2.cs.cmu.edu/~ascend`

[Asselmeyer 96] T. Asselmeyer, W. Ebeling, *Unified Description of Evolutionary Strategies over Continuous Parameter Spaces*, May 1996, `http://citeseer.nj.nec.com/asselmeyer96unified.html`.

[Ausiello et al. 99] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, M. Protrasi, *Complexity and Approximation Combinatorial Optimization Problems and their Approximability Properties*, Springer, Berlin, Heidelberg, 1999.

[Azarm 96] S. Azarm, *Multiobjective Optimum Design*, 1996, `http://www.glue.umd.edu/~azarm/optimum_notes/multi/multi.html`

[Azarm et al. 99] S. Azarm, B. J. Reynolds, S. Narayanan, *Comparison of two multiobjective optimization techniques with and within genetic algorithms*, Proceedings of the 1999 ASME Design Engineering Technical Conference, September 1999.

[Baldick et al.] R. Baldick, A. B. Kahng, A. Kennings, I. L. Markov, *Efficient Optimization by Modifying the Objective Function: Applications*

to Timing-Driven VLSI Layout, http://citeseer.nj.nec.com/446578.html.

[Barda 87] O. H. Barda, *Etude comparative des méthodes multicritères dans le cadre d'un problème de localisation*, Report, Paris-Dauphine University, 1987.

[Barnett 97] L. Barnett, *Tangled Web: Evolutionary Dynamics on Fitness Landscapes with Neutrality* MSc thesis, School of Cognitive Sciences, University of Sussex, Brighton, 1997.

[Bartak 99] R. Bartàk, *Constraint programming: in pursuit of the holy grail*, Proceedings of WDS99, Prague, June 1999, http://kti.ms.mff.cuni.cz/~bartak/html/publications.html.

[Battiti 94] R. Battiti, G. Tecchiolli, *The reactive tabu search*, ORSA Journal on Computing, volume 6, number 2, pages 126–140, 1994.

[Ben-Tal et al. 96] A. Ben-Tal, J. Tind, *Duality with Multiple Criteria and Multiple Resources*, 1996, http://citeseer.nj.nec.com/ben-tal96duality.html.

[Boese et al. 94] K. D. Boese, A. B. Kahng, *Best-so-far vs. where-you-are: implications for optimal finite-time annealing*, Systems and Control Letters, volume 22, number 1, pages 71–80, January 1994, http://citeseer.nj.nec.com/56460.html.

[Boese et al.] K. D. Boese, A. B. Kahng, S. Muddu, *A new adaptative multi-start technique for combinatorial global optimizations*, Operation Research Letters, volume 16, number 2, 1994, pages 101–113, http://nexus6.cs.ucla.edu/~boese/research.html.

[Bonomi 88] E. Bonomi, J.-L. Lutton, *Le recuit simulé*, Pour la Science, number 129, pages 68–77, July 1988.

[Borges et al. 00] C. C. H. Borges, H. J. C. Barbosa, *A non-generational genetic algorithm for multiobjective optimization*, 2000 Congress on Evolutionary Computation, volume 1, pages 172–179, July 2000.

[Bourles] H. Bourles, *Optimisation et commande optimale*, Lecture notes, ENS Cachan, 1997.

[Bowlin 98] W. F. Bowlin, *Measuring performance: an introduction to data envelopment analysis (DEA)*, Journal of Cost Analysis, pages 3–27, 1998.

[Boyan et al. 97] J. A. Boyan, A. W. Moore, *Using prediction to improve combinatorial optimization search*, Sixth International Workshop on Artificial Intelligence and Statistics (AISTATS), 1997, http://www.cs.cmu.edu/~jab/cv/cv50.html#pubs.

[Brans et al. 86] J. P. Brans, Ph. Vincke, B. Mareschal, *How to select and how to rank projects: The PROMETHEE method*, European Journal of Operational Research, volume 24, pages 228–238, 1986.

[Bruyere et al. 86] M. Bruyère, A. Vallée, Ch. Collette, *Extended Fuel Cycle Length*, Framatome Note, FRADOC 11, September 1986.

[Burke et al. 92] L. I. Burke, J. P. Ignizio, *Neural networks and operations research: an overview*, Computer Operations Research, volume 19, number 3/4, pages 179–189, 1992.

[Cardon et al. 00] A. Cardon, T. Galinho, J.-P. Vacher, *Genetic algorithms using multi-objectives in a multi-agent system*, Robotics and Autonomous Systems, volume 33, pages 179–190, 2000.

[Clerc 00] M. Clerc, *Discrete Particle Swarm Optimization Illustrated by the Travelling Salesman Problem*, February 2000, http://www.mauriceclerc.net.

[Charon et al. 96] I. Charon, A. Germa, O. Hudry, *Méthodes d'optimisation combinatoire*, Masson, 1996.

[Chekroun 85] E. Chekroun, *Aide à la décision*, Sirey, 1985.

[Coello 96] C. A. Coello Coello, *An Empirical Study of Evolutionary Techniques for Objective Optimization in Engineering Design*, PhD thesis, Department of Computer Science, Tulane University, New Orleans, April 1996, http://www.lania.mx/~ccoello/EMOO.

[Coello 98] C. A. Coello Coello, *An Updated Survey of G.A. Based Multiobjective Optimization Techniques*, Technical report Lania-RD-98-08, Laboratorio Nacional de Informática Avanzada, Xalapa, Veracruz, Mexico, December 1998, http://www.lania.mx/~ccoello/EMOO.

[Coello 99] C. A. Coello Coello, *A Survey of Constraint Handling Techniques Used with Evolutionary Computation Methods*, Technical report Lania-RI-99-04, Laboratorio Nacional de Informática Avanzada, Xalapa, Veracruz, Mexico, 1999, http://www.lania.mx/~ccoello/EMOO.

[Coello et al. 98] C. A. Coello Coello, A. D. Christiansen, *Two new GA-based methods for multiobjective optimization*, Civil Engineering Systems, volume 15, number 3, pages 207–243, 1998, http://www.lania.mx/~ccoello/EMOO.

[Coello et al. 99] C. A. Coello Coello, A. D. Christiansen, *MOSES: a multiobjective optimization tool for engineering design*, Engineering Optimization, volume 31, number 3, pages 337–368, 1999, http://www.lania.mx/~ccoello/EMOO.

[Coello et al. 00] C. A. Coello Coello, *Handling preferences in evolutionary multiobjective optimization: a survey*, 2000 Congress on Evolutionary Computation, volume 1, pages 30–37, July 2000, http://www.lania.mx/~ccoello/EMOO.

[Coello 01] C. A. Coello Coello, *A short tutorial on evolutionary multiobjective optimization*, In E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, and D. Corne (eds.), First International Conference on Evolutionary Multi-Criterion Optimization, pages 21–40, Lecture Notes in Computer Science 1993, Springer, Berlin, Heidelberg, 2001, http://www.lania.mx/~ccoello/EMOO/EMOObib.html.

[Coleman et al. 93] T. Coleman, D. Shalloway, Z. Wu, *Isotropic effective energy simulated annealing searches for low energy molecular cluster states*, Computational Optimization and Applications, number 2, pages 145–170, 1993, http://citeseer.nj.nec.com/coleman92isotropic.html.

[Collette et al. 00] Y. Collette, P. Siarry, H.-I. Wong, *A systematic comparison of performance of various multiple objective metaheuristics using a common set of analytical test functions*, Foundations of Computing and Decision Sciences, volume 25, number 4, pages 249–272, 2000.

[Collins 96] T. Collins, *Genotype-Space Mapping: Population Visualization for Genetic Algorithms*, Technical report KMI-TR-39, Knowledge Media Institute, September 1996.

[Cooper et al. 00] W. W. Cooper, L. M. Seiford, K. Tone, *Data Envelopment Analysis: Comprehensive Text with Models, Applications, References and DEA-Solver Software*, Kluwer Academic, Dordrecht, 2000.

[Corne et al. 00] D. W. Corne, J. D. Knowles, M. J. Oates, *The Pareto envelope-based selection algorithm for multiobjective optimization*, In 6th International Conference on Parallel Problem Solving from Nature (PPSN VI), pages 839–848, September 2000, Paris.

[C.S.E.P. 95] The Computer Science Education Project, *Mathematical Optimization*, Electronic book, 1995, `http://csep1.phy.ornl.gov/csep.html`.

[Cvetkovic et al. 99] D. Cvetkovic, I. C. Parma, *Use of preference G.A.-based multi-objective optimization*, Proceedings of the Genetic and Evolutionary Conference (GECCO99), volume 2, pages 1504–1509, Orlando, Florida, July 1999.

[Cvetkovic 00] D. Cvetkovic, *Evolutionary Multi-Objective Decision Support Systems for Conceptual Design*, PhD thesis, School of Computing, Faculty of Technology, University of Plymouth, November 2000.

[Daniels 78] R. W. Daniels, *An Introduction to Numerical Methods and Optimization Techniques*, North-Holland, New-York, 1978.

[Darwen et al. 95] P. Darwen, X. Yao, *A dilemma for fitness sharing with a scaling function*, Proceedings of the Second IEEE International Conference on Evolutionary Computation, Piscataway, New Jersey, pages 166–171, 1995.

[Das et al. 98] I. Das, J. Dennis, *Normal boundary intersection: a new method for generating Pareto optimal points in multicriteria optimization problems*, SIAM Journal on Optimization, volume 8, number 3, pages 631–657, 1998, `http://www.caam.rice.edu/~indra/researchwork.html`.

[Das 99] I. Das, *A preference ordering among various Pareto optimal alternatives*, Structural Optimization, volume 18, number 1, pages 30–35, 1999, `http://www.caam.rice.edu/~indra/researchwork.html`.

[Dauer et al. 80] J. P. Dauer, R. J. Krueger, *A multiobjective optimization model for water resources planning*, Applied Mathematical Modelling, pages 171–175, June 1980.

[Davis 91] L. Davis, *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, 1991.

[Deb 98a] K. Deb, *Multiobjective Genetic Algorithms: Problem Difficulties and Construction of Test Problems*, Technical report CI-49/98, Department of Computer Science, University of Dortmund, 1998.

[Deb 98b] K. Deb, *Non-Linear Goal Programming Using Multi-Objective Genetic Algorithms*, Technical report CI-60/98, Department of Computer science,University of Dortmund, October 1998.

[Deb 99] K. Deb, *An overview of multi-objective evolutionary algorithms*, Journée J.E.T., Slides, May 1999.

[Deb et al. 00a] K. Deb, T. Goyal, *Controlled Elitist Non-Dominated Sorting Genetic Algorithms for Better Convergence*, Technical report, KanGAL report 200004, Indian Institute of Technology, Kanpur, 2000.

[Deb et al. 00b] K. Deb, A. Pratap, T. Meyarivan, *Constrained Test Problems for Multi-Objective Evolutionary Optimization*, Technical report, KanGAL 200005, Institute of Technology of India, Kanpur, 2000.

[Deb et al. 00c] K. Deb, E. Zitzler, L. Thiele, *Comparison of Multiobjective Evolutionary Algorithms: Empirical Results*, Technical report Computer Engineering and Communication Networks Laboratory (TIK), Swiss Federal Institute of Technology, Zurich, 2000.

[Deb 01] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, Wiley, 2001.

[Deb et al. 01a] K. Deb, T. Goel, *A hybrid multi-objective evolutionary approach to engineering shape design*, In E. Zitzler, K. Deb, L.

Thiele, C. A. Coello Coello, and D. Corne (eds.), First International Conference on Evolutionary Multi-Criterion Optimization, pages 385–399, Lecture Notes in Computer Science 1993, Springer, Berlin, Heidelberg, 2001, http://www.lania.mx/~ccoello/EMOO/EMOObib.html.

[Deb et al. 01b] K. Deb, A. Pratap, T. Meyarivan, *Constrained test problem for multi-objective evolutionary optimization*, In E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, and D. Corne, editors, First International Conference on Evolutionary Multi-Criterion Optimization, pages 284–298, Lecture Notes in Computer Science 1993, Springer, Berlin, Heidelberg, 2001, http://www.lania.mx/~ccoello/EMOO/EMOObib.html.

[Deb et al. 01c] K. Deb, T. Goel, *Controlled elitist non-dominated sorting genetic algorithms for better convergence*, In E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, and D. Corne (eds.) , First International Conference on Evolutionary Multi-Criterion Optimization, pages 67–81, Lecture Notes in Computer Science 1993, Springer, Berlin, Heidelberg, 2001, http://www.lania.mx/~ccoello/EMOO/EMOObib.html.

[Deb et al. 01d] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, *A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II*, In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo and H.-P. Schwefel (eds), Proceedings of the Parallel Problem Solving from Nature VI Conference, pages 849–858, Springer, Berlin, Heidelberg, 2000, http://www.lania.mx/~ccoello/EMOO/EMOObib.html.

[Delbos 75] M. Delbos, *Le positionnement multidimensionnel*, Internal report, EDF-DER, number HI/1943/02, September 1975.

[Deng et al. 96] H. L. Deng, W. Gouveia, J. Saales, *The CWP Object-Oriented Optimization Library*, 1996, http://timna.mines.edu/cwpcodes/coool.

[Diamantaras et al. 99] K. I. Diamantaras, K. Hornik, M. G. Strintzis, *Optimal linear compression under unreliable representation and robust PCA neural models*, IEEE Transactions on Neural Networks, volume 10, number 5, pages 1186–1195, 1999.

[Di Gaspero et al. 00] L. Di Gaspero, A. Schaerf, *EasyLOCAL++: an Object Oriented Framework for Flexible Design of Local Search Algorithms*, Technical report UDMI/13/2000/RR, 2000, http://citeseer.nj.nec.com/digaspero00easylocal.html.

[Dorigo et al. 96] M. Dorigo, V. Mariezzo, A. Colorni, *The ant system: optimization by a colony of cooperating agents*, IEEE Transactions on Systems, Man, and Cybernetics, part B, volume 26, number 1, pages 1–13, 1996.

[Doye et al.] J. P. K. Doye, D. J. Wales, *On the thermodynamics of global optimization*, Physical Review Letters, volume 8, pages 1357–1360, 1998, http://www-wales.ch.cam.ac.uk/~jan/abstracts/prl98.html.

[Dozier et al. 98] G. Dozier, S. McCullough, A. Homaifar, L. Moore, *Multi-objective evolutionary path planning via fuzzy tournament selection*, IEEE International Conference on Evolutionary Computation, pages 684–689, Piscataway, New Jersey, May 1998.

[Dreyfus et al. 01] G. Dreyfus, J.-M. Martinez, M. Samuelides, M. B. Gordon, F. Badran, S. Thiria, L. Hérault, *Réseaux de neurones: méthodologie et applications*, Eyrolles, 2001.

[Duch et al. 98] W. Duch, J. Korczak, *Optimization and global minimization methods suitable for neural networks*, Neural Computing Survey, 1998, `http://citeseer.nj.nec.com/duch98optimization.html`.

[Duch 99] W. Duch, *Alternative to gradient based neural training*, In Fourth Conference on Neural Networks and their Applications, Zakopane, Poland, May 1999, `http://citeseer.nj.nec.com/duch99alternatives.html`.

[Duvivier et al. 96] D. Duvivier, Ph. Preux, E.-G. Talbi, *Climbing up NP hard hills*, Fourth International Conference on Parallel Problem Solving from Nature, September 1996.

[Edelman 97] S. Edelman, N. Intrator, *Learning as Extraction of Low-Dimensional Representations*, Mechanisms of Perceptual Learning, Psychology of Learning and Motivation series, pages 574–583, Academic Press, Berlin, 1997.

[Ehrgott 97] M. Ehrgott, *A characterization of lexicographic max-ordering solutions*, Methods of Multicriteria Decision Theory: Proceedings of the 6th Workshop of the DGOR Working Group on Multicriteria and Decision Theory, Egelsbach, Häsel-Hohenhausen, pages 193–202, 1997.

[Ehrgott 00] M. Ehrgott, *Multicriteria Optimization*, Lecture Notes in Economics and Mathematical Systems 491, Springer, Berlin, Heidelberg, 2000.

[Ehrgott et al. 00] M. Ehrgott, X. Gandibleux, *An Annotated Bibliography of Multiobjective Combinatorial Optimization*, Technical report 62/2000, Fachbereich Mathematik, University of Kaiserslautern, Germany, 2000.

[Elmohamed 97] S. Elmohamed, P. Coddington, G. Fox, *A Comparison of Annealing Techniques for Academic Course Scheduling*, Technical report NPAC SCCS-777, January 1997.

[Engrand 97] P. Engrand, *Approche d'optimisation multiobjectif basée sur l'adoucissement simulé, et son application à la gestion des combustibles nucléaires*, Icone 5, International Conference on Nuclear Engineering, volume 10, pages 1–8, 1997.

[Engrand et al. 98] P. Engrand, X. Mouney, *Une méthode originale d'optimisation multiobjectif*, Internal report, EDF-DER, number HT-14/97/035/A, March 1998.

[Eschenauer et al. 90] H. Eschenauer, J. Koski, A. Osyczka, *Multicriteria design optimization: Procedures and Applications*, Springer, Berlin, Heidelberg, 1990.

[Ferber 95] J. Ferber, *Les systèmes multi-agents: vers une intelligence collective*, InterEditions, 1995

[Fleming et al. 85] P. J. Fleming, A. P. Pashkevich, *Computer aided control system design using a multiobjective optimization approach*, Proceedings of the IEEE Control 85 Conference, pages 174–179, January 1985.

[Fleming 93] P. J. Fleming, *Genetic algorithms for multiobjective optimization: formulation, discussion and generalisation*, Genetic Algorithms: Proceedings of the Fifth International Conference, 1993.

[Flexer 99] A. Flexer, *On the self-organizing maps for clustering and visualization*, In Third European Conference on Principles of Data Mining and Knowledge Discovery, PKDD'99, Prague, pages 80–88, Lecture Notes in Artificial Intelligence 1704, Springer, Berlin, Heidelberg, 1999.

[Fonseca et al 93] C. M. Fonseca, P. J. Fleming, *Genetic algorithms for multiobjective optimization: formulation, discussion and generalization*, Proceedings of the Fifth International Conference on Genetic Algorithms, pages 416–423, San Mateo, California, 1993.

[Fudenberg et al. 92] D. Fudenberg, J. Tirole, *Game Theory*, MIT Press, 1992.

[Gandibleux 99] X. Gandibleux, *Journée de travail PM2O Programmation mathématique multiobjectif*, LAMIH-ROAD, Valenciennes, September 1999.

[Gandibleux 00] X. Gandibleux, *Journée de travail PM2O Programmation mathématique multiobjectif*, Tours, November 2000.

[Gandibleux 01] X. Gandibleux, *Journée de travail PM2O Programmation mathématique multiobjectif*, Paris, May 2001.

[Gaudier 99] F. Gaudier, *Modélisation par réseaux de neurones, application à la gestion du combustible dans un réacteur*, Technical report CEA-R-5854, CEA/Saclay, 1999.

[Gert et al. 97] I. P. Gert, J. L. Underwood, *The logic of search algorithms: theory and applications*, Constraint Programming Conference, CP97, 1997.

[Giesy 78] D. P. Giesy, *Calculation of Pareto optimal solutions to multiple objective problems using threshold of acceptability constraints*, IEEE Transactions on Automatic Control, volume AC-23, number 6, 1978.

[Glover et al.] F. Glover, M. Laguna, M. Marti, *Scatter Search*, Preprint, `http://bus.colorado.edu/Faculty/Laguna/Papers/ss2.html`.

[Glover 86] F. Glover, *Future paths for integer programming and links to artificial intelligence*, Computational and Operations Research, volume 13, number 5, pages 533–549, 1986.

[Glover 90] F. Glover, *Artificial intelligence, heuristic frameworks and tabu search*, Managerial and Decision Economics, volume 11, pages 365–375, 1990.

[Glover et al. 97] F. Glover, M. Laguna, *Tabu Search*, Kluwer Academic, Dordrecht, 1997.

[Goicoechea et al. 79] A. Goicoechea, L. Duckstein, M. M. Fogel, *Multiple objective under uncertainty: an illustrative application of PROTRADE*, Water Resources Research, volume 15, number 2, pages 203–210, 1979.

[Goldberg 94] D. Goldberg, *Algorithmes génétiques*, Addison-Wesley, 1994.

[Gomez-Skarmeta et al. 98] A. F. Gómez-Skarmeta, F. Jiménez, J. Ibáñez, *Pareto-optimality in fuzzy modeling*, 6th European Congress on Intelligent Techniques and Soft Computing (EUFIT'98), pages 694–700, September 1998.

[Guvenir et al. 95] H. A. Güvenir, *A genetic algorithm for multicriteria inventory classification*, Artificial Neural Nets and Genetic Algorithms. Proceedings of the International Conference, pages 6–9, Vienna, April 1995.

[Haimes et al. 75] Y. Y. Haimes, W. A. Hall, H. T. Freedman, *Multiobjective Optimization in Water Resources Systems*, Elsevier Scientific, 1975.

[Han et al. 01] Q. Han, J. Han, *Revised filled function methods for global optimization*, Applied Mathematics and Computation, number 119, pages 217–228, 2001.

[Hansen 97a] M. P. Hansen, *Experiments on the Usage of Hashing Vectors in Multiobjective Tabu Search*, August 1997, `http://www.imm.dtu.dk/~mph/papers`.

[Hansen 97b] M. P. Hansen, *Tabu search for multiobjective optimization: MOTS*, MCDM97, January 1997, `http://www.imm.dtu.dk/~mph/papers`.

[Hansen 97c] M. P. Hansen, *Generating a Diversity of Good Solutions to a Practical Combinatorial Problem Using Vectorized Simulated Annealing*, Preprint, 1997, http://www.imm.dtu.dk/~mph/papers.

[Hansen et al. 98] P. Hansen, N. Mladenović, *Variable neighborhood search: Principles and applications*, Les cahiers du GERAD, G-98-20, May 1998.

[Hansmann et al. 94] U. H. E. Hansmann, Y. Okamoto, *Comparative Study of Multicanonical and Simulated Annealing Algorithms in the Protein Folding Problem*, Technical report SC-94-20-NWU-5/94, July 1994.

[Hartmann 99] A. K. Hartmann, *Introduction to Complexity Theory*, October 1999

[Hati et al. 87] S. K. Hati, R. G. Lamb, *Application of game theory in the design of optimal air pollution control measures*, Atmospheric Environment, volume 21, number 8, pages 1833–1841, 1987.

[Henrion 74] C. Henrion, *L'aide à la décision, état de l'art et des problèmes*, Informatiques et gestion, number 60, 1974.

[Hillier et al. 95] F. S. Hillier, G. J. Lieberman, *Introduction to Operations Research*, McGraw-Hill International, 1995.

[Hoffmann et al. 95] T. Hoffmann, J. Buhmann, *Multidimensional scaling and data clustering*, Advances in Neural Information Processing, volume 7, pages 104–111, 1995.

[Hong et al. 86] L. Hong, C. Ting, *Group decision making with multiple objectives and its interactive solution approach*, IFAC Large scale systems: Theory and applications, 1986.

[Hooker 95] J. N. Hooker, *Testing heuristics: we have it all wrong*, Journal of Heuristics, pages 33–42, May 1995, http://citeseer.nj.nec.com/hooker95testing.html.

[Hordijk] W. Hordijk, *A measure of landscapes*, Evolutionary Computation, volume 4, number 4, pages 335–360, 1996, http://citeseer.nj.nec.com/hordijk95measure.html.

[Horn et al. 93] J. Horn, N. Nafpliotis, D. E. Goldberg, *Multiobjective optimization using the niched Pareto genetic algorithm*, Technical report, IlliGAl Report 93005, University of Illinois, Urbana, 1993.

[James 95] R. James, *A Framework for Search Heuristics*, 1995, http://www.esc.auckland.ac.nz/Organisations/ORSNZ/Conf95/papers/RossJames.pdf.

[Jaszkiewicz 01] A. Jaszkiewicz, *Multiple Objective Metaheuristic Algorithms for Combinatorial Optimization*, Habilitation thesis, Poznan University of Technology, 2001.

[Jimenez et al. 99] F. Jiménez, J. L. Verdegay, A. F. Gómez-Skarmeta, *Evolutionary techniques for constrained multiobjective optimization problems*, Proceedings of the 1999 Genetic and Evolutionary Computation Conference, pages 115–116, Orlando, Florida, july 1999.

[Jin et al. 01] Y. Jin, T. Okabe, B. Sendhoff, *Adapting weighted aggregation for multiobjective evolution strategies*, EMO 2001, pages 96–110, 2001.

[Jones et al. 95] T. Jones, S. Forrest, *Fitness distance correlation as a measure of problem difficulty for genetic algorithms*, Proceedings of the Sixth International Conference on Genetic Algorithms, pages 184–192, 1995, http://citeseer.nj.nec.com/jones95fitness.html.

[Kahng et al. 97] A. B. Kahng, R. Sharma, *Studies of Clustering Objectives and Heuristics for Improved Standard-Cell Placement*, Manuscript, 1997, http://citeseer.nj.nec.com/152880.html

[Kanzow 98] C. Kanzow, *Global Optimization Techniques for Mixed Complementarity Problems*, Technical Report MP-TR-1998-09, July 1998, `http://citeseer.nj.nec.com/kanzow98global.html`.

[Keeney 82] R. L. Keeney, *Decision analysis: an overview*, Operations Research, volume 30, number 5, pages 803–837, 1982

[Keeney et al. 93] R. L. Keeney, H. Raiffa, *Decisions with Multiple Objectives: Preferences and Value Tradeoff*, Cambridge University Press, 1993.

[Kennedy et al. 95] J. Kennedy, R. Eberhart, *Particle swarm optimization*, Proceedings of the IEEE International Conference on Neural Networks, pages 1942–1948, 1995.

[Kermanshahi et al. 89] B. S. Kermanshahi, Y. Yasuda, R. Yokoyama, *Environmental Marginal Cost Evaluation by Non-Inferiority Surface for Optimal Power Sispatching*, Technical report PE-89-177, 1989.

[Kim et al.] D. G. Kim, P. Husbands, *Mapping Based Constraint Handling for Evolutionary Search; Thurston's Circle Packing and Grid Generation*, Preprint.

[Kingdon et al. 95] J. Kingdon, L. Dekker, *The shape of space*, Proceedings of the first IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, London, pages 543–548, 1995.

[Klock et al. 97] H. Klock, J. M. Buhmann, *Multidimensional Scaling by Deterministic Annealing*, Lecture notes in computer science 1223, pages 246–260, Springer, Berlin, Heidelberg, 1997.

[Klock et al. 99] H. Klock, J. M. Buhmann, *Data visualization by multidimensional scaling: a deterministic annealing approach*, Pattern Recognition, volume 33, number 4, pages 651–669, 1999.

[Knowles et al. 99] J. Knowles, D. W. Corne, *The Pareto archived evolution strategy: a new baseline algorithm for Pareto multiobjective optimization*, 1999 Congress on Evolutionary Computation, pages 98–105, July 1999.

[Knowles et al. 00] J. D. Knowles, D. W. Corne, *A comparison of diverse approaches to memetic multiobjective combinatorial optimization*, Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program, pages 103–108, Las Vegas, July 2000.

[Knowles et al. 01] J. D. Knowles, R. A. Watson, D. W. Corne, *Reducing local optima in single-objective problems by multi-objectivization,* In E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, D. Corne (eds.), First International Conference on Evolutionary Multi-Criterion Optimization, pages 268–282, Lecture Notes in Computer Science 1993, Springer Berlin, Heidelberg, 2001.

[Koppen et al. 98] M. Köppen, S. Rudlof, *Multiobjective optimization by Nessy algorithm*, Proceedings of the 3rd Online Workshop on Soft Computing, pages 357–368, London, 1998.

[Koski 85] J. Koski, *Defectiveness of weighting method in multicriterion optimization of structures*, Communications in Applied Numerical Mathematics, volume 1, pages 333–337, 1985.

[Koza et al. 99] J. R. Koza, F. H. Bennett III, D. Andre, M. A. Keane, *Genetic Programming III: Automatic Programming and Automatic Circuit Synthesis*, Morgan Kauffman, 1999.

[Krishnamachari et al. 00] B. Krishnamachari, X. Xie, B. Selman, S. Wicker, *Performance Analysis of Neighborhood Search Algorithms*, Preprint, January 2000, `http://citeseer.nj.nec.com/293033.html`.

[Kruskal 64a] J. B. Kruskal, *Nonmetric multidimensional scaling: a numerical method*, Psychometrika, volume 29, number 2, June 1964.

[Kruskal 64b] J. B. Kruskal, *Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis*, Psychometrika, volume 29, number 1, March 1964.

[Laguna et al. 99] M. Laguna, R. Marti, *GRASP and path relinking for 2-layer straight line crossing minimization*, INFORMS Journal on Computing, volume 11, number 1, 1999, `http://bus.colorado.edu/faculty/laguna/publications.html`.

[Lamagna 87] E. A. Lamagna, *Infeasible computation: NP complete problems*, ABACUS, volume 4, number 3, pages 18–33, 1987.

[Laumanns et al. 00] M. Laumanns, E. Zitzler, L. Thiele, *A unified model for multi-objective evolutionary algorithms with Elitism*, 2000 Congress on Evolutionary Computation, volume 1, pages 46–53, Piscataway, New Jersey, July 2000.

[Lauriere 78] J. L. Laurière, *A language for stating and solving combinatorial problems*, Artificial Intelligence, number 10, pages 29–127, 1978

[Lee et al. 00] Y.-H. Lee, B. J. Berne, *Global optimization: quantum thermal annealing with path integral monte carlo*, Journal of Physics and Chemistry A, volume 104, pages 86–95, 2000.

[De Leeuw 93] J. De Leeuw, *Fitting Distances by Least Squares*, Technical report, UCLA Statistics Series 130, 1993.

[De Leeuw et al. 97] J. De Leeuw, P. J. F. Groenen, *Inverse multidimensional scaling*, Journal of Classification, volume 14, pages 3–21, 1997.

[Levin] M. S. Levin, *Algorithm Systems for Combinatorial Optimization: Hierarchical Multistage Framework*, Preprint, `http://citeseer.nj.nec.com/267726.html`.

[Lin 76] J. G. Lin, *Multiple-objective problems: Pareto-optimal solutions by proper equality constraints*, IEEE Transactions on Automatic Control, volume 5, number AC21, pages 641–650, october 1976.

[Lin 77] J. G. Lin, *Proper inequality constraints and maximization of index vectors*, Journal of Optimization Theory and Applications, volume 21, number 4, pages 505–521, April 1977.

[Lin et al. 98] C. Lin, J. I. Yang, K. J. Lin, Z. D. Wang, *Pressurized water reactor loading pattern design using the simple tabu search*, Nuclear Science and Engineering, volume SC1, pages 61–71, 1998.

[Lister 93] R. Lister, *Annealing Networks and Fractal Landscapes*, IEEE International Conference on Neural Networks, volume 1, pages 257–262, San Francisco, March 1993.

[Liu 01] X. Liu, *A class of generalized filled functions with improved computability*, Journal of Computational and Applied Mathematics, number 137, pages 61-69, 2001.

[Liu 02] X. Liu, *A computable filled function used for global minimization*, Applied Mathematics and Computation, number 126, pages 271–278, 2002.

[Lombard 79] J. Lombard, *Revue de quelques méthodes d'aide à la décision*, Commission of the European Communities, 2-85206-127-9, Techniques et documentation (FRA), pages 69–88, 1979.

[Lootsma 87] F. A. Lootsma, *Modélisation du jugement humain dans l'analyse multicritère du moyen de comparaison par nos pairs*, R.A.I.R.O. Recherche opérationnelle, volume 21, number 3, pages 241–257, 1987.

[Loukil et al. 00] T. Loukil, J. Teghem, Ph. Fortemps, *Solving multi-objective production scheduling problems with tabu search*, Control and Cybernetics, volume 29, number 3, pages 819–828, 2000.

[Ma et al. 94] J. Ma, J. E. Straub, *Simulated Annealing using the classical density distribution*, Journal of Chemical Physics, volume 101, number 1, July 1994.

[Mathias et al. 94] K. E. Mathias, L. D. Whitley, *Transforming the search space with gray coding*, IEEE Conference on Evolutionary Computation, volume 1, pages 513–518, 1994, `http://www.cs.colorado.edu/~genitor/Pubs.html`.

[Mathieu et al. 96] P. Mathieu, J.-P. Delahaye, *Expériences sur le dilemme itéré des prisonniers*, Internal publication of the LIFL, number IT-233, March 1996, `http://www.lifl.fr/IPD/ipd.html#french`.

[Maystre et al. 94] L.-Y. Maystre, J. Pictet, J. Simos, *Méthodes multicritères ELECTRE,* Presses Polytechniques et Universitaires Romandes, 1994.

[McAllester et al. 97] D. McAllester, B. Selman, H. Kautz, *Evidence for invariants in local search*, Proceedings of the Fourteenth National Conference on Artificial Intelligence, Providence, Rhode Island, pages 321–326, 1997.

[McReady et al. 95] W. G. McReady, D. H. Wolpert, *What Makes an Optimization Problem Hard?*, Techical Report SFI-TR-95-0, Santa Fe Institute, April 1995.

[Menon et al. 95] A. Menon, K. Mehrotra, C. K. Mohan, S. Ranka, *Optimization using replicators*, Proceedings of the Sixth International Conference on Genetic Algorithms, pages 209–216, 1995.

[Menon et al. 97] A. Menon, K. Mehrotra, C. K. Mohan, S. Ranka, *Replicators, majorization and genetic algorithms: new models and analytical tools*, Foundations of Genetic Algorithms, number 4, 1997.

[Mertens] S. Mertens, *Computational Complexity for Physicists*, Lecture notes, `http://odysseus.nat.uni-magdeburg.de/~mertens/notes/complex.ps.gz`.

[Merz 00] P. Merz, *Memetic Algorithms for Combinatorial Optimization Problems: Fitness Landscape and Effective Search Strategies*, PhD thesis, December 2000.

[Merz et al. 98] P. Merz, B. Freisleben, *On the effectiveness of evolutionary search in high-dimensional NK-landscapes*, Proceedings of the 1998 IEEE International Conference on Evolutionary Computation, pages 741–745, 1998, `ftp://ftp-informatik.uni-siegen.de/pub/papers/pmerz/icec98.ps.gz`.

[Messac et al. 00] A. Messac, G. J. Sundararaj, R. V. Tappeta, J. E. Renaud, *Ability of objective functions to generate points on nonconvex Pareto frontiers*, AIAA Journal, volume 38, number 6, pages 1084–1091, 2000.

[Michalewicz 95a] Z. Michalewicz, *A survey of constraint handling techniques in evolutionary computation methods*, In (eds.), Proceedings of the Fourth Annual Conference on Evolutionary Programming, MIT Press, Cambridge, MA, pages 135–155, 1995

[Michalewicz 95b] Z. Michalewicz, *Genetic algorithms, numerical optimization, and constraints*, Proceedings of the sixth International Conference on Genetic Algorithms, pages 151–158, San Mateo, California, July 1995.

[Miettinen et al. 98] K. M. Miettinen, M. M. Mäkelä, *Proper Pareto Optimality in Nonconvex Problems – Characterization with Tangent and Normal Cones*, Technical report, Jyväskylä University, November 1998.

[Miettinen 99] K. M. Miettinen, *Nonlinear Multiobjective Optimization*, Kluwer academic, Dordrecht, 1999.

[Mobius et al.] A. Mobius, A. Diaz-Sanchez, B. Freisleben, M. Schreiber, A. Fachat, K. H. Hoffmann, P. Merz, A. Neklioudov, *Two Physically Motivated Algorithms for Combinatorial Optimization: Thermal*

Cycling and Iterative Partial Transcription, http://citeseer.nj.nec.com/119625.html.

[Monclar 95] F.-R. Monclar, *Résolution coopérative de problèmes: notes bibliographiques*, Internal report, EDF-DER, number HT-37/95/019, 1995

[Monclar 98] F.-R. Monclar, *Résolution coopérative de problèmes: ELICO et son application à la supervision des réseaux électriques*, PhD thesis, Montpellier University (LIRMM), March 1998.

[Monclar 99] F.-R. Monclar, *Deux exemples d'utilisation d'algorithmes "méta-heuristiques" pour des applications à EDF*, Internal report, EDF-DER, 1999.

[MOPGP] M.O.P.G.P. Website about the goal programming method, http://www.sms.port.ac.uk/mpg/homepage.html.

[More et al. 95] J. J. Moré, Z. Wu, *Global Smoothing and Continuation for Large-Scale Molecular Optimization*, preprint MCS-P539-1095, October 1995, http://citeseer.nj.nec.com/114806.html.

[Morita et al. 01] H. Morita, X. Gandibleux, N. Katoh, *Experimental feedback on biobjective permutation scheduling problems Solved with a population heuristic*, FCDS, volume 26, number 1, pages 43–50, 2001.

[Mouney 98] X. Mouney, *Implémentation dans le logiciel FORMOSA d'une méthode multi-objectif basée sur le recuit simulé*, Internal Report, EDF-DER, 1998.

[Muller 94] C. Muller, *Introduction aux méthodes neuronales d'optimisation*, Internal report, EDF-DER number HI-23/94/012, December 1994.

[Multisimplex] Users manual of the MultiSimplex software package, http://www.multisimplex.com

[Munro et al. 86] J. Munro, P. H. Chaung, *Optimal plastic design with imprecise data*, Journal of Engineering Mechanics, volume 112, number 9, pages 888–903, 1986.

[Mynard 97] L. Mynard, *Exploration locale oscillante heuristiquement ordonnée*, PhD thesis LAFORIA-LIP6, December 1997.

[Nelder et al. 65] J. A. Nelder, R. Mead, *A simplex method for function minimization*, Computer Journal, volume 7, pages 308–313, 1965.

[Niimura et al.] T. Niimura, Y. Ueki, T. Matsumoto, R. Yokoyama, *A Fuzzy Approach for Multi-Objective Optimal Generation Dispatch*, Preprint.

[Norbis et al. 88] M. I. Norbis, J. McGregor-Smith, *A multiobjective, multi-level heuristic for dynamic resource constrained scheduling problems*, European Journal of Operational Research, volume 33, number 1, pages 30–41, 1988.

[Opricovic] S. Opricovic, *An Extension of Compromise Programming to the Solution of Dynamic Multicriteria Problems*, Preprint.

[Osyczka 92] A. Osyczka, *Computer Aided Multicriterion Optimization System (CAMOS)*, International Software, 1992.

[Othmani 98] I. Othmani, *Optimisation multicritère: Fondements et Concepts*, PhD thesis, Grenoble University, May 1998.

[Papadimitriou 97] C. H. Papadimitriou (ed.), *NP Completeness: A Retrospective*, Proceedings of ICALP 97, Springer, Berlin, Heidelberg,1997.

[Parks et al. 98] G. T. Parks, I. Miller, *Selective breeding in a multiobjective genetic algorithm*, PPSN V, pages 250–259, Amsterdam, 1998.

[Parks et al. 99] G. T. Parks, A. Suppapitnarm, *Multiobjective optimization of PWR reload core designs using Simulated Annealing*, Mathematics and Computation, Reactor Physics and Environmental Analysis in Nuclear Applications, Madrid, pages 1435–1444, September 1999.

[Pasquier-Dorthe et al. 95] J. Pasquier-Dorthe, H. Raynaud, *Un outil d'aide à la décision multicritère*, Revue Française de Gestion, number 106, pages 11–21, 1995.

[Pentzaropoulos et al. 93] G. C. Pentzaropoulos, D. I. Giokas, *Cost-performance modeling and optimization of network flow balance via linear goal programming analysis*, Computer Communications, volume 16, number 10, pages 645–652, 1993.

[Peralska et al.] E. Peralska, D. De Ridder, R. P. W. Duin, M. A. Kraaijveld, *A new method of generalizing Sammon mapping with application to algorithm speed up*, Fourth Annual Conference of the Advanced School for Computing and Imaging, pages 221–228, June 1999, `http://www.ph.tn.tudelft.nl/~dick/publications.html`.

[Peretto] P. Peretto, *Réseaux de neurones et optimisation combinatoire*, Preprint.

[Perrin et al. 97] E. Perrin, A. Mandrille, M. Oumoun, C. Fonteix, I. Marc, *Optimisation globale par stratégie d'évolution: technique utilisant la génétique des individus diploïdes*, Recherche Opérationnelle, AFCET-Gauthier-Villars, volume 31, number 2, pages 161–201, 1997.

[Peterson et al. 93] C. Peterson, B. Soderberg, *A new method for mapping optimization problems onto neural networks*, International Journal of Neural Systems, volume 1, number 1, pages 3–22, 1993.

[Pohlheim] H. Pohlheim, *Visualization of Evolutionary Algorithms: Real-World Application of Standard Techniques and Multidimensional Visualization*, Preprint, `http://citeseer.nj.nec.com/287543.html`.

[Potter 02] M. A. Potter, *An NK-Landscape Generator*, 2002, `http://www.cs.gmu.edu/~potter`.

[Pouillot 92] S. Pouillot, *Etude comparative des techniques multi-agents et des blackboard*, Internal report, EDF, number HI-57/7959, May 1992.

[Press et al. 02] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, *Numerical Recipes in C++*, Cambridge University Press, 2002.

[Preux et al. 99] Ph. Preux, E.-G. Talbi, *Towards hybrid evolutionary algorithms*, International Transactions in Operational Research, volume 6, pages 557–570, 1999.

[Quentin 82] R. Quentin, *Quatre techniques d'aide à la décision*, Cadréco, number 63, pages 9–12, September 1982.

[Radcliff et al. 95] N. J. Radcliff, P. D. Surry, *Fundamental Limitations on Search Algorithms: Evolutionary Computation in Perspective*, Lecture Notes in Computer Science 1000, Springer, Berlin, Heidelberg, 1995, `http://citeseer.nj.nec.com/radcliffe95fundamental.html`.

[Reardon 97a] B. J. Reardon, *Fuzzy Logic vs Niched Pareto Multi-Objective Genetic Algorithm Optimization: Part I: Schaffer's F2 Problem*, Technical report LA-UR-97-3675, Los Alamos National Laboratory, 1997.

[Reardon 97b] B. J. Reardon, *Fuzzy Logic vs Niched Pareto Multi-Objective Genetic Algorithm Optimization: Part II: a Simplified Born-Mayer Problem*, Technical report LA-UR-97-3676, Los Alamos National Laboratory, 1997.

[Rebreyend 99] P. Rebreyend, *Algorithmes génétiques hybrides en optimisation combinatoire*, PhD thesis, ENS Lyon, January 1999.

[Reidys et al.] C. M. Reidys, P. F. Stadler, *Combinatorial Landscapes*, `http://citeseer.nj.nec.com/426169.html`.

[Remez 86] C. Remez, *L'intelligence dévoilée ou la logique floue*, Microsystèmes, May 1986.

[Reuss 85a] P. Reuss, *Eléments de neutronique*, Lecture notes I.N.S.T.N., 1985.

[Reuss 85b] P. Reuss, *Que sais-je? La neutronique*, Presse Universitaire Française, 1985.

[Robert 85] F. Robert, *Meilleure approximation en norme vectorielle et minimum de Pareto*, Mathematical Modelling and Numerical Analysis, volume 19, number 1, pages 90–110, 1985.

[Roy 85] B. Roy, *Méthodologie multicritère d'aide à la décision*, Economica, 1985.

[Roy et al. 81] B. Roy, P. Vincke, *Multicriteria analysis: survey and new directions*, European Journal of Operational Research, 1981.

[Roy et al. 88] B. Roy, D. Bouyssou, *Aide à la décision*, Encyclopedia des sciences de la gestion (AFCET/Interfaces), number 65, pages 4–13, March 1988.

[Roy et al. 93] B. Roy et D. Bouyssou, *Aide multicritère à la décision: Méthodes et cas*, Economica, 1993.

[Rudolph 98] G. Rudolph, *On a multiobjective evolutionary algorithm and its convergence to the Pareto set*, Proceedings of the 5th IEEE Conference on Evolutionary Computation, pages 511–516, Piscataway, New Jersey, 1998.

[Sammon 69] J. M. Sammon, *A nonlinear mapping for data structure analysis*, IEEE Transactions on Computers, volume C-18, number 5, pages 401–409, 1969

[Sait et al. 99] S. M. Sait, H. Youssef, *Iterative Computer Algorithms with Applications in Engineering: Solving Combinatorial Optimization Problems*, IEEE Computer Society, 1999.

[Sakawa et al. 77] N. Sakawa, Y. Sawaragi, *Multiobjective optimization in water resources problem for a single river basin – a case study of the Kakogawa river basin*, Proceedings of the First KNKI IRDP Workshop, pages 197–221, 1977.

[Sakawa et al. 85] M. Sakawa, F. Seo, *Interactive multiobjective decision making by the sequential proxy optimization technique (SPOT) and its application to environmental systems*, IFAC Control Science and Technology, volume 10, number 50-2, pages 119–124, 1985.

[Sakawa et al. 88] M. Sakawa, H. Yano, *An interactive fuzzy satisficing method for multiobjective linear programming problems with fuzzy parameters*, Fuzzy Sets and Systems, volume 28, pages 129–144, 1988.

[Sakawa 02] M. Sakawa, *Genetic Algorithms and Fuzzy Multiobjective Optimization*, Kluwer Academic, Dordrecht, 2002.

[Sarma et al. 96] J. Sarma, K. De Jong, *An analysis of the effects of neighborhood size and shape on local selection algorithms*, In Proceedings of the Fourth PPSN, LNCS 1141, Springer, Berlin, Heidelberg, pages 236–244, 1996.

[Sawaragi et al.] Y. Sawaragi, S. Ikeda, H. Nakayama, T. Tanino, C. Okumura, *An Interactive Optimization Method for a Water Resource Problem with Multiple Objectives*, preprint.

[Schaerf et al. 99] A. Schaerf, M. Lenzerini, M. Cadoli, *LOCAL++: A C++ Framework for Local Search Algorithms*, Technical report 11-99, Dipartimento di Informatica e Sistemistica, University of Rome, May 1999, `http://www.dis.uniroma1.it/pub/AI/papers/scha-lenz-cado-99.ps.gz`.

[Schaffer et al. 89] J. D. Schaffer, R. A. Caruana, L. J. Eshelman, R. Das, *A study of control parameters affecting online performance of genetic algorithms for function optimization*, ICGA-89, pages 51–60, San Mateo, California, 1989.

[Scharlig 85] A. Scharlig, *Décider sur plusieurs critères: panorama de l'aide à la décision multicritère*, Presses Polytechniques Romandes, 1985.

[Scharlig 96] A. Scharlig, *Pratiquer ELECTRE et PROMETHEE: Un complément à Décider sur plusieurs critères*, Presses Polytechniques Romandes, 1996.

[Schelstraete et al.] S. Schelstraete, W. Schepens, H. Verschelde, *Energy Minimization by Smoothing Techniques: a Survey*, `http://citeseer.nj.nec.com/411692.html`.

[Schoenauer et al. 98] M. Schoenauer, Z. Michalewicz, *Sphere operators and their applicability for constrained parameter optimization problems*, Seventh Annual Conference on Evolutionary Programming (EP'98), 1998.

[Sefrioui et al.] M. Sefrioui, J. Periaux, *Nash Genetic Algorithms: Examples and Applications*, Preprint.

[Sen] S. Sen, *Stochastic Programming: Computational Issues and Challenges*, Encyclopedia of OR/MS.

[Serquin et al. 97] Y. Serquin, F. Beaudouin, B. Murier, *Multi-Attribute Utility Theory Toward a More General Framework*, Internal report, EDF-DER, number HP-28/97/042/A, April 1997.

[Shao et al. 95] C.-S. Shao, R. H. Byrd, E. Eskow, R. B. Schnabel, *Global Optimization for Molecular Clusters Using a New Smoothing Approach*, 1995, `http://citeseer.nj.nec.com/shao95global.html`.

[Sharpe 00] O. J. Sharpe, *Towards a Rational Methodology for Using Evolutionary Search Algorithms*, PhD thesis, University of Sussex, Brighton, January 2000.

[Shatilla et al. 99] Y. A. Shatilla, D. C. Little, J. A. Penkrot, R. A. Holland, *Biased multi-objective function optimization in Westinghouse advanced loading pattern search code ALPS*, Mathematics and Computation, Reactor Physics and Environmental Analysis in Nuclear Applications, volume 2, pages 282–295, Madrid, September 1999.

[Siarry et al. 89] P. Siarry, G. Dreyfus, *La méthode du recuit simulé. Théorie et applications*, IDSET, ESPCI, 1989.

[Siarry 94] P. Siarry, *La méthode du recuit simulé en électronique*, Habilitation report, Paris-Sud University (Orsay), 1994.

[Siarry 99] P. Siarry, *Optimisation et classification de données*, Lectures from the DEA GBM courses at Paris 12 University, 1999.

[Smith 95] B. M. Smith, *A Tutorial on Constraint Programming*, Technical report 95-14, April 1995.

[Smith 99] K. A. Smith, *Neural networks for combinatorial optimization: A review for more than a decade of research*, INFORMS Journal on Computing, volume 11, number 1, 1999 `http://www.bsys.monash.edu.au/staff/kate/kate.html`.

[Spenle et al. 96] D. Spenlé, R. Gourhant, *Guide du calcul en mécanique*, Hachette Technique, 1996.

[Spierenburg 97] J. A. Spierenburg, *Dimension Reduction of Images Using Neural Networks*, Master's Thesis, Department of Computer Science, Leiden University, July 1997.

[Srinivas et al. 93] N. Srinivas, K. Deb, *Multiobjective Optimization Using Non-Dominated Sorting in Genetic Algorithms*, Technical report, Department of Technical Engineering, Indian Institute of Technology, Kanpur, 1993.

[Stadler et al. 92] P. F. Stadler, W. Schnabl, *The Landscape of the Travelling Salesman Problem*, 1992, `http://citeseer.nj.nec.com/stadler92landscape.html`.

[Steuer 76] R. E. Steuer, *Multiple objective linear programming with interval criterion weights*, Management Science, volume 23, number 3, pages 305–316, 1976.

[Stutzle et al. 00] T. Stutzle, A. Grun, S. Linke, M. Ruttger, *A Comparison of Nature Inspired Heuristics on the Traveling Salesman Problem*, 2000, `http://citeseer.nj.nec.com/388289.html`.

[Szidarowsky et al. 84] F. Szidarowsky, L. Duckstein, I. Bogardi, *Multiobjective management of mining under water hazard by game theory*, Journal of Operational Research, volume 15, pages 251–258, 1984.

[Tabak 79] D. Tabak, *Computer based experimentation with multicriteria optimization problems*, IEEE Transactions on Systems, Man and Cybernetics, volume SMC9, number 10, pages 676–679, 1979.

[Tagami et al. 99] T. Tagami, T. Kawabe, *Genetic algorithm based on a Pareto neighborhood search for multiobjective optimization*, Proceedings of the 1999 International Symposium on Nonlinear Theory and its Applications, pages 331–334, 1999, `http://citeseer.nj.nec.com/320632.html`.

[Taillard] E. D. Taillard, *Comparison of Non-Deterministic Iterative Methods*, Preprint, `http://www.eivd.ch/ina/collaborateurs/etd/articles.dir/articles.html`.

[Talbi 98] E.-G. Talbi, *A Taxonomy of Hybrid Metaheuristics*, Technical report TR-183, LIFL, Lille 1 University, May 1998, `http://www.lifl.fr/~talbi/papers.html`.

[Tarvainen 86] K. Tarvainen, *On the generating of Pareto optimal alternatives in large scale systems*, IFAC Large Scale Systems: Theory and applications, Pergamon, Oxford, pages 519–524, august 1986.

[Teghem et al. 00] J. Teghem, D. Tuyttens, E. L. Ulungu, *An interactive heuristic method for multi-objective combinatorial optimization*, Computer & Operations Research, volume 27, pages 621–634, 2000.

[Thiollet 99] B. Thiollet, *Méthode du gradient discret pour l'optimisation de la recherche de plans de rechargement: recommandations pour la maquette*, Internal report, EDF-DER, number HI-72/99/002/A, February 1999.

[Tipping et al. 97] M. E. Tipping, C. M. Bishop, *Mixtures of Principal Component Analysis*, Technical report, NCRG/97/003, Neural Computing Research Group, Aston University, Birmingham, 1997, `http://www.research.microsoft.com/users/mtipping/pages/publications.htm`.

[Tipping et al. 98] M. E. Tipping, D. Toux, *Shadow targets: a novel algorithm for topographic projections by radial basis functions*, NeuroComputing, volume 19, pages 211–222, 1998, `http://www.research.microsoft.com/users/mtipping/pages/publications.htm`.

[T'kindt et al. 00] V. T'kindt, J.-C. Billaut, *L'ordonnancement multicritère*, Presses Universitaires de Tours, 2000.

[Toshinsky et al. 00] V. G. Toshinsky, H. Sekimoto, G. I. Toshinsky, *A method to improve multiobjective genetic algorithm optimization of a self-fuel-providing LMFBR by niche induction among nondominated solutions*, Annals of Nuclear Energy, volume 25, number 5, pages 397–410, 2000.

[Turinsky 99] P. J. Turinsky, *Mathematical optimization of incore nuclear fuel management decisions: status and trends*, ATW Atomwirtschaft-Atomtechnik Internationale Zeitschrift für Kernenergie, volume 44, number 7, pages 454–458, 1999.

[Tuyttens et al. 00] D. Tuyttens, J. Teghem, Ph. Fortemps, K. Van Nieuwenhuyze, *Performance of the MOSA method for the bicriteria assignment problem*, Journal of Heuristics, volume 6, number 3, pages 295–310, 2000.

[Ulungu et al. 94] E. L. Ulungu, J. Teghem, *Multi-objective combinatorial optimization problems: a survey*, Journal of Multi-criteria Decision Analysis, volume 3, pages 83–104, 1994.

[Ulungu et al. 95] E. L. Ulungu, J. Teghem, *The two phases method: an efficient procedure to solve bi-objective combinatorial optimization problems*, Foundations of Computing and Decision Sciences, volume 20, number 2, 1995.

[Ulungu et al. 99] E. L. Ulungu, J. Teghem, P. H. Fortemps, D. Tuyttens, *MOSA method: a tool for solving multiobjective combinatorial optimization problems*, Journal of Multicriteria Decision Analysis, volume 8, number 4, pages 221–236, 1999.

[Vaessens et al. 92] R. J. M. Vaessens, E. H. L. Aarts, J. K. Lenstra, *A Local Search Template*, PPSN 2, 1992, `http://citeseer.nj.nec.com/vaessens92local.html`.

[Vallin et al. 00] P. Vallin, D. Vanderpooten, *Aide à la décision: Une approche par les cas*, Ellipses, 2000.

[Van Den Bergh et al.] F. Van Den Bergh, A. P. EngelBrecht, *Effects of Swarm Size on Cooperative Particle Swarm Optimizers*, `http://citeseer.nj.nec.com/447945.html`.

[Van Den Bergh et al. 00] F. Van Den Bergh, A. P. Engelbrecht, *Cooperative learning in neural networks using particle swarm optimizers*, South African Computer Journal, number 26, pages 84–90, 2000, `http://citeseer.nj.nec.com/vandenbergh00cooperative.html`.

[Van Hentenryck et al. 96] P. Van Hentenryck, D. McAllester, D. Kapur, *Solving polynomial systems using a branch and prune approach*, SIAM Journal of Numerical Analysis, volume 34, number 2, 1996.

[Van Kemenade 98] C. H. M. Van Kemenade, *Analysis of NK-XOR Landscape*, 1998, `http://citeseer.nj.nec.com/vankemenade98analysis.html`.

[Van Laarhoven et al. 92] P. J. M. Van Laarhoven, E. H. L. Aarts, *Simulated Annealing: Theory and Applications*, Kluwer Academic, Dordrecht, 1992.

[Van Veldhuizen et al. 98] D. A. Van Veldhuizen, G. B. Lamont, *Multiobjective Evolutionary Algorithm Research: a History and Analysis*, Technical report TR-98-03, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright Patterson AFB, Ohio, October 1998.

[Van Veldhuizen 99] D. A. Van Veldhuizen, *Multiobjective Evolutionary Algorithms: Classifications, Analyses and New Innovations*, PhD thesis, Graduate School of Engineering. Air Force Institute of Technology, Wright Patterson AFB, Ohio, January 1999.

[Van Veldhuizen et al. 00] D. A. Van Veldhuizen, G. B. Lamont, *On measuring multiobjective evolutionary algorithms performances*, 2000 Congress on Evolutionary Computation, volume 1, pages 205–211, Piscataway, New Jersey, July 2000.

[Varanelli 96] J. M. Varanelli, *On the Acceleration of Simulated Annealing*, PhD thesis, Faculty of the School of Engineering and Applied Sciences, University of Virginia, May 1996.

[Vavasis] S. A. Vavasis, *Complexity Issues in Global Optimization: A Survey*, Preprint.

[Vern 92] J. L. Vern, *La logique floue: concepts et définitions*, Electronique Radio Plans, December 1992

[Vincke 89] Ph. Vincke, *L'aide multicritère à la décision*, Ellipses, 1989

[Visee et al. 98] M. Visée, J. Teghem, M. Pirlot, E. L. Ulungu, *Two-phases method and branch and bound procedures to solve the bi-objective knapsack problem*, Journal of Global Optimization, volume 12, pages 139–155, 1998.

[Wendt 97] O. Wendt, W. Konig, *Cooperative Simulated Annealing: How Much Cooperation is Enough?*, Technical report, 1997, `http://www.wiwi.uni-frankfurt.de/~wendt/IWI9719/iwi9719.ps.gz`.

[Wenzel et al.] W. Wenzel, K. Hamacher, *A Stochastic Tunneling Approach for Global Minimization of Complex Potential Energy Landscapes.*

[De Werra 90] D. De Werra, *Heuristics for graph coloring*, Computing Supplement, volume 7, pages 191–208, 1990.

[Westhoff et al. 96] F. H. Westhoff, B. V. Yarbrough, R. M. Yarbrough, *Complexity, organisation, and Stuart Kauffman's The Origin of Order*, Journal of Economic Behavior and Organization, volume 29, pages 1–25, 1996.

[Whales et al.] D. J. Whales, J. P. K. Doye, *Global Optimization by Basin-Hopping and the Lowest Energy Structures of Lennard-Jones Clusters Containing up to 110 Atoms.*

[Whitley 91] L. D. Whitley, *Fundamental principles of deception in genetic search*, Foundations of Genetic Algorithms, pages 222–241, 1991, `http://www.cs.colorado.edu/~genitor/Pubs.html`

[Whitley et al. 95] L. D. Whitley, K. Mathias, S. Rana, J. Dzubera, *Building better test functions*, International Conference on Genetic Algorithms, 1995,`http://www.cs.colorado.edu/~genitor/Pubs.html`.

[Whitley et al. 96] D. Whitley, K. Mathias, S. Rana, J. Dzubera, *Evaluating evolutionary algorithms*, Journal of Artificial Intelligence, volume 85, number 1–2, pages 245–276, 1996, `http://citeseer.nj.nec.com/whitley96evaluating.html`.

[Williamson 99] D. P. Williamson, *Lecture Notes on Approximation Algorithms*, Fall 1998, Internal report RC 21409, IBM Research Division, T. J. Watson Research Center, Yorktown Heights, New York, February 1999.

[Wolpert et al. 95] D. H. Wolpert, W. G. McReady, *No Free Lunch Theorem for Search*, Technical report SFI-TR-95-02-010, Santa Fe Institute, February 1995, `http://citeseer.nj.nec.com/wolpert95no.html`.

[Wolpert et al. 96] D. H. Wolpert, W. G. McReady, *No free lunch theorem for optimization*, IEEE Transactions on Evolutionary Computation, 1996, `http://citeseer.nj.nec.com/wolpert97no.html`.

[Wong et al. 98] H. I. Wong, S. Crouzet, *Optimisation des plans de rechargement nucléaire*, Internal report, EDF-DER, number HI-14/98/034/A, December 1998.

[Wu et al. 01] J. Wu, S. Azarm, *Metrics for quality assessment of a multiobjective design optimization solution set*, Transactions of the ASME, volume 123, pages 18–25, 2001.

[Xu et al. 98] J. Xu, S. Y. Chiu, F. Glover, *Fine-tuning a tabu search algorithm with statistical tests*, International Transactions on Operational Research, volume 5, number 3, pages 233–244, 1998.

[Xu et al. 99] H. Xu, B. J. Berne, *Multicanonical jump walk annealing: an efficient method for geometric optimization*, Journal of Chemical Physics, pages 299–306, 1999.

[Yamamoto et al. 99] A. Yamamoto, H. Hashimoto, *Applications of temperature parallel simulated annealing to loading pattern optimizations of pressurized water reactors*, Mathematics and Computation, Reactor

Physics and Environmental Analysis in Nuclear Applications, volume 2, pages 1445–1458, Madrid, September 1999.

[Yao 96] X. Yao, *An overview of evolutionary computation*, Chinese Journal of Advanced Software Research, volume 3, number 1, pages 12–29, 1996.

[Zadeh 65] L. A. Zadeh, *Fuzzy sets*, Information and Control, volume 8, pages 338–353, 1965.

[Zionts et al. 76] S. Zionts, J. Wallenius, *An interactive programming method for solving the multiple criteria problem*, Management Science, volume 22, number 6, pages 652–653, 1976.

[Zitzler et al. 98] E. Zitzler, L. Thiele, *An Evolutionary Algorithm for Multiobjective Optimization: the Strength Pareto Approach*, Technical report 43, Computer Engineering and Communication Networks Laboratory (TIK), Swiss Federal Institute of Technology, May 1998.

[Zitzler et al. 99] E. Zitzler, K. Deb, L. Thiele, *Comparison of Multiobjective Evolutionary Algorithms: Empirical Results*, Technical report 70, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, December 1999.

Index