Lehrstuhl für Leichtbau
Technische Universität München

Extended Evolutionary Algorithms for Multiobjective and Discrete Design Optimization of Structures

Harald Langer

Vollständiger Abdruck der von der Fakultät für Maschinenwesen der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr.-Ing. Wolfgang A. Wall
Prüfer der Dissertation:	1. Univ.-Prof. Dr.-Ing. Horst Baier
	2. Univ.-Prof. Dr.-Ing., Dr.-Ing. habil. Rudolf Schilling

Die Dissertation wurde am 15.06.2005 bei der Technischen Universität München eingereicht und durch die Fakultät für Maschinenwesen am 27.10.2005 angenommen.

Preface

This thesis was written during the years I worked at the Institute of Lightweight Structures (Lehrstuhl für Leichtbau) at the Technische Universität München.

I would like to thank Prof. Dr.-Ing. Horst Baier, the director of the institute and my supervisor, for giving me the chance to work on this interesting topic, his support during these years, and last but not least his patience.

I am also very grateful to all my colleagues, who helped and supported me during these years. Especially I would like to thank Guido Locatelli, Michael Müller, Christoph Zauner and Leri Datashvili not only for their help during this time but also for their friendship. They have been a great support not only for my work at the institute but also on other challenges. I learned a lot from them. My special thanks are directed to Tim Pühlhofer and Martin Huber for their support and the critical and fruitful discussions we had on the field of optimization. They have been a great help. Also I like to thank Jesko Klammer for his steady and always prompt help at the edition of this thesis.

I would also like to express my gratitude to all the students I had the chance to work with, especially to Javier Martin Recio for his committed and persistent work at the beginning of my research.

Last but not least I would like to thank my family, my friends, and especially my brother Detlef for their support during these years.

München, June 2005 Harald Langer

Abstract:

An optimization algorithm for multiobjective problems with continuous and discrete design variables is developed and investigated. Primary development goals have been to enlarge applicability to a very broad spectrum of tasks in the field of structural design while simultaneously providing sufficient efficiency to be viable for complex and computationally costly problems. The algorithm is based on the concept of evolutionary algorithms and features special adaptations for the simultaneous treatment of continuous and discrete variables as well as for the handling of multiple objectives. Core elements for increasing the efficiency are the grain-parallelization on a computer cluster as well as the extension of the conventional evolutionary algorithm by response surface methods. The performance of the algorithm has been evaluated by numerical experiments with suitable mathematical test problems as well as representative structural design problems.

Kurzfassung:

Für Mehrzielprobleme mit kontinuierlichen und diskreten Variablen ist ein Optimierungsalgorithmus entwickelt und untersucht worden. Primäre Entwicklungsziele waren die Anwendbarkeit in einem möglichst breiten Aufgabenspektrum auf dem Gebiet des Strukturentwurfs sowie gleichzeitig eine auch für komplexe, rechnerisch aufwendige Probleme ausreichende Effizienz. Der Algorithmus basiert auf dem Konzept der evolutionären Algorithmen und zeichnet sich durch spezielle Anpassungen für die simultane Handhabung kontinuierlicher und diskreter Designparameter sowie mehrerer Zielfunktionen aus. Kernelemente zur Effizienzsteigerung sind die Parallelisierung auf einem Computer-Cluster sowie die Erweiterung des konventionellen evolutionären Algorithmus um Response-Surface-Methoden. Die Eigenschaften und Leistungsfähigkeit des Algorithmus sind anhand numerischer Experimente mit geeigneten mathematischen Testfunktionen sowie repräsentativen Strukturentwurfsproblemen untersucht worden.

Contents

Nomenclature

Abbreviations

CTE	Coefficient of Thermal Expansion
CFRP	Carbon Fiber Reinforced Plastic
DOE	Design of Experiments
EA	Evolutionary Algorithms
ES	Evolutionary Strategies
FE	Finite Elements
FEM	Finite Element Method
GA	Genetic Algorithm
GAME	Genetic Algorithm for Multicriteria Engineering
MOEA	Multiobjective Evolutionary Algorithm
NSGA	Deb's Nondominated Sorting Genetic Algorithm
PAES	Knowles's Pareto Archived Evolutionary Strategy
RSA	Response Surface Approximation
RSM	Response Surface Method
SPEA	Zitzler's Strength Pareto Evolutionary Algorithm
SQP	Sequential Quadratic Programming

Symbols

Symbols	Description
$\mathbf{A}$	state space system matrix
$\mathbf{B}$	state space actuator matrix

Symbols	Description
$\mathbf{cost}$	vector with cost function values $\mathbf{cost} = \mathbf{f}(\mathbf{x}) = [\mathbf{f}(\mathbf{x}), \mathbf{gf}(\mathbf{x}), \mathbf{hf}(\mathbf{x})]$
$\mathbf{Cost}$	matrix with cost function values for the whole population P: $\mathbf{Cost} = \mathbf{f}(\mathbf{P})$
$\mathbf{C}$	children matrix, state space sensor matrix
Cl, $\mathbf{Cl}$	cluster (set of individuals), structure of a set of clusters
$\mathbf{d}_s$	discrete variable step size vector
$\mathbf{D}$	design space, damping matrix, state space feedthrough matrix (see contents)
$\mathbf{D}_f$	feasible design space
$\mathbf{D}_{dp}$	set of design points for the RSA built-up
e, $\mathbf{e}$	residual (approximation error), residual vector
f, $\mathbf{f}$	objective function, objective function vector
f, $\mathbf{f}$	alternatively: generalized cost function vector $\mathbf{f} = [\mathbf{f}, \mathbf{gf}, \mathbf{hf}]$ including both objective- and constraint functions
fit, $\mathbf{fit}$	fitness, fitness vector
gf, $\mathbf{gf}$	inequality constraint function, inequality constraint function vector
$\mathbf{g}$	goal vector
gen	current generation number
hf, $\mathbf{hf}$	equality constraint function, equality constraint function vector
$\mathbf{H}$	controller gain matrix
$\mathbf{i}$	individual i
I	area moment of inertia
$\mathbf{K}$	stiffness matrix
$\mathbf{lb}$	vector with lower bounds for the design variables
m	mass
$\mathbf{M}$	mass matrix
mat	material
n	number of individuals
n_c	number of children
n_{dp}	number of design points

Symbols	Description
n_{dv}	number of design variables
n_{ec}	number of equality constraints
n_{gen}	number of generations
n_{ic}	number of inequality constraints
n_{ob}	number of objective functions
n_{rc}	number of regression coefficients
$N(\mu, \sigma)$	normally distributed random number with a mean value of μ and a standard deviation of σ
$\mathbf{p}$	priority vector
$\mathbf{P}$	population matrix
$\bar{\mathbf{P}}$	nondominated population
$\mathbf{P}_a$	actuator influence matrix
$\mathbf{P}_{ini}$	initial population
p_m	probability for mutation
$\mathbf{P}_{par}$	parents matrix
p_r	probability for crossover
$\mathbf{P}_s$	sensor influence matrix
r, $\mathbf{r}$	rank of an individual, rank vector
r_{cp}	ratio between the number of children and the population size (n_c/n)
t	generation number
$\mathbf{ub}$	vector with upper bounds for the design variables
$U(a, b)$	uniformly distributed random number within the limits $[a, b]$
W	section modulus
x	design variable, gene
$\mathbf{x}$	design variable vector, chromosome vector
$\mathbf{X}$	regression matrix
$\mathbf{X}_{dp}$	matrix of design variable values of the design points for the RSA
y, $\mathbf{y}$	response value, response value vector
$\bar{y}$, $\bar{\mathbf{y}}$	approximate response value, approximate response value vector
$\mathbf{Y}_{dp}$	matrix of response values of the design points for the RSA

Symbols	Description
$z,\ \mathbf{z}$	objective function value, objective function value vector
$z,\ \mathbf{z}$	alternatively: generalized cost function value, vector of generalized cost function values
α	coefficient of thermal expansion
$\beta,\ \beta$	regression coefficient, regression coefficient vector
ε	error term
σ, σ	standard deviation, standard deviation (vector) for mutation
σ	normal stress (see context!)
τ	shear stress
ω	eigenfrequency

Indices

Index	Description
x	scalar value
$\mathbf{x}$	vector
$\mathbf{X}$	matrix
$(\ )^T$	matrix transpose
$\oslash$	mean value

1 Introduction

1.1 Background

The search for better solutions is one of the essential driving elements in cultural evolution of mankind. At all times men tried to make things better, no matter what level they have already achieved. On the contrary stagnation has often been and still is one of the main risks for future of a culture.

In the context of engineering or mathematics a formalized form of appearance of this search for better solutions or even the best possible solution is known as optimization. And here it has recently gained even more importance due to severely increased competition resulting from shortened product cycles and globalized markets.

Whenever engineers set off designing a structure, a component, or a complex system, a general goal is almost certainly to achieve an optimal design. The goal of an optimal design is easily claimed, but the complexity of real world structures or systems as well as their similarly complex application scenarios offer considerable objections for realizing this goal. There are two major challenges for performing an optimization task. The first one is to gain a thorough and complete concept of what optimality means with respect to the considered design task. That means that all relevant aspects have to be reflected in the objectives and constraint formulation. Neglecting a single relevant constraint could render the solution useless. It also means that all relevant design parameters and load cases have to be identified. Often systems are so complex, that this is not a simple task at all. There may be load cases that cannot be derived from the primary task of a structure like e.g. people stepping on the structure. Or load case are not known precisely or so numerous that only a selection can be taken into account.

The second major challenge for an optimization task is selecting or finding a appropriate optimization method. There is a huge variety of optimization methods available. They run from stochastic methods to deterministic methods. The choice is mainly driven by the classification of the design task.

Looking on optimization in engineering practice, especially structural optimization, optimization is typically applied in the final phase of product design when the general concept or configuration of a structure or system has already been defined. Structural optimization then often means adapting 'only' sizing parameters having no influence on the overall concept or configuration of the design. That does not diminish the importance of optimization. It still can make the difference between outstanding performance and a useless solution. Sometimes it is even absolutely necessary because otherwise no feasible solution could be found at all.

Nevertheless the potential improvement is thus limited by decisions already made in earlier development stages. It is therefore desirable to apply optimization in the early design phase dealing with the general configuration. At present the driving forces in the early design process are not automated optimization methods but mainly the experience of the design team, intuition, and last not least thorough contemplation over the design task by the responsible engineers.

What are the reasons for the current limited application of optimization in design? Among a broad spectrum of often very specific reason two are worth to be highlighted, which both relate to the nature of the early design stages. One is the difficulty to transform the inherent decision structures of the early design phase into computer executable simulation models. And the other one is the deficient generality of many common optimization methods. As an example of a complex structure the x-ray view of the wing box of a commercial aircraft wing is shown in figure (1.1). It gives a good impression about the complexity of the respective design task and the associated optimization problem.

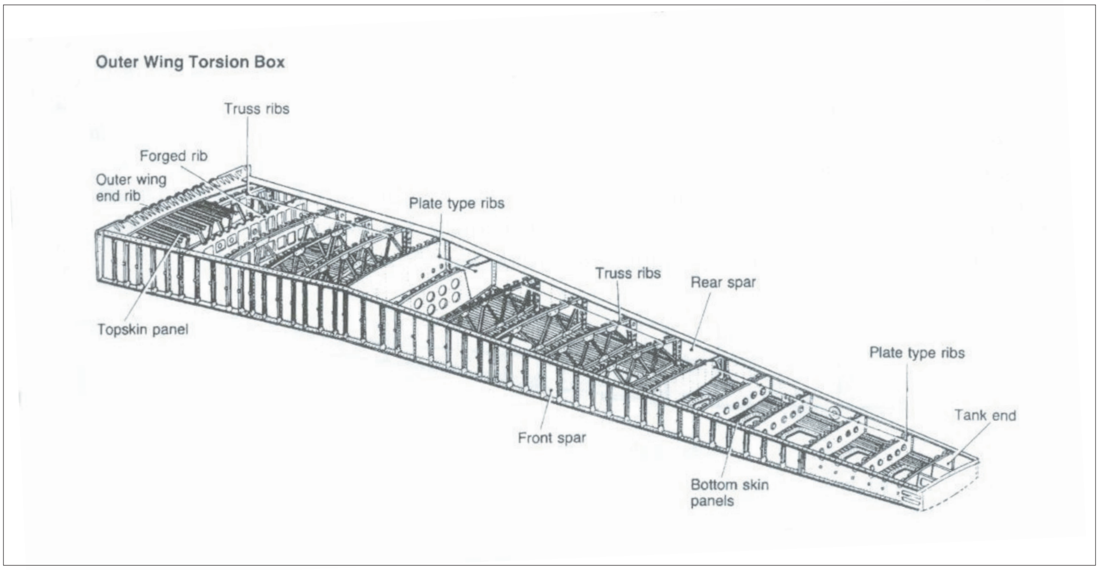


Figure 1.1: X-ray view of the wing box of a commercial aircraft wing (picture taken from ([Fie99]))

The design task of such a wing box includes aspects like the geometric and structural architecture, equipment and component allocation, material selection, and the selection of certain types and numbers of structural members. Sometimes this also means to consider a varying number of design variables, e.g. adding a new stiffening element to a structure may also require additional design parameter for dimensioning it. Also the decision between general structural concept like e.g. a stringer stiffened design or a sandwich design for the skin panels may result in complete different set of design variables. Similarly for each substructure an additional decision tree may unfold.

Such configuration design optimization tasks are typically characterized by an underlying widely branched decision tree and combinatorial problems. With respect to the potential optimization algorithms this means that they must be capable of handling truly discrete design variables and a varying number of design variables. Referring to the example above for a stiffened structure there are discrete variables like the type and number of stiffeners as well as continuous variables like the cross-section dimensions of the stiffeners. Also the number of design variables changes depending on the number and type of stringer. Most conventionally applied gradient based optimization methods do not provide the required generality and are often already ruled out by the presence of discrete variables.

Another aspect is that these kind of problems are mostly multiobjective and multidisciplinary tasks. For the example of the wing there are aerodynamical objectives, structural objectives, and also cost objectives incorporating aspects related to material, manufacturing and maintenance costs. Another example is the design of adaptive structural systems, for which both structural objectives as well as objectives related to the control system have to be pursuit. Multiple objectives are not a problem per se, the difficulties start if the different objectives are incommensurable or conflicting e.g. a high stiffness of a structure vs. minimal mass. The most common conflict is the one between some performance characteristics of the structure or system and its cost. With reference to the optimization algorithm this means that it should be capable of handling multiple objectives. Although by employing aggregating functions single objective algorithms can be easily adapted for this problem class, it should be clearly stated that the preferred result for a multiobjective problem is the complete Pareto-frontier and not only a single Pareto-optimal point. Only having the Pareto-frontier at hand the interactions between the different objectives become apparent to the decision maker enabling better design decisions.

Finally independent from aspects concerning optimization algorithms let quickly address the second problem for applying optimization in the early design stage. Optimization is always a twofold problem of having a suitable optimization method and, often not so obvious, of providing a suitable optimization model. Especially in the early design phase, when the general structural architecture is to be chosen, it is often very difficult to transform the respective decision tree or combinatorial problems into computer executable simulation models. Often alternative structural concepts require completely different FEM models, or with respect to e.g. manufacturing cost models are hardly available at all. Without going into further detail it can be stated that for this field of optimzation a considerably high effort in modelling is required, which often keeps optimization methods from being viewed as a viable option. Therefor a compromise has to be found between applying optimization as early as possible in the design process and the resulting costs to do this.

Concerning this work it aims at the first obstacle to provide an optimization method general enough to be applied in the early design phase. In the following paragraph the goals of this work are defined explicitly.

1.2 Goals of this Work

The goal of this work has been to develop, implement and evaluate an optimization methods that is general and flexible enough to be applicable to an utmost range of problems. The main emphasis for this algorithms is not efficiency but generality. It aims especially at the application at configuration problems in the early stages of the design process and thus should correspond to the needs and requirements discussed above.

In detail the optimization algorithm to be developed should therefore be

- applicable to nonlinear constrained optimization problems
- capable of simultaneously handling discrete and continuous design variables
- capable of handling disconnected design spaces
- capable of handling a varying number of design variables
- capable of handling multiple objectives and providing the Pareto-frontier
- efficient enough in terms of computation cost and time to be applicable to real world engineering problems

According to these requirements currently existing algorithms are discussed and evaluated. In consequence it has been decided to use the concept of an *evolutionary algorithm* (EA) as starting point. As stated above the algorithm to be developed should not cause computational cost that would be prohibitive for many real world engineering problems. EAs, though, generally suffer from high computational cost. So further goals have been to develop perspectives to increase the efficiency. Within this work it is therefore aimed at improving operators or developing additional features that decrease the computational effort.

Furthermore the new algorithm is to be evaluated with suitable benchmark examples to verify the basic functioning of specific elements and its performance. With respect to its performance it is also to be compared to another state-of-the-art algorithm. Finally it is to be applied to design problems from the field of lightweight structures.

1.3 Overview of this work

In the following chapter the basics relevant for development of the *Genetic Algorithm for Multicriteria Engineering* (GAME) are introduced. First the general definition of an optimization task is introduced and an overview of existing optimization algorithms is given. These algorithms are discussed with respect to the requirements stated above and it is derived why an evolutionary algorithm has been chosen as the basis for GAME. Furthermore the basics of evolutionary algorithms are introduced. As GAME is a multiobjective algorithm also some fundamentals of multiobjective optimization are described. Finally the basics of response surface methods are explained since those are an integral part of GAME.

A thorough investigation on evolutionary algorithms for multiobjective optimization has been performed. The results of this study about this relatively young field of research are given in chapter (3). Up to date methods and representative algorithms of the current state of art are discussed.

In chapter (4) GAME is explained in detail. The flow chart is introduced and the specific operators are explained. Special emphasis is put on the fitness assignment, the constraint handling, as well as the integration of the response surface methods. Concluding some aspects with reference to implementation and parallelization are given.

In chapter (5) specific issues concerning the performance and characteristics of GAME are investigated by numerical experiments. With these benchmark problems it is verified if GAME meets the development goals. Among these issues are the influence of the different algorithm parameters, the performance of the chosen constraint handling method compared to conventional penalty function methods, the effect of the RSA integration, and finally the overall performance compared to another prominent example of a multiobjective EA, the Nondominated Sorting Genetic Algorithm II (NSGA II).

In chapter (6) GAME is applied to several structural design problems. First GAME is applied to the optimal design of actively damped beam and plate structures. Here the optimal positioning and sizing of the piezo actuators had to be determined while simultaneously determining different control loop parameters. The second problem is the optimal design of a stringer stiffened plate, in which the optimal configuration for a maximum buckling load is to be determined. The last problem is the design of a high precision CFRP beam which is subject to structural as well as thermal loads. Finally a summary and conclusions are given.

2 Basics

2.1 General Definition of an Optimization Task

The idea of optimization is to achieve the best possible design of a system or process in terms of one or more criteria within the scope of given requirements and limited resources. The criteria which serve as a measure of a design's quality are called *objectives*. The requirements a design has to satisfy are called *constraints*. The quantities which influence or define a certain design are called *design variables*. The general formulation of a optimization task is given in (2.1) :

$$\begin{aligned}
&min\ \mathbf{z} = \mathbf{f}(\mathbf{x}),\ \mathbf{f} = [f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_{n_{ob}}(\mathbf{x})] \\
&\text{with: } \mathbf{x} = [x_1, x_2, ..., x_{dv}] \\
&\text{subject to:} \\
&\mathbf{gf}(\mathbf{x}) \leq 0,\ \mathbf{gf} = [gf_1(\mathbf{x}), gf_2(\mathbf{x}), ..., gf_{n_{ic}}(\mathbf{x})] \\
&\mathbf{hf}(\mathbf{x}) = 0,\ \mathbf{hf} = [hf_1(\mathbf{x}), hf_2(\mathbf{x}), ..., hf_{n_{ec}}(\mathbf{x})] \\
&x_{i_{lb}} \leq x_i \leq x_{i_{ub}},\ i = 1, 2, ..., n_{dv},\ i \in \mathbb{N}
\end{aligned} \tag{2.1}$$

The function or functions $\mathbf{f}(\mathbf{x})$ reflecting the functional correlation of the objectives and the design parameters are called objective functions. Accordingly $\mathbf{gf}(\mathbf{x})$ and $\mathbf{hf}(\mathbf{x})$ are called inequality constraint and equality constraint functions. The boundaries of the design space $\mathbf{lb} = [x_{i_{lb}}]$ and $\mathbf{ub} = [x_{i_{ub}}]$ are known as side constraints. They limit the allowed range of the design variables $\mathbf{x}$ and thus define the design space $\mathbf{D}$. Designs which satisfy all constraints are called *feasible* designs, those, who do not, *infeasible*. All feasible solutions $\mathbf{x}$ form the so called feasible set $\mathbf{D_f}$:

$$\mathbf{D_f} = \{\mathbf{x}\epsilon\mathbf{D}|\ \mathbf{gf}(\mathbf{x} \leq 0)\ \wedge\ \mathbf{hf}(\mathbf{x}) = 0\} \tag{2.2}$$

If only one objective function exists, the problem is called a *single objective problem*, if more than one objective functions exist, a *multiobjective problem*. For the latter case additional aspects arise, which will be explained in chapter (2.4).

The different aspects and complexity of such an optimization task in the field of structural optimization are illustrated for an example of a cantilever beam.

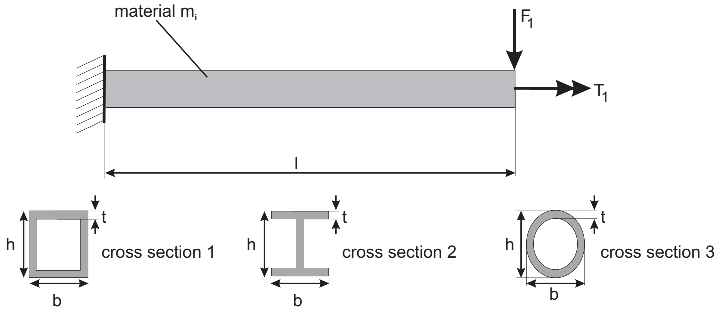


Figure 2.1: Example: Cantilever beam subject to two load cases

The cantilever beam shown in figure (2.1) is subject to two load cases, a tip force $(\mathbf{F}_1)$ and a torsional moment $(\mathbf{T}_1)$, which do not occur simultaneously.

The goal is to minimize the mass and the tip deflection of a cantilever beam that is able to withstand the two load cases. I.e. for neither load case the maximum bending stress $\sigma_b(\mathbf{x})$ occurring anywhere in the beam should exceed the allowable stress $\sigma_p(\mathbf{x})$ of the respective material and the maximum shear stress $\tau_s(\mathbf{x})$ not the allowable shear stress $\tau_p(\mathbf{x})$ accordingly. Furthermore its first eigenfrequency ω_1 is required to be above a certain limit $\hat{\omega}$.

For designing the beam different cross section cs (cs_1 - box, cs_2 - I-section, cs_3 - ellipse) and different materials mat (mat_1 - steel, mat_2 - aluminium, mat_3 - quasi-isotropic CFRP) are available. For each respective cross section the dimensions can be varied in terms of height h, width b, as well as wall thickness t of the respective cross section. These three variables cannot be chosen independently from each other, e.g. the wall thickness is limited by the height and width.

In accordance with equation (2.1) the optimization task can be formulated like below:

$$\begin{aligned}
&min\ \mathbf{z} = [z_1, z_2] = \mathbf{f}(\mathbf{x}) = [m(\mathbf{x}), u_t(\mathbf{x})]\\
&\text{with: } \mathbf{x} = [mat, cs, h, b, t]\\
&\text{subject to:}\\
&gf_1(\mathbf{x}) = \sigma_b(\mathbf{x}) - \sigma_p(\mathbf{x}) \leq 0,\\
&gf_2(\mathbf{x}) = \tau_s(\mathbf{x}) - \tau_p(\mathbf{x}) \leq 0,\\
&gf_3(\mathbf{x}) = \omega_1(\mathbf{x}) - \hat{\omega} \leq 0,\\
&gf_4(\mathbf{x}) = t - b/2 \leq 0,\\
&gf_5(\mathbf{x}) = t - h/2 \leq 0,\\
&and\\
&1 \leq mat \leq 3,\\
&1 \leq cs \leq 3,\\
&0 < t, b, h.
\end{aligned} \tag{2.3}$$

Although small, this example covers already a large part of the challenges that can occur in structural optimization. First there are two conflicting objectives. It is intuitively clear that the minimization of both objectives simultaneously will not be possible. Thus there will be no unique solution but an set of best compromise solutions, the Pareto optimal set. So this is a decision problem, for which the designer has to perform a trade-off analysis and finally take a decision to which objective he addresses more priority. The potential algorithms must be capable of dealing with multiple objectives and provide the Pareto optimal set.

Among the 5 design variables the three dimensioning variables h, b, and t are continuous, but mat and cs are truly discrete. Hence this problem is a combinatorial problem requiring the potential optimization algorithm to be capable of searching truly discrete design spaces.

Finally objectives and constraint function both include nonlinear functions, requiring the optimization algorithm to be able to work with nonlinearities.

Although a huge variety of optimization algorithms exist, it will become clear in the following that this task is a considerable challenge and it is not easy to find an algorithm satisfying all the above requirements.

2.2 Optimization Methods - Overview

The methods for solving the optimization problem defined in chapter (2.1) are various. Within this thesis and probably most common in engineering in general, an optimization method is understood as an automated, iterative process consisting of two main elements: a *simulation model*, which provides objective- and constraint function results for a given set of design variables, and an *optimization algorithm*, that provides some kind of steering logic for changing the design variables in order to improve the solution. Generally this iterative process sets off with an arbitrarily chosen start point or set of points. After evaluating this initial guess by employing the simulation model, the optimization algorithm then changes the design variables based on some internal logic. This is the essential task of the algorithm. The new design or set of designs are again evaluated and the result is presented to some kind of stopping criteria, which is usually a convergence criteria. If this criteria hints that an optimum has been achieved the process is stopped, otherwise the next iteration is started. In figure (2.2) this fundamental concept of nearly all optimization methods is illustrated.

The availability of an simulation model is an essential prerequisite for optimization process. In the context of optimization a decisive question for the simulation model are the computational cost. Dependant on the chosen algorithm a large number of evaluations is required. The resulting high computational cost can therefor be prohibitive for viewing an optimization as a viable design tool at all.

The other essential part is the optimization algorithm. Although the history of mod-

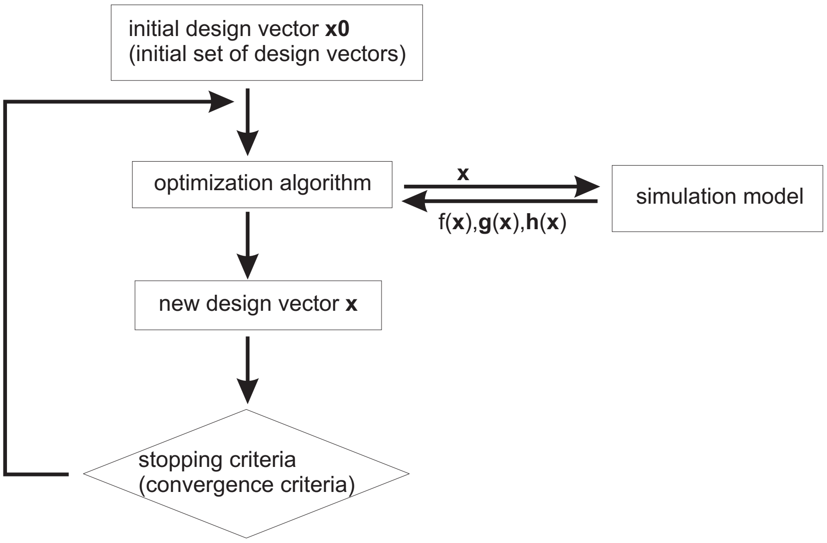


Figure 2.2: Basic flow chart for an automated, iterative optimization method

ern optimization is relatively short, approximately 30 years, a considerable amount of optimization algorithms have been developed. There are many categories which can be used to classify the algorithms. Two very basic perspectives from which one can look on the variety of algorithms are the *application level* and the *algorithm level*. Criteria on application level are e.g.:

- generality: the range of problems, an algorithm can be applied on (linear, nonlinear, unconstrained, constrained, continuous, discrete)
- efficiency: the computational cost, number of function evaluation, computation time
- robustness and reliability: the probability, that an algorithm solves the problem successfully in the context of slight deviations of the algorithms assumptions

Criteria on the algorithm level are e.g.:

- Usage of derivatives: derivative free methods, gradient methods, methods employing second order partial derivatives
- Usage of approximations: algorithms that work directly with objective and constraint functions, algorithms that work with approximation of objectives, constraints or derivatives
- Treatment of constraints: Integration of constraints by setting up a substitute objective function (penalty transformation) or direct solution of the constrained problem
- nature of the search process: deterministic, stochastic

In figure (2.3) examples of the most common representatives of optimization algorithms are shown. Since most problems in engineering optimization are nonlinear, pure linear algorithms like linear programming or the Simplex-Method are skipped. The examples are ordered according to the following categories: sequential unconstrained optimization techniques, methods solving the nonlinear constrained problem directly, methods using approximation models, and gradient free methods.

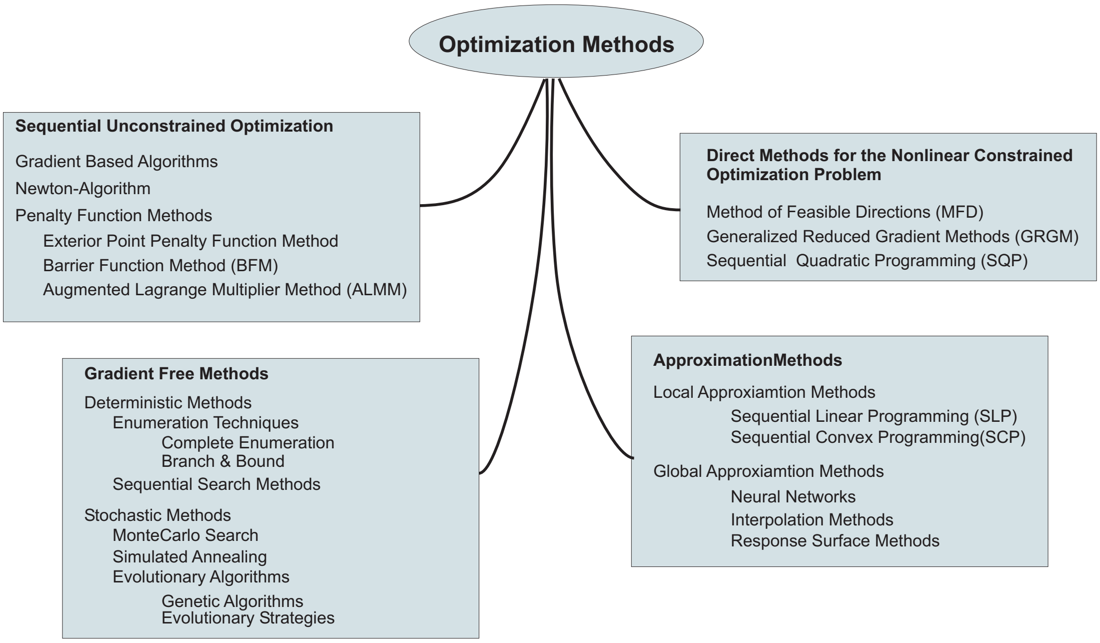

Optimization Methods
Sequential Unconstrained Optimization
Gradient Based Algorithms
Newton-Algorithm
Penalty Function Methods
Exterior Point Penalty Function Method
Barrier Function Method (BFM)
Augmented Lagrange Multiplier Method (ALMM)
Direct Methods for the Nonlinear Constrained Optimization Problem
Method of Feasible Directions (MFD)
Generalized Reduced Gradient Methods (GRGM)
Sequential Quadratic Programming (SQP)
Gradient Free Methods
Deterministic Methods
Enumeration Techniques
Complete Enumeration
Branch & Bound
Sequential Search Methods
Stochastic Methods
MonteCarlo Search
Simulated Annealing
Evolutionary Algorithms
Genetic Algorithms
Evolutionary Strategies
ApproximationMethods
Local Approxiamtion Methods
Sequential Linear Programming (SLP)
Sequential Convex Programming(SCP)
Global Approxiamtion Methods
Neural Networks
Interpolation Methods
Response Surface Methods

Figure 2.3: Overview of the most common optimization methods

A detailed introduction to the above mentioned algorithms can be found in [BSS94], [Van84], [Kir94], and [ea82] for the mathematical algorithms, in [Gol89] and [Rec73] for the Evolutionary Algorithms, and in [BD87] for the Response Surface Approximation. The available algorithms are discussed in the following subsection with respect to previously stated requirements for GAME.

2.2.1 Discussion with Respect to the Intended Field of Application

When looking at this broad range of optimization algorithms one can notice a goal conflict between generality and efficiency, i.e. the spectrum of problems to which the algorithm can be applied to and the respective computational costs. And this trade-off is directly linked to the extent the respective algorithms exploit and use information about the design space.

The most efficient representatives like e.g. SQP exploit first and second order derivative information. This information is used to determine **a priori** in which search direction to move and how far in order to create an improved solution. For this search method it is sure that each newly created solution will be a better one. Less efficient algorithms like evolutionary algorithms (EAs) do not employ gradient information and rely only on objective function values. New solutions are created stochastically, although certain heuristics are used. Only after the new solution are evaluated it can be assessed **a posteriori** if the new designs turn out to be an improvement or not. Consequently a considerably higher number of trials are required.

But in order to be able to exploit higher order derivative information and thus achieve this high efficiency, SQP and similar algorithms make higher demands on the characteristics of the design space and the objective functions. E.g. for SQP the objective and constraint functions are required to be continuously differentiable. Therefore they cannot be applied to discrete or combinatorial problems. Also for a reliable convergence to the global optimum the objective and constraint functions are required to be convex. Furthermore 'noisy' objective- or constraint functions may cause problems for the proper computation of the gradients. Gradient free methods like EAs offer relief of all these requirements and therefore are applicable to a much wider spectrum of problems.

The goal of this thesis is to develop an optimization algorithm for application in the early design phase namely configuration design. The specific requirements have been posted in chapter (1.2). Although it may seem difficult to choose an algorithm that serves as a starting point for the desired algorithm due to the huge variety of algorithms, in fact it is not. One of the most dominant requirements is the capability to handle discrete design variables, this rules out all calculus based algorithms relying on gradient information in some way. So the choice is left to gradient free. Although orders of magnitude less efficient, the most efficient methods among them in a relative sense are branch and bound methods and evolutionary algorithms. One essential difference is that branch and bound is a deterministic method guaranteeing that the global optimum will be reached whereas EAs are stochastic in nature. Thus the respective solutions generally cannot be reproduced exactly and also no guarantee is given, that the solutions are the true optimum. Therefore often the term *near optimal solutions* is used.

Branch and bound methods originate from linear integer programming. Although recently expansions to nonlinear problems have been developed, they rely mainly on linearization techniques. An essential element for their working principle is that they employ a continuous relaxation of the integer problem, i.e. the integer constraint is dropped for the design variables and the problem is treated as an equivalent continuous one. But this is not possible for truly discrete variables like e.g. material selection, so this is an serious drawback with respect to the intended scope of applicability.

EAs do not suffer from such limitations. Concerning the design parameter space, EAs can be adapted to any specific requirement ranging from truly discrete variables, disconnected search spaces to a varying number of design variables. Also they virtually make no demands with respect to the characteristics of the objective functions. So from the viewpoint of generality EAs are an excellent starting point for the desired algorithm. Furthermore from an programming point of view they are of low complexity and very easy to implement.

The most serious drawback with respect to the applicability to real world problems are the high computational costs, a characteristic common to all gradient free methods. But in contrast to other gradient free methods like the completely stochastic Monte Carlo search EAs offer a significantly higher efficiency, since they are only a semi-stochastic method that features an inherent logic to direct the search. Nevertheless, again, it has to be stated that compared to gradient based algorithms the efficiency is orders of magnitude lower. With typical population sizes and number of generations easily thousands of objective function evaluations are reached. Furthermore as will be shown in chapter (5.3) the required computational cost for EAs rise with the number of design variables, a characteristic e.g. gradient based algorithms not necessarily show. In case their gradients are computed by differential quotients, they also share this characteristic though.

Nevertheless in conclusion evolutionary algorithms seem to be most appealing among the gradient free methods. For the intended field of application the ability to handle truly discrete variables is essential. Thus a lower efficiency is accepted in favor of the gained generality. So the concept of EAs have been chosen as the basic starting point for the development of GAME.

But keeping this major drawback in mind it is one of the important development goals to increase the efficiency and lower the computational burden of the conventional EA to broaden the field of potential application. This goal is pursuit by especially adapted operators, the integration of RSA methods (chapter 4.8) and, last but not least, the grain-parallelization of GAME on a cluster (chapter 4.11). Although grain-parallelization does not reduce the overall computational effort (still the number of function evaluation remains the same), it will reduce the overall computation time, maybe the even more important criterium.

2.3 Evolutionary Algorithms

Based on the above reasoning evolutionary algorithms have been decided to be the core element of GAME. Therefore their origin, basic working principle as well as their two most popular representatives are introduced here in detail. Furthermore their applicability as an optimization tool is discussed to illustrate advantages and disadvantages. EAs are a relatively new type of algorithm being present in the scientific community for approximately 30 years. The first representatives of this class of search algorithms have been Fogel's Evolutionary Programming (EP) ([FOW66]), Rechenberg's Evolutionary Strategies (ES) ([Rec73]), and Holland's Genetic Algorithms (GA) ([Hol75]). Today a huge variety of different implementations exists. The most relevant for optimization in the engineering context are the ES and GAs.

2.3.1 Role Model: Natural Evolution

When trying to find an equivalent to the concept of optimization in nature, one will quickly hit on the principle of evolution, or more popular known as 'The survival of the fittest' (Charles Darwin, 1859). Presented in a simplified way, the individuals in a population, may it be animals or other organism, compete to pass their genes on to the next generation. The fundamental thesis of evolution is that individuals which are adapted better to the challenges of their natural environment (meant in a very general way), will have a higher probability of reproduction. A measure of how good an individual is adapted to its specific environmental requirements is the so called *fitness*. Fitness is an aggregated measure that can be composed of very different things and is by no means limited to pure physical strength as sometimes popularly assumed. As various as the challenges are in nature as various the factors determining the individuals fitness can be: e.g. the variety of food an individual is able to live on, the ability to camouflage against potential predators, the ability to withstand germs, viruses, or diseases, the ability to adapt to environmental changes. According to the theory of evolution the number of offspring an individual is able to generate in its life is dependent on its fitness. The selection of an individual as a parent is directly dependent on its fitness.

The characteristics of an individual are stored in the genes on its chromosomes. During reproduction two main processes are essential for the generation of the offspring: recombination (or crossover) and mutation. The first process exchanges sections of the chromosomes between the parents and therefore provides a mixture of the parent's characteristics. The second, mutation, is a pure random change of the genes and so will result in some completely new characteristics not present in any of the parents. Selection as well as reproduction are stochastic processes by nature. The reason, why the average fitness is improving while the cycle of selection and reproduction continues, is that first selection favors high fitness individuals and second high fitness individuals have a higher probability to generate even fitter offspring.

2.3.2 Working Principle

Evolutionary algorithms now try to incorporate the principles of natural evolution and genetics into a numerical search and optimization algorithm. In figure (2.4) the general flowchart of an EA is given. The most striking difference to traditional optimization methods is that EAs work with a set of design points, the population, simultaneously and not a single design point. At the beginning an initial population is generated at random. Based on the objective- and constraint-function values a fitness is determined for each individual. According to a given selection scheme individuals are selected as parents. In general these selection schemes rely on the fitness as a measure for the probability to be selected as a parent. The offspring is generated by applying recombination and mutation operations. Finally the next generation is set up based on

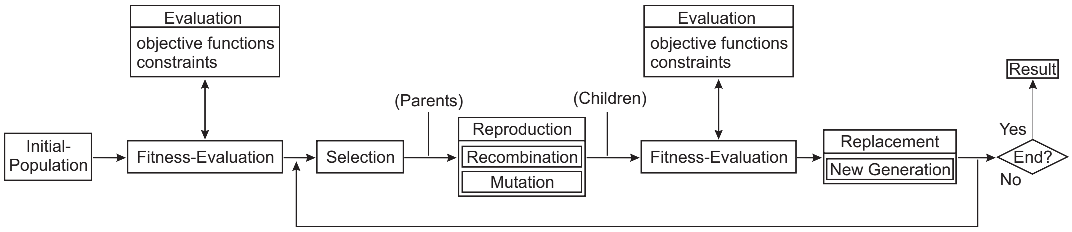


Figure 2.4: General EA flowchart

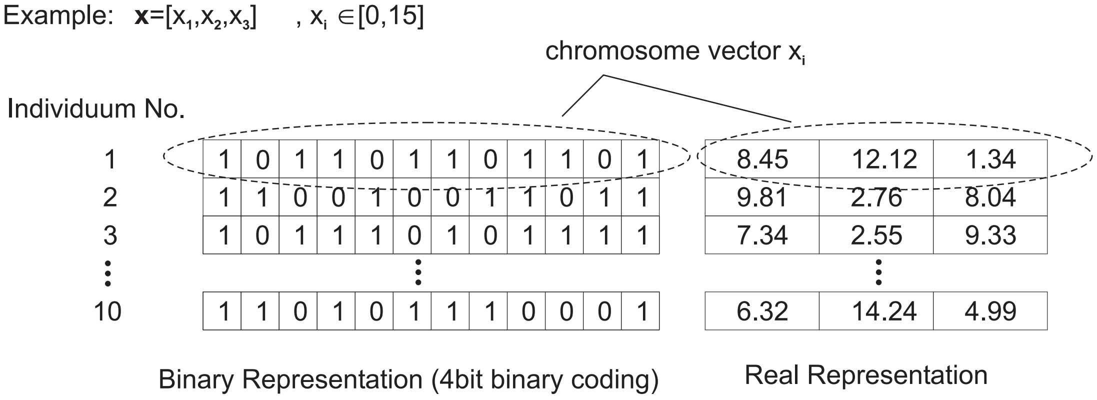
a replacement scheme, that controls which individuals of the last generation survive and which will be replaced by the offspring. This process is repeated until some stopping criteria is met.

The interaction of the different operators to function as a search method for better solutions is based on two principles: *exploration and exploitation*. Recombination and mutation generate new solutions and thus promote the exploration of the design space. Selection and replacement provide means that better solutions are favored for reproduction and survival and thus promote the exploitation of the design space. In the following paragraphs important aspects and the different operators are introduced in more detail.

Design Variable Representation

In nature the 'design information' is stored in *genes* using a 4 letter alphabet consisting of 4 bases (Adenin, Cytosin, Guanin, Thymin). These genes are stored on *chromosomes*. The place, where a gene is located on the chromosome, is called a *locus*. The different values a certain gene can have are called *alleles*. The representation of an individual on the chromosome level is called the *genotype*, the real design the *phenotype*. Based on this natural example the design variables in EAs are typically aggregated in so called '*strings*' as equivalents of the chromosomes. In practise, however, these chromosome strings or chromosome vectors are nothing else but the conventional design vector and thus are designated here also as $\mathbf{x}$. Nevertheless, it not necessarily is the original design vector. In classical *genetic algorithms* (GA) e.g. a binary encoding is used following nature's example of using a small alphabet.

Without loss of generality these strings can be assumed as line vectors, though other representation like tree structures have been published (Koza 1992). A population therefore is a set of vectors, which usually can be represented as a matrix. In figure (2.5) an example population of 10 individuals is plotted both as a binary and a real-valued representation.


Example: **x**=[x_1,x_2,x_3] , $x_i \in$[0,15]
chromosome vector x_i
Individuum No.
Binary Representation (4bit binary coding)
Real Representation

Individuum No.															
1	1	0	1	1	0	1	1	0	1	1	0	1	8.45	12.12	1.34
2	1	1	0	0	1	0	0	1	1	0	1	1	9.81	2.76	8.04
3	1	0	1	1	1	0	1	0	1	1	1	1	7.34	2.55	9.33
⋮						⋮								⋮	
10	1	1	0	1	0	1	1	1	0	0	0	1	6.32	14.24	4.99

Figure 2.5: Example for binary and real-valued representation of a population

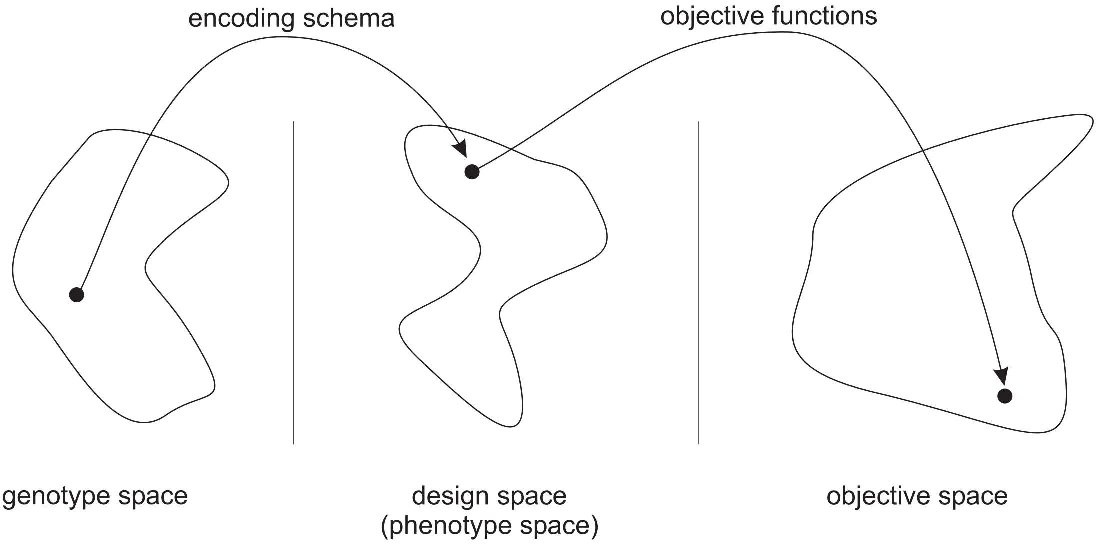


Figure 2.6: Relation between the genotype space, design space, and the objective space

Because of the chromosome encoding here a third space, the genotype space, has to be taken into account in addition to the design space and the objective space known from classical methods. The relation between the three spaces is illustrated in figure (2.6). In case of real-valued representation of the design parameters the genotype space and the design space are actually identical. *Evolutionary strategies* (ES) work with real-valued strings, but recently real-valued parameter representation is also common in GAs. The different codings reflect the different philosophies behind the respective algorithms. GA try to closely mimic natural genetics while ES put more emphasis on the evolutionary process in the phenotype space. Also they were originally aimed at continuous parameter optimization.

A binary coding also means a discretization of the design space even for the continuous design variables. For those a desired accuracy has to be chosen. A higher accuracy means that more bits have to be taken for the respective design variable. This results in a longer string vector which again increases the dimensionality of the search space. In a real coding discrete variables can be introduced as integers. This has the advantage that only a single locus is needed independent of the resolution.

Fitness Assignment

EAs are inherently unconstrained search algorithms. The only measure taken into account is fitness. In general fitness is a scalar measure representing the probability of being chosen as a parent. Since better solutions should be favored, fitness should reflect the overall quality of a certain design. A better design should have a higher fitness. This is certainly very simple for an unconstrained single objective problem. The problem arises in the presence of constraints. For determining fitness fit_i of each individual $\mathbf{i}$ both objective and constraint information have to be mapped into a scalar measure:

$$fit_i(\mathbf{x}_i) = fitnessfunction([\mathbf{f}(\mathbf{x}_i), \mathbf{gf}(\mathbf{x}_i), \mathbf{hf}(\mathbf{x}_i)])$$

Because constraint handling is an essential aspect for EAs, the most common methods will be introduced in chapter (2.3.3) in more detail. Once an aggregated scalar quality measure is at hand the most common methods to convert this into a fitness value, i.e. selection probability, are linear mapping, exponential mapping, or ranking methods. Exponential mapping provides an additional mean for increasing the selection pressure, since it increases the fitness of the better individuals in a superproportional way. Ranking methods are based on rearranging the individuals in order from best to worst. Thus the fitness is based only on the relative position in the ranking list, the absolute differences in objective values are ignored. Requiring only objective- and constraint-function values and no partial derivations for computing the fitness values is one of the essential advantages of EAs. It is this that makes them applicable to discrete or combinatorial problems without any problems.

Selection and Replacement

The selection operator takes care of the selection of individuals as parents. The replacement operator is responsible for the building of the next generation, i.e. it decides which individuals of the old generation and which of the children are chosen for the next generation. Selection and replacement are very much like mirrored operators. While selection generally favors the best individuals for reproduction, replacement generally hinders the worst individuals from being passed to the next generation. Not necessarily both operators need to regard fitness information. For the successful working of an EA it is only necessary that one of the two operators works based on fitness information and favors better solutions in some way. E.g. GAs employ a stochastic selection with a fitness based probability while the replacement operator ignores fitness totally and simply replaces the complete population with the offspring. In order not to loose the best ones, often a so called *elitism operator* is added, which directly copies the best individuals into the next generation. Vice versa in ES, here the selection operator ignores fitness while the replacement operator selects only the best solutions from the former generation and the offspring for the next generation.

The two most common realizations for selection operator in GAs are the *Roulette Wheel selection* and the *tournament selection*, which are illustrated in figure (2.7). For the Roulette Wheel selection all individuals are assigned an arc section on a virtual roulette wheel whose size is proportional to their fitness. So the probability to be selected in a virtual spin of the wheel is directly proportional to the fitness. For the tournament selection first two or more individuals are randomly chosen from the population as participants oft the tournament disregarding their fitness values. Then simply the one with the higher fitness is selected as a parent. The decisive parameter is the tournament size t_s, it directly controls the selection pressure towards high fitness solutions. A low tournament size promotes population diversity but also leads to a slower convergence.

The selection operator as well as the replacement operator play a crucial role for successful working of the algorithm. They are the elements which are responsible that the better individuals prevail in the course of generation and therefore that the search process drives the population to better solutions.

Recombination

The recombination or crossover operator creates new individuals, the children, by simply exchanging genes between the parents. Sections of the design strings are randomly cut out at one or more positions and exchanged between the parents. This process is illustrated for the example of a two-point-crossover (2 cuts) for a binary coded GA in figure (2.8). For real coded EAs it works equivalently. Although not directly obvious, in case of a binary coding crossover is in fact a mutation. Because the binary strings can be cut anywhere, i.e. in general not a the locus where the coding of one design variable ends, the resulting individual features design variable values, which have not been present in neither of its parents. This has to be kept in mind for judging the dominant role of recombination in traditional GAs.

For real-valued representations also arithmetic recombination operators are common, e.g. children are formed by a linear combination of the parents (2.4).

$$\begin{aligned} &parent\ individuals: && \mathbf{x}_i, \mathbf{x}_j \\ &child: && \mathbf{c} = a\mathbf{x}_i + (1-a)\mathbf{x}_j,\ a = U([0,1]),\ (random\ number) \end{aligned} \tag{2.4}$$

The basic reasoning behind recombination is that combining the genes of good individuals has a higher probability of generating an even better offspring. Since it

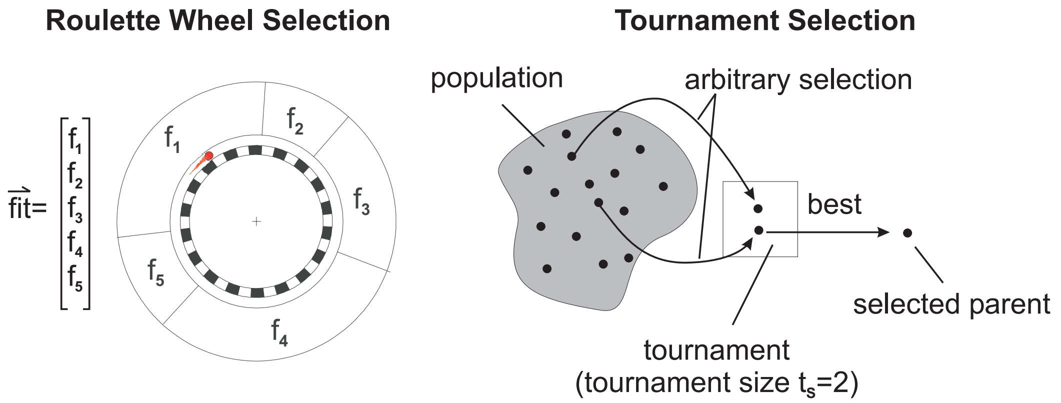


Figure 2.7: Roulette Wheel Selection scheme

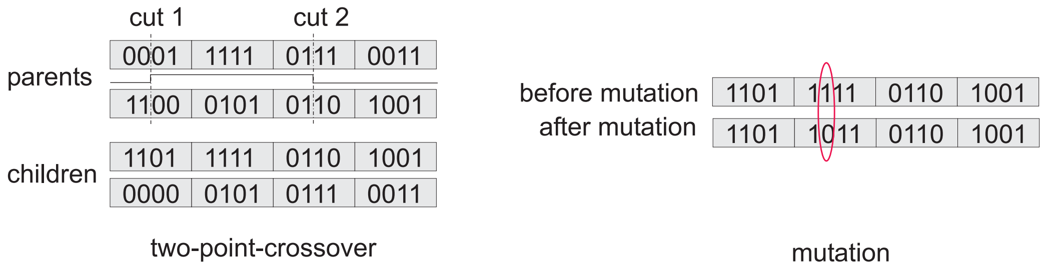


Figure 2.8: Illustration of the crossover operator (two-point-crossover) and mutation operator for GAs

recombines design variable settings which are already present in the current population, recombination can be characterized as an exploiting operator. Thus it increases convergence speed but also inherits the danger of reducing population diversity. In general real recombination may only seem reasonable for tasks with subsystems, i.e. groups of design variables that define more or less independent elements of the complete design.

Mutation

The mutation operator changes design variables arbitrarily within the allowed design variable domain and a certain probability distribution. Again this operator is applied only with a certain probability. Mutation is clearly an exploring operator. It generates designs not present in the current population and thus helps to explore the design space. By adding new solutions to the population it also promotes population diversity. Thus this operator is crucial for the search process.

For binary coding mutation simply means flipping 0 to 1 or vice versa at randomly chosen loci. Real codings allow for more sophisticated mutation operators which can also inherit adaptive elements. In ES the mutation operator adds normally distributed random changes with a certain standard variation (σ) to the design variables of an individual:

$$\begin{aligned}
&\mathbf{x} = [x_1, x_2, ..., x_{n_{dv}}],\ \sigma = [\sigma_1, \sigma_2, ..., \sigma_{n_{dv}}],\\
&\mathbf{x}^{mut} = \mathbf{x} + \mathbf{N}(\mathbf{0}, \sigma) = [x_1 + N(0,\sigma_1), x_2 + N(0,\sigma_2), ..., x_{n_{dv}} + N(0, \sigma_{n_{dv}})],\\
&\quad N(0,\sigma_i):\ \ normally\ distributed\ random\ number\\
&\qquad\qquad\quad mean\ value:\ 0\\
&\qquad\qquad\quad standard\ deviation:\ \sigma_i,\ i = 1..n_{dv}
\end{aligned} \tag{2.5}$$

Thus ES mutation favors small changes on the phenotype level, or to put it in other terms: natura non fecit saltus - nature does not jump. The underlying reasoning is that changing a good individual slightly has a higher chance of achieving an improvement than arbitrary jumps in the design space. A drawback of this mutation operator is that it keeps ES from being applicable to discrete problems, since it requires continuous design variables or at least quasi-continuous design variables.

The ES mutation is no static operator, several rules for adapting the standard deviation σ_i during the evolution run have been developed. The most simple is starting with large σ_i at the beginning and then let them gradually decrease over the generations. This reflects the assumption, that at the beginning it makes sense to broadly explore the search space and later, when the search can be assumed to have converged around an optimum, a smaller standard deviation allows a more accurate determination of the optimum. A typical realization of this is a linear decrease of the standard deviation:

$$\sigma_i(gen) = \frac{\sigma_{i_{n_{gen}}} - \sigma_{i_0}}{n_{gen}}(gen) + \sigma_{i_0},\ i = 1, 2, ..., n_{dv} \tag{2.6}$$

Here σ_{i_0} is the initial standard deviation vector, $\sigma_{i_{n_{gen}}}$ the desired final standard deviation vector. Another approach is to change the standard deviations in dependence on the success of the search process like suggested by Schwefel ([SHF94]). Finally the adaption of the standard deviation can itself be made a subject of the evolutionary process. The chromosome vector is augmented by the standard deviations and so a self adaption of the standard deviations is performed by the evolutionary process:

$$\begin{aligned} &augmented\ chromosome\ vector: && \mathbf{x} = [x_1, x_2, ..., x_{n_{dv}}, \sigma_1, \sigma_2, ..., \sigma_{n_{dv}}] \\ &twofold\ mutation\ process: && \sigma_i^{mut} = \sigma_i e^{(N(0,\Delta))} \\ & && x_i^{mut} = x_i + N(0, \sigma_i^{mut}),\ i = 1, 2, ..., n_{dv} \\ & && \mathbf{x}^{mut} = [x_1^{mut}, ..., x_{n_{dv}}^{mut}, \sigma_1^{mut}, ..., \sigma_{n_{dv}}^{mut}] \end{aligned} \tag{2.7}$$

The standard deviation Δ for mutating the standard deviations σ_i is generally chosen small. The evaluation of the offspring now serves not only the purpose to judge if the mutation was successful but also if the currently applied standard deviation presents a good choice. For this reason the standard deviations are mutated first and the offspring is created with the new standard deviations. Otherwise the new standard deviation were not applied until the next generation and a direct correlation to the results of the current offspring is lost. Nevertheless this correlation between the quality of the offspring and the quality of the chosen standard deviation is rather loose, since the actual change $\Delta\mathbf{x}$ employed for the creation of the offspring could have been generated by many values of the standard deviations. It is still a random process.

The self adaption introduces a meta-level in the evolutionary search process. In order to judge a certain standard deviation value as a successful rule a statistically relevant number of 'experiments' has to be performed. Thus a higher number of experiments, i.e. mutations, is required compared to the standard ES. In practice this translates to a higher number of children and population size and so means increasing the computational cost even further. If this self adaption is really a promising approach is investigated in chapter (5.5).

The combination of real coding and this ES mutation operator has shown to be a very efficient search tool in comparison to a binary coding. Especially it allows substantially smaller population sizes than binary coded EAs. For this reason it has been decided to incorporate it in GAME. In consequence GAME also features a real coded variable representation. Due to the presence of discrete variables in GAME it can of course only be applied to the continuous variables.

2.3.3 Constraint Handling in Evolutionary Algorithms

Since EA are inherently unconstrained methods additional effort is required to apply them to constrained problems. A convincing solution similar to calculus based algorithms directly solving the nonlinear constrained problem like SQP is not available up to date. In the following the most common approaches for constraint handling are summarized.

Lethal Factor

The most simple and naive approach is to reject all infeasible solutions, so that violated constraints pose a kind of *lethal factor*. Although this approach, sometimes also called *'death penalty'*, is very easy to implement, it will work only for very simple, loosely constrained problems. It will certainly fail for highly constrained problems where a high percentage of the population or even the complete initial population is infeasible.

Penalty Function Methods

Most popular are *penalty function methods*, which originally were developed for unconstrained mathematical algorithms like gradient methods. The objective function f and the constraint functions gf and hf are aggregated into a single function, in which the constraints are added to the objective as a "penalty" given by the penalty function γ. The constrained problem as given in equation (2.1) is replaced by a substitute problem:

$$min\ \varphi(x, \alpha) = f(\mathbf{x}) + \gamma(\mathbf{gf}(\mathbf{x}), \mathbf{hf}(\mathbf{x}), \alpha) \tag{2.8}$$

I.e. if the constraints are violated the objective function value is degraded. The problem is solved for this aggregated function. Depending on the setting of α the optimization will converges more or less far away from the true solution. A setting of α causing a large penalty leads to a dominating influence of the constraints, so rather the constraints are minimized than the objective. Thus the penalty parameter α is adapted at each iteration for decreasing the height of the penalty in order to allow convergence closer to the true optimum. Theoretically for infinite iteration the optimization should converge to the true solution. In reality one will achieve only near optimal solutions, which can be considered a disadvantage of this approach. The two most common representatives are the *exterior point penalty function method* and the *barrier function method*. The latter one requires feasible solutions, which cannot be guaranteed in EAs. So commonly the first one is employed. A possible realization of γ is:

$$\gamma(\mathbf{gf}(\mathbf{x}), \mathbf{hf}(\mathbf{x}), \alpha) = \alpha(\sum_{i=1}^{n_{ic}} min(0, gf_i(\mathbf{x}))^2 + \sum_{i=1}^{n_{ec}}(hf_i(\mathbf{x}))^2),\ \alpha > 0 \tag{2.9}$$

As a disadvantage might be considered that when the optimization is stopped after a finite number of iteration (which will always be the case in practise) the solution might be an infeasible solution of the original problem.
Penalty transformation methods have the advantage that they are very general methods. One difficulty with these methods is to determine a proper penalty so that on the one hand feasible solutions are achieved and one the other hand these solutions are not too far from the true optimum. Certain problems can react extremely sensitive to the choice of the proper penalty parameters.

Constraint Handling by Specially Adapted Representation and Operators

Another approach is using a special representation of the design variables and adapted operators. One realization of this approach can be found in *GENOCOP* developed by Michalewicz ([Mic99]), but his solutions requires all constraints to be linear. Here first all linear equality constraints are eliminated resulting in an elimination of an equal number of design variables. So part of the search space is already removed before the search. The remaining constraints are all linear inequalities. In order to handle these GENOCOP first starts with a completely feasible initial population, which can either be found by sampling the design space or be provided by the user. During the evolutionary run then especially adapted reproduction operators are applied which alter the individuals by linear combinations, so the feasibility is ensured due to the linearity of the constraints. But it is this restriction to linear constraints that hinders this approach from being applicable to a broad range of problems.

Constraint Tournament Selection

Constraint tournament selection is an approach developed by Deb ([Deb00]), which is based on the principle that feasible solutions are always superior to infeasible solutions. In a binary tournament the following rules are applied:

1. if one solutions is feasible and the other one infeasible, the feasible solutions is selected
2. if both solutions are feasible, the one with the better objective function is selected,
3. if both solutions are infeasible, the one with the lower overall constraint violation is selected

The computation of the 'overall' constraint violation requires an aggregation of all constraints and thus a normalization in order to avoid bias. One attractive feature of this approach is that no penalty parameter is necessary. Furthermore this approach is generally applicable and easy to implement.

Multiobjective Methods

Finally there exist several approaches that try to redefine the constrained problem as a multiobjective problem like e.g. Fonseca and Fleming ([FF98a]). As this method is employed by GAME, a detailed description is given later in chapter (4.4).

Discussion

In order to evaluate the performance the different constraint handling methods (lethal factor approach, penalty function approach, multiobjective approach) are compared by numerical experiments in chapter (5.4). For the multiobjective approach the one of Fonseca and Fleming has been selected. The constraint tournament selection has been skipped as it basically included in Fonseca's and Flemings's approach. The results showed that the multiobjective approach is the most promising one and therefore has been selected for GAME.

2.4 Multiobjective Optimization

Multiobjective problems feature certain characteristics which differ considerably from single objective problems. Because these special characteristics are directly employed in the fitness assignment operator as well as in the constraint handling method of GAME, they are described in the following in detail. In contrast to single objective problems the quality of a design has to be compared in several dimensions, which is the fundamental challenge of multiobjective optimization.

In practice the objectives are generally conflicting, i.e. improving one objective has degrading effects on another objective. For this case there is no single solution representing the minimum for all objectives simultaneously. But there exists a set of solutions, the Pareto-optimal set, which are optimal in a wider sense. These solutions have in common that no other solutions exist in the design space that are simultaneously better in all objectives. In the following section the specific definitions, characteristics, and principles of multiobjective optimization are introduced in detail.

2.4.1 Concept of Optimality for Multiobjective Optimization

It is essential for any optimization to have a measure to discriminate between better and worse. In the context of multiobjective optimization the concept of *domination* can be used for this purpose:

Pareto-Dominance: A design vector $\mathbf{x}_1$ is said to dominate a design vector $\mathbf{x}_2$, $\mathbf{x}_1 \preceq \mathbf{x}_2$, if and only if the corresponding objective vector $\mathbf{z}_1 = \mathbf{f}(\mathbf{x}_1)$ is partially less than $\mathbf{z}_2 = \mathbf{f}(\mathbf{x}_{"})$, $\mathbf{z}_{1\,p} < \mathbf{z}_2$, i.e.:

$$(\forall\, i\epsilon\{1, .., n_{ob}\} : z_{1_i} \leq z_{2_i}) \;\wedge\; (\exists\, i\epsilon\{1, .., n_{ob}\} : z_{1_i} < z_{2_i}) \tag{2.10}$$

Sometimes equation (2.10) is referred to as weak Pareto-dominance. In contrast to this strong Pareto-dominance is defined as:

(Strong)-Pareto-Dominance: A design vector $\mathbf{x}_1$ is said to strongly dominate a design vector $\mathbf{x}_2$, $\mathbf{x}_1 \prec \mathbf{x}_2$, if and only if

$$\forall\, i\epsilon\{1, .., n_{ob}\} : z_{1_i} < z_{2_i} \tag{2.11}$$

Using definition (2.10) the two solution vectors $\mathbf{z}_1$ and $\mathbf{z}_2$ are indifferent to each other, if neither $\mathbf{z}_1$ dominates $\mathbf{z}_2$ nor vice versa. To illustrate the concept of *Pareto dominance*, or simply *dominance*, one can take a look at the solution $\mathbf{c}$ in figure (2.9). Solution $\mathbf{c}$ dominates the solutions $\mathbf{d}$, $\mathbf{e}$, and $\mathbf{f}$ in the upper right rectangle, whereas solution $\mathbf{c}$ is dominated by the solutions $\mathbf{a}$ and $\mathbf{b}$ in the lower left rectangle. The other solutions like e.g. $\mathbf{g}$ are indifferent to $\mathbf{c}$.

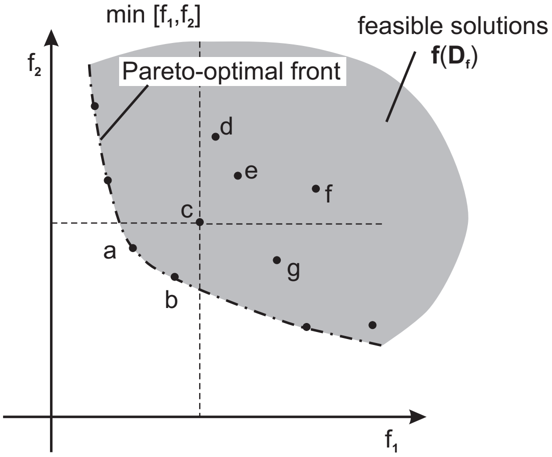


Figure 2.9: Example of a problem with 2 objective functions for illustrating the concept of Pareto optimality

Based on this concept of domination, an optimality criterion can be derived for multiobjective optimization problems. Again referring to figure (2.9), one can easily observe, that there is a set of solutions, which are not dominated by any other. They are so called non-dominated solutions and form a non-dominated front, which is indicated by dash-dotted line.

Non-dominated design, Pareto-optimal design: A design vector $\mathbf{x}^*\epsilon\mathbf{D}_f$ is called non-dominated with respect to a set $\mathbf{X} \subseteq \mathbf{D}_f$, if and only if $\nexists \mathbf{x}\epsilon\mathbf{X} : \mathbf{x} \prec \mathbf{x}^*$. If

$\mathbf{X} = \mathbf{D}_f$ the design $\mathbf{x}^*$ is called a **Pareto-optimal** design.

In figure (2.9) e.g. the solution $\mathbf{a}$ represents a non-dominated solution. Similarly this can be formulated for sets of design vectors:

Non-dominated set, Pareto-optimal set: A set of designs $\mathbf{X}^* \subseteq \mathbf{X} \subseteq \mathbf{D}_f$ is called the non-dominated set of $\mathbf{X}$, if and only if $\forall \mathbf{x}^* \epsilon \mathbf{X}^* : \nexists \mathbf{x} \epsilon \mathbf{X} : \mathbf{x} \prec \mathbf{x}^*$.

The fundamental goal of a multiobjective optimization is to determine the Pareto optimal set. It is important to notice that the evaluation of a design as a non-dominated design is a relative measure with respect to finite set of designs.
The Pareto-optimal set is also called the Pareto-front or -frontier. Going back to figure (2.9) and assuming the gray marked area would resemble the complete set of feasible solutions, then the solutions $\mathbf{a}$ and $\mathbf{b}$ are Pareto-optimal. The set of all Pareto-optimal solutions forming the Pareto-frontier is marked as a dash-dotted line. Of course the above phenomena apply only for conflicting objectives. If the objectives are not conflicting one can easily show that the multiobjective problem degenerates to a single-objective one. The Pareto-optimal set then comprises just a single optimum.

2.4.2 Search and Decision Making for Multiobjective Optimization Problems

As noted above in a strict sense the solution of a multiobjective optimization problem is determining the Pareto-optimal set. Generally this Pareto-optimal set contains an infinite number of solutions. In practice the designer has to come up with a single design in the end and therefore has to take a decision which design is best suited. So a multiobjective optimization problem is in contrast to a single-objective optimization problem not only a search, but also a *decision making problem*. The multiobjective optimization problem is therefore a twofold-problem: search and decision making ([Hor97]). The search problem is associated to the determination of the Pareto-optimal set, the decision making problem is associated to the problem to decide in favor for a compromise solution, which implies a trade-off analysis between the different objectives. For taking this decision the designer has to employ higher level preference information.
The way how to combine this two elements can be classified into three categories according to ([Hor97], [HM79]):

- **Decision making before search (a priori methods)**: Based on some kind of preference information the different objectives are aggregated in a single objective. A decision on the relative importance of the respective objectives is taking prior to the search. The problem degenerates to a single-objective problem.

- **Search before decision making (a posteriori methods)**: The search is performed without any preference information, all objectives are equally important. The goal is the determination of the Pareto-optimal set. Based on some kind of preference information the designer then takes a final decision which compromise solution is to take. Although the Pareto-optimal frontier generally consists of a infinite number of solutions, in practise it will only be possible to determine a finite set of Pareto-optimal solutions with respect to computational effort. So besides finding Pareto-optimal solutions for this category the search process has to maintain a sufficiently diverse set of Pareto-optimal designs. This second goal is essential for providing a profound basis for trade-off-analysis and a final design decision.
- **Decision making during search (interactive methods)**: During the search process the designer can articulate and change preferences depending on the current results of the search.

2.4.3 Traditional Approaches for Solving Multiobjective Optimization Problems

Most classical multiobjective optimization methods fall into the first category "preference decision making prior search" and therefore convert the multiobjective optimization problem into a single-objective one. The most common aggregating approach is weighted sum approach, which is briefly introduced and discussed in the following:

Weighted Sum Approach

The weighted sum approach is the most simple and intuitive way of aggregating multiple objectives into a single one. The composite objective function is formed as linear combination of all objectives:

$$\begin{aligned}
&min\ \bar{z} = \bar{f}(\mathbf{x}) = \sum_{i=1}^{n_{ob}} w_i f_i(\mathbf{x}),\ \sum_{i=1}^{n_{ob}} w_i = 1, w_i \geq 0 \\
&\text{subject to:} \\
&\mathbf{gf}(\mathbf{x}) \leq 0,\ \mathbf{gf} = [gf_1(\mathbf{x}), gf_2(\mathbf{x}), ..., gf_{n_{ic}}(\mathbf{x})] \\
&\mathbf{hf}(\mathbf{x}) = 0,\ \mathbf{hf} = [hf_1(\mathbf{x}), hf_2(\mathbf{x}), ..., hf_{n_{ec}}(\mathbf{x})] \\
&x_{i_{lb}} \leq x_i \leq x_{i_{ub}},\ i = 1, 2, ..., n_{dv},\ i \in \mathbb{N}
\end{aligned} \tag{2.12}$$

The w_i are the weighting factors for each objective. In figure (2.10) the approach is demonstrated for an example of two objective functions. The feasible region of the

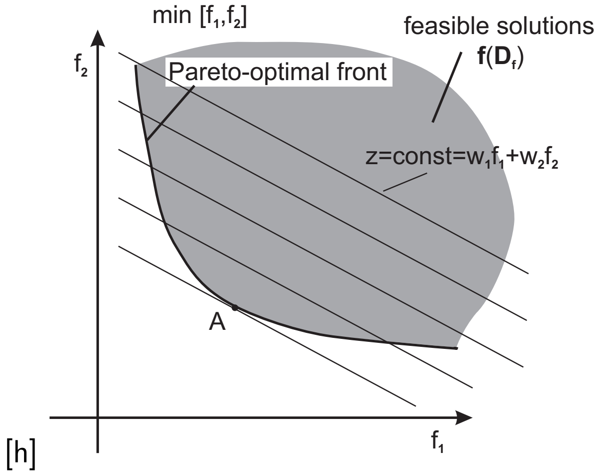


Figure 2.10: Illustration of the weighted sum approach for an example with two objective functions

design space D_f is plotted in the objective space. For a given choice of the weighting factors w_1 and w_2 the isolines for constant $\bar{z}$-values are plotted, which form straight lines. The gradient of these isolines is determined by the ratio of the weights w_1/w_2. Graphically the optimization process can be imagined as moving a straight line parallel to the isolines towards lower $\bar{z}$-values until it is tangent to the Pareto-Frontier. The tangent point A is then the resulting optimal design.

The decision making problem here is the choice of the weights. They represent the relative preference between the objectives, which is not always easy to determine. Other classical methods are the constraint approach, the weighted-metric approach, and the Benson-method.

Discussion of the Classical Methods

The biggest advantage of these methods are directly connected to the transformation into a single-objective problem. For solving the problems the large variety of algorithms for single-objective optimization at hand can be used without modifications. The possibility of the using sophisticated single-objective algorithms also provides the advantage of a high efficiency and well defined convergence properties.

Disadvantages are that generally only one Pareto-optimal solution is obtained per optimization run. In order to obtain multiple Pareto-optimal solutions to approximate the complete Pareto-front multiple runs with different weight settings have to be performed. An additional difficulty here is that in general a uniformly spaced weight vector not necessarily results in uniformly spaced solutions on the Pareto-front.

Another serious drawback is that some methods like the weighed sum approach have considerable difficulties to obtain all Pareto-optimal solutions for non-convex objective spaces, so it may not be possible to obtain the entire Pareto-front. In figure 2.11 on the right an example for such a case is illustrated. The point A is clearly part of

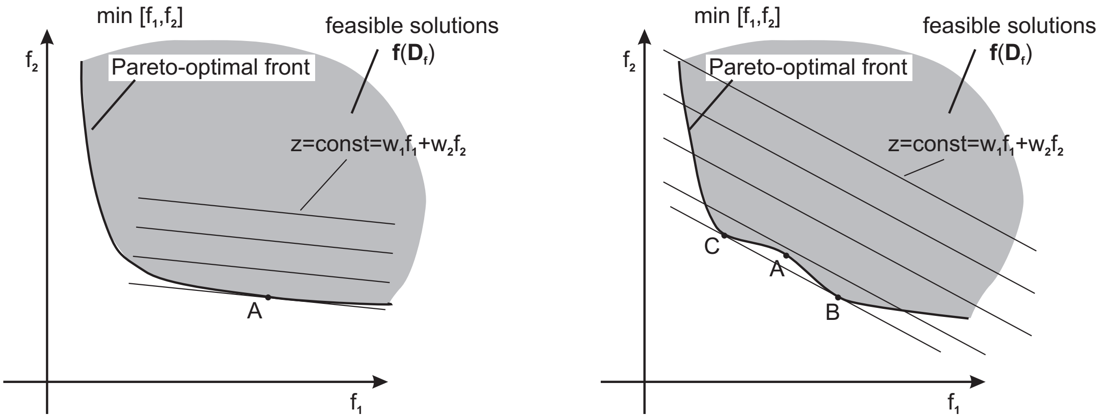


Figure 2.11: Illustration of potential problems with the weighted sum approach

the Pareto-front, but no choice of the weights w_1 and w_2 will result in A being the minimum of the composite function $\bar{f}$.

A general drawback of these methods is that the preferences have to be articulated prior to the search and thus without or only limited knowledge about the interaction of the objectives. In figure 2.11 on the left an example is plotted, in which a slight shift in weights towards the objective f_1 results in a huge improvement without considerable sacrifices in f_2. But this kind of trade-off is only possible when having the complete Pareto-front present. The choice of the weights might perfectly reflect some reasoning about the relative importance of the objectives, but is blind to the design possibilities that open up in awareness of the interaction between the objectives.

For this reasoning it has been decided to use an approach of the second category 'Search before decision making' in GAME. The population based search enables an advantageous combination of this approach with specially adapted search operators, which are introduced in chapters (3.2.1) and (4.4).

2.5 Response Surface Approximation

Response Surface Approximations (RSA) are a method for building empirical approximation models for an unknown functional correlation between the design- (or input-) variables and the response- (or output-) variables of a process or system. RSAs are used as an integral part of the optimization algorithm presented in this thesis in order to increase the efficiency compared to conventional evolutionary algorithms. Therefore this method is introduced here in detail.

In equation (2.13) the general concept is shown. $\bar{f}$ is the response surface function, $\mathbf{x}$ the design vector and $\bar{y}$ the approximated response value.

$$\bar{y} = \bar{f}(\mathbf{x}), \mathbf{x} = [x_i], i = 1..n_{dv} \tag{2.13}$$

Generally these approximation models or response surfaces (RS) are deduced from a set of design points and their response values D_{dp} in equation (2.14) which are resulting either from experiments or simulations.

$$D_{dp} = \{\mathbf{X}_{dp}, \mathbf{Y}_{dp} \mid \mathbf{X}_{dp} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{n_{dp}} \end{bmatrix}, \mathbf{Y_{dp}} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n_{dp}} \end{bmatrix} \tag{2.14}$$

In figure (2.12) an example is shown with two variables x_1 and x_2 and one response value. On the left side the available set of design points and response values is shown, on the right the resulting response surface is shown. How to choose these design points and how to choose a suitable approximation model for building the RSA are the major topics of a more general theory called Response Surface Methodology (RSM), which e.g. is presented in [MM95]. Originally RSM is a methodology for setup and data exploration of experiments, in [MM95] it is defined as a "collection of statistical and mathematical techniques useful for developing, improving, and optimizing processes". The RSM encompasses methods to choose the design points (design of experiments - DOE), methods to choose suitable model functions and also methods to find the optimal designs. The advantages of having an RSA available are obvious. The outcome of experiments can now be predicted without actually performing a single, often costly and time consuming experiment.

Though originally coming from the field of experiments, RSA methods are recently also applied in the field of optimization. Although here the functional correlation between design variables and objective values is exactly known, the application of RSA can be advantageously. The cost of optimizing a complex structure or multidisciplinary system is mainly determined by the computational effort for evaluating the respective simulation models. If the actual evaluation of such a large simulation model is replaced by a simple mathematical approximation model like a 2nd order polynomial enormous time savings can be achieved. Beside saving computation costs, it is also good to have well defined gradient information. Many problems are characterized by noisy objective functions spoiling the successful working of gradient based optimization algorithms. For this cases the smoothening character of the RSA is welcomed.

In the context of this work the RSA is employed for increasing the efficiency of a multiobjective evolutionary algorithm. The set of design points for building the response surface are constituted by the current population. Therefore no DOE methods need to be applied and are not introduced here. Furthermore only simple 2nd order polynomials are be employed as approximation functions. The reason for this is, that the RSA are spanned only over small parts of the design spaces due to a clustering process (see chapter 4.8). In the following sections the basics for building these approximation models is introduced in detail.

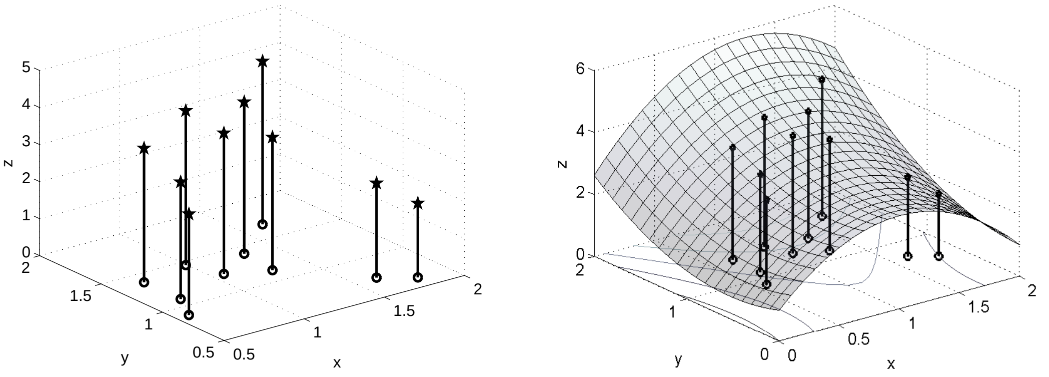


Figure 2.12: Illustration of building an RSA from a set of design points for an examples with 2 design variables x_1 and x_2

2.5.1 General Concept

The response value or objective function value y of a system with the n_{dv} design variables may be given as:

$$y = f(\mathbf{x}), \mathbf{x} = [x_i], i = 1..n_{dv} \tag{2.15}$$

The true system response y may then be approximated by a RSA with the value $\bar{y}$:

$$y = \bar{y} + \varepsilon,\ \bar{y} = \bar{f}(\mathbf{x}), \mathbf{x} = [x_i], i = 1..n_{dv} \tag{2.16}$$

$\bar{f}$ is the RSA function, ε is the approximation error. Often the design variables are normalized for numerical reasons, which can be assumed without loss of generality.

2.5.2 Second Order Polynomial Approximation Functions

In order to built a RSA an approximation model function has to be chosen. In the scientific literature a broad variety of model function can be found. Sometimes the physical background of the problem is known, so an adapted model function can be chosen. In other cases, it is aimed at an utmost generality for covering a broad range of possibly objective functions. Here mainly polynomial or spline models are applied. Another research direction pursued at the LLB is to optimize the choice of the model function ([Gle04]). But the most common model function are low order polynomials like linear functions and second order polynomials:

$$\bar{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_{n_{dv}} x_{n_{dv}} = \beta_0 + \sum_{i=1}^{n_{dv}} \beta_i x_i \tag{2.17}$$

$$\bar{y} = \beta_0 + \sum_{i=1}^{n_{dv}} \beta_i x_i + \sum_{i=1}^{n_{dv}} \sum_{j=i}^{n_{dv}} \beta_{ij} x_i x_j \tag{2.18}$$

The unknown coefficients β_0, β_i and β_{ij} are determined by linear regression. With these polynomials a considerably broad spectrum of objective function classes can be approximated sufficiently. In figure (2.13) the possible response surface shapes for RSA using 2nd order polynomials are shown for an example with 2 design variables. In the context of optimization having a RSA by itself does not help much. The goal is to find the optimal design by applying an optimization method on the RSA. And here another highly desirable advantage of using 2nd order polynomials is apparent. When using 2nd order based algorithms like SQP (Sequential Quadratic Programming, [Sch85b]) the optimization process converges in a single iteration and performs reliably.

2.5.3 Linear Regression

The approximation model in equation (2.17) basically resembles exactly the linear regression model. The name "linear" draws from the fact that the model is a linear function of the unknown coefficients. Higher order models can still be treated by linear regression by substitution of the higher order terms. As an example consider the 2nd order polynomial model for 2 design variables:

$$\bar{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \beta_{12} x_1 x_2 \tag{2.19}$$

Let $x_3 = x_1^2$, $x_4 = x_2^2$,and $x_5 = x_1 x_2$ as well as $\beta_3 = \beta_{11}$, $\beta_4 = \beta_{22}$,and $\beta_5 = \beta_{12}$. Of course x_3, x_4, and x_5 are no independent variables, their are functions of x_1 and x_2. This substitution is not exactly necessary, but serves a better illustration, that the higher order model can still be solved by linear regression. Because the linear regression a is based on a certain set of design points with fixed choices for x_1 and x_2, there will be certain x_3, x_4, and x_5. With this substitution (2.19) is turned into a linear regression model:

$$\bar{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 \tag{2.20}$$

The regression coefficients β_k $(k\epsilon[1, n_{rc}])$ are now determined in a way that the error ε between the true response value y and the estimated on $\bar{y}$ is minimized. The standard procedure for this is called "least squares fit". For this the number of design points has to be at least the number of regression coefficients: $n_{dp} > n_{rc}$. For the 2nd order RSA model the number of regression coefficients n_{rc} results in:

$$n_{rc} = 1 + 2 * n_{dv} + \frac{n_{dv}!}{2!(n_{dv} - 2)!} \tag{2.21}$$

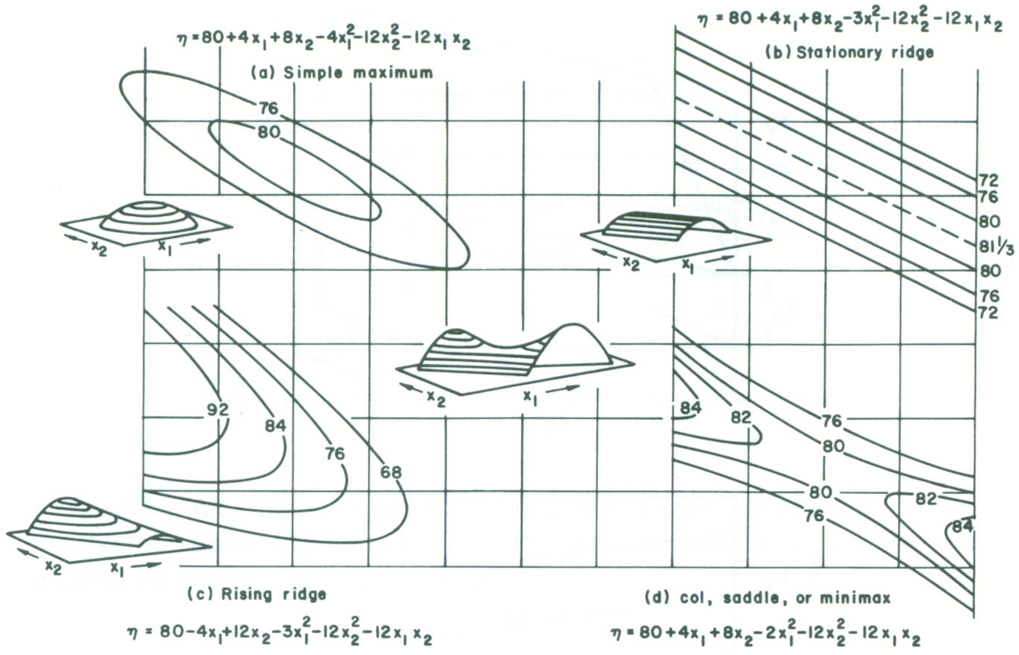

η = 80+4x₁+8x₂−4x₁²−12x₂²−12x₁x₂
(a) Simple maximum
76
80
x₂
x₁
η = 80+4x₁+8x₂−3x₁²−12x₂²−12x₁x₂
(b) Stationary ridge
72
76
80
81⅓
80
76
72
x₂
x₁
92
84
76
68
x₂
x₁
84
82
76
80
80
76
82
84
x₂
x₁
(c) Rising ridge
η = 80−4x₁+12x₂−3x₁²−12x₂²−12x₁x₂
(d) col, saddle, or minimax
η = 80+4x₁+8x₂−2x₁²−12x₂²−12x₁x₂

Figure 2.13: Possible response surface shapes for RSAs using 2nd order polynomials for an example with 2 design variables x_1 and x_2 ([BD87])

For a set of n_{dp} design point the data for computing the regression coefficients can be presented as follows:

design point no.	x_1	x_2	$\cdots$	x_j	$\cdots$	$x_{n_{rc}}$	y
1	x_{11}	x_{12}	$\cdots$	x_{1j}	$\cdots$	$x_{1n_{rc}}$	y_1
2	x_{21}	x_{22}	$\cdots$	x_{2j}	$\cdots$	$x_{2n_{rc}}$	y_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
i	x_{i1}	x_{i2}	$\cdots$	x_{ij}	$\cdots$	$x_{in_{rc}}$	y_j
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
n_{dp}	$x_{n_{dp}1}$	$x_{n_{dp}2}$	$\cdots$	$x_{n_{dp}j}$	$\cdots$	$x_{n_{dp}n_{dv}}$	$y_{n_{dp}}$

(2.22)

The index $i\epsilon[1, n_{dp}]$ denotes the ith design point $\mathbf{x}$ and its respective response value y, the index $j\epsilon[1, n_{rc}]$ denotes the jth regression coefficient. With equation (2.16) the regression model can be written as:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_{n_{rc}} x_{in_{rc}} + \varepsilon_{(i)} = \beta_0 + \sum_{j=1}^{n_{rc}} \beta_j x_{ij} + \varepsilon_i,\ i\epsilon[1, n_{dp}] \tag{2.23}$$

In matrix notation the above equation (2.23) results in:

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon$$

with:

$$\mathbf{y} = \begin{bmatrix} y_{(1)} \\ y_{(2)} \\ \vdots \\ y_{(n_P)} \end{bmatrix}, \quad \varepsilon = \begin{bmatrix} \varepsilon_{(1)} \\ \varepsilon_{(2)} \\ \vdots \\ \varepsilon_{(n_P)} \end{bmatrix}, \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{n_V} \end{bmatrix}, \tag{2.24}$$

$$\mathbf{X} = \begin{bmatrix} 1 & x_{1(1)} & x_{2(1)} & \cdots & x_{n_V(1)} \\ 1 & x_{1(2)} & x_{2(2)} & \cdots & x_{n_V(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{1(n_P)} & x_{2(n_P)} & \cdots & x_{n_V(n_P)} \end{bmatrix}$$

The least squares fit minimizes the sum of the error squares ε^2:

$$\begin{aligned} L &= \sum_{i=1}^{n_{dp}} \varepsilon_i^2 \\ &= \sum_{i=1}^{n_{dp}} \left(y_i - \beta_0 - \sum_{j=1}^{n_{rc}} \beta_j x_{ij} \right)^2 \end{aligned} \tag{2.25}$$

Equation (2.25) denotes in matrix notation:

$$\begin{aligned} L &= \varepsilon^T\varepsilon = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) \\ &= \mathbf{y}^T\mathbf{y} - \beta^T\mathbf{X}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\beta + \beta^T\mathbf{X}^T\mathbf{X}\beta \\ &= \mathbf{y}^T\mathbf{y} - 2\beta^T\mathbf{X}^T\mathbf{y} + \beta^T\mathbf{X}^T\mathbf{X}\beta \end{aligned} \tag{2.26}$$

For minimizing L the following equation has to be satisfied:

$$\left.\frac{\partial L}{\partial \beta}\right|_{\mathbf{b}=[b_0, b_1, \ldots, b_{n_V}]^T} = -2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\mathbf{b} = \mathbf{0} \tag{2.27}$$

Solving this equation for the optimal regression coefficients $\mathbf{b}$ results in:

$$\mathbf{b} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{2.28}$$

Substituting these coefficients in equation (2.18) gives the following expression for the approximated response values $\bar{y}$:

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{b} \tag{2.29}$$

The error between the true response values y_i and the approximated ones $\bar{y}_i$ at the design points are called the residuals:

$$\mathbf{e} = \mathbf{y} - \hat{\mathbf{y}} \tag{2.30}$$

3 Multiobjective Optimization with Evolutionary Algorithms

The applicability to multiobjective problems is an essential goal for the development of GAME. Since the core of GAME is an EA, the methods and possibilities to apply EAs in multiobjective optimization have been investigated in a literature research ([PR01]). The results are summarized in this chapter.
The application of EAs in multiobjective optimization still is a relatively new and active field of research. First the basic reasoning for applying EAs to multiobjective problems is discussed. Then the essential adaptations of the evolutionary operators are presented. Finally some multiobjective evolutionary algorithms (MOEAs) representing the current state-of-the-art are introduced.

3.1 Basic Reasoning for Applying Evolutionary Algorithms to Multiobjective Optimization Problems

The solution of a multiobjective optimization problem consists of multiple solutions, the Pareto-optimal set. As EAs work with set of design points, the population, anyway, the idea to use this set of design points to search for multiple solutions simultaneously is nearby. The basic thought is to adapt the EA in such a way that the population converges towards the Pareto-frontier. This is illustrated in figure (3.1). In this way an EA can determine the Pareto-frontier in a single run. This constitutes the essential advantage of applying EAs to multiobjective optimization problems. When trying to determine the Pareto-frontier with a reasonable spacial resolution with one of the conventional techniques (see chapter 2.4.3), numerous runs are required. So the main disadvantage of EAs, the higher computation cost compared to e.g. gradient based algorithms, is lessened, while keeping the advantages like being able to deal with discrete design variables.
In order to realize this basic idea certain adaptations have to be made to the standard EA implementation. As stated before in chapter (2.4.2) two main goals have to be regarded for a multiobjective search: find Pareto-optimal solutions and maintain a sufficient diversity for covering the whole Pareto-frontier.

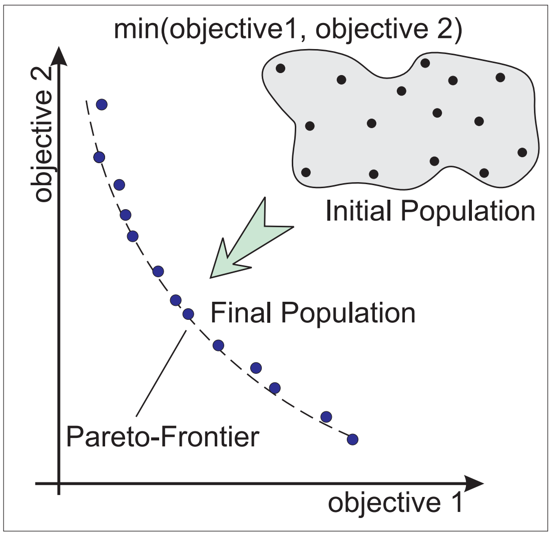


Figure 3.1: General concept of applying EAs to multiobjective problems: Convergence of the population towards the Pareto-frontier

For accomplishing the first goal, the search has to be directed towards the Pareto-frontier. The key element for driving the search in an EA is the fitness assignment, thus this operator is the key operator to be modified. In order to have the population converge towards the Pareto-frontier fitness assignment must assure that non-dominated individuals are preferred and are assigned the same fitness.

For the second goal additional information reflecting the similarity of the solutions has to be employed in some way. This can be integrated in the standard evolutionary operators like fitness assignment or selection or additional elements can be added to the standard EA.

Possible ways for these adaptations are introduced in the following section.

3.2 Key Aspects of the Adaptation for Multiobjective Optimization Problems

3.2.1 Adaptations of Fitness Assignment and Selection

In order to have the population converge towards the Pareto-front the fitness-assignment as well as the selection have to be modified. In his PhD-thesis ([Zit99]) Zitzler generally distinguishes three approaches:

Fitness Assignment with Switching Objectives

Within this class of fitness assignment schemes, not a constant fitness assignment scheme is applied for the complete population over all generation. Instead for the selection of the parents it is switched between several substitute objective functions. These different substitute objective functions can simply be the single objective functions of the original problem or they can be built by aggregating approaches like e.g. lexicographic ordering with randomly changing orders. The switching between these different substitute functions can be done for every new generation. Or the different substitute functions are applied to different subsets of the population. In fact, taken to the extremes, for the selection of each parent of a specific generation a different substitute objective function could be applied. One example for this kind of fitness assignment scheme is Schaffer's *Vector Evaluated Genetic Algorithm* (VEGA)([Sch85a]). At each generation the population is split into n_{ob}-subpopulations. In each subpopulation selection is then performed according to only one single objective without considering the other ones, in each subpopulation according to a different objective respectively. After selection the parent individuals are shuffled together again to perform crossover and mutation in the usual way. Other examples for this class are Fourman ([Fou85]), who proposed comparing individuals with regard to a specific or random order of objectives, or Kursawe ([Kur91]), who suggested assigning a probability to each objective which controls whether it becomes the sorting criterion for the next selection phase or not.

The basic thought behind all these schemes is that by repeatedly switching the objectives finally Pareto-optimal solutions are generated. Nevertheless these selection schemes have several drawbacks. They have a bias towards 'extreme' solutions. Solutions, which are rather 'compromise' designs showing equally good performance in several or all objectives will not survive in these fitness assignment schemes. This effect also known from genetics is called 'speciation', i.e. within the population certain 'species' arise which specialize in certain objectives.

Fitness Assignment with Aggregating Methods and Varying Parameters

These fitness assignment schemes are based on the traditional aggregating approaches combining all objectives into a single scalar substitute objective function, the most popular of which being the well known weighted sum approach as introduced in chapter (2.4.3). For the traditional approach the weights are held constant during the optimization run. Within this class of fitness assignment schemes the weights are changed during the run. These changes can be done systematically or at random. The different weights can be applied to different subsets of the population or even each individual can be assigned different weights. Thus the optimization process aims at different directions simultaneously. For the successful working of this approach a proper normalization of all objectives is critical. Because this is not always possible, this poses a drawback of this fitness assignment scheme. One example of this class has been published by Hajela (HLGA, [HL92]), where the weights have been explicitly included in the chromosome string.

Fitness Assignment Based on the Concept of Pareto-Dominance

The last class of fitness assignment schemes directly employs the concept of Pareto-domination for the fitness assignment. This concept has first been proposed by Goldberg in his book on genetic algorithms in 1989 ([Gol89]). Goldberg proposed a ranking scheme in which all non-dominated individuals of a population are assigned rank 1. Then these individuals are temporarily removed from the population. The non-dominated individuals of the remaining population are assigned rank 2. This procedure is repeated until the complete population is ranked. In this way it is ensured that all non-dominated individuals are assigned the highest fitness and, equally important, the same fitness. Thus the population is directly driven towards the Pareto-frontier and no artificial fix-ups are needed.
This fundamental concept of a Pareto based ranking scheme has been the basis and the inspiration for many other researchers who developed similar fitness assignment schemes for their own MOEAs. One example is the *Multi-Objective Genetic Algorithm* (MOGA) developed by Fonseca and Fleming ([FF98a]). In MOGA the rank of each individual is defined by the number of other individuals in the population by which it is dominated. So nondominated individuals are assigned rank 0 and thus the highest fitness. This approach is illustrated in figure (3.2). These Pareto based ranking schemes differ fundamentally from the above mentioned methods with respect to the fact, that fitness is determined as a relative measure. The fitness is not determined based on absolute objective function values but on the relation between each individual and the rest of the population. So no scaling or normalization is required. These Pareto based fitness assignment schemes are most popular in the current development of MOEAs. Nevertheless, in their overview on MOEAs ([FF95]) Fonseca and Fleming stated, referring to independent publications of again Fonseca and Fleming([FF93])

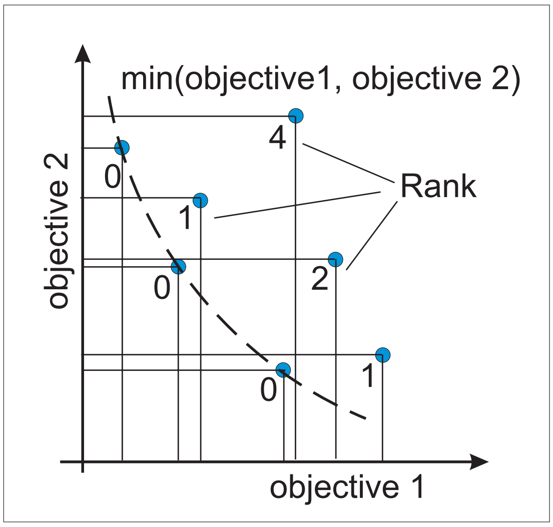


Figure 3.2: Pareto-Ranking of a sample population for an example problem with 2 objectives

as well as Horn and Nafpliotis ([HN93]): "Pure Pareto-EAs cannot be expected to perform well on problems involving many competing objectives and may simply fail to produce satisfactory solutions due to the large dimensionality and size of the trade-off surface. As the number of actually competing objectives increases, more and more of the search space can be expected to conform with the definition of Pareto-optimality, which would make the theoretical problem of finding non-dominated solutions easier. Unfortunately, in the total absence of preference information, the EA will face the impossible task of finding a satisfactory compromise in the dark, which can only happen by pure chance".

Discussion

The fitness assignment schemes that directly employ Pareto-dominance are the most direct realization of the original idea of having the population converging against the Pareto-front. Furthermore they guarantee that all nondominated solutions are rated equally and do not prefer extreme solutions. Accordingly most of the state-of-the-art MOEAs are based on this fitness assignment scheme. Thus it has been decided that GAME also employs a Pareto-ranking based fitness assignment.

3.2.2 Methods for Maintaining Population Diversity

In order to have a well founded basis for a trade-off analysis an MOEA should achieve a sufficiently uniform distribution of the solutions on the Pareto-front. In the following the main examples of methods supporting diversity are briefly introduced:

Fitness Sharing

Fitness sharing is the most commonly applied concept for maintaining diversity and has been proposed by Goldberg and Richardson ([GR87]) in 1987. It originally aims at establishing stable subpopulation in multi-modal single objective problems. The basic idea is that the fitness of a certain individual is degraded if there a many very similar individuals in the close neighborhood. The more individuals are located in the neighborhood of a certain individual, the more its fitness value is degraded. An individual $\mathbf{j}$ belongs to the neighborhood of an individual $\mathbf{i}$ if it is within a given distance, the so called niche radius, σ_{share}: $d(\mathbf{i},\mathbf{j}) < \sigma_{share}$. The distance metric $d(\mathbf{i},\mathbf{j})$ is computed by an appropriate distance measure. This distance measure can be defined in the design variable space or, more seldom, in the objective function space. The most common distance metric is the Euclidean distance, which again requires a proper normalization. The resulting *shared* fitness $fit_s(\mathbf{x}_i)$ of an individual $\mathbf{x}_i$ is defined as follows:

$$fit_s(\mathbf{i}) = \frac{fit(\mathbf{i})}{\sum\limits_{j=1}^{n} s(d(\mathbf{i},\mathbf{j}))} = \frac{fit(\mathbf{i})}{n_c(\mathbf{i})},\ i,j\epsilon[1,n] \tag{3.1}$$

The shared fitness is equal to the original fitness divided by the *niche count* $n_c(\mathbf{i})$, which is the sum of all sharing function values $s(d(\mathbf{i},\mathbf{j}))$ between the individual $\mathbf{x}_i$ and all other individuals in the population. The sharing function reflects the similarity between two individuals. If two individuals are further apart than a given niche radius σ_{share}, the sharing function value is typically zero, while within the radius the values increase with decreasing distance. A typical realization for a sharing function is:

$$s(d(\mathbf{x}_i,\mathbf{x}_j)) = \left\{ \begin{array}{l} 1 - \left(\frac{d(\mathbf{i},\mathbf{j})}{\sigma_{share}}\right)^{\alpha},\ d(\mathbf{i},\mathbf{j}) < \sigma_{share} \\ 0,\ d(\mathbf{i},\mathbf{j}) \geq \sigma_{share} \end{array} \right\} \tag{3.2}$$

Though the concept is straight forward, there are some difficulties. So a value for the sharing radius σ_{share} has to be determined, which is critical for the performance.

Crowding

This method was proposed by DeJong in his PhD-thesis in 1975 ([DeJ75]). In his model he used a so called overlapping population, i.e. that only a part G, called generation gap, is allowed to reproduce at each generation. The crowding model aiming at preserving diversity is integrated in the replacement operator. For each offspring to be inserted in the population, CF solutions are chosen at random from the population (CF is called crowding factor). From these solutions the most similar one is replaced by the offspring. Similarity is again measured by an appropriate distance metric and can be evaluated in the design space as well as the objective space. Because only similar individuals are replaced, dissimilar individuals have a higher chance of survival. In this way crowding of solutions anywhere in the search space is prevented and so diversity in the population is encouraged.

Immigration

Another method to maintain population diversity is to substitute limited parts of the population with randomly generated new individuals. Basically these new individuals can be considered as immigrants as they have no common roots with the current generation. This concept of immigration has been applied e.g. by Fonseca and Fleming ([FF98a]).

Isolation by Distance

This type of methods for maintaining diversity is based on the use of several separated and distinct populations. These populations can evolve independently and so globally diversity is supported. At a relatively low rate individuals are allowed to migrate between the different populations. This migration basically resembles the idea of immigration introduced above. This concept is e.g. employed by Poloni ([Pol95]). Similar to the concept of having multiple subpopulations is having a single population but with a specially defined spatial structure so that several niches can evolve simultaneously. E.g. Laumanns, Rudolph, and Schwefel proposed a population which was structured by an underlying graph, a two dimensional torus ([LRS98]).

Overspecification

Overspecification is an approach where an individual's chromosome contains has a redundant set of genes for each design variable, an active and an inactive set. The active part specifies the current design, whereas the inactive part has no function. But dependent on a specific decision logic the inactive part can become the active one and vice versa. Basically this concept resembles the existence of dominant and recessive alleles of diploid organisms in nature, which also feature redundant sets of chromosomes. In nature always the dominant allele will be realized unless both parents feature the recessive allele. Within this method of course an arbitrary logic can be applied for switching between the active and inactive part. The inactive part is also subject to recombination and mutation, but because the phenotype is built only based on the active set, it is not subject to the selection process. Thus it can evolve completely nondirectional and is not effected by e.g. genetic drift. Because the inactive part can become the active one at any time, new individuals can appear in the population which are not resulting from the past selection process. Thus this method helps to maintain diversity. An example for this concept of diploidy can be found in the MOEA proposed by Kursawe ([Kur91]).

Discussion

Among the above mentioned methods overspecification, isolation by distance and immigration have the disadvantage that they only indirectly support the diversity on the Pareto-frontier. They do not directly consider the current distribution of the solutions. The effectivity of these methods can therefore be regarded as low. Crowding and fitness sharing are more promising as they take into account distance information reflecting the actual distribution. But none of the methods acts directly on the distribution of solutions on the current nondominated front. Thus it has been decided to employ another methods that does so. This method is integrated in a specially adapted elitism operator which will be introduced in the next section.

3.2.3 Elitism

Elitism means that the best individuals of one generation are directly passed on to the next generation without having to 'survive' the stochastic selection and replacement process. Elitism aims at avoiding loss of the best solutions due to stochastic operators and assures a monotone convergence. In the context of multiobjective optimization the best individuals are all non-dominated individuals. As the goal of a multiobjective optimization is to determine the Pareto-frontier, it is obvious, that one wants to keep all non-dominated solutions. So the need for some kind of elitism operator is so obvious that one can wonder why first generation MOEA did not consider elitism right from the start.
Elitism can be implemented in several ways. The most basic approach to implement elitism in a multiobjective EA is to take all nondominated individuals and directly pass them into the next generation. Another way is based on the replacement operator of the ES, i.e. the old population and the offspring are merged and the n best individuals of this union set are selected to form the next generation. Alternatively an external archive can be used, in which all nondominated individuals are stored. The selection pool then comprises the population and the external archive. After reproduction the external archive is updated. The offspring is added to the archive and all members are ranked again. Those who might have become dominated are deleted. The three concepts are illustrated in figure (3.3). Nevertheless the first two alternatives which are directly transferred from single-objective optimization show some drawbacks in the context of multiobjective optimization. In contrast to single-objective problems for which the best is only one single solution, for multiobjective problems the best, i.e. the nondominated set, can constitute a large part of the population. For the first alternative of copying the best directly into the next generation such a case would mean that a large part of the new generation is already occupied and only a few new individuals of the offspring will have the chance to be enter. This is a drawback with respect to population diversity as well as convergence speed. For the second alternative of building the new generation by taking the best, i.e. the nondominated, from a union

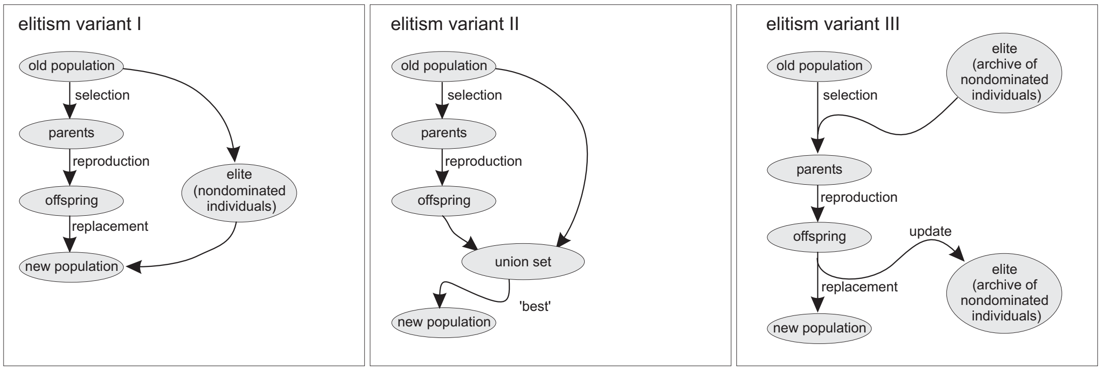


Figure 3.3: Alternatives of implementing elitism in MOEAs

set of the old population and the offspring, another problem can arise. If the union set contains more nondominated individuals than the population size, the question arises which one to take. A random choice is an option. But recalling the goal of achieving a widely spread Pareto-front, this option has again disadvantages, because it makes no difference if solutions are almost equal or very different. Thus a method should be applied that takes into account diversity information. This applies also for the last alternative of an external archive, which has become the most popular realization of elitism in the context of multiobjective evolutionary optimization. Generally this external archive could store all nondominated solutions found during the evolutionary run. But a finite size for the external archive opens the possibility of combining elitism with the support of diversity on the Pareto-front. At some point when the number of nondominated solutions in the archive exceeds the given limit, the question arises which solutions to keep and which to delete. The goal of achieving a good distribution of solutions across the Pareto-frontier forms a basis to draw these decisions. In his PhD-thesis Zitzler ([Zit99]) controlled the size of the external set by a clustering method. This method deletes the most similar members of the set until the desired size is reached. By this way the external archive not only serves as a elitism operator, but also as a method for promoting population diversity. The elitism operator in GAME is based on this concept and uses a similar approach.

3.3 Multiobjective Evolutionary Algorithms - Current State of the Art

In order to determine the current state-of-the-art concerning multiobjective optimization with evolutionary algorithms a vast literature research has been performed ([PR01]). The goals of this literature research have been to:

- get a profound overview on work related to MOEA (past and presence)
- determine the current state-of-the-art of MOEA

As a result of the literature research in general it can be stated that applying evolutionary algorithm to multiobjective optimization problems is still a young and active field of research as can be seen in the growing number of publication per year in the recent years (see figure 3.4).

3.3.1 Overview on the Development of Multiobjective Evolutionary Algorithms

Several papers have been found, in which researchers tried to give a an overview on the scope of MOEA research at different times. An early overview can be found

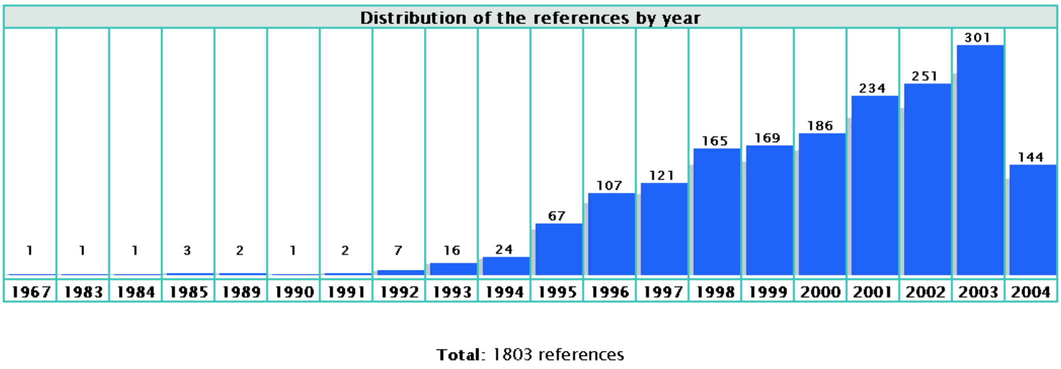


Figure 3.4: Number of MOEA related publication per year (source: EMOO Repository by C.A. Coello Coello ([Coe]), date: August 2004)

in ([FF95]) by Fonseca and Fleming, in which they discuss different approaches for fitness assignment like aggregating approaches and Pareto-ranking based approaches. Furthermore they show directions for future research. A more recent survey can be found in ([Coe99a]) by Coello Coello, in which he also discusses strengths and weaknesses of the most popular MOEAs and describes their application. Furthermore he gives a detailed view on future research perspectives. Also a profound source for getting an overview on MOEA techniques is ([Deb02]), in which the complete field of multiobjective optimization with evolutionary algorithms is described ranging from basic MOP theory to applications. The most up to date resource for MOEA is undoubtable the excellent web repository of evolutionary multiobjective optimization (EMOO, [Coe]). In this continuously and reliably updated online resource MOEA related publications are gathered worldwide and listed in several categories.

Based on these sources a brief review on the development of MOEA is given. The first actual implementation of a real multiobjective EA was Schaffer's Vector Evaluated Genetic Algorithm (VEGA), which he published in the mid 1980s ([Sch85a]). Before mainly standard EA were applied with preference based fitness assignment methods like aggregating functions (weighted-sum-approach), lexicographic ordering, or target-vector approaches.

A major step forward in the development of MOEAs has been initiated by a proposal by Goldberg in his book on genetic algorithms in 1989 ([Gol89], see also chapter (3.2.1)). Goldberg proposed a fitness assignment scheme that is based on the concept of Pareto-dominance. In his review on the current state of MOEAs ([Coe03]), Coello Coello distinguishes two generations in the development of MOEAs. Goldberg's proposal for a Pareto based fitness assignment scheme became basically the standard for the first generation of MOEAs. Many researchers realized different implementations of this concept for developing MOEAs. Other representatives for this first generation of MOEAs are the *Non-dominated Sorting Genetic Algorithm* (NSGA) by Srinivas and Deb ([SD94]) and the *Niched-Pareto Genetic Algorithm* (NPGA) by Horn et al.

([HNG94]). Whereas Srivinas more or less directly implemented Goldberg's proposal, Horn used a tournament based selection scheme, where two randomly chosen individuals compete against an also randomly chosen subset of the population (typically 10% of the population size). If one is dominated by any individual of the comparison set and the other not, the non-dominated one wins, if both are non-dominated or dominated, i.e. there is a tie, the winner is determined on the basis of fitness sharing. Having a Pareto-dominance based fitness assignment scheme deals only with the first goal of a multiobjective optimization. Even in his first sketch Goldberg already noted, that an algorithm based on a pure implementation of this concept will face problems if not special measures will be included to maintain a satisfactory diversity. First generation MOEA typically used niching or fitness sharing for that purpose.

The second generation of MOEAs are characterized by the introduction of elitism. As stated in chapter (3.2.3) in practise (although by no means necessarily) the concept of elitism for MOEAs is typically associated with the existence of an additional population or external set, in which all currently non-dominated individuals are stored. As the goal of a MOEA is to determine the Pareto-frontier, one wants to keep all nondominated solutions and does not want to loose non-dominated solutions due to stochastic operators or even more simple due to the limits of the population size. So as stated before it is a good question why elitism has not been used by the early MOEAs despite the underlying logic is so clear and obvious. Typical representees of these second generation MOEAs are e.g. the Strength Pareto Evolutionary Algorithm (SPEA) and its improved version SPEA2 developed by Zitzler and Thiele ([Zit99]), the Pareto Archived Evolution Strategy (PAES) by Knowles and Corne ([KC00]), the Nondominated Sorting Genetic Algorithm II (NSGA-II) by Deb et al. ([DPAM02]), the Niched Pareto Genetic Algorithm 2 (NPGA2) by Erickson et al. ([EMH01]). These algorithms also represent the current state-of-the-art according to Coello Coello, who published an overview on the current state of MOEA ([Coe03]) in 2003.

3.3.2 Determination of the Current State-Of-The-Art of Multiobjective Evolutionary Algorithms

It is understood that there is no such thing as a 'best algorithm' in general. Comparing the performance of algorithms generally is a difficult problem. One aspect is that the performance usually depends strongly on the problem category. So in general it may only be possible to compare algorithms for a very specific problem class. Another decisive aspect for comparing performance is to find suitable performance measures. Commonly these measures include the quality of the solutions, computational effort, robustness, and reliability aspects. Though the application of these categories to single-objective problems addressed by e.g. a gradient based algorithm is straight forward, it is substantially more complex in case of a MOEA. First of all MOEAs are stochastic search algorithms, so several runs and a statistical evaluation have to be performed in order to get meaningful results. Measuring computational effort is rather straight forward, usually the number of objective function evaluations is taken as a measure. Comparing the quality of the solutions of MOEAs is much more difficult. Because the solution of a MOEA run is a set of points here comparing the results means comparing nondominated sets.

The development of significant and valid measures for comparing nondominated sets is still subject to ongoing research. A quality measure has to answer the questions, if one set is better than another one and how much is it better. The quality measure has to aggregate the information of a solution set in a few characteristic key numbers. Certainly aspects like the distance of solutions from the true Pareto-frontier, the extent of coverage of the true Pareto-front, or the homogeneity of the distribution on the front have to be considered. In literature most commonly unary measures or combinations of such unary measures can be found. Examples are the generational distance measure ([Vel99]), which is the average distance between each nondominated individual and the closest true Pareto-optimal vector, or the hypervolume measure ([ZT99]), which is the volume of the objective space dominated by the respective nondominated set. In order to address the problem of comparing nondominated sets by unary quality measures, Zitzler et al. performed an investigation of all currently known unary quality indicators with respect to their significance ([ZLT$^+$02]). One of their conclusions was that in general the use of unary quality indicators is limited. Binary indicators, which are subject of ongoing research, are regarded as a more promising alternative. In another investigation on this topic ([ZTL$^+$02]) Zitzler et al. came to similar results. In their conclusions it is stated that no unary measure exists that is able to determine that one nondominated set is better than another one. Most quality measures proposed to judge that one nondominated set is better than another one, at best would allow to say, that one set is not worse than another one. Furthermore it has been stated that binary quality measures would overcome these limitations. An example of such a measure, the binary ε-quality measure, was proposed by the authors.

Despite these difficulties to define valid quality measures, several researchers performed comparisons between different MOEAs. In his PhD-thesis ([Vel99]) Veldhuizen defined a test function suite covering a broad and relevant spectrum of aspects with regard to multiobjective problems. Based on this test suite and a set of 6 quality criteria he compared MOGA, MOMGA (proposed by Veldhuizen), NSGA, and NPGA. Although because of the different test problems and criteria it is not easy to come to an clear ranking, it can be stated that NSGA performed worse than the other tested MOEAs.

Zitzler also performed a systematic comparison of several MOEAS for a set of test problems in his PhD-thesis ([Zit99]). These test problems encompassed the multiobjective knapsack problem, a multiobjective travelling salesman problem as well as 6 continuous test problems. The latter ones were based on a proposal by Deb ([Deb98]), who identified several characteristics which may cause problems for a MOEA to have its population successfully converging towards the Pareto-front or to maintain population diversity. Those features comprise multimodality, deception, isolated optima, nonconvexity, discreteness, and non-uniformity. For each of these potential hazards a specific test function has been constructed. Zitzler's comparison encompassed the following algorithms: SPEA (Zitzler an Thiele, 1999), NSGA (Srivinas and Deb, 1994), VEGA (Schaffer, 1985), NPGA (Hajela and Lin, 1992), HLGA (Horn, Nafpliotis, and Goldberg, 1994), FFGA (MOGA, Fonseca and Fleming, 1993), and a pure random search for reference purposes. The comparison resulted in the following ranking:

1. SPEA
2. NSGA
3. VEGA
4. NPGA, HLGA
5. FFGA

Zitzler stated a clear performance gap between SPEA and NSGA as well as the remaining algorithm. The comparison also emphasized the importance of elitism, since the only MOEA directly incorporating elitism, SPEA, clearly outperformed the other ones. Furthermore adding elitism to the other MOEAs substantially improved their performance. Another finding of Zitzler was, that for an example of two objectives SPEA performed better than a single-objective EA employing a weighted-sum-approach although the single-objective EA required more than 100 times the computation effort of SPEA. Zitzler took this as an indication that elitist MOEA can find better solutions in a single run than traditional approaches in several runs. Obviously in a continuative effort, Zitzler published another, more comprehensive version of this comparison in collaboration with Deb ([ZDT99]), which basically confirms these findings. Again the importance of elitism is emphasized. Also in accordance to the previous comparison adding the elitism strategy of SPEA could significantly improve the performance of MOEAs not featuring elitism before. Especially NSGA, augmented with elitism, was found to equal the performance of SPEA. These findings basically led to the improved versions NSGAII and NPGA 2.

Another performance comparison was conducted by Knowles and Corne ([KC00]), in which they compared different versions of their PAES with NPGA and NSGA. Additionally modified versions of NPGA and NSGA were considered, which employed an nondominated archive and elitism. In contrast to the archive in SPEA in PAES the members of the archive do not take part in the further selection process, the archive serves only as a storage. For the comparison the performance assessment method proposed by Fonseca and Fleming ([FF96]) was employed. The comparison was performed for 6 test problems, which were Schaffer's test functions F1, F2, and F3 ([Sch85a]) as well as Fonseca and Fleming's F1 ([FF95]). A fifth test function was self-defined and as the 6th test function served the adaptive distributed database management problem. As a result of this comparison Knowles and Corne found that PAES in its most basic version is a very capable MOEA. Only the improved version of NSGA with a nondominated archive as well as elitism was slightly better with respect to its overall performance.

3.3.3 Discussion

The review on the current state-of-the-art of MOEAs gave valuable hints for the development of GAME. Most of these algorithms employ fitness schemes based on the Pareto-ranking scheme proposed by Goldberg. Also the importance of elitism for the performance of MOEAs can clearly be seen. Zitzler's SPEA showed an elegant method to combine elitism and the promotion of population diversity. Finally it has been decided to employ NSGAII as a reference algorithm for evaluating the performance of GAME. In chapter (5.7) this comparison is performed with several test functions.

4 The Genetic Algorithm for Multicriteria Engineering

4.1 Overview

The *Genetic Algorithm for Multicriteria Engineering* (GAME) has been developed based on the goals stated in chapter (1.2). These requirements ask for an optimization algorithm that is capable of solving a very broad range of tasks, e.g. it should be able to work simultaneously with discrete and continuous design variables, be applicable to multiobjective problems, and last but not least feature a higher efficiency compared to conventional EAs.

In this chapter GAME is introduced and the different implemented methods and operators to achieve the goals are described in detail. The general flowchart of GAME is shown in figure (4.1).

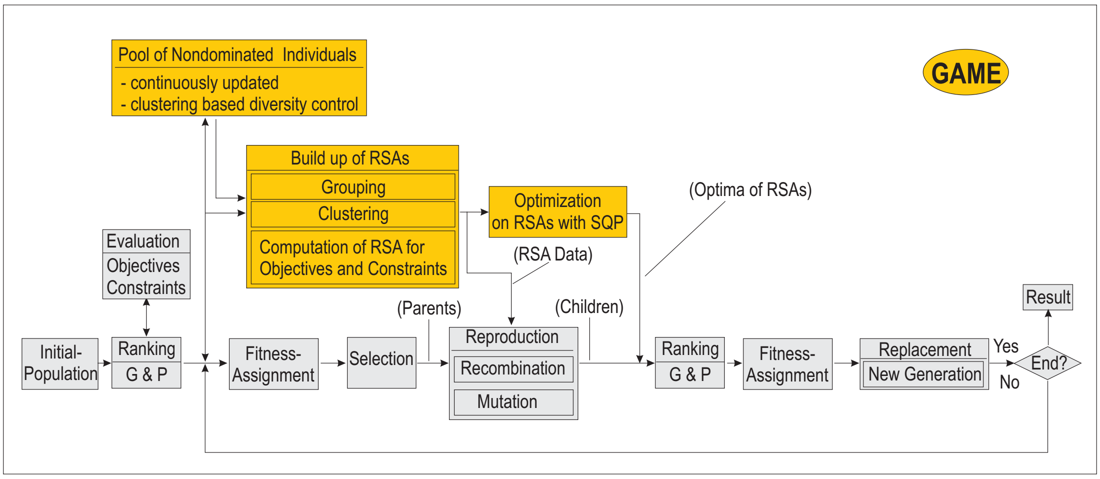


Figure 4.1: Flowchart of GAME

Based on a review of optimization algorithms in chapter (2.2.1) it has been decided to use EAs as the basis for the development of GAME. So on the one hand GAME resembles the flow chart of a conventional EA. On the other hand GAME features additional elements like the external pool of nondominated individuals or a new branch with an RSA based optimization in parallel to the conventional reproduction operators. Both elements will be explained later.

One goal is the applicability of GAME to multiobjective problems. For this purpose GAME features a special fitness assignment scheme which is based on a Pareto-ranking method similar to the ones of NSGA ([SD94]) and MOGA ([FF98a]). This scheme is described in chapter (4.4).

One critical aspect for EAs is the handling of constraints. In GAME the handling of constraints is integrated in a generalized Pareto-ranking scheme, which is based on the goals and priority approach of Fonseca and Fleming ([FF98a]). In chapter (5.4) this constraint handling method is compared to conventional approaches like the lethal factor approach or a penalty function method (see chapter (2.3.3)) and proved to be superior.

When reviewing the state of the art (see chapter (3.3)), elitist EAs proved to be significantly superior to non-elitist EAs. Thus GAME incorporates elitism. In GAME elitism is realized by an external archive storing all nondominated individuals. Besides working as an elitism operator the external archive also supports population diversity and thus pursues also the second goal of a multiobjective EA of achieving a uniform distribution of solutions on the Pareto front. This is achieved by a special update mechanism described in chapter (4.5).

GAME is especially adapted for handling discrete and continuous design variables simultaneously. According to their different nature, discrete and continuous design parameters are treated differently with respect to variable coding as well as the evolutionary operators. Usually when common operators for all variables are used, the presence of discrete variables prohibits the use of continuous operators like the ES mutation operator. A separate treatment solves this problem and allows a broader spectrum of operators. The applied reproduction operators are introduced in chapters (4.7.2) and (4.7.1).

Besides generality the other decisive design goal has been to increase the efficiency and lower the computational costs of conventional EAs. One approach implemented in GAME pursuing this goal is parallelization. Since the evaluation of all individuals is totally independent from each other, EAs are fortunately ideal candidates for grain-parallelization. The respective objective function values for each individual can be computed in parallel on different computers. In GAME this approach is realized by employing a PC-Cluster as described in chapter (4.11). Parallelization does not lower overall computational costs but the total computation time, maybe the even more decisive goal.

Directly addressing the reduction of the computational cost is the augmentation of the conventional EA flowchart with a new branch featuring an RSA based optimization. This integration of RSA methods is one of the essential ideas realized in GAME to increase efficiency and performance.

Optimization is all about exploration and exploitation of information about the design space. During the evolutionary run a high number of individuals is evaluated and so a large amount of information about the design space is gathered. The here presented strategy is based on the idea to further exploit this information already being at hand.

The thought to employ RSA for this purpose origins from a simple analogy: both EAs and RSA methods work with a set of points. While usually design of experiments methods are used to choose design points for the built-up of the RSA, here the individuals of the current and past generations are used. In this way the RSA exploits information that is already present and causes no additional computational costs. Once the RSAs are set up for the objectives and constraints, a SQP based optimization is run on these RSAs. The optimal solutions found are then added to the offspring of the conventional reproduction operators and are also evaluated. So the approximated objective function values are replaced by the true ones. This RSA integration is a robust process because the optima found by the RSA optimization still have to prove their quality during the further evolution process. This compensates for poor optima due to low quality RSA or numerically failed optimization.

While an EA sees only points in the design space, the RSA builds up knowledge over certain areas of the design space. This spacial knowledge can be exploited by a subsequent gradient based optimization as implemented in GAME or also by special directional mutation operators. The reason why this RSA integration increases efficiency is that the children originating from the RSA based optimization can be assumed to be superior to children stemming from random operations and thus increase convergence speed. A basic prerequisite for this approach to actually increase efficiency is that the computation time of the algebraic model functions of the RSA is significantly lower than those for the evaluation of the objective functions. But in the context of structural optimization, in which commonly FEM models or time domain simulations are used, this can be assumed to be the case.

One critical aspect is that GAME works with discrete and continuous variables simultaneously. Because RSA can only be set up for continuous or quasi-continuous design variables, the population has to be split in subsets with consistent discrete variables before RSA methods can be applied. The details of the RSA integration are described in chapter (4.8). In chapter (5.6) it is verified to what extent the assumed advantages of the RSA integration actually materialize.

Before explaining the different elements of GAME first the pseudo code of the algorithms is given in the following. Due to the special implemented fitness assignment scheme GAME does not discriminate between objective- and constraint-functions. In the context of GAME they are subsumed under the more general term *cost-functions*, designated also as $\mathbf{f} = [\mathbf{f}, \mathbf{gf}, \mathbf{hf}]$. So the evaluation of each individual $\mathbf{i}$ can be rewritten as:

$$\mathbf{cost}_i = \mathbf{f}(\mathbf{x}_i)$$

The fitness assignment thus results to:

$$fit_i = fitnessfunction(\mathbf{cost}_i)])$$

With this definition the pseudo code reads as:

```
Input:    $n, \bar{n}$, population size, external archive size
          $n_{gen}$, total number of generations
          $\mathbf{lb}$, $\mathbf{ub}$, $\mathbf{ds}$, lower bound vector, upper bound vector, vector with the discrete
          step sizes for the discrete design variables
          $\sigma$, vector with standard deviations (mutation operator)
          $\mathbf{g}$, $\mathbf{p}$, goal vector, priority vector
Output:   $\mathbf{P}_{sol}$, final nondominated set
Step 1:   Create a random initial population $\mathbf{P}_i$ within the limits $\mathbf{lb}$ and $\mathbf{ub}$. Set
          $t = 1$, $\mathbf{P}_t = \mathbf{P_i}$, and $\bar{\mathbf{P}}_t = \varnothing$
Step 2:   Evaluate all individuals: $\mathbf{Cost}_t = \mathbf{f}(\mathbf{P}_t)$
Step 3:   Perform a goals and priorities based ranking:
          $\mathbf{r}_{\mathbf{P}_t} = ranking(\mathbf{Cost}_t, \mathbf{g}, \mathbf{p})$
Step 4:   Update external population:
          - determine all nondominated individuals $\mathbf{P}_{nd}$ in $\mathbf{P}_t$:
            $\mathbf{P}_{nd} = \{\mathbf{i}\epsilon\mathbf{P}_t | r_{\mathbf{P}_t}(\mathbf{i}) = 0\}$,
          - if $|\bar{\mathbf{P}}_t| + |\mathbf{P}_{nd}| < \bar{n}$: $\bar{\mathbf{P}}_t = \bar{\mathbf{P}}_t \cup \mathbf{P}_{nd}$,
          - else $\bar{\mathbf{P}}_t = clustering(\bar{\mathbf{P}}_t, \mathbf{P}_{nd})$
Step 5:   Form the selection pool: $\mathbf{P}_{sel} = \mathbf{P}_t \cup \bar{\mathbf{P}}_t$
          Compute the ranking: $\mathbf{r}_{\mathbf{P}_{sel}} = ranking(\mathbf{f}(\mathbf{P}_{sel}), \mathbf{g}, \mathbf{p})$
          Compute the fitness: $\mathbf{fit} = fitnessfunction(\mathbf{r}_{\mathbf{P}_{sel}})$
Step 6:   Select parents: $\mathbf{P}_{par} = selectionfunction(\mathbf{fit})$
Step 7:   Generate children by recombination:
          $\mathbf{C} = crossoverfunction(\mathbf{P}_{par})$
Step 8:   Mutate children: $\mathbf{C} = mutationfunction(\mathbf{C})$
Step 9:   Built RSAs: $RSA_{Data} = RSAfunction(\mathbf{P}_t, \bar{\mathbf{P}}_t, \mathbf{Cost}_t, \mathbf{C\bar{o}st}_t)$
Step 10:  Find optima of the RSAs: $\mathbf{P}_{RSA_{opt}}$
Step 11:  Evaluate the offspring:$\mathbf{Cost}_{offspring} = \mathbf{f}([\mathbf{C}, \mathbf{P}_{RSA_{opt}}])$
Step 12:  Built the replacement pool $\mathbf{P}_{rep} = \mathbf{P}_t \cup \mathbf{C} \cup \mathbf{P}_{RSA_{opt}}$,
          $\mathbf{r}_{rep} = ranking([\mathbf{Cost}_t, \mathbf{Cost}_{offspring}], \mathbf{g}, \mathbf{p})$
          $\mathbf{fit}_{rep} = fitnessfunction(\mathbf{r}_{rep})$
          Set t=t+1 and built the next generation:
          $\mathbf{P}_t = replacementfunction(\mathbf{P}_{rep}, \mathbf{fit}_{rep})$
Step 13:  If $t < n_{gen}$ go to step 4
```

4.2 Design Variable Coding

Internally GAME discriminates three categories of design variables: continuous design variables, discrete design variables of quasi-continuous nature, and truly discrete design variables. Discrete design variables of quasi-continuous nature are such discrete design variables which inherit some kind of internal ordering. E.g. some structural members like pipes may only be available in certain discrete diameters, but they can be ordered from small to large. In contrast truly discrete design variables are characterized by the lack of any inherent order, e.g. a design variable representing the choice between different material. The discrimination between these three categories is done with respect to their handling by the reproduction operators. So each category can be treated differently. If common operators for all variables were used, the presence of discrete variables would prohibit the use of the continuous operators like the ES mutation operator. A separate treatment avoids this problem. The use of specialized reproduction operators increases the overall efficiency and convergence speed of the algorithm. The operators are explained in detail in chapter (4.7).
For all continuous design variables a real coding is used whereas an integer representation is used for discrete parameters. For each variable the user has to provide a lower and an upper bound, for the discrete variables additionally the discrete step size has to be given. Discrete variables represent a finite set, which is internally mapped on a integer set starting from 1. The integer coding has been preferred to a binary representation because it takes only a single locus on the chromosome string independent of its resolution as do the continuous variables.
In the following an example of the design variable coding is shown. In a 3-dimensional design vector $\mathbf{x} = [x_1, x_2, x_3]$ may x_1 be a continuous design variable within the limits $\mathbf{D}_{x1} = [-10, 10]$, x_2 a quasi-continuous discrete variable within the limits $\mathbf{D}_{x2} = [3, 5]$ and a discrete step size $ds_2 = 0.1$, and x_3 a truly discrete variable within the limits $\mathbf{D}_{x3} = [1, 10]$ and a discrete step size $ds_3 = 1$. On this level no differences between quasi-continuous and truly discrete variables show up. Mapping both to their respective integer set reads as:

$$D'_{x2} = [1, 2, ..., floor(\tfrac{5-3}{0.1}) + 1] = [1, 21]$$

$$D'_{x3} = [1, 2, ..., floor(\tfrac{10-1}{1}) + 1] = [1, 10]$$

The chromosome string of the individual $\mathbf{x} = [2.934, 4.2, 7]$ would thus result to:

$$\mathbf{x}' = [2.934, 12, 7]$$

4.3 Generation of the Initial Population

Similar to almost all EAs the generation of the initial population in GAME is a completely random process, that generates the n desired individuals within their allowed bounds. This process employs an even probability distribution for the complete range. Because wether objective nor constraint functions are taken into account the resulting initial population can and most likely will contain infeasible solutions.

4.4 Fitness Assignment and Constraint Handling

As GAME is designed for constrained multiobjective problems the design of the fitness assignment operator is an especially challenging task. The fitness assignment has to provide a single scalar value reflecting the overall quality of a design. So the fitness assignment of GAME first has to incorporate constraint information, and secondly it has to handle multiple objectives.

Concluding from the investigations in chapter (3) Pareto-dominance ranking approaches have turned out to be the most promising methods for considering multiple objectives. Nevertheless these approaches do not consider any constraints. For this purpose most commonly penalty function approaches are employed although their application in a ranking scheme is more difficult than in an aggregation approach. A general drawback of penalty function transformations is that their performance is very sensitive with respect to a proper choice of function type and penalty parameters. Thus for a robust algorithm their use is not really desirable, a parameterless approach is preferable.

In GAME therefore a modified version of the Pareto-dominance ranking approach for constrained multiobjective problems is applied, which extends the conventional ranking scheme in a very elegant way to a generalized ranking scheme which considers objectives and constraints. This generalized ranking scheme was developed by Fonseca and Fleming ([FF93]) and is based on a so called *goals and priorities* approach for evaluating individuals. Furthermore it offers a broad flexibility and can address a wide range of optimization tasks. The decision for this approach is also confirmed by the results of the numerical experiments in chapter (5.4), in which the performance of this scheme has been compared to a penalty function scheme and the so called 'lethal factor' scheme. The results of this comparison showed the superiority of the generalized ranking approach.

In the following the chosen generalized ranking approach is introduced. Similar to a conventional ranking scheme the actual fitness of one individual is still determined by its rank. But the procedure how the rank is determined is different. First the rank is not any more equal to the number of individuals which dominate it, but the number of individuals which are *preferable* to it. If one individual is preferable to another one is determined by a comparison based on a so called *goals and priorities* approach. The proposed scheme requires the assignment of a goal and a priority to each objective, which can be aggregated into a goal vector and a preference vector: $\mathbf{g} = [g_1, g_2, ..., g_{n_{ob}}]$, $\mathbf{p} = [p_1, p_2, ...p_{n_{ob}}]$ The goals represent the values which have to be attained for each objective functions, so the algorithm tries find solutions that satisfy ($\mathbf{z}(\mathbf{x}) \leq \mathbf{g}$). The priorities (integers!) represent different preference levels or levels of importance between the respective objectives, a higher value relates to a higher priority.

One major difference to the conventional optimization task formulation is that this approach does not explicitly discriminate between objective and constraint functions. So for simplicity only the term objective function will be applied in the following.

In the goals and priorities approach the difference between objective and constraint functions are reflected by the assignment of different priority levels. Although many possible choices exist, for a typical constrained multiobjective optimization task there will be two priority levels. The constraints will be assigned the highest priority 2 while the objectives will be assigned the next lower priority 1. This reflects the simple reasoning that the satisfaction of the constraints can be considered to be of a higher priority than achieving a good objective function value. A good but infeasible solution is of no use.

The choice of the goal values is also straight forward. For the constraint functions simply the right sides of the respective inequalities can be taken as goals, whereas the true objectives (functions to be minimized) can be assigned $-\infty$ as goal values. The latter choice reflects that for an objective function to be minimized no real goal value can stated.

The goal and priority information is used for the comparison of two solutions, to determine which one is preferable compared to the other one. To illustrate this a comparison of two solution vectors $\mathbf{u} = \mathbf{f}(\mathbf{x}_u)$ and $\mathbf{v} = \mathbf{f}(\mathbf{x}_v)$ is performed in the following. The goals and priorities can be aggregated into a modified goal vector $\mathbf{g}$, which is named *preference vector*:

$$\mathbf{g} = [\mathbf{g_1}, \mathbf{g_2}, ..., \mathbf{g_p}] = [(g_{1,1}, ..., g_{1,n_1}), ..., (g_{p,1}, ..., g_{p,n_p})],$$
$$p \epsilon \mathbb{N}^+, \; n_i \epsilon [0, n_{ob}] \, for \, i = [1, ..., p], \; \sum_{i=1}^{p} n_i = n_{ob}$$

In the preference vector the goal values of the objective functions are ordered in subvectors $\mathbf{g}_i$ according to their priority level, starting with the lowest priority level 1 up to the highest priority p. Without loss of generality the different objective functions in $\mathbf{f}$ can be arranged in the same order:

$$\mathbf{f} = [\mathbf{f_1}, \mathbf{f_2}, ..., \mathbf{f_p}] = [(f_{1,1}, ..., f_{1,n_1}), ..., (f_{p,1}, ..., f_{p,n_p})]$$

The same can be done for the solution vectors $\mathbf{u}$ and $\mathbf{v}$. The subvectors $\mathbf{g}_i$ of the preference vector $\mathbf{g}$ with $i = 1, ...p$ associate priorities i and the goals g_{i,j_i} with $j_i = 1, ..., n_i$ to the corresponding objective function f_{i,j_i}. Higher values of i indicate higher priorities.

Generally each subvector $\mathbf{u}_i$ will be such that a number $k_i \epsilon \{0, ..., n_i\}$ of its components meet their goals while the remaining do not. Thus, again without loss of generality, $\mathbf{u}$ can be reordered so that the following applies for $\mathbf{u}$ and $i = 1, ..., p$:

$$\exists k_i \epsilon \{0, ..., n_i\} | \forall l \epsilon \{1, ..., k_i\}, \, \forall m \epsilon \{k_i + 1, ..., n_i\} \, : \, (u_{i,l} \leq g_{i,l}) \wedge (u_{i,m} > g_{i,m})$$

The first $i = 1, ..., k_i$ objectives of $\mathbf{u}$ meet their respective goal values, while the other ones do not. The vectors $\mathbf{v}_i$ and $\mathbf{g}_i$ are then reordered in the same way as $\mathbf{u}$. For simplicity Fonseca and Fleming proposed the following notation: the first k_i components of $\mathbf{u}_i$, $\mathbf{v}_i$, and $\mathbf{g}_i$ are denoted as $u_i^{\overset{\mathbf{u}}{\smile}}$,$v_i^{\overset{\mathbf{u}}{\smile}}$, and $g_i^{\overset{\mathbf{u}}{\smile}}$, i.e. in all subvectors $\mathbf{u}_i$, $\mathbf{v}_i$, and $\mathbf{g}_i$ the components are ordered according to the fact wether the respective components of $\mathbf{u}_i$ meet their goals or not. Correspondingly the last $n_i - k_i$ components of these vectors are denoted $u_i^{\overset{\mathbf{u}}{\frown}}$,$v_i^{\overset{\mathbf{u}}{\frown}}$, and $g_i^{\overset{\mathbf{u}}{\frown}}$. The smile ($\overset{\mathbf{u}}{\smile}$) and the frown ($\overset{\mathbf{u}}{\frown}$) indicate the components in which $\mathbf{u}$ either does meet or does not meet the goals.

If $\mathbf{u}$ is preferable to $\mathbf{v}$ is now defined as follows:

A vector $\mathbf{u} = [\mathbf{u}_1, ..., \mathbf{u}_p]$ is preferable to $\mathbf{v} = [\mathbf{v}_1, ..., \mathbf{v}_p]$

with respect to a given preference vector $\mathbf{g} = [\mathbf{g}_1, ..., \mathbf{g}_p]$

$\mathbf{u} \underset{\mathbf{g}}{\prec} \mathbf{v}$, if and only if :

$$p = 1 \ \Rightarrow \ (\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}}\ {}_p < \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \ \vee \{(\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \wedge [(\mathbf{v}_p^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}}) \vee (\mathbf{u}_p^{\overset{\mathbf{u}}{\smile}}\ {}_p < \mathbf{v}_p^{\overset{\mathbf{u}}{\smile}})]\}$$

$$p > 1 \ \Rightarrow \ (\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}}\ {}_p < \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \ \vee \{(\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}} = \mathbf{v}_p^{\overset{\mathbf{u}}{\frown}}) \wedge [(\mathbf{v}_p^{\overset{\mathbf{u}}{\smile}} \not\leq \mathbf{g}_p^{\overset{\mathbf{u}}{\smile}}) \vee (\mathbf{u}_{1,...,p-1} \underset{\mathbf{g}_{1,...,p-1}}{\prec} \mathbf{v}_{1,...,p-1})]\}$$

So for deciding if $\mathbf{u}$ is preferable to $\mathbf{v}$, both are first compared with respect to those objective components of $\mathbf{u}$ with the highest priority ($i = p$), in which $\mathbf{u}$ does not satisfy the goals, $\mathbf{u}_p^{\overset{\mathbf{u}}{\frown}}$, while disregarding those for which $\mathbf{u}_p$ satisfies the respective goals, $\mathbf{u}_p^{\overset{\mathbf{u}}{\smile}}$. If those components of $\mathbf{u}$ are partially less than the respective ones of $\mathbf{v}$, $\mathbf{u}$ is preferable to $\mathbf{v}$. In case both vectors meet all goals with this priority, or if they violate some or all of them, but in exactly the same way, it is checked if those components of $\mathbf{v}$ for which $\mathbf{u}$ satisfies the goals, $\mathbf{u}_p^{\overset{\mathbf{u}}{\smile}}$, violate the goals. If this is the case, $\mathbf{u}$ is preferable to $\mathbf{v}$. If this is not the case, i.e. $\mathbf{v}$ also satisfies the goals for those objective in which $\mathbf{u}$ does, the next priority level $(p - 1)$ is considered. This process is continued until the lowest priority 1 is reached. On this last priority level 1 the comparison is finally decided by a classic check for Pareto dominance.
Because satisfied high priority components are left out from the comparison, vectors which are equal in all but these components express virtually no trade-off information given the corresponding preferences.
This comparison scheme for determining preference is parameterless and also requires no scaling of the objectives because no aggregation of any objectives is done. The approach is also very flexible and can be adapted to all common optimization tasks. So for an unconstrained multiobjective optimization task the preference vector may be set as follows: $\mathbf{g} = [\mathbf{g}_1] = [(-\infty, -\infty, ..., -\infty)]$. Only one priority level is needed here, since all objectives are equal. The goals are all $-\infty$, because all function are to be minimized. For a constrained multiobjective optimization task typically two priority levels are needed. The n_c constraint functions are assigned the priority 2. Their corresponding goal values are simply the right sides of the respective inequalities. The

true objective functions are handled as above. Thus the preference vector results to: $\mathbf{g} = [\mathbf{g}_1, \mathbf{g}_2] = [(-\infty, ..., -\infty), (g_{2,1}, g_{2,2}, ..., g_{2,n_c}])]$. This set up also encompasses a constrained single objective optimization task. Correspondingly preference vectors for tasks like goal programming or constraint satisfaction can be set up.

The final fitness assignment is more or less unchanged. Based on the above defined comparison scheme for each individual the number of other individuals is determined which are preferable to it. This number is called its rank r. Thus all nondominated individuals of a certain generation are assigned $r = 0$. The ranking scheme is illustrated in figure (4.2) for three small example populations for a problem with two objective functions. On the left both objectives are to be minimized and have the same priority ($\mathbf{g} = [(\mathbf{g}_1)] = [(-\infty, -\infty)]$), in the middle both objectives are again of the same priority but have to satisfy certain goals ($\mathbf{g} = [(\mathbf{g}_1)] = [(g_{1,1}, g_{1,2})]$), and on the right both objectives have to meet certain goals but the second objective has a higher priority than the first one ($\mathbf{g} = [(\mathbf{g}_1), (\mathbf{g}_2)] = [(g_{1,1}), (g_{2,2})]$).

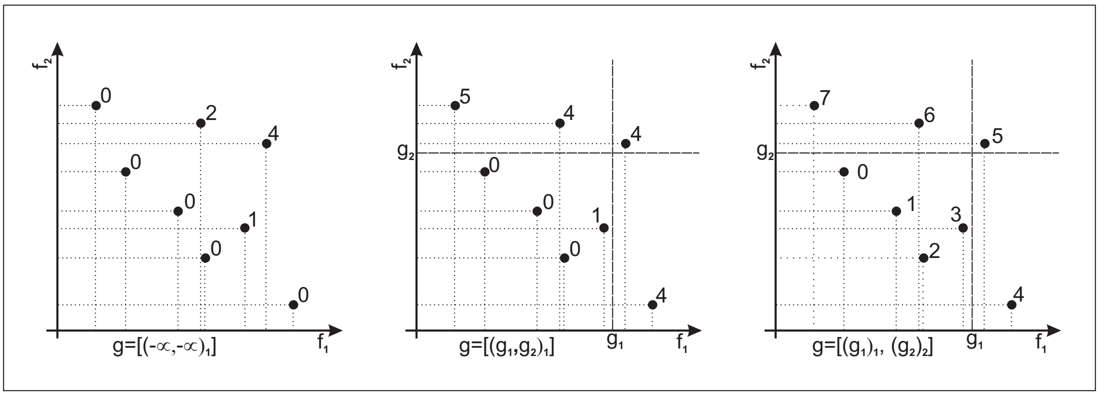


Figure 4.2: Illustration of the ranking based on the goals and priorities scheme

Based on the rank the actual fitness values fit_i for each individual $\mathbf{i}$ is computed. In GAME the fitness values directly reflect the probability for being selected as a parent and is therefore a value between $[0, 1]$. The fitness values are normalized, so that $\sum_{i=1}^{n} fit_i = 1$. The relation between rank and fitness can be chosen as linear or exponential.

In conclusion the generalized ranking scheme by Fonseca and Fleming, which is employed by GAME, provides a very elegant fitness assignment method for constrained multiobjective optimization problems. The concept of goals and priorities is characterized by being a parameterless method and by offering a broad flexibility to be adapted to nearly all commonly occurring optimization tasks. Problems with this approach might occur in the presence of a high number of constraints on the same priority level, when a large percentage is assigned an equal rank and so no sufficient selection pressure is generated.

4.5 Elitism and External Population

In chapter (3) the essential role of elitism for the performance of MOEAs was emphasized by many researchers. In GAME elitism is realized by an finite, external archive, in which only nondominated solutions are stored. For simplicity the term *nondominated* is applied to all individuals, to which no other individual is preferable. This may not be correct in a strict sense with respect to the original definition, but it is the logical equivalent for the constrained task. In GAME this external archive of nondominated solutions has two purposes. On the one hand it serves as an elitism operator by keeping the nondominated solutions, on the other hand it serves a diversity preservation tool. So it is employed in a similar way like in PAES and SPEA. The role as an elitism operator is straight forward. As long as the maximum allowable number $\bar{n}$ of individuals is not exceeded, simply all nondominated individuals of each generation are added to the archive. Subsequently it is checked wether formerly nondominated individuals of the archive now have become dominated and therefore have to be removed. In this way the currently best are preserved for the next generation without the danger of being sorted out by stochastic selection and replacement operators. Actually in GAME the external archive and the population do not really resemble two separate populations, because for the decisive steps fitness assignment and selection both sets are merged. Basically both can be considered as a single population, in which a certain subset, the nondominated individuals, is treated differently in order to promote elitism and population diversity.

The latter aspect is described in the following. Since the external archive is of finite size, at some point adding new nondominated solutions will exceed the given limit. So an procedure is required to reduce the set. This process should not only reduce the number of individuals in the external archive, but simultaneously pursue the second major goal in multiobjective optimization: to achieve a uniformly distributed nondominated set. From this goal a logic for the reduction of the temporary external population is easily derived: In order to promote diversity, delete those solutions which are most similar to each other.

Therefore a so called clustering process is employed similar to the one in SPEA. This clustering technique originally stems from multivariate statistical analysis to identify groups of similar objects.

This is done by building up a binary hierarchical cluster tree. This tree basically reflects the similarity of the individuals in terms of some distance criterion, in most cases the Euclidean distance. First all individuals form a single cluster by themselves, so $\bar{n}$ clusters exist. After computing the distances between all clusters, the two closest cluster, or at this time the two closest individuals, are merged to form a new cluster. Now $\bar{n} - 1$ clusters exist. Subsequently all cluster distances are recomputed. For clusters consisting of several solutions the centroid position is taking as a reference point for computing the distances. Again the two closest clusters are merged. This process continues until there are only two clusters left which will finally build the last cluster. This process can be illustrated in a binary tree. The binary cluster tree poses a powerful tool to reduce the nondominated set to a given number of individuals. One simply has to cut the tree at the desired level, so that the number of clusters equals the number of desired individuals. Afterwards the number of individuals in each cluster has to be reduced to one. For GAME the most central individual is selected in each cluster. In figure (4.3) an example of building such a binary hierarchical cluster tree is shown for a problem with 2 design variables and a population of 10 individuals. In the left figure the spatial distribution of the individuals in the two dimensional objective space is shown. In the middle figure the respective binary hierarchical cluster tree can be seen. On the bottom the first level the clustering process is shown where each individual form a cluster. The iterative clustering process is then marked by u-shaped lines. The height of these u-shaped lines represents the distance of the merged cluster. For this example the external set is to be reduced to 5 individuals. So the binary hierarchical cluster tree is cut at that level where only 5 clusters exist. In the left figure the respective cluster numbers are shown for each individual. For finally building the reduced set in each cluster only the most centroid element is kept and all other are deleted. In the right figure the resulting reduced set is shown. Though the example is extreme with respect to the very low size population and high reduction rate, the effect of achieving a more uniform distribution can clearly be observed.

In the following the pseudo-code of the clustering procedure to update the external archive is given:

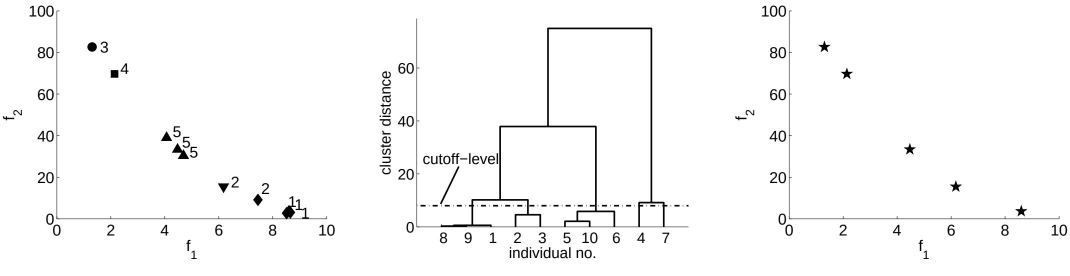


Figure 4.3: Illustration of the clustering procedure for reducing a sample population from 10 to 5 individuals for a problem with 2 objective functions

Input: $\bar{n}$, size of the external population

$\bar{\mathbf{P}}_{\mathbf{t}}$, external archive population

$\mathbf{P}_{nd}$, nondominated individuals of $\mathbf{P}_t$

Output: $\bar{\mathbf{P}}_{t+1}$, updated external population

Step 1: Built a temporary external archive: $\bar{\mathbf{P}}' = \bar{\mathbf{P}}_t \cup \mathbf{P}_{nd}$,

Step 2: Assign each solution $\mathbf{i}\epsilon\bar{\mathbf{P}}'$ to a distinct cluster Cl_k, $\mathbf{Cl} = \{Cl_1, Cl_2, ..., Cl_{\bar{n}'}\}$

Step 3: If $|\mathbf{Cl}| \leq \bar{n}$ go to step 6, otherwise go to step 4

Step 4: Compute the cluster distances d_c for all pairs of cluster. The cluster distance between two cluster Cl_k and Cl_l is defined as:

$$d_c(Cl_k, Cl_l) = \frac{1}{|Cl_k||Cl_l|} \sum_{\mathbf{i}\epsilon Cl_k, \mathbf{j}\epsilon Cl_l} d(\mathbf{i}, \mathbf{j}),$$

where d is the normalized Euclidean distance between the individuals $\mathbf{i}$ of cluster Cl_k and $\mathbf{j}$ of cluster Cl_l wether in the design space or objective space

Step 5: Determine the clusters Cl_k and Cl_l, which have the minimal distance d_c and merge them to single cluster: $C = Cl_k \cup C_l$. This reduces $\mathbf{Cl}$ by one cluster. Go to step 3

Step 6: For each cluster choose the most centroid element to represent the cluster and delete all other. The updated external population is then formed by the individuals of all clusters: $\bar{\mathbf{P}}_{t+1} = \mathbf{Cl}$

In contrast to a conventional fitness sharing approach the approach here affects only the nondominated individuals, while the rest of the population is left as it is. This can be considered as a drawback, but since it addresses directly the distribution of individuals on the nondominated front it can be considered a very efficient tool for achieving a good distribution. Furthermore towards the end of the evolution run, it can be expected that a large percentage of the population is part of the nondominated front.

Though the actual clustering process is parameterless, the size of the external archive has to be given. A too small archive can degrade the elitism effect, while a too large archive may not support the population diversity enough. In general the external archive will be chosen smaller than the population, typical values are $\bar{n} \sim 0.5...0.75n$.

4.6 Selection

The selection operators of GAME are similar to ones employed in conventional EAs. During the selection process $2*n_c$ individuals are selected from the population to form the parent pool or parent couples, $P_{par} = [(\mathbf{p}_{11}, \mathbf{p}_{12}); (\mathbf{p}_{21}, \mathbf{p}_{22}); ..., (\mathbf{p}_{n_c1}, \mathbf{p}_{n_c2})]$. The selection is based on the individual's fitness values. For selection GAME features two possibilities: global stochastic remainder selection and tournament selection. The first one is a strongly modified roulette wheel selection method, that reduces stochastic noise. It copies the integer part of the expected number of copies of each individual directly into the parent pool, $n_{i_{copies}} = floor(fit_i * 2 * n_c)$. Subsequently a conventional roulette wheel selection is applied to fill up the parent pool to the desired size, in which the selection probability is fit_i for each individual. Afterwards the final parent couples are randomly combined. The selection pressure for this method depends strongly on the fitness distribution function wether linear or exponential. For the tournament selection n_t (tournament size) individuals are selected from the selection pool (population + external archive) at random without considering any fitness information. Among those the one with the best fitness is chosen as a parent. This process is repeated until the parent pool is filled up. Again the parent couples are random combination from the selected individuals. Here the selection pressure can be tuned by changing the tournament size. A large tournament size increases the selection pressure.

4.7 Reproduction Operators

Besides the fitness assignment the reproduction operators, recombination and mutation, are the second decisive operator block in an EA. They are responsible for creating new solutions, the so called offspring. In general recombination basically recombines genes from both parents to form a new individuum. The resulting child therefore features characteristics from both parents. Mutation is an arbitrary change in the genes of an individuum, so the resulting child will feature characteristics that cannot be found in either parent. In contrast to crossover mutation introduces truly new genetic information into the genepool, whereas recombination only works with information already present in the genepool. Therefore recombination can be considered mainly an 'exploiting' operator, that tries to get the best out of the available information. Mutation can be considered an 'exploring' operator, which tries to find better solutions by generating completely new genetic information not present in the current genepool.

In which case more importance has to be put on which strategy strongly depends on the problem at hand. Generally for problems, in which the structure or system to be optimized features an internal substructuring, i.e. comprises of different subsystems or components, recombination is an especially effective strategy. This also applies for truly combinatorial problems. Otherwise, if all design variables are more or less sizing variables, pure recombination will result in a high convergence rate, but also a high genetic drift and premature convergence. For such problems mutation is a more effective operator, since it will continuously introduce new individuals into the population. So depending on the problem structure a balance between both strategies has to be found.

Looking at the actual realization of recombination and mutation operators in specific algorithms a broad range of alternatives exists. But to judge their working principle and effectiveness, it is important to judge them in context of other algorithm specific features like e.g. the variable coding.

Traditionally in binary coded GAs the emphasis is put on the recombination operator, while the mutation operator is only applied at a very low rate. But taking a closer look at the working principle of traditional one- or multi-point-crossover in binary coded GAs, one has to state, that the effect may represent a recombination on the genotype level, but on the phenotype level it actually resembles a mutation. Binary strings are cut at arbitrary positions, not regarding the borders between code sections of the different design variables. When decoded the outcome will feature characteristics present in none of the two parents. So the effect is similar to a mutation, an arbitrary mutation. The offspring will most likely not reside anywhere close to its parents, since this mutation does not favor small changes in any way. In conclusion binary coded GA are mainly driven by mutation and not recombination though it might look vice versa at first glance.

Mutation also is the key operator in the real coded ES, originally it has been the only one. Here mutation is defined by adding normally distributed random changes to the respective individual. Small changes are therefore preferred to larger ones. Furthermore typically the number of children is significantly larger than the population size in ES. So each individual will statistically have multiple children. In combination with the ES mutation strategy this has the effect that the resulting children of each individual will resemble several slightly changed, mutated, clones of the respective individual residing in its neighborhood. The best of which will survive and be passed in to the next generation in all likelihood. In this way ES mutation resembles a low efficient hill climber, who does not know which search direction to take. So it has to check the complete neighborhood first before it decides to move in a specific direction. Nevertheless, it is a very robust 'hill climber' since it does not rely on gradient information.

In conclusion mutation can be considered the key operator for generating offspring in both GA and ES. The reason why true recombination does not seem to be existential may be due to the fact that most optimization problems do not feature a sufficiently complex substructure of loosely coupled subsystems. Nevertheless the application scenario of GAME includes the preliminary configuration design potentially including a complex set of subsystems, so recombination will be considered. But based on the considerations above mutation is considered as the main operator in GAME.

4.7.1 Recombination Operator

For the recombination operator it is discriminated between the different design variable types. For all discrete variables a uniform crossover is applied, i.e. the resulting child is a random recombination of its parents, each gene is taken with an equal probability from either one of the two parents. For continuous design variables an arithmetic crossover is applied. The resulting child is generated by a linear combination of its parents:

Input: $P_{par} = [(\mathbf{p}_{11}, \mathbf{p}_{12}); ..., (\mathbf{p}_{n_c 1}, \mathbf{p}_{n_c 2})]$, parents

Output: $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_{nc}]$, children

Step 1: For all n_c couples of parents generate the children $\mathbf{c}_i = [\mathbf{c}_i^{cont}, \mathbf{c}_i^{discr}]$ in the following way:

for the continuous subspace: $\{\mathbf{c}_i^{cont} = \{x_{i,j} | j \epsilon [1, n_{dv}] \wedge x_{i,j} \; is \; continuous\}$:

$\mathbf{c}_i^{cont} = \mathbf{p}_{i1}^{cont} + r(\mathbf{p}_{i2}^{cont} - \mathbf{p}_{i1}^{cont}), \; r = U([0, 1])$

for the discrete subspace: $\{\mathbf{c}_i^{discr} = \{x_{i,j} | j \epsilon [1, n_{dv}] \wedge x_{i,j} \; is \; discrete\}$:

$\mathbf{c}_i^{discr} = [x_{i,j}(\mathbf{p}_{i,r}^{discr})], \; r = round(s) + 1, \; s = U([0, 1])$

4.7.2 Mutation Operator

Experiences with ES show that the combination of real coded design variables and ES-mutation leads to higher convergence rates and also requires significantly lower population sizes than comparable binary GAs. So mutation by adding normally distributed random changes has been the first choice for GAME. As described in chapter (4.2) GAME employs a mixed integer/real coding for representing the design variables, so the ES-mutation can only be applied for the continuous variables. For the discrete design variables it is discriminated between truly discrete and quasi continuous discrete variables. For the latter ones also a normally distributed random change is added, but subsequently rounded to the next possible discrete value. For truly discrete variables mutation is implemented as a pure random jump within the valid set of possible alternatives.

The most important parameter for the ES like mutation operator is the standard deviation σ_i, which has to be defined for each design variable x_i. It basically determines the scatter radius for the mutated offspring around the respective individuum. These standard deviations can be kept constant during the complete evolution run, but it is advantageously to adapt it by specific control laws. In the beginning a higher standard deviation is desirable in order to have a wide spread search in the design space. In a later stage of the evolution run a smaller standard deviation may be advantageously, since it can be assumed that the population has already roughly converged to the optimum. A smaller standard deviation then concentrates the search on the optimal region and hinders moving away again. In literature a variety of potential control laws can be found ranging from simple linear adaptations over success dependent methods to self-adaptive mechanisms. In GAME it has been decided to employ a simple control laws that linearly decreases the standard deviation in dependence of the generation. In chapter (5.5) this method has been compared to the self-adaptive method. The results show that the chosen method is competitive and that the more sophisticated self-adaptive method does not show significant advantages, at least in the application scenarios considered here. The pseudo code for the mutation operator is given in the following (regard the internal integer coding for discrete variables!):

```
 Input:  $\mathbf{C} = [\mathbf{x}_i]$, children
         $\sigma_0$,$\sigma_{n_{gen}}$, vector of the initial and final standard deviations
         $t$, generation
         $\mathbf{lb}, \mathbf{ub}$, lower and upper bounds
         $\mathbf{ds}$, discrete step sizes for discrete variables
Output:  $\mathbf{C}^{mut} = [\mathbf{x}_i^{mut}]$, mutated children
Step 1:  Compute actually applied standard deviations:
         $\sigma_j(t) = \frac{\sigma_{n_{gen},j} - \sigma_{0,j}}{n_{gen}} t + \sigma_{0,j},\ j = 1, 2, ..., n_{dv}$
Step 2:  For all $i = 1..n_c$ children mutate the $j = 1..n_{dv}$ design variables $x_{i,j}$:
         $x_{i,j}^{mut} = x_{i,j} + N(0, \sigma_j(t))$, $x_{i,j}$ continuous:
         $x_{i,j}^{mut} = x_{i,j} + round(N(0, \frac{\sigma_j(t)}{ds_j}))$, $x_{i,j}$ discrete, but quasi continuous:
         $x_{i,j}^{mut} = U([1, floor(\frac{ub_j - lb_j}{ds_j}) + 1])$, $x_{i,j}$ truly discrete
```

4.8 Integration of Response Surface Approximations

The basic idea of integrating RSA methods in GAME is to increase efficiency by further exploiting the information stored in the population by means of response surface approximations. For this purpose a new branch has been added to the conventional EA flow chart. This new branch is basically a bypass that is installed in parallel to the reproduction operators, as can be seen in figure (4.1). So for the next generation on the one hand there are children resulting from the recombination/mutation process like before, but on the other hand additional children resulting from the RSA branch. The way the RSA branch works and how the additional children are generated is explained in the following.

The RSA branch consists of a two step procedure. In the first step RSAs for all objective- and constraint functions are built up. In contrast to traditional RSA methods, that use design of experiments methods (DOE) for choosing points to built the RSA (e.g. [Mye71]), in GAME the individuals of the current and past generations being already at hand are used. In the second step a gradient based optimization is run on the RSAs. The optimal solutions found are then fed back into the population, evaluated using the real objective- and constraint functions, and the yet approximated objective- and constraint functions values are replaced by the true ones.

The main advantage of this parallel implementation is that it is a very robust one. Since the normal evolutionary process remains completely independent of the RSA branch, the success of overall optimization process does not depend on the success of the RSA based solutions. The optima found by the RSA based optimization still have to prove their quality in the further evolution process. This compensates for poor optima due to low quality RSA or numerically failed optimization. Even if the RSA branch fails completely to find reasonable solutions, GAME performs still at least as well as a conventional EA.

Though the basic reasoning for integrating is rather simple, some challenges arise for the actual implementation since GAME works with discrete and continuous variables simultaneously. RSAs do only work with continuous or quasi-continuous design variables. The solution strategy implemented in GAME is to split up the population in subsets with consistent discrete variables before the RSAs can be set up. This process is illustrated in figure (4.4). The RSAs are then built only on these continuous subsets. If reasonable, the quasi-continuous variables can be included in the RSAs, too. In this case the respective variables of the found solutions will be rounded to the next feasible discrete values before adding them to the population. Depending on the actual number of individuals residing in these corresponding continuous subsets linear, quadratic or mixed quadratic polynomials are used as model functions for the RSA. These low order approximation models have been chosen for several reasons. The first one is that they, although being simple, already cover a considerable range of possible response surface shapes. The second, even more important reason is that gradient based optimization algorithm, like the implemented SQP, converge very fast and reliable on these types of functions.

The distribution of the individuals used for the RSA built-up will be random to a certain degree and thus may not be ideal for setting up a highly accurate RSAs. Therefore it is not likely that a RSA spanning over the complete design space will be very accurate. Nevertheless it is first tried to set up the RSAs over the whole design space. If this produces RSAs with insufficient accuracy, an alternative strategy is applied that tries to reach a higher accuracy by subdividing the design space into smaller subsets and span the RSAs only over these smaller subsets. The reasoning behind this is that there is a higher probability that simple model functions are able to provide accurate approximations on smaller areas. Basically this concept of using low order model functions and splitting up the design space resembles to a certain

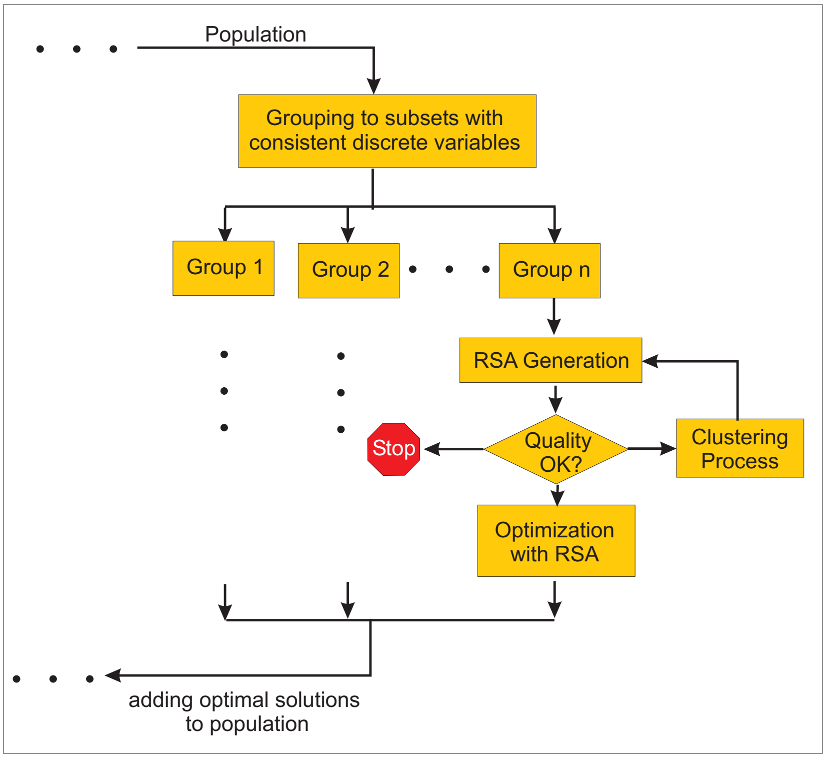


Figure 4.4: Flowchart of the RSA set up

extent the FEM approach. Preliminary work on how increasing the accuracy of RSA by subdivision techniques has been done in ([Tha00]).

The procedure applied in GAME to built these subsets is basically the same clustering process introduced in chapter (4.5) for the external archive. This time it is applied in the design space and used to find groups of individuals with a higher similarity. Similar to the clustering applied for the external archive first a binary hierarchical cluster tree is set up. In order to finally determine the subsets a desired similarity level has to be provided in order cut the cluster tree at his level. A higher desired similarity (i.e. lower cluster distances) leads to more cluster with less individuals, a lower similarity will lead to less subsets containing correspondingly more individuals. A lower bound for the desired similarity is provided by the minimum number of design points a cluster must contain, so that at least a linear approximation model can be set up. In figure (4.5) an example of this clustering process is shown for a sample population for a problem with one objective function and two design variables. In the upper left figure a three-dimensional surface plot illustrates the shape of the objective function. The dot objects represent the population. This example problem features a multimodal objective function, which definitely cannot be approximated by the available model functions over the complete design space. In the upper right figure the resulting binary hierarchical cluster tree is shown. Taking a look on the distribution of individuals, the population shows a natural distribution into three relatively separated regions. This division is clearly reflected by the cluster tree. And expectedly the clustering process successfully manages to split up the population in exactly this three regions. In the

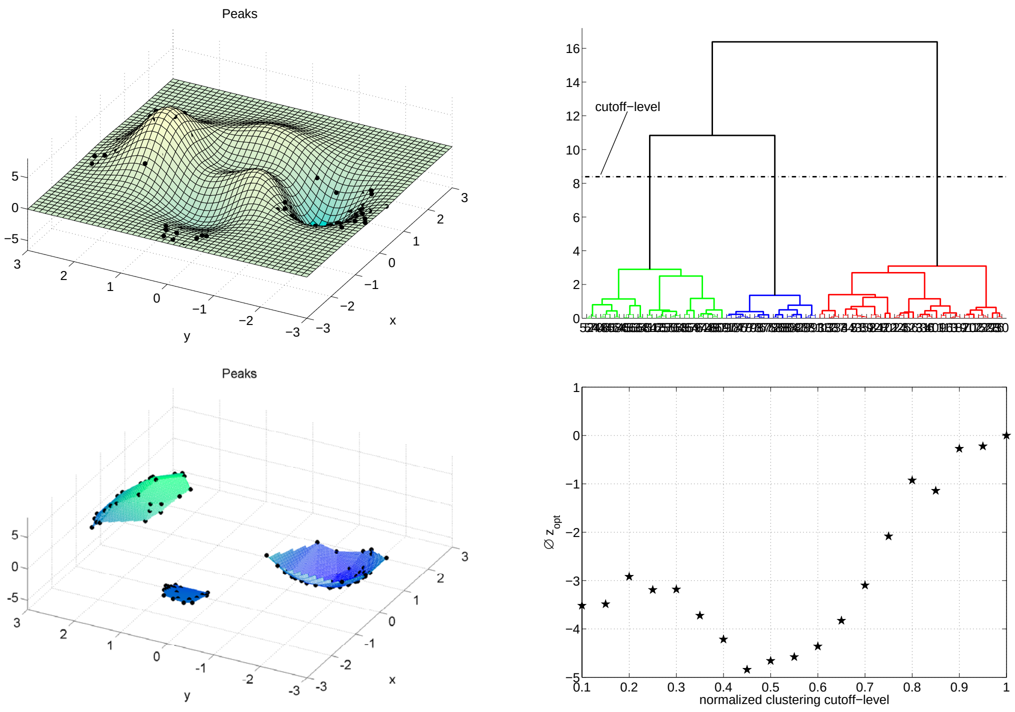


Figure 4.5: Illustration of the clustering process of the population for an example with one objective function of two variables

lower left figure the resulting RSAs are displayed after the clustering process. It can clearly be seen, that the clustering successfully enables the set-up of accurate RSAs even with the chosen simple model functions. For this example a cut-off level of 0.75 of the maximum cluster distance has been applied. A general drawback of the clustering procedure is that for an automatic application this cut-off level, or similarity measure, has to be given a priori without any knowledge about the actual distribution of design points.

In order to investigate how different choices for the cut-off level effect the performance of the RSA based optimization, an experiment has been performed with the above example. For each setting of the cut-off level 100 runs have been performed with randomly generated populations. In the lower right figure the average optimum resulting from the RSA based optimization is displayed in dependence of the relative cut-off level (normalized by the maximum resulting cluster distance). The results verify the underlying assumption that dividing up the search space into smaller subset provides more accurate RSA and thus better optima of the subsequent optimization. A lower cut-off level (i.e. a higher degree of similarity) leads to a better optimum. This trend can be observed down to a certain limit. If the cut-off level is lowered further the resulting optima become worse again. This reveals an inherent drawback of requiring a higher similarity of the individuals. Although the accuracy of the RSA theoretically increases with a higher similarity, this higher similarity leads to a higher number of cluster containing less individuals due to the finite size of the population.

Thus beyond a certain limit the accuracy of the RSA will decrease again due to the low number of points. At the lowest cut-off level settings a convergence of the achieved optima can be observed. This can be addressed to the fact that the implemented clustering procedure will always limit the subdivision to clusters containing at least the minimum required number of points for a mixed quadratic model.

Due to the random distribution of the individuals it is hard to set up an automatic procedure determining an optimal cut-off level. In practice settings in the range of 0.5 to 0.75 have shown the best results.

In case even after this clustering process no sufficient accuracy can be achieved, the RSA process is quit for this subset. After all RSAs have been built, a gradient based optimization is run on these RSAs. All objective and constraint functions of the original problems are replaced by their respective RSA models. The optimization is performed for every cluster of every continuous subset separately. For this purpose the MATLAB internal SQP-algorithm *fmincon* is employed. Since SQP is a applicable to constrained problems, the integration of constraints is no problem. But because SQP is a single-objective method, multiple objectives have to be aggregated using the weighted sum approach. In order not to aim only at a single point of the Pareto-frontier, multiple runs are performed with different, randomly set weights. The resulting optima of all these optimization runs are then fed back into the normal evolutionary cycle by adding them to the offspring created by recombination and mutation.

Splitting up the design space into continuous subspaces and then further splitting up the design space by clustering can result in a considerably high number of sets. In the left plot in figure (4.6) the possible number of subsets is shown as a function of the discrete variables as well as the number of discrete values the respective variable can be assigned. The result shows a highly exponential behavior. For the cantilever beam example there are two truly discrete variables (material, cross section type) with 3 possible realizations each. Thus there exist 9 continuous subspaces. But for an example with 4 discrete variables with 5 possible realizations each there exist already 625 continuous subspaces.

The number of subspaces and clusters has direct consequences on the requirements of the populations size. In order to build up RSAs in all subspaces a sufficient number of designs points must reside in all respective subspaces. The number of necessary design points for building a RSA is again dependent on the model function and the number of design variables. In figure (4.6) on the right side this is shown for linear, quadratic, and quadratic mixed model functions. It can be seen that especially for the mixed quadratic model the number of required design points quickly rises with the number of design variables. For the cantilever beam example with three continuous design variables thus for a linear model 4, for a quadratic model 7, and for a mixed quadratic model 10 design points a required at minimum. In case of a problem with 20 design variables already 21, 41 and 231 design points are required respectively.

Assuming a uniform distribution of the design points across the design space a minimum population size can be estimated by simply multiplying the possible number of

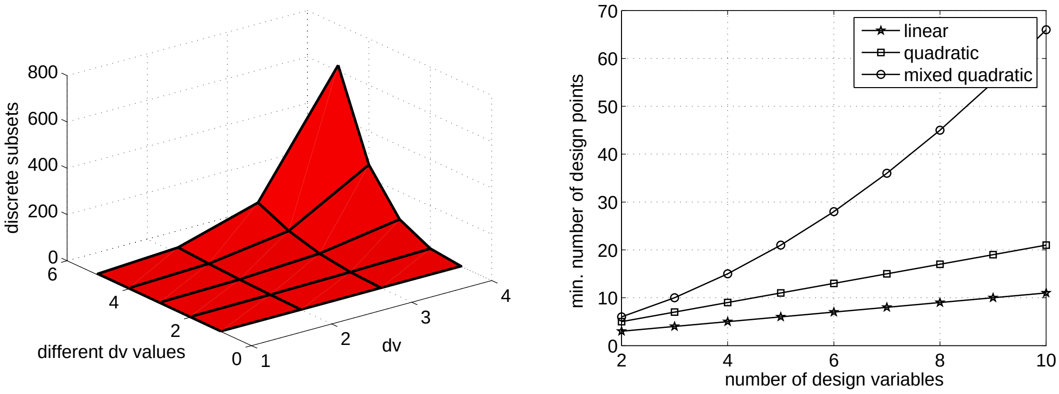


Figure 4.6: Number of continuous subsets in dependence on the number of discrete design variables and their number of possible values (left), required number of design points in dependence on the RSA model function and number of design variables (right),

subspaces and the required number of design points for the respective model function. Thus in the presence of discrete variables quickly a very large required population size can be reached.

So employing the RSA might lead to a required population size that conflicts with the original goal of reducing the computational costs by integrating the RSA. Unfortunately this is a basically unsolvable dilemma of dimensionality.

One possible solution is to employ a larger population only in the initial generations. In the later stages of the optimization it can be assumed that large portions of the search space will have been ruled out as unattractive and so the number of subspaces will decrease also. Thus only a smaller population may be sufficient then. But it may be reasonable not to scale up the population size at all. It might also be acceptable that the RSA branch is not applied during the first generations but only during the later stages when the population is concentrated on a smaller part of the design space. So in the first generations the search is mainly a classic evolutionary search. This approach is pursued in GAME. The underlying reasoning is that it is exactly this initial phase of the search, the quick extracting of promising subsets of the design space, at which evolutionary search is very good. In the later phase of the search process, when the individuals can be assumed to be already in the vicinity of the true optima, the advantages of the RSA based optimization with respect to local search and precise location of the optimum are especially effective.

Nevertheless, the effort for computing a high number of RSA based optimization runs will still be relatively small. Here the advantage of using only low order model functions shows. The SQP algorithm converges in only a few iterations, ideally in only one iteration.

Another possibility of exploiting the information provided by the RSA is to use it by EA operators. Currently under investigation is e.g. an approach approximating the feasible design space and using this information in the mutation operator for increasing the rate of feasible offspring.

4.9 Replacement

For building the next generation the replacement operator has to decide which individuals of the current generation are kept and which are replaced by offspring or the solutions of the RSA branch. In GAME a deterministic replacement operator similar to the one used in ES is applied. All n individuals of the current generation $\mathbf{P}_t$, the n_c children $\mathbf{C}$, and the n_{RSA} RSA based solutions $\mathbf{P}_{RSA_{opt}}$ are inserted in a common replacement pool. The new generation $\mathbf{P}_{t+1}$ is then built by taking the n best individuals from the pool with respect to their fitness values.

4.10 Implementation

GAME has been realized in the multi-purpose mathematics and numerical simulation environment MATLAB. The main reasons for this is because MATLAB provides a highly flexible and easy to use environment for realizing complex algorithms. Furthermore it offers a broad range of mathematical function library and specialized tool boxes. Another advantage are the powerful visualization features in MATLAB.
As GAME is a research tool for evaluating the proposed optimization approach, algorithm speed is not the first priority. A C++ implementation would increase the speed significantly. But in the application environment GAME is designed for, the computation time for the objective functions, e.g. FEM models, can be assumed being at least one order of magnitude higher. So the algorithm's speed contributes only little to the overall computation time.
Implementing GAME in MATLAB also requires the objective functions to be executable in MATLAB. But since all command line executable functions can also be integrated in MATLAB, this poses no real handicap. In the context of adaptive lightweight structures many simulation models are MATLAB models anyway. Furthermore other applications can easily be integrated in MATLAB. The only requirements are that these applications can be executed in batch mode, work with accessible input files, and provide well documented result files. This holds true for most of the currently available FEM tools like e.g. ANSYS.

4.11 Parallelization

Cutting down the total computation time by grain parallelization is essential for making EAs a viable option for many engineering problems. Since the evaluation of the individuals is independent from each other, the most obvious and most effective strategy is to simply evaluate all individuals on different computers in parallel. Exactly this approach is pursued in GAME and realized on a Linux-PC-cluster. In figure (4.7) the realized approach is illustrated. The main algorithm GAME is run on the so called master computer. The parallelization affects only the objective function evaluation module, the rest remains unchanged. For the parallel evaluation of all individuals first for each one a MATLAB batch file and an associated script file are generated. The batch file basically consists of a MATLAB script that loads the design vector of the respective individuum into the MATLAB workspace, evaluates the objective function, and finally saves the results to a specified file. The script file is the file actually executed by the queueing system. Besides calling the MATLAB batch file on the assigned node, it also takes care of the necessary file handling. It copies the MATLAB batch file and other additional files to the proper directories on the specific node. Afterwards it copies the result files back to the master.

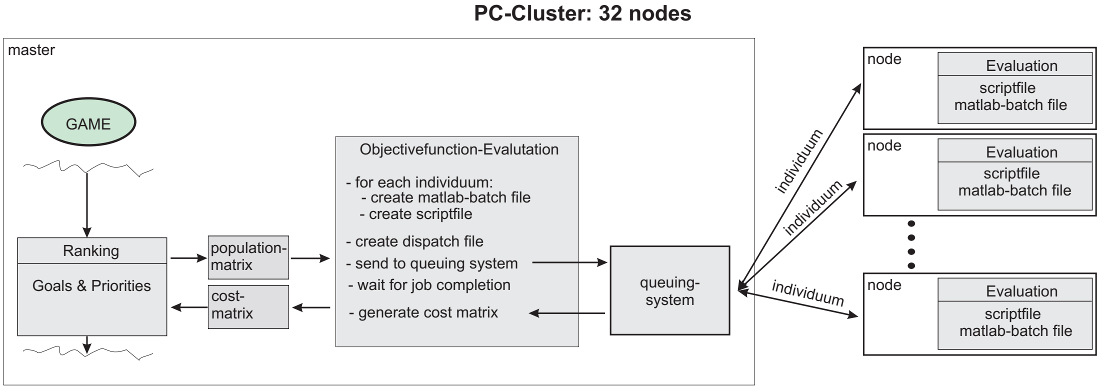


Figure 4.7: Illustration of the realized parallelization approach in GAME for application on PC clusters

After the creation of all script- and batch-files the complete job-queue is set up in a so called dispatch file, which is basically a batch file submitting all script files to the queueing-system. This dispatch file is then executed and the queuing system starts to distribute the jobs on the available nodes. After all jobs are completed the resulting cost matrix containing the objective function values is returned to the calling GAME function.

Currently this parallelization approach is implemented on a Linux-PC-cluster using a XPBS-queueing system (see figure (4.8)). But the chosen approach is flexible to be adapted to other cluster systems easily. The potential performance increase by employing parallelization depends on the number of available nodes, the performance of

Figure 4.8: Linux-cluster at the LLB

the nodes, and also on the communication costs. This communication effort is mainly caused by the copying of the different input- and result-files. This effort increases with the number of nodes, but also the file size has a decisive influence.

In order to evaluate the performance increase a test has been performed with an example problem that included the evaluation of an FEM model. The evaluation of an FEM model required approximately 30 minutes computation time on a single computer. For the test a population of 250 individuals was to be evaluated. The test was run with different numbers of nodes. In figure (4.9) the resulting overall computation time is displayed in dependence of the number of employed nodes. The overall computation time has been normalized with the time required on a single computer. It can be seen that the performance increase is almost proportional to the number of employed nodes. The communication effort is obviously negligible for the number of nodes considered here. In conclusion it can be stated that the chosen parallelization approach is an effective method to significantly reduce overall computation time.

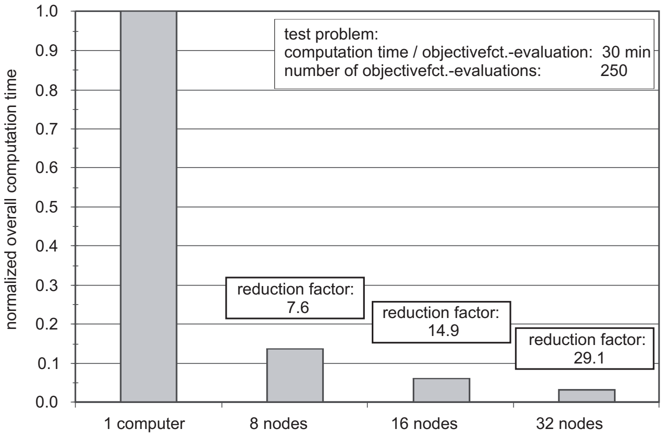


Figure 4.9: Illustration of the reduction in overall computation time depending on the number of nodes for an example problem

5 Numerical Experiments

In this chapter the performance and characteristics of GAME are investigated. The different aspects to be investigated are:

- the influence of the different algorithm parameters like population size, number of children, and number of generations
- the performance of the chosen constraint handling method compared to conventional penalty function methods
- the performance of different methods to adapt the standard deviation of the mutation operator
- the performance of the RSA integration
- the overall performance compared to another prominent representative of a multiobjective evolutionary algorithm, the NSGA II

Since EAs are stochastic search methods, meaningful conclusions, if a certain parameter setting or implemented method is better than another one, cannot be drawn from a single run. So a statistically relevant number of runs has to be performed. Since this will not be possible for real world test problems due to the required time, analytical test problems have been employed for this investigation. The selected test problems have been chosen so that a sufficient range concerning the level of difficulty as well as special characteristics are covered. Furthermore such test problems have been selected that have already been employed in benchmark studies by other researchers so that comparisons to other publications are possible. These problems are introduced in the following section.

5.1 Test Problems

5.1.1 Test Problem ZDT

The test problem ZDT is taken from a series of test problems designed by Zitzler, Deb, and Thiele for benchmark tests of MOEAs ([Deb02]). These are scalable, unconstrained, algebraic, multiobjective problems with 2 objective functions. The complexity of the search can be scaled by changing the number of variables n_{dv} and thereby varying the dimensionality of the design space. The first problem of this problem family is used for the experiments here. It is the most simple one and features a convex Pareto frontier. In equation (5.1) the problem statement is given:

$$\begin{aligned} min \quad & f_1(\mathbf{x}); \\ & f_2(\mathbf{x}) = g(\mathbf{x})h(f_1(\mathbf{x}), g(\mathbf{x})) \end{aligned} \tag{5.1}$$

The subfunctions g and h are:

$$ZDT1 = \begin{cases} f_1(\mathbf{x}) & = & x_1 \\ g(\mathbf{x}) & = & 1 + \frac{9}{n_{dv}-1} \sum_{i=2}^{n_{dv}} x_i \\ h(f_1, g) & = & 1 - \sqrt{f1(\mathbf{x})/g(\mathbf{x})} \end{cases}, \tag{5.2}$$

$$for\ ZDT1: \ n_{dv} = 10, \ x_i \epsilon [0,1]$$

For $ZDT1$ the global Pareto frontier is found for $0 \leq x_1 \leq 1$ and $x_i = 0, i = 2, ..., n_{dv}$, which corresponds to $g(\mathbf{x}) = 1$.

5.1.2 Test Problem OSY

This problem proposed by Osyczka and Kundu ([OK95]) is a constrained test problem. It is a problem with 6 design variables and has two objective and 6 constraint functions. The major challenge of this problem is that it is relatively severely constrained with varying active constraints along the Pareto front.

$$\begin{aligned} min \quad & f_1(\mathbf{x}) = -(25(x_1-2)^2 + (x_2-2)^2 + (x_3-1)^2 + (x_4-4)^2 + (x_5-1)^2); \\ & f_2(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2; \\ subject\,to: \quad & c_1(\mathbf{x}) = x_1 + x_2 - 2 \geq 0; \\ & c_2(\mathbf{x}) = 6 - x_1 - x_2 \geq 0; \\ & c_3(\mathbf{x}) = 2 - x_2 + x_1 \geq 0; \\ & c_4(\mathbf{x}) = 2 - x_1 + 3x_2 \geq 0; \\ & c_5(\mathbf{x}) = 4 - (x_3-3)^2 - x_4 \geq 0; \\ & c_6(\mathbf{x}) = (x_5-3)^2 + x_6 - 4 \geq 0; \\ \\ & 0 \leq x_1, x_2, x_6 \leq 10, 1 \leq x_3, x_5 \leq 5, 0 \leq x_4 \leq 6 \end{aligned} \tag{5.3}$$

This problem has also been applied as a single objective problem employing only objective function f_1 and turning f_2 into an additional constraint ($f_2 \leq 100$). The true optimum for this problem is $x_{opt} = [5\ 1\ 5\ 0\ 5\ 4.899]$ and yields an objective function value of $f(\mathbf{x}) = -274$ with constraints c_1, c_3, and c_5 being active.

5.1.3 Test Problem TNK

The test problem TNK by Tanaka ([Tan95]), also taken from [Deb02], has two continuous design variables ($x_1, x_2 \epsilon [-\pi, \pi]$), two objective functions, and two constraints:

$$\begin{aligned} min \qquad & f_1(\mathbf{x}) = x_1; \\ & f_2(\mathbf{x}) = x_2; \\ subject\,to: \quad & c_1(\mathbf{x}) = x_1^2 + x_2^2 - 1 - 0.1\,cos(16\,arctan(x_1/x_2)) \geq 0; \\ & c_2(\mathbf{x}) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5 \end{aligned} \tag{5.4}$$

This problem features a disconnected Pareto front and the difficulty that all Pareto optimal solutions are placed on a nonlinear constraint. This problem has also been applied as a single objective problem employing only objective function f_1 and turning f_2 into an additional constraint ($f_2 \leq 0.9$). The true optimum for this problem is $x_{opt} = [0.4632\ 0.9]$ and yields an objective function value of $f(\mathbf{x}) = 0.4632$ with constraints c_1 and f_2 being active.

5.1.4 Test Problem CTP1

The scalable test problem CTP1 designed by Deb ([Deb02]) is part of a test problem series for which the constraints were especially designed to interfere with the Pareto front of the unconstrained problem. The different test problems introduce different kinds of difficulties near the Pareto front. For CTP1 each constraint cuts away certain parts of the unconstrained frontier and therefore becomes part of the Pareto front itself. The difficulty for the optimizer here is that for each part of the constrained Pareto front the solutions are required to exactly meet the respective constraint. Since each constraint is a nonlinear function of the design variables it will be difficult for the algorithm to discover and maintain such solutions on a nonlinear boundary. The problem is scalable in terms of the number of design variables and the number of constraints. With an increasing number of constraints the difficulty will also rise.

$$\begin{aligned} min \qquad & f_1(\mathbf{x}_I) = f_1(\mathbf{x}_I); \\ & f_2(\mathbf{x}) = g(\mathbf{x}_{II})\,exp(-f_1(\mathbf{x}_I)/g(\mathbf{x}_{II})); \\ subject\,to: \quad & c_1(\mathbf{x}) = f_2(\mathbf{x}) - a_j\,exp(-b_j f_1(\mathbf{x}_I)) \geq 0;\ j = 1, .., n_{ic} \end{aligned} \tag{5.5}$$

Here $x = [x_I, x_{II}]$ and $f_1(\mathbf{x}_I)$ and $g(\mathbf{x}_{II})$ can be any multivariate function. The number of inequality constraints can be varied. The constants a_j and b_j are determined via a special formula:

- set $j = 0$, $a_j = b_j = 1$, $\Delta = 1/(n_{ic} + 1)$, and $\alpha = 1$;
- calculate $\beta = a_j exp(-b_j \alpha)$;
- calculate $a_{j+1} = (a_j + \beta)/2$ and $b_{j+1} = -1/\alpha ln(\beta/a_{j+1})$;
- set $\alpha = \alpha + \Delta$ and $j = j + 1$.

For the test here the functions f_1 and g have been chosen as:

$$\begin{aligned} f_1(\mathbf{x}) &= x_1 \\ g(\mathbf{x}) &= 1 + \tfrac{9}{n_{dv}-1} \textstyle\sum_{i=2}^{n_{dv}} x_i \end{aligned} \tag{5.6}$$

Again, a single objective version has also been used in which the second objective function has been turned into an additional constraint ($f_2 \leq 1$). The true optimum for this problem is $x_{opt} = [0.001, 0, 0, 0, 0.0016, 0, ..., 0]$ and yields an objective function value of $f(\mathbf{x}) = 0.001$ with constraint c_1 being active.

5.1.5 Test Problem Cantilever Beam

As a more realistic structural optimization test problem the optimization of a cantilever beam has been chosen, which is illustrated in figure (5.1). This $2[m]$ long beam is subject to a tip force $F = 100000[N]$. The optimization task is to minimize

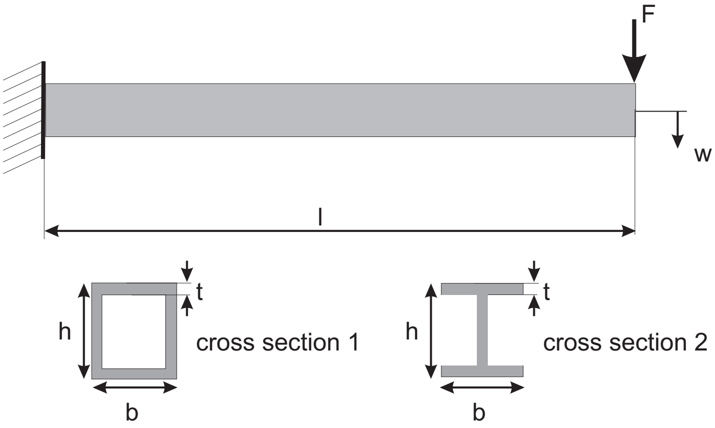


Figure 5.1: Example: cantilever beam

the mass and the end deflection while satisfying constraints with respect to strength, stability constraints (lateral buckling/buckling) and the first eigenfrequency:

$$\begin{aligned}
min: \quad & m(\mathbf{x}), \\
& w_l(\mathbf{x}), \\
& \mathbf{x} = [cs, mat, h, b, t] \\
subject\,to: \quad & \sigma_{max_{Mises}}(\mathbf{x}) \leq \sigma_{allow.}(\mathbf{x}_2), \\
& \frac{M(\mathbf{x})}{M_{l.buckl.,crit}} \leq 1, \\
& \frac{\sigma(\mathbf{x})}{\sigma_{buckl.,crit}} \leq 1 \\
& \omega_1(\mathbf{x}) \geq \omega_{min}
\end{aligned} \tag{5.7}$$

The design variables encompass two truly discrete variables (cross-section type cs (I- or box-profile [1,2]) and material mat (steel, aluminum, titanium [1,2,3])) and three continuous variables (width $b\epsilon[0.020, 0.200][m]$, height $h\epsilon[0.020, 0.200][m]$, and sheet thickness $t\epsilon[0.001, 0.020][mm]$ of the profile). The maximum allowable stress is dependent on the chosen material ($\mathbf{x}(2)$). The eigenfrequency is required to be above $10[Hz]$. The cantilever beam problem has also been used as a single objective problem minimizing only the mass and turning the tip deflection into an additional constraint ($w_l(\mathbf{x}) \leq w_{max} = 0.05m$). The true optimum for this problem is $x_{opt} = [1, 2, 0.121, 0.200, 0.001]$ and yields an objective function value of $f(\mathbf{x}) = 2.335$ with constraint $w_l(\mathbf{x})$ being active.

5.2 Performance Measures

For the performance assessment of algorithms appropriate performance measures have to be defined. In general this can be a quite challenging task. For the study here three main categories have been employed: the quality of the achieved optimum, the computational cost for achieving this optimum, and the probability of achieving this optimum. The last one is especially important for evolutionary algorithms since they are stochastic methods.

5.2.1 Measures for the Quality of the Optimum

For judging the quality of the achieved optimum two cases have to be distinguished: single objective and multiobjective problems. Judging the quality of a single objective problem is straight forward: the feasible solution with the smaller objective function value is the better one. Judging the result of a multiobjective optimization is significantly more difficult and challenging. The result of a multiobjective optimization is the Pareto-optimal set, or at least the final nondominated set. So for evaluating the performance sets have to be compared instead of single solutions. These solution sets consist of a finite number of solutions with a considerably spread in terms of closeness to the true Pareto-frontier, distribution across the front, and also the extent of coverage of the true Pareto-front. General criteria for judging these nondominated sets are easily formulated, e.g. how close the solutions are to the true Pareto frontier and how well they are distributed across the front. But the transformation in numerical measures is quite challenging. It can be quickly seen that it is impossible to develop a single, meaningful, scalar measure reflecting the absolute quality of a solution set. This fails already for the simple case of a set, whose solution are very close to the true Pareto-frontier but cover only a small part of the true front, and another set, whose solutions are a little more away from the true front but cover it almost uniformly. It is impossible to say which one is better. The judging of the solutions sets of a multiobjective problem is a multiobjective problem itself. So at least two aspects, closeness to the front and the distribution, have to be considered. With respect to the first criteria another substantial obstacle is that the true Pareto front is generally not known. Several researchers tried to develop measures. Within the scope of the studies here the following three measures have been applied: hypervolume, spacing, and set coverage metric. These are defined in the following:

- Hypervolume:
  Hypervolume HV is a measure that calculates the volume in the objective space which is covered by the members of the nondominated set $\bar{\mathbf{P}}$. This measure has been introduced first by Zitzler ([ZT98]). For each solution $\mathbf{i}\epsilon\bar{\mathbf{P}}$ a hypercube $\mathbf{v}_i$ is constructed using a reference point W and the solution $\mathbf{i}$ as diagonal corner points. The reference point W can simply be defined by constructing a vector of the worst objective function values. The hypervolume then is the union all hypercubes: $HV = volume(U_{i=1}^{|\bar{\mathbf{P}}|}\mathbf{v}_i)$. In figure (5.2) this measure is illustrated for an example with two objective functions.

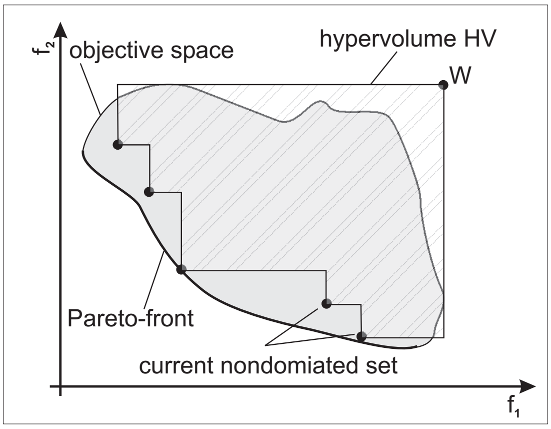


Figure 5.2: Illustration of computing the hypervolume for an example problem with two objectives

As can be seen a nondominated set closer to the true Pareto front corresponds to a higher hypervolume value. Also a more uniform distribution will lead to a higher hypervolume value. In conclusion the hypervolume can be regarded as a measure of how close the nondominated set is to the true Pareto front. Since this metric is sensitive to the scaling of the objectives, commonly the objectives are normalized prior to the computing of the measure.

- Spacing:
  The spacing measure was proposed by Schott ([Sch95]) and is a measure for judging the distribution of the solutions on the Pareto front. The spacing measure S is defined as follows:

$$\begin{aligned} S &= \sqrt{\tfrac{1}{|\bar{\mathbf{P}}|} \textstyle\sum_{i=1}^{|\bar{\mathbf{P}}|} (d_i - \bar{d})}, \\ d_i &= \min_{k\epsilon\bar{\mathbf{P}}\wedge k\neq i} \textstyle\sum_{m=1}^{n_{ob}} |f_m^i - f_m^k|, \\ \bar{d} &= \textstyle\sum_{i=1}^{|\bar{\mathbf{P}}|} d_i/|\bar{\mathbf{P}}| \end{aligned} \tag{5.8}$$

  The measure uses the relative distance between two consecutive solutions on the nondominated front. The distances d_i are the minimum values of the sum of the absolute difference in the objective function values between the ith solution and any other solution in the nondominated set. It has to be noted, that this is not the Euclidean distance between the respective two solutions. The spacing measure S is the standard deviation of those distances d_i. If the solutions are more or less uniformly spaced on the nondominated front the spacing measure S will be very small. A smaller value of S is therefore better. The spacing measure S reflects only the distribution of the solutions on the front, but not the extent of the spread of the solutions. Like the hypervolume this measure is sensitive to the scaling of the objectives and therefore prior normalization is essential.

- Set Coverage Metric:
  This measure was propossed by Zitzler ([Zit99]) and compares two solutions sets $\bar{\mathbf{P}}_1$ and $\bar{\mathbf{P}}_2$ by directly employing the concept of domination. The set coverage metric $SCM(\bar{\mathbf{P}}_1, \bar{\mathbf{P}}_2)$ computes the percentage of solutions in $\bar{\mathbf{P}}_2$ which are weakly dominated by solutions in $\bar{\mathbf{P}}_1$:

$$SCM(\bar{\mathbf{P}}_1, \bar{\mathbf{P}}_2) = \frac{|\{\mathbf{i}\epsilon\bar{\mathbf{P}}_2 | \exists \mathbf{j}\epsilon\bar{\mathbf{P}}_1 : \mathbf{j} \preceq \mathbf{i}|}{|\bar{\mathbf{P}}_2|} \tag{5.9}$$

  If all members of $\bar{\mathbf{P}}_2$ are weakly dominated by $\bar{\mathbf{P}}_1$, $SCM(\bar{\mathbf{P}}_1, \bar{\mathbf{P}}_2) = 1$. Vice versa $SCM(\bar{\mathbf{P}}_1, \bar{\mathbf{P}}_2) = 0$, if no members of $\bar{\mathbf{P}}_2$ are weakly dominated by $\bar{\mathbf{P}}_1$. It has to be noted, that domination is no symmetric operator, so $SCM(\bar{\mathbf{P}}_1, \bar{\mathbf{P}}_2)$ is not generally equal to $1 - SCM(\bar{\mathbf{P}}_1, \bar{\mathbf{P}}_2)$. So it is necessary to compute both. SCM is no absolute measure, it is only useful for directly comparing two solution sets.

For a comprehensive comparison of two set, all three measures have to be considered. Evolutionary algorithms are stochastic algorithms. So the results are not exactly reproducible. Thus a statistically sufficient number of experiments has to be conducted in order to draw meaningful conclusions about the performance of a certain parameter setting or a certain operator. Within this study the resulting mean values of the respective metrics are taken for judging the quality of the achieved optimum or nondominated sets. For judging the probability of reaching these values the standard deviation is chosen. This metric gives valuable information if the results show a large amount of scattering and therefore how likely the results of a sample run will be at the expected performance level.

5.2.2 Measures for the Computational Costs

GAME is developed for the design optimization of structures or structural systems. In most cases the computing of the objectives and constraints therefore will include the evaluation of complex simulation models like FEM models. In comparison with computational costs for these models the costs of the internal computations of GAME will be orders of magnitude lower. The overall computation cost will be dominated by the costs for the objective function evaluations, the algorithm specific computation costs can be neglected. With sufficient accuracy the overall computation cost are therefore directly proportional to the number of objective function evaluations. For the studies here therefore the number of objective function evaluations is taken as a measure for the computational costs.

5.3 Influence of Different GAME Parameters on the Performance

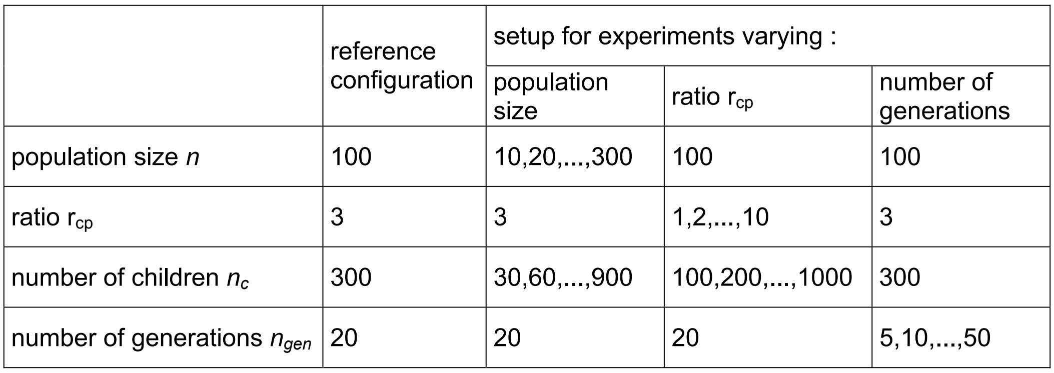
The most important parameters for an evolutionary algorithm like GAME are the population size (n), the number of generations (n_{gen}), and the number of children per generation (n_c) or better the ratio between the number of children and the populations size ($r_{cp} = n_c/n$). This ratio basically determines the selections pressure SP if an ES-replacement operator is used, as is it the case for GAME. All these parameters directly determine the computation costs. So it is an interesting question how certain choices for these parameters relate to the performance of the search, how they relate to certain problem characteristics like the dimensionality of the design space, and if there are optimal settings with respect to efficiency. This is investigated in the following numerical experiments with different test problems.

Since the effects of these three parameters on the performance are wether independent nor linear, a three dimensional grid of test settings would have to be evaluated for determining the complete interdependency of the three parameters. Due to time

constraints this was not possible. Instead it has been decided to vary each parameter around a certain reference configuration. This basically resembles a sensitivity analysis. It is clear that this approach provides only a limited insight in contrast to a full scale study, but, as can be seen later, provides sufficient information for drawing basic conclusion on how to choose these parameters.
The experiments have been conducted with the single objective configurations of the test problems TNK, OSY, and CTP1 as well as with multiobjective test problem ZDT. For CTP1 three configuration have been used with 10, 15, and 20 design variables. This has been done to investigate the dependency between the computational cost for a certain performance level and the dimensionality of the design space. The number of constraints has been set to 20. For ZDT a configuration with 10 design variables has been used.
In the following sections the setup of the experiments is introduced, the results are presented and discussed.

5.3.1 Experiment Configuration

For investigating their influence the three parameters population size (n), the number of generations (n_{gen}), and the ration (n_c/n) have been varied around a reference configuration. The reference configuration as well as the configurations of the respective experiments are shown in table (5.1).



	reference configuration	setup for experiments varying :		
		population size	ratio r_{cp}	number of generations
population size n	100	10,20,...,300	100	100
ratio r_{cp}	3	3	1,2,...,10	3
number of children n_c	300	30,60,...,900	100,200,...,1000	300
number of generations n_{gen}	20	20	20	5,10,...,50

Table 5.1: Setup for the experiments varying the different parameters

In each experiment only the respective parameter has been changed, all other parameters have been kept constant. To single out the effect of increasing the population size it has also been decided to keep the ratio n_c/n constant as an potentially decisive parameter for the convergence behavior. Thus the number of children grows with population size. For the different configurations of the CTP1 test problem the population size has been varied only up to 200 individuals due to computational costs. For the experiments varying the ratio r_{cp} there are two parameters potentially to be varied. For the experiments here the population size has been kept constant ($n = 100$) and the number of children has been adapted according to the chosen ratio: ($n_c = r_{cp} * n$). For each configuration 100 experiments have been performed in order to achieve statistically meaningful results.

5.3.2 Experiment Results

Experiment Results - Influence of the Population Size

First the results for the experiments with single objective test problems are shown in the figures (5.3), (5.4), and (5.5). For all three single objective test problems the results are displayed in the following pattern: on the left the mean value ($\varnothing$) of the achieved optima is shown in dependence of the population size, in the middle the mean value of the generation at which the optimum has been found, and on the right the absolute best optimum, which has been achieved with respective population size. For CTP1 additionally the standard deviations (σ) of the achieved optima and the generations at which they have been found are displayed.

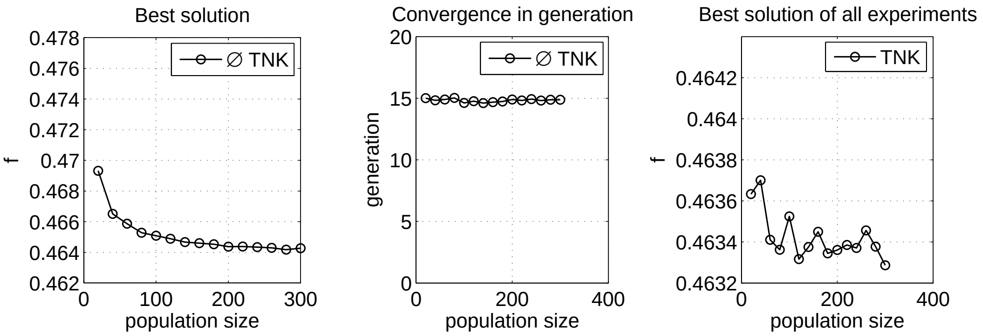


Figure 5.3: Experiment results for varying the population size for TNK

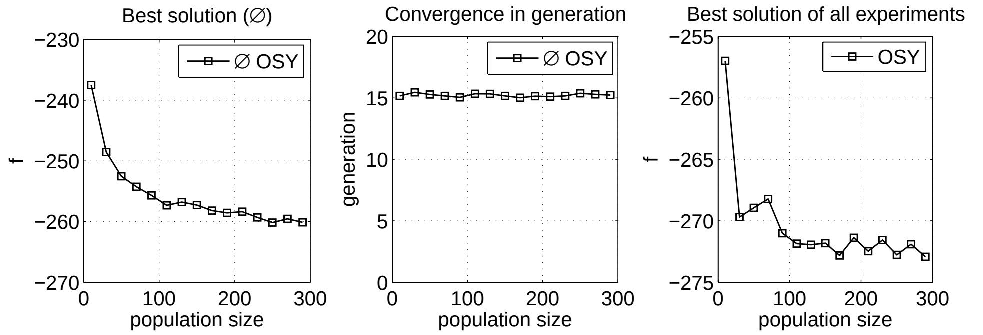


Figure 5.4: Experiment results for varying the population size for OSY

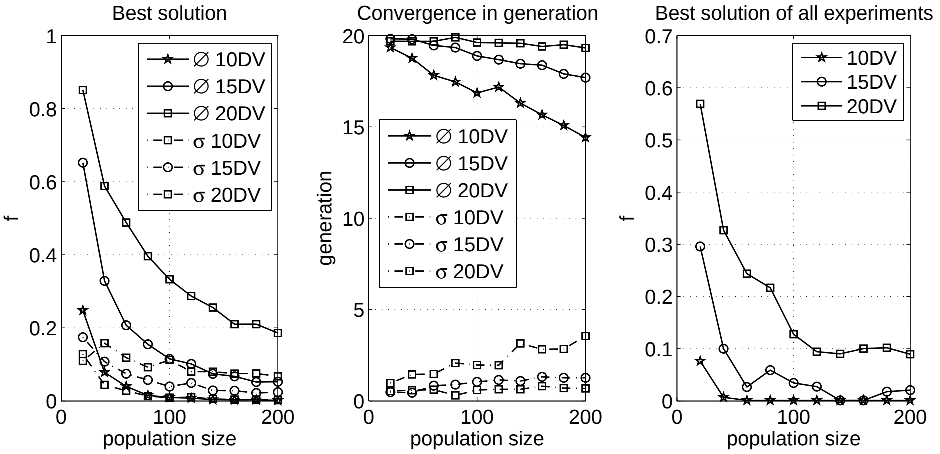


Figure 5.5: Experiment results for varying the population size for CTP1

The results for the multiobjective test problem ZDT are shown in the following figures. In figure (5.6) the mean value (left) as well as the standard deviation (right) of the hypervolume measure is plotted as a function of the population size. Similarly in figure (5.7) the mean value (left) and the standard deviation (right) of the spacing measure is plotted in dependence of the population size. For comparing the results between two different population sizes n_i and n_j with the set coverage metric first the SMC has been computed for all pairs of solutions sets for the two respective population sizes, $SMC(\bar{\mathbf{P}}_{n_i,k}, \bar{\mathbf{P}}_{n_j,l}), SMC(\bar{\mathbf{P}}_{n_j,l}, \bar{\mathbf{P}}_{n_i,k}),\ \ k, l\epsilon[1, 100]$. Then for all those SMC values the respective mean values are computed: $S\bar{M}C(\bar{\mathbf{P}}_{n_i}, \bar{\mathbf{P}}_{n_j})$ and $S\bar{M}C(\bar{\mathbf{P}}_{n_j}, \bar{\mathbf{P}}_{n_i})$. This computation is performed for all possible combinations of population sizes, for which experiments have been performed. In figure (5.8) the resulting mean values $S\bar{M}C$ are displayed in a matrix pattern. On the left side the values of $S\bar{M}C(\bar{\mathbf{P}}_{n_i}, \bar{\mathbf{P}}_{n_j})$ are displayed, on the right side the values of the respective permuted sets $S\bar{M}C(\bar{\mathbf{P}}_{n_j}, \bar{\mathbf{P}}_{n_i})$. The values are color coded employing the color bar at the right side of each figure. In both figures the y-axis corresponds to the population size n_i and the x-axis to the population size n_j.

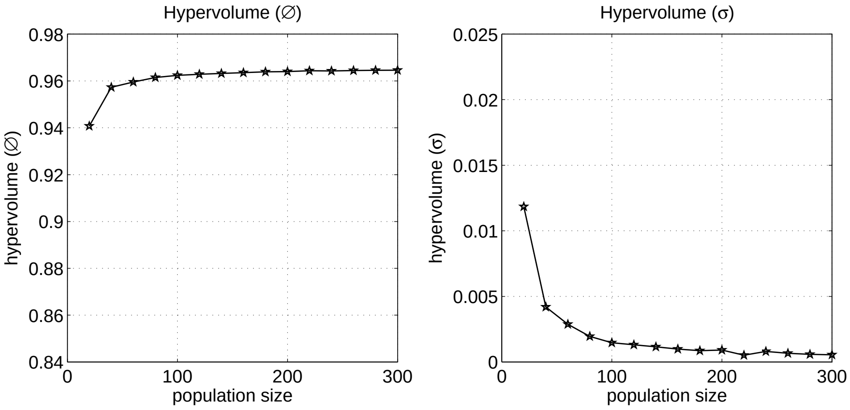


Figure 5.6: Hypervolume measure results for varying the population size (mean value ($\oslash$) and standard deviation (σ)) for ZDT

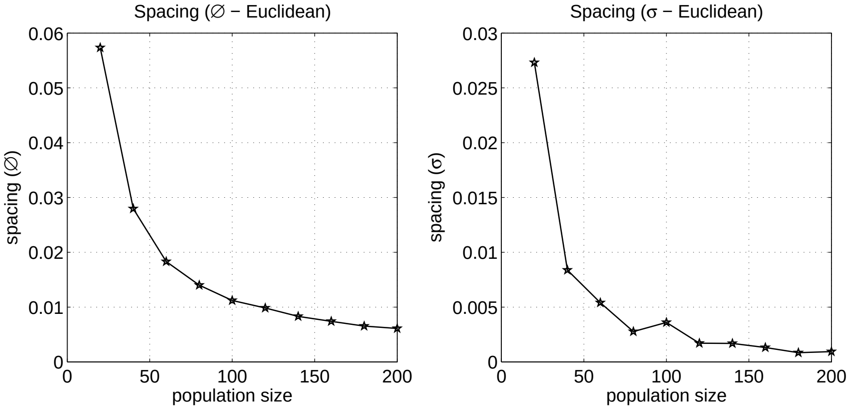


Figure 5.7: Spacing measure results for varying the population size (mean value ($\oslash$) and standard deviation (σ)) for ZDT

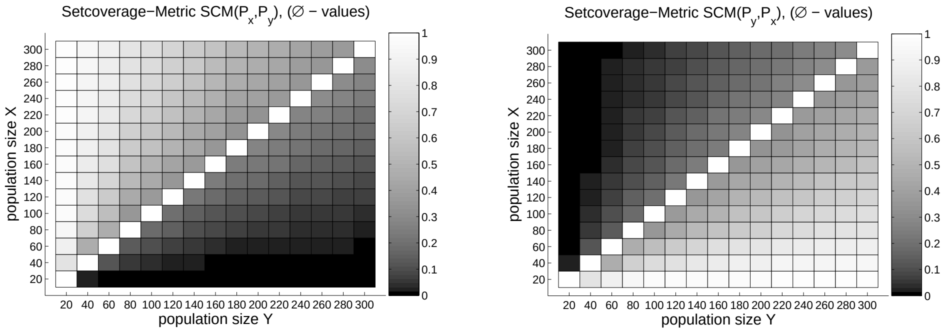


Figure 5.8: Set coverage measure results for varying the population size $S\bar{M}C(\bar{\mathbf{P}}_{population\ size\ i}, \bar{\mathbf{P}}_{population\ size\ j})$ and $S\bar{M}C(\bar{\mathbf{P}}_{population\ size\ j}, \bar{\mathbf{P}}_{population\ size\ i})$ for ZDT

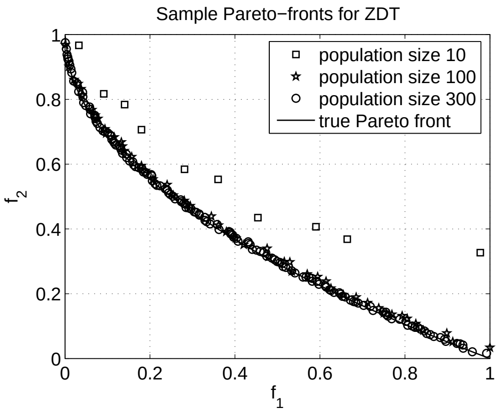


Figure 5.9: Sample nondominated fronts for populations with 10, 100, and 300 individuals for ZDT

To get an impression how the results for different population sizes relate to actual differences in nondominated frontiers three typical fronts are displayed for the populations with 10, 100, and 300 individuals in figure (5.9). The sample fronts reflect clearly the results of the hypervolume metric. The front for a population size of 100 clearly dominates the one for a population size of 10 as could be expected from the results of the hypervolume metric. And also expectedly the front for 300 individuals is not significantly better than the one for 100. Both are already very close to the true Pareto frontier.

The results of the experiments varying selection pressure and the number of generations are displayed in exactly the same way, so no further comments are given.

Experiment Results - Influence of the Ratio r_{cp}

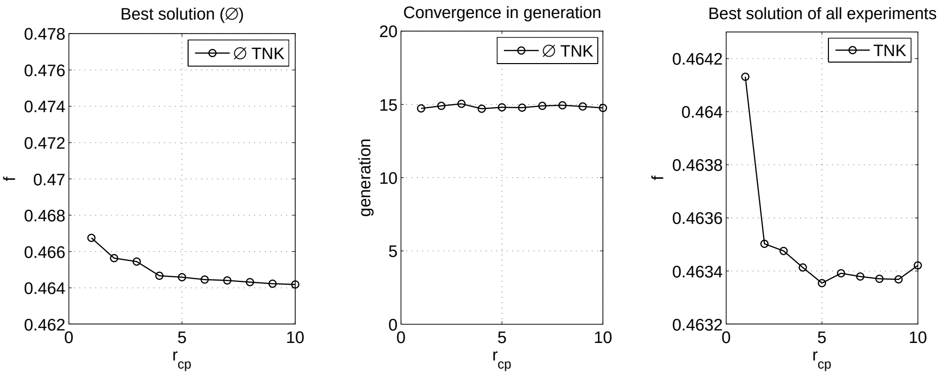


Figure 5.10: Experiment results for varying the ratio r_{cp} for TNK

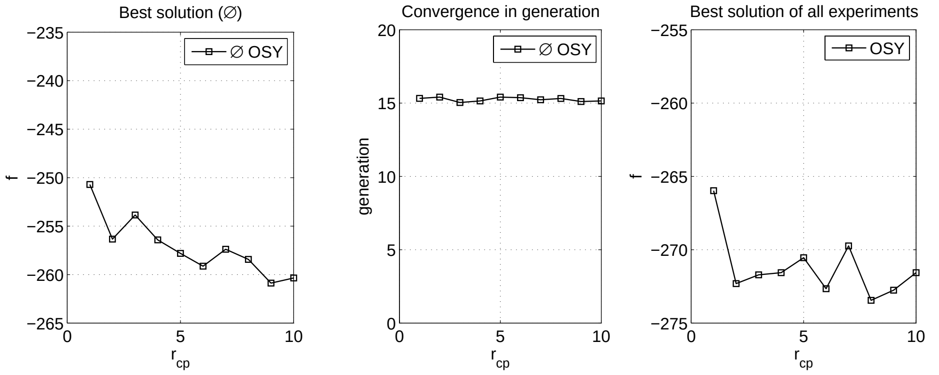


Figure 5.11: Experiment results for varying the ratio r_{cp} for OSY

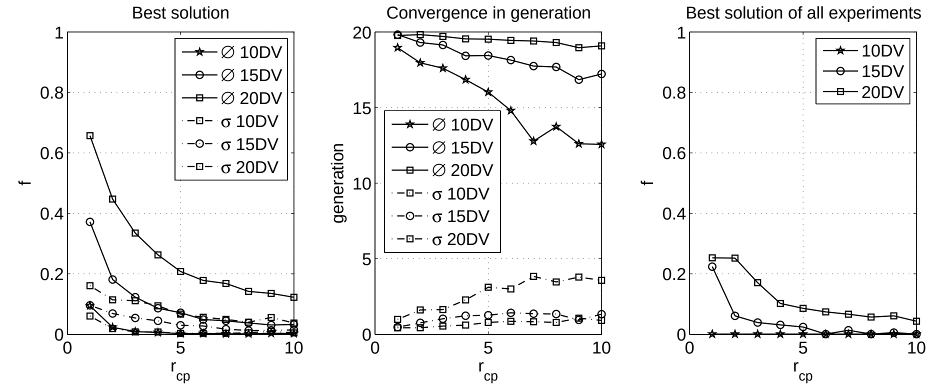


Figure 5.12: Experiment results for varying the ratio r_{cp} for CTP1

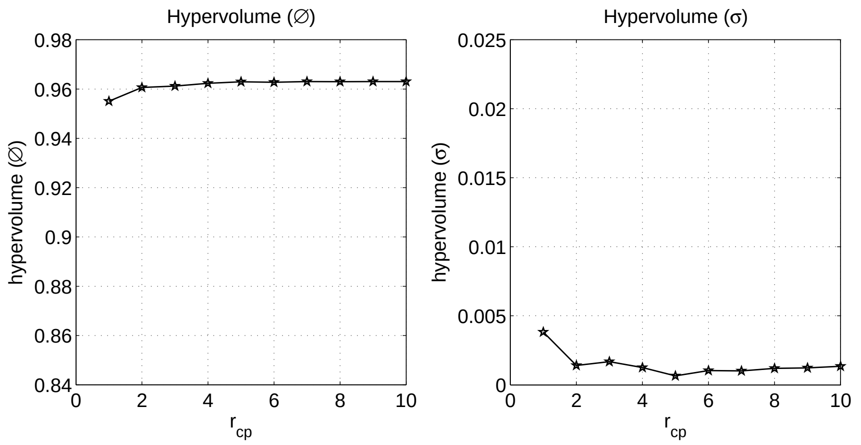


Figure 5.13: Hypervolume measure results for varying the ratio r_{cp} (mean value ($\oslash$) and standard deviation (σ)) for ZDT

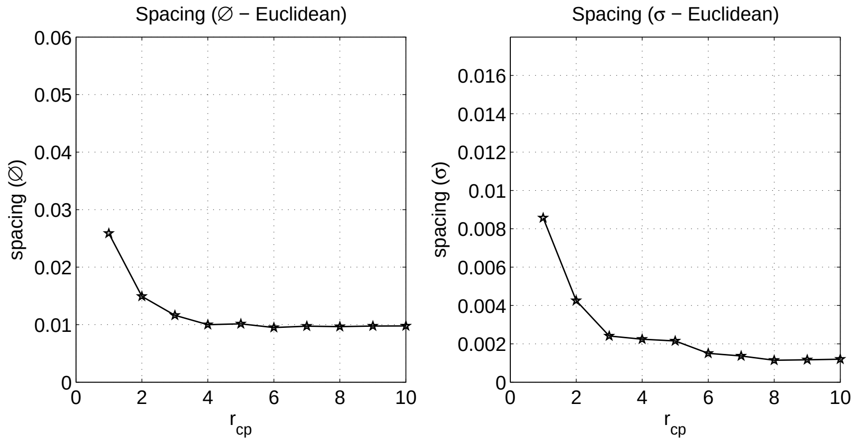


Figure 5.14: Spacing measure results for varying the ratio r_{cp} (mean value ($\oslash$) and standard deviation (σ)) for ZDT

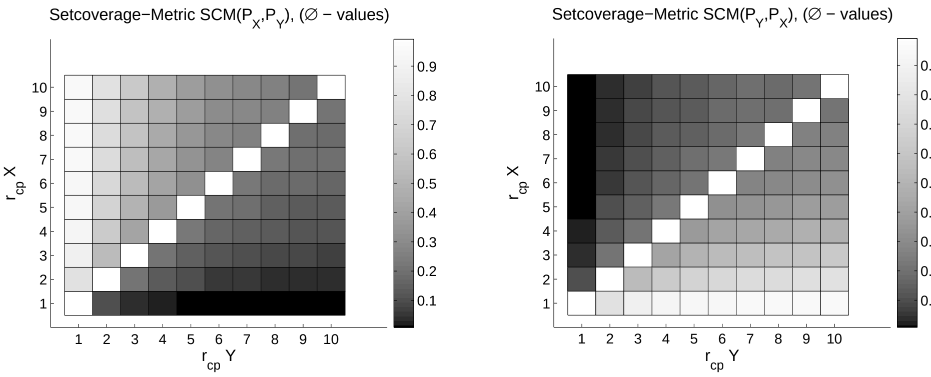


Figure 5.15: Set coverage measure results for varying the ratio r_{cp} $S\bar{M}C(\bar{\mathbf{P}}_{r_{cp}\,i}, \bar{\mathbf{P}}_{r_{cp}\,j})$ and $S\bar{M}C(\bar{\mathbf{P}}_{r_{cp}\,j}, \bar{\mathbf{P}}_{r_{cp}\,i})$ for ZDT

Experiment Results - Influence of the Number of Generations

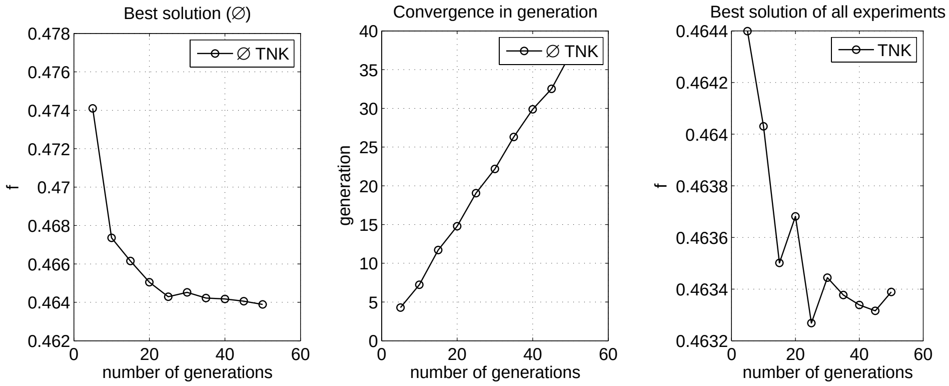


Figure 5.16: Experiment results for varying the number of generations for TNK

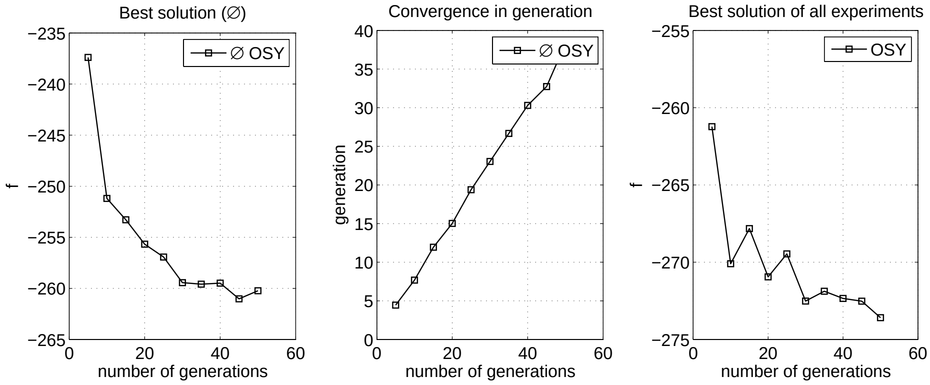


Figure 5.17: Experiment results for varying the number of generations for OSY

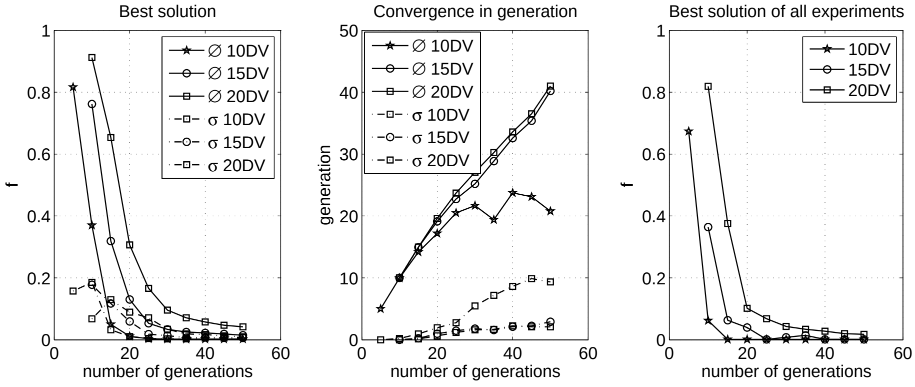


Figure 5.18: Experiment results for varying the number of generations for CTP1

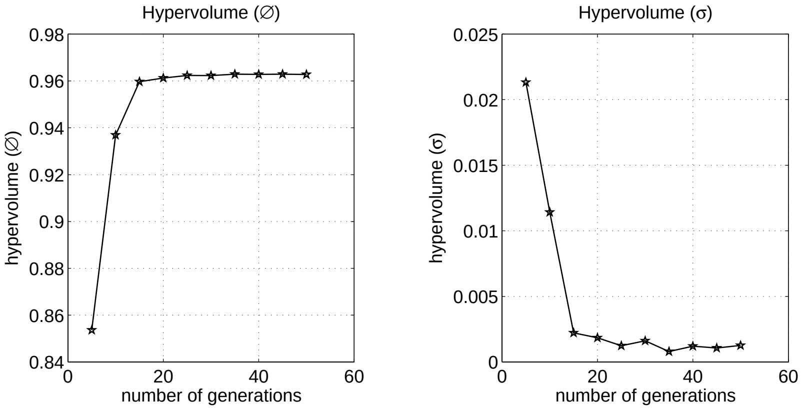


Figure 5.19: Hypervolume measure results for varying the number of generations (mean value ($\oslash$) and standard deviation (σ)) for ZDT

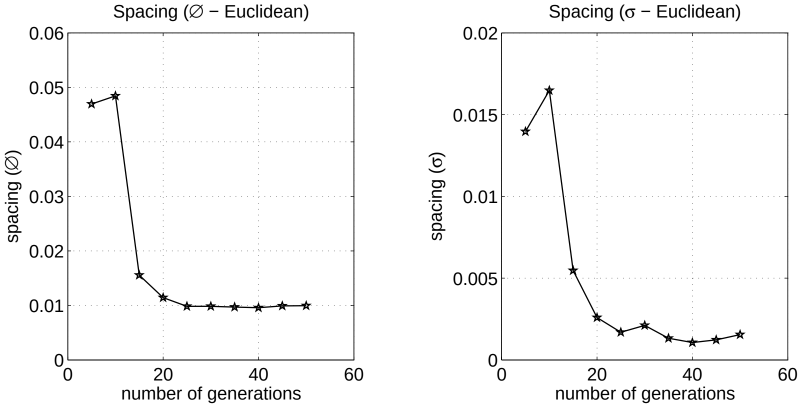


Figure 5.20: Spacing measure results for varying the number of generations (mean value ($\oslash$) and standard deviation (σ)) for ZDT

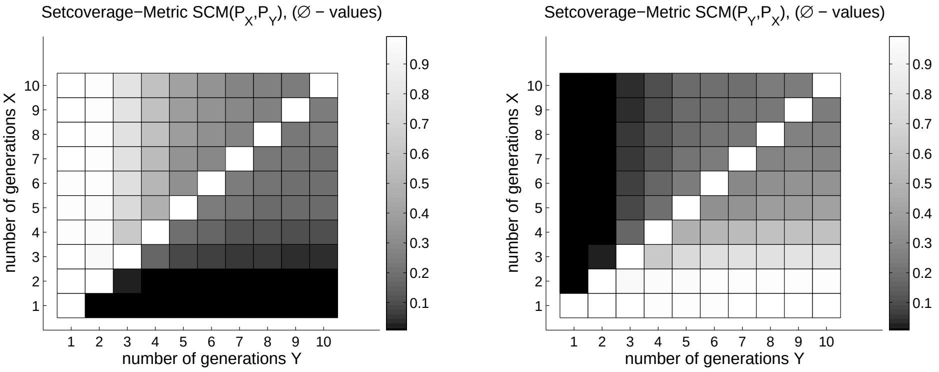


Figure 5.21: Set coverage measure results for varying the number of generations $S\bar{M}C(\bar{\mathbf{P}}_{n_{gen}\,i}, \bar{\mathbf{P}}_{n_{gen}\,j})$ and $S\bar{M}C(\bar{\mathbf{P}}_{n_{gen}\,j}, \bar{\mathbf{P}}_{n_{gen}\,i})$ for ZDT

5.3.3 Discussion

Experiments Varying the Population Size

Increasing the population size leads to better solutions. This is the most obvious finding and can be observed for all test problems. Of course, there is a definite limit for these improvements, the true optimum or true Pareto front. Accordingly for all performance measures a convergence or at least a convergence tendency for a higher population size can be found.

Since increasing the population size means a more massive search process in the design space and a more dense sampling of the design space, this characteristics that a larger population provides better results could be expected. On the other hand a larger population increases the computational costs, so the most interesting question is how large should the population be to provide satisfying results.

In order to find an answer several other important factors have to be considered. One is definitely the dimensionality of the design space. For a given population size a design space with more dimensions is more sparsely populated than one with less dimensions. For the two variable problem TNK the solutions do not improve significantly with population sizes above 200 individuals, for the six variable problem OSY above roundabout 250 individuals. The interaction between the dimensionality of the the design space, the population size, and the performance can directly be seen for the different configurations of CTP1. Convergence for the 10 variable configuration can be observed for a population size of about 100, for the 15 variable version for a population size of about 200, and for the 20 variable version no convergence at all could be observed within the limits of considered population sizes. For a population size of 200 the average best solution for the 10 variable version is 15 times better than the one for the 15 variable version and 114 times better than the one for the 20 variable version.

Being aware of the fact that only a few examples have been investigated, nevertheless this is a strong hind that a very strong interdependency between the dimensionality of problem, the population size, and the quality of the achieved solution exists. If the same performance as for the 10 variable CTP1 problem and a population size of 20 is desired for the 15 or 20 variable CTP1 problem, the population has to be increased to 60 or even 140 individuals. Thus an increase in problem dimensionality of 1.5 respectively 2 requires an increase in population size by a factor of 3 or 7 for equivalent performance. This can be considered as a surprisingly high value. Although this may be an extreme example the results show that the computational effort for an EA based optimization is dependent on the design space dimensionality. This is an unappreciated characteristic which may become a disqualifying factor for the application of EAs for very high dimensional problems.

In addition to achieving better solutions also the standard deviation of the results is significantly reduced by increasing the population size. Again this is confirmed by the

results for all test problems. So the search process becomes less stochastic and more reliable.
When looking at the influence of the population size on the generation in which the best solution is found the conclusions are not that clear. For TNK and OSY this number seems to be nearly independent of the population size and stays almost constant. For the different CTP1 configurations increasing the population size clearly leads to an earlier finding of the optimum. This tendency is the stronger the more the respective configuration has already reached a convergence with respect to quality of the achieved solution. So for the 10 variable configuration the strongest decrease can be observed. From this it may can be concluded that increasing the population will first lead to improved solutions. After the quality of the achieved solutions has converged sufficiently against the true optima further increasing the population size will then lead to an earlier finding of the optima.
The above findings are also confirmed by the results of the experiments for the multiobjective ZDT problem. The desired solution of a multiobjective problem is the true Pareto front, i.e. generally an infinite number of points. If a multiobjective problem is solved with an evolutionary algorithm this front can only be approximated by a finite set of points, the population. It can therefore be expected that a larger population is likely to perform better. And this is clearly reflected by the results of the experiments. All performance measures show better results with an increasing population. Again the limit for this improvement is the true Pareto front, so a convergence can be observed. This applies for the hypervolume metric as well as for the spacing measure. The reason why the spacing measure improves with a growing population is not only due to a more massive search but mainly due to the fact that the applied diversity preservation method (clustering, see chapter 4.5) works significantly better with a higher number of individuals.

Experiments Varying the Ratio r_{cp}

Within these experiments increasing the ratio r_{cp} means increasing the number of children. Therefore the number of individuals taking part in the search process is increased and consequently the computational costs. This might seem to be similar to increasing the population size. But one major difference is that increasing the number of children means that the search is still performed on the same basis (population size). Although more individuals are present, these individuals are actually children of the same parents. That means that the search does not become broader in the design space but more intense. Nevertheless this more intense search can be expected to better exploit the search space and thus find better optima.
The results confirm this expectation for all test problems. The performance improvement is especially strong in the lower r_{cp} range and towards higher values of r_{cp} a convergence can be observed.
Due to the intensified search a quicker finding of the optimum could be expected, but

this can only be observed for the CTP1 test problem with 10 and 15 design variables. For the TNK and OSY test problems no correlation can be observed, the value stays more or less constant. Similar as for increasing the population size increasing r_{cp} will primarily lead to better solution. Only after a convergence is reached there, increasing r_{cp} will lead to a quicker finding of the optimum.

The danger of premature convergence or being trapped in local optima for higher r_{cp} could not be observed. Even when tested with functions offering multiple local optima (not presented here) this tendency could not be observed. This shows that the diversity preservation method implemented in GAME does work well and compensates for the potential tendency of premature convergence.

Increasing r_{cp} means increasing the computational costs. So one has to find a trade off between improving the solutions and the resulting costs for that. Based on all experiments in conclusion choosing a choice of $r_{cp} = 5$ might be a good compromise. The major part of the potential improvement is already achieved up to this value. This can be seen for the results of the single objective problems as well as for the multiobjective problem ZDT. For ZDT the results of the hypervolume, spacing, and set coverage metric support this choice. For the latter metric the percentage of the dominated solutions stays almost constant for r_{cp} values higher than $SP = 5$.

Experiments Varying the Number of Generations

For the last parameter, the number of generations, it can also be stated that higher values leads to the finding of better solutions for all test problems. With more generations the evolutionary search has more time to explore the design space, so this result complies with the expectations. Also expectedly a convergence of this improvements can be observed as the results approach the true optima. The number of generations at which this convergence occurs is problem dependent. One parameter for this is the dimensionality of the search space as can be seen for the different configurations of the CTP1 test problem. The convergence for 10 design variables occurs already at about 20 generations, while for the configurations with 15 and 20 design variables it occurs at about 45 and 50 generations respectively.

Since a similar convergence behavior could be observed in all experiments one might suppose that checking the improvement from generation to generation would be a viable stopping criteria for the evolution. But one has to be aware that the results plotted here are statistical mean values of 100 experiments. For a single instance the best solution can be constant for several generations and then starts improving again. A stopping of the search if no improvement with respect to the previous generation has been achieved inherits a high risk of prematurely stopping the search far away from the true optimum. Observing the improvement process for a certain period of generations is a more reasonable approach, but still this number of generations has to be chosen arbitrarily.

Conclusions of the Parameter Studies

The most prominent conclusion drawn from the experiments reconfirms a universal truth: the more (computational) effort is invested, the better the results will be. Increasing each one of the three parameters increases the computational costs, and in each case the results became better. Since the true optimum or Pareto front poses the limit for this improvement process, a convergence of this improvement process can be observed for all three parameters. So the second conclusion is that increasing the parameters beyond a certain value will lead only to increased cost, but not to substantially better results.

The first conclusion is comprehensible but unsatisfactory, because it does not answer the question how to choose these parameters in practise. The second one also offers no help because the values at which the convergence occurs are generally unknown. In practise computational resources are limited in most cases, so e.g. the time budget only allows for a certain number of function evaluations. So the essential question is how to choose these parameters to achieve the best possible results with respect to given computational resources. Answering this question is very difficult since it is highly dependent on the dimensionality of the search space, the number of constraints, and most important the characteristic of the problem (e.g. convexity, multi-modality, degree of nonlinearity). So even in the above examples it could be observed that for the test problem CTP1 with 10 design variables and 20 constraints a smaller population size was sufficient then for the problem OSY with only 6 design variables and 6 constraints.

Since no direct rules can be deducted for choosing the parameters, it is tried to at least deduce some hinds for choosing them. Because the above experiments are basically a sensitivity analysis like study around one single reference configuration ($n = 100, r_{cp} = 3, n_{gen} = 20$), it is reasonable to evaluate the results with respect to the question to which parameter the performance is most sensitive in relation to the invested additional computational costs, i.e. by increasing which parameter the performance is improved most with respect to the additional cost. For doing so the results of the above experiments are plotted against a common scale: the number of objective function evaluation as a measure for the computational costs. In figure (5.22) this is shown for the test problems TNK and OSY, in figure (5.23) for the different configuration of CTP1 and in figure (5.24) for the multiobjective test problem ZDT. The plots reveal that at least for these test problems the performance is nearly only dependent on the number of function evaluations and almost independent of the parameter by which the additional computational effort is caused. It can be seen that for the test problems OSY, TNK, and ZDT it does not make any significant difference which parameter is increased. No matter by which parameter the computation costs are increased, the achieved optimum improves nearly in the same way. For the test problem CTP1 these findings are basically confirmed, but here the performance seems to be more sensitive to changes in the number of generations than to changes

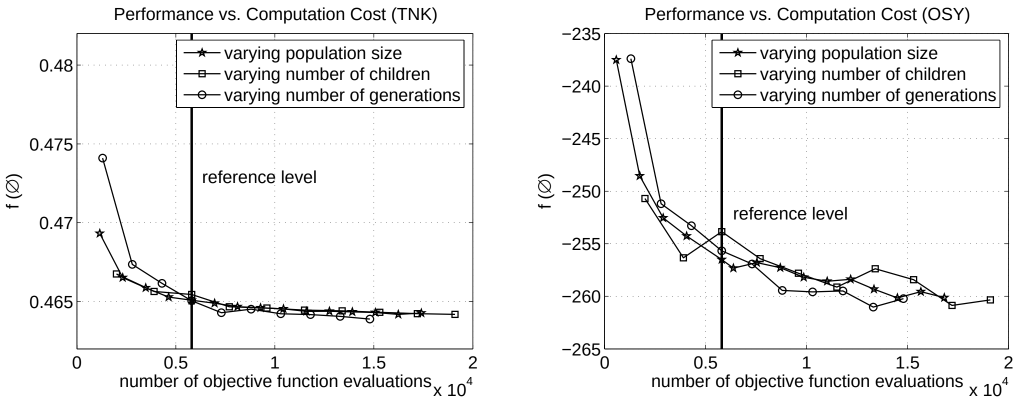


Figure 5.22: Performance vs. computational effort for the test problems TNK and OSY with respect to varying population size, r_{cp}, and the number of generations

in population size or selection pressure. And this advantage of letting the evolution run over more generations increases for configurations with more variables as can be seen for the configurations with 15 and 20 design variables. One explanation for this might be that for this problem good solutions are found in such a way that a particular individuum has to evolve along certain constraint borders by mutation. A process that needs time.

As a final investigation on this topic GAME has been used on a meta level to optimize its own parameters with respect to maximize performance and minimize computational costs. As test problem CTP1 has been chosen with a 15 design variable configuration. The design variables are the population size, the ratio r_{cp}, and the number of generations. The objectives have been the number of function evaluations and the average best solutions of 50 runs:

$$min\ \mathbf{z} = \mathbf{f}(\mathbf{x}),\ \mathbf{f}(\mathbf{x}) = [number\, of\, function\, evaluations(\mathbf{x}), z^*(\mathbf{x})]$$

with: $\mathbf{x} = [n, r_{cp}, n_{gen}]$

with:

$$z^*(\mathbf{x}) = \oslash(z_i^*(\mathbf{x}));\ i = 1, 2, ..., 50;\ z_i^*(\mathbf{x})\, best\, feasible\, solution\, of\, the\, problem\, CTP1$$

$$5 \leq n \leq 300$$

$$1 \leq r_{cp} \leq 10$$

$$5 \leq n_{gen} \leq 50$$

In figure (5.25) the results of this optimization are shown as a Pareto plot of the average best solution z^* versus the number of function evaluations.

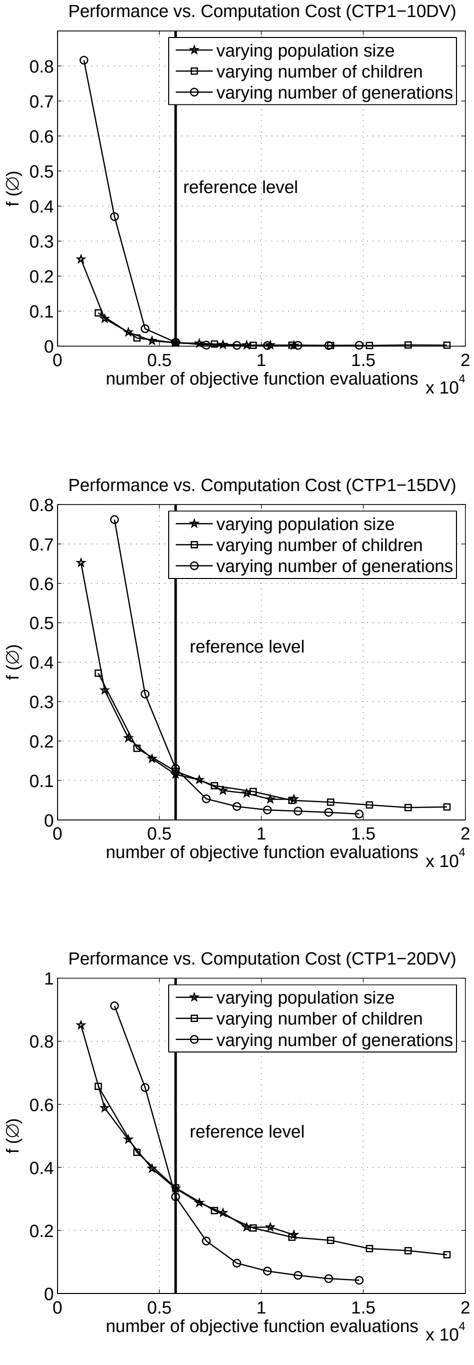


Figure 5.23: Performance vs. computational effort for the test problem CTP1 with 10, 15, and 20 design variables with respect to varying population size, r_{cp}, and the number of generations

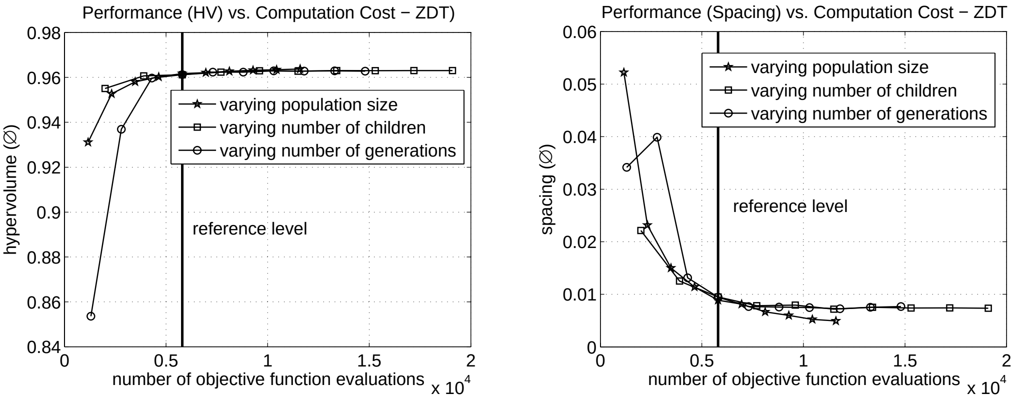


Figure 5.24: Performance vs. computational effort for the test problem ZDT with respect to varying population size, r_{cp}, and the number of generations

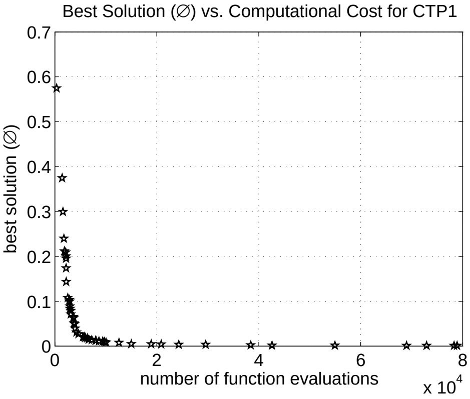
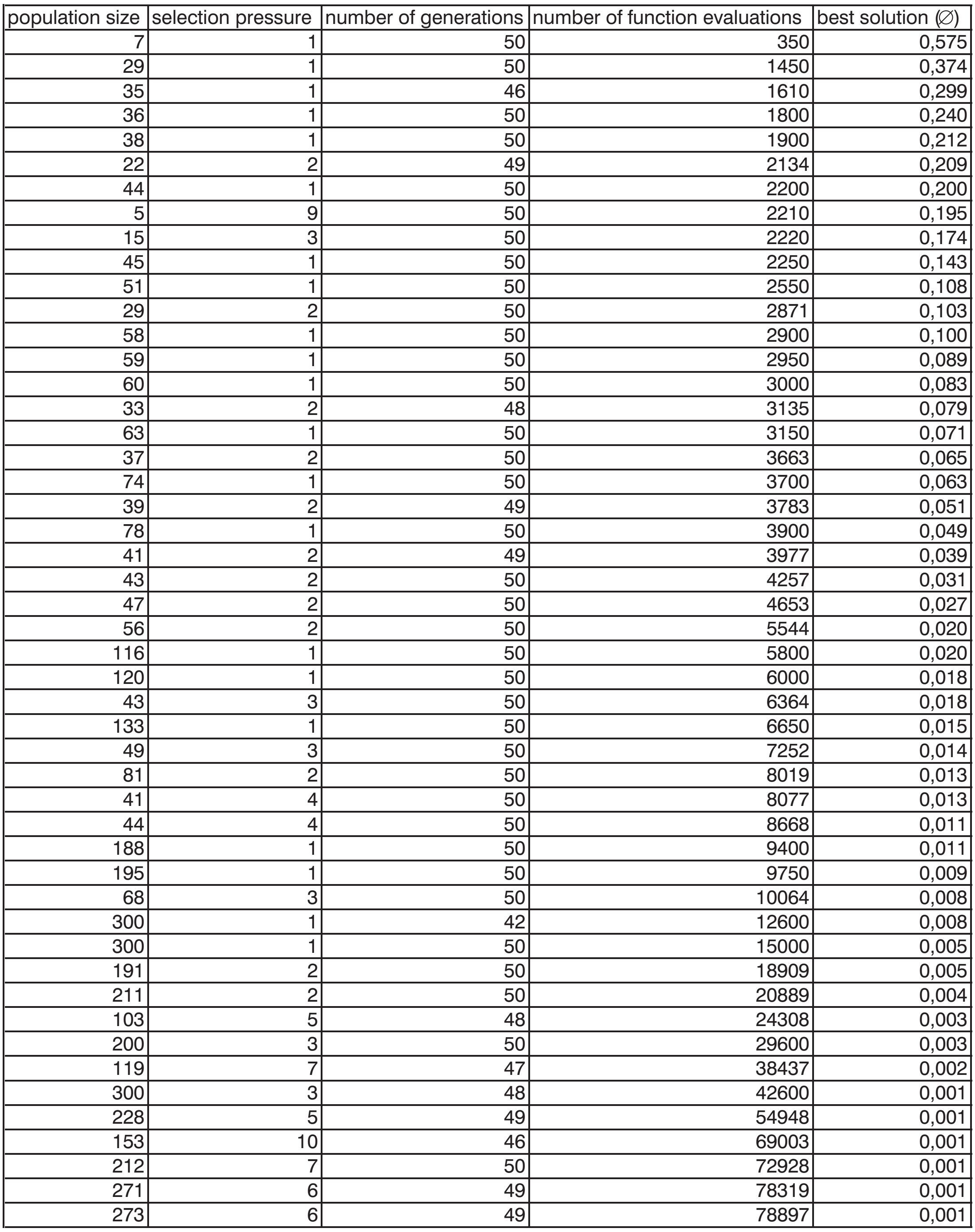


Figure 5.25: Pareto plot: average best solution vs. computation cost for CTP1

The Pareto front has almost a kink like shape. It shows that it is possible to improve the expected best solution down to values around 0.01 with relatively little additional costs. But to improve further one has to invest substantially more computational effort. With respect to the small potential of further improvement it makes no sense to invest more than roughly 15.000 function evaluations. This will lead to an expected solution of around $z^* = 0.005$. To reach the true optimum of 0.001 one has to invest at minimum 40.000 function evaluation, a surprisingly high number. This reveals again one of the major drawbacks of EAs: the weakness to determine the true optimum in the final search phase.

In table (5.2) it is shown how these Pareto optimal solutions relate to the design variables. In the table the solutions are sorted in an descending order according to the average solution quality z^* and therefore an ascending number of function evaluations. The data in table (5.2) gives answer to the question how the additional computation cost relate to the respective choices of the design variables. In accordance with the prior simulation results for CTP1 it can be seen that increasing the number of generations is obviously a very efficient parameter for achieving better results. For all Pareto optimal solutions the value for the number of generations has been driven



population size	selection pressure	number of generations	number of function evaluations	best solution (∅)
7	1	50	350	0,575
29	1	50	1450	0,374
35	1	46	1610	0,299
36	1	50	1800	0,240
38	1	50	1900	0,212
22	2	49	2134	0,209
44	1	50	2200	0,200
5	9	50	2210	0,195
15	3	50	2220	0,174
45	1	50	2250	0,143
51	1	50	2550	0,108
29	2	50	2871	0,103
58	1	50	2900	0,100
59	1	50	2950	0,089
60	1	50	3000	0,083
33	2	48	3135	0,079
63	1	50	3150	0,071
37	2	50	3663	0,065
74	1	50	3700	0,063
39	2	49	3783	0,051
78	1	50	3900	0,049
41	2	49	3977	0,039
43	2	50	4257	0,031
47	2	50	4653	0,027
56	2	50	5544	0,020
116	1	50	5800	0,020
120	1	50	6000	0,018
43	3	50	6364	0,018
133	1	50	6650	0,015
49	3	50	7252	0,014
81	2	50	8019	0,013
41	4	50	8077	0,013
44	4	50	8668	0,011
188	1	50	9400	0,011
195	1	50	9750	0,009
68	3	50	10064	0,008
300	1	42	12600	0,008
300	1	50	15000	0,005
191	2	50	18909	0,005
211	2	50	20889	0,004
103	5	48	24308	0,003
200	3	50	29600	0,003
119	7	47	38437	0,002
300	3	48	42600	0,001
228	5	49	54948	0,001
153	10	46	69003	0,001
212	7	50	72928	0,001
271	6	49	78319	0,001
273	6	49	78897	0,001

Table 5.2: Design variable and objective function data for the Pareto optimal solutions

up to the upper limit. So the increase of the computation costs is reflected mainly in the values of the population size and the selection pressure. For both parameters some spreading can be observed, a clear relationship can not be determined. E.g. a solution level of 0.020 can be reached with a population size of $n = 56$ and a $r_{cp} = 2$ as well as with $n = 116$ and $r_{cp} = 1$. This finding is also in accordance with those of the prior experiments. The achieved performance is primarily dependent on the invested computational effort and relatively robust with respect to how this given effort translates into an actual choice of the specific parameters. Nevertheless, it is interesting to see, that the actual values for the selection pressure are mainly at the lower bound of the possible range, most often even as low as one.

So in conclusion these findings are very interesting and encouraging because they show that the performance is not extremely sensitive to a proper choice of the parameters. So the potential user does not have to spent much effort on wondering how to transform a limited available computational resource into proper choices for n, r_{cp}, and n_{gen}. A somewhat reasonable choice will work without being at risk of spoiling performance significantly. The above findings also suggests that a higher number of generation should be favored over population size and selection pressure. Also they hint that lower values of selection pressure do not endanger the efficiency of the evolutionary search.

5.4 Comparison of Different Approaches for Constraint Handling

EAs are inherently unconstrained optimization methods, which evaluate candidate solutions only by one scalar measure: fitness. So the constraint handling is an essential challenge for the development of an EA. In GAME constraints are treated as additional objectives and aggregated in the fitness value via the goals and priorities approach (see chapter4.4).

In general penalty function approaches (see chapter 2.3.3) are most common for constraint handling in EAs ([Coe02],[Coe99b]). Another common and most simple approach is the so called 'lethal-factor' or 'death penalty' approach, which simply rejects all infeasible solutions. In order to evaluate the performance of the constraint handling approach of GAME, it has been compared to these two common representatives. For the penalty function the one published in ([JH94]), which is defined as (nomenclature according to equation 2.1):

$$
\begin{aligned}
&fit(\mathbf{x}) = f(\mathbf{x}) + (Cgen)^{\alpha} SVC(\beta, \mathbf{x}),\\
&with\ C, \alpha, \beta\, user\ supplied\ constants\ and\ SVC(\beta, \mathbf{x})\, defined\, as:\\
&SVC(\beta, \mathbf{x}) = \sum_{i=1}^{n_{ic}} D_i^{\beta}(\mathbf{x}) + \sum_{j=1}^{n_{ec}} D_j(\mathbf{x}),\\
&with\\
&D_i(\mathbf{x}) = \begin{cases} 0 & g_i(\mathbf{x}) \leq 0 \\ |g_i(\mathbf{x})| & g_i(\mathbf{x}) > 0 \end{cases} \qquad 1 \leq i \leq n_{ic}\\
&D_j(\mathbf{x}) = \begin{cases} 0 & -\varepsilon \leq h_j(\mathbf{x}) \leq \varepsilon \\ |h_j(\mathbf{x})| & \{(h_j(\mathbf{x}) < -\varepsilon) \vee (h_j(\mathbf{x}) > \varepsilon)\} \end{cases} \qquad 1 \leq j \leq n_{ec}
\end{aligned}
\tag{5.10}
$$

According to the authors the user supplied constants have been set to $C = 0.5$, $\alpha = 2$, and $\beta = 2$. The above penalty function is a dynamic penalty function, since the penalty increases with the generation number gen.

5.4.1 Experiment Configuration

For the comparison the test problems OSY, CTP1, and the cantilever beam have been used. Again for the ease of a simpler comparison of the results the single objective versions of the test problems have been employed for this comparison. The CTP1 problem has been applied in two configurations: 10 variables and 10 constraints, and 30 variables and 50 constraints. The criterium for judging the respective constraint handling methods has been the quality of the best feasible solution in the final population. The configuration for the experiments is given in table (5.3). The only element of the algorithm that changes is the constraint handling method. All components of the algorithm have been kept unchanged. For each method 100 runs have been performed.

	configuration
population size *n*	20 (CTP1: 200)
ratio r_{cp}	4
number of children n_c	80 (CTP1: 800)
number of generations n_{gen}	20

Table 5.3: Setup for the experiments varying the different parameters

5.4.2 Experiment Results

In table (5.4) the results are listed. In contrast to barrier function methods penalty function methods may result in infeasible solutions close to the true optimum. This is typically the case if they are applied with gradient based search methods. But when applied in EAs this drawback is compensated by the scattering characteristic of the mutation operator. So in the population there will be both infeasible and feasible solutions near the true optimum. For the evaluation of the performance of each constraint handling method the best feasible solution in the final generation is selected, although in the case of the penalty function method there may be better solutions in terms of the aggregate objective function. For each constraint handling method the mean value $\varnothing$ of the best feasible solution found as well as the respective standard deviation σ for all 100 experiments are shown. Additionally the percentage of feasible solutions in the final population is given.

5.4.3 Discussion

The constraint handling method implemented in GAME (goals and priorities approach) proves to perform better than the two alternatives for all experiments. For the cantilever beam the average value for the best feasible solution in GAME is about 13% better than the one provided by the dynamic penalty method and about 18% better than the one of the lethal-factor method. The standard deviation of the solutions found by GAME is in neither case higher than the one of the two alternatives and is in each case below 6%. For the test problem OSY GAME also clearly outperforms the penalty function approaches. An interesting detail is that here the lethal-factor achieves a better result than the dynamic penalty. On the other hand the lethal-factor shows a very low percentage of feasible solutions. This reflects the underlying logic of the lethal-factor approach. The low percentage of feasible solutions for the lethal-factor reconfirms that this method basically supports the 'lucky punch' and not systematically favors solutions with a less severe constraint violation. The high percentage of feasible solutions for the dynamic penalty shows a homogeneous population, but obviously the penalty was not strong enough to force the best ones close enough to the feasible region. This coincides with the inherent logic of the dynamic penalty function approach, that has the solutions converging from the infeasible side of the constraint boundary. For the more complex problem CTP1 again the goals and priorities approach shows the best results. For the first configuration GAME reaches a solution, which is nearly 60% better than the one of the dynamic penalty approach. The lethal-factor approach fails completely to find feasible solutions. For the second configuration with 50 constraints only GAME finds feasible solutions, both penalty approaches fail. After increasing the population size to 1000, the number of children to 4000, and the number of generation to 30 the dynamic penalty approach was finally able to find feasible solutions. But even with this significantly higher computational

test function		GAME	dynamic penalty	lethal factor
OSY	∅	-237.4	-165.0	-210.8
	σ	17.4	92.9	33.8
	% feas.	87	86	11
CTP1(10/10)	∅	0.0047	0.0117	NaN
	σ	0.0168	0.0011	NaN
	% feas.	100	100	0
CTP1(30/50)	∅	0.3918	NaN	NaN
	σ	0.0816	NaN	NaN
	% feas.	100	0	0
Beam	∅	8.938	10.316	11.013
	σ	0.508	0.571	0.254
	% feas.	100	100	100

Table 5.4: Table with the results of the constraint handling experiments

effort the average optimum was still of relatively low quality (0.4521).
In conclusion the experiments show that the implemented constraint handling method in GAME clearly outperforms the considered penalty approaches. The logic behind the goals and priorities approach to concentrate the search on feasibility first seems to pay off here.
It has to be stated, that here only two typical examples of penalty function approaches have been considered. There exists a broad variety of other implementations, which may perform better. A better tuning of the penalty parameters may also have shown better results. But on the other hand this is maybe the most significant advantage of the goals and priorities approach implemented in GAME, that it does not depend on any user supplied parameters.

5.5 Influence of the Standard Deviation Adaptation

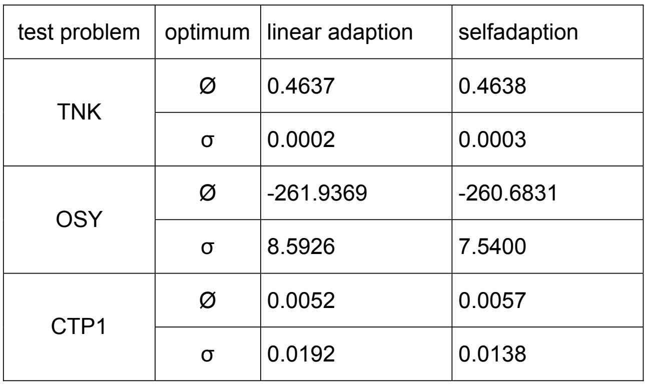
Mutation is the central reproduction operator in GAME. For all continuous or quasi-continuous design variables an ES based mutation is applied adding normally distributed random changes to the respective variables. The standard deviation of the applied normal distribution can be changed according to different laws during the evolution run (see chapter 2.3.2). In GAME a linear decreasing adaptation is implemented. In this section it is investigated how this implementation performs in comparison to a more advanced technique, the self-adaptation of the standard deviation.

5.5.1 Experiment Configuration

For comparing the different methods of adapting the standard deviation experiments have been performed with the single-objective versions of the test problems TNK, OSY, and CTP1 (10 design variable version) as well as with multiobjective test problem ZDT (10 design variable version). The population size has been set to $n = 100$, the number of children to $n_c = 300$, and the number of generations to $n_{gen} = 20$. The different adaptation methods have been implemented as defined in equation (2.6) for the linear adaption and in equation (2.7) for the self-adaptation. For each method 100 runs have been performed.

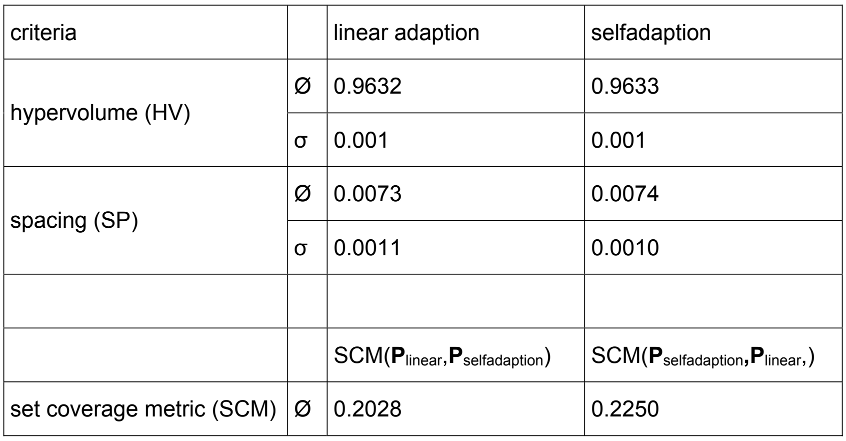
5.5.2 Experiment Results

In table (5.5) the results for the single objective test problems are listed, in table (5.6) the results for the multiobjective test problem ZDT.



test problem	optimum	linear adaption	selfadaption
TNK	Ø	0.4637	0.4638
	σ	0.0002	0.0003
OSY	Ø	-261.9369	-260.6831
	σ	8.5926	7.5400
CTP1	Ø	0.0052	0.0057
	σ	0.0192	0.0138

Table 5.5: Mean values ($\oslash$) and standard deviations (σ) for the achieved optima of the test problems TNK, OSY, and CTP1 for the different standard deviation adaptation methods



criteria		linear adaption	selfadaption
hypervolume (HV)	Ø	0.9632	0.9633
	σ	0.001	0.001
spacing (SP)	Ø	0.0073	0.0074
	σ	0.0011	0.0010
		SCM($\mathbf{P}_{\text{linear}}$,$\mathbf{P}_{\text{selfadaption}}$)	SCM($\mathbf{P}_{\text{selfadaption}}$,$\mathbf{P}_{\text{linear}}$,)
set coverage metric (SCM)	Ø	0.2028	0.2250

Table 5.6: Mean values ($\oslash$) and standard deviations (σ) for the different multiobjective criteria (HV, SP, SCM) of the test problems ZDT for the different standard deviation adaptation methods

5.5.3 Discussion

The most obvious conclusion of all results is that employing a self-adapting technique for the standard deviation of the mutation operator shows no advantages in comparison to a simple linear self adaptation. Within stochastic errors the results are nearly identical. There are three likely reasons for this. The first one refers to the general working principle of the self-adaptation. The self-adaptation technique is an indirect method, the quality of a certain standard deviation is not judged directly, but by the evolutionary success of the resulting individuals. But due to the stochastic nature of the mutation a good individual not necessarily is the result of a good choice for the standard deviation. Although with a lower probability a good individual can also be generated with a bad choice for the standard deviation. So the self adapting process

is a feedback loop that relies on statistical evidence. This can only work if a significantly large number of offspring is produced with the same or comparable setting of the standard deviation. With a selection pressure of three this prerequisite is given to some extent, but still the performance is not better than the one of a simple linear adaptation law as the experiments show.
Another reason is that it is doubtful if complicated adaptation techniques are necessary at all. Because this implies that in different areas of the design space different choices for the standard deviation are advantageous. Though theoretical comprehensible in practise the relevance might be overrated as these experiments show.
Finally the simple linearly decreasing adaptation of the standard deviation is a very effective adaptation law, because it reflects the concept of convergence: A widespread search at the beginning when the solutions can be assumed to be far away from the optimum and an exploration of the design space is required to single out promising areas, and a more focussed search towards the end of the optimization run when the solutions can be assumed to be close to the optimum.
In conclusion the chosen linear adaptation method for varying the standard deviation of mutation operator has performed well in comparison with a more sophisticated adaptation law.

5.6 Performance of the Integration of Response Surface Methods

The main goal for integrating RSA methods has been to further exploit the information about the design space available during the evolution and thus increase the performance and efficiency of the evolutionary algorithm. So on the one hand this aims at achieving better solutions and on the other hand at decreasing the computational effort for reaching a given solution quality in comparison with a standard EA.

5.6.1 Experiment Configuration

In order to verify if these goals are reached, experiments with several test problems, single- and multiobjective, have been performed. For comparing the performance effects of the RSA integration GAME has been applied with and without the RSA. In order to evaluate the efficiency increase the experiments have been performed for different population sizes. The test problems for the single-objective problems are the respective versions of TNK, OSY, CTP1, and the cantilever beam. For the CTP1 test problem a configuration with 10 design variables and 20 constraints has been chosen. For the multiobjective problems the test problems ZDT, TNK, OSY, CTP1, and the cantilever beam have been employed. In table (5.7) the configurations of the respective experiments are shown. To reduce the computational effort the experiments for the configurations with the RSA branch have been limited to the population size at which a clear picture about the effect of the RSA application could be drawn. Again for each configuration 100 experiments have been conducted.

	configuration
population size *n*	20,40,...,200
ratio r_{cp}	3
number of children n_c	60,120,...,600
number of generations n_{gen}	20

Table 5.7: Configuration of the experiments evaluating the influence of the RSA integration

5.6.2 Experiment Results

In figures (5.26), (5.27), (5.28), and (5.29) the results for the single objective test problems are displayed. The results are displayed in the same way as defined in chapter (5.3.2).

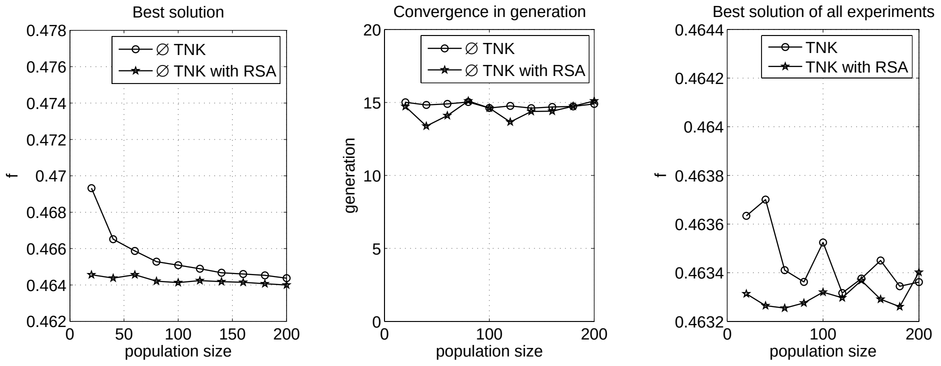


Figure 5.26: Experiment results for test problem TNK

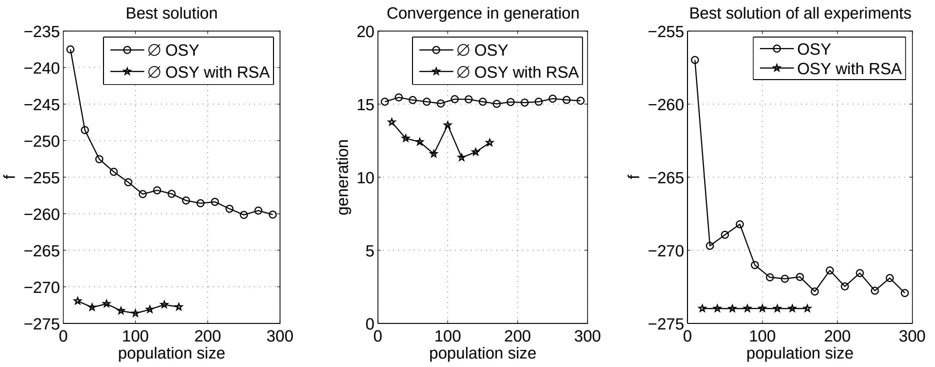


Figure 5.27: Experiment results for test problem OSY

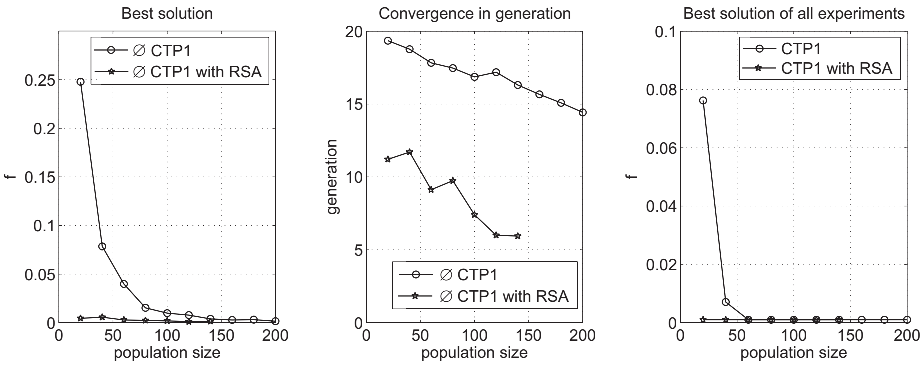


Figure 5.28: Experiment results for test problem CTP1

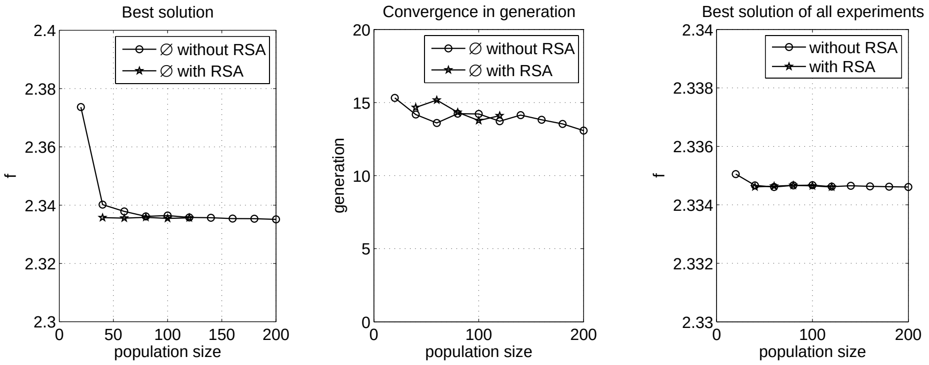


Figure 5.29: Experiment results for the cantilever test problem

The results of the multiobjective test problems are basically displayed the same way as described in chapter (5.3.2) with the exception of the set coverage metric. Here first for a certain population size n_i the solutions sets $\bar{\mathbf{P}}^{RSA}_{n_i,l}$ of all experiments with the RSA and $\bar{\mathbf{P}}_{n_i,k}$ of all experiments without the RSA have been compared:

$SMC(\bar{\mathbf{P}}_{n_i,k}, \bar{\mathbf{P}}^{RSA}_{n_i,l}), SMC(\bar{\mathbf{P}}^{RSA}_{n_i,l}, \bar{\mathbf{P}}_{n_i,k}),\ k, l\epsilon[1, 100]$. Then for all those SMC values the respective mean values are computed: $S\bar{M}C(\bar{\mathbf{P}}_{n_i}, \bar{\mathbf{P}}^{RSA}_{n_i})$ and $S\bar{M}C(\bar{\mathbf{P}}^{RSA}_{n_i}, \bar{\mathbf{P}}_{n_i})$. These mean values are then plotted in dependence of the population size n_i. The results of the unconstrained multiobjective test problem ZDT are shown in figures (5.30) and (5.31). The results for the constrained multiobjective test problems TNK, OSY, CTP1, and the cantilever beam are displayed in the figures (5.32), (5.33), (5.34), (5.35), (5.36), and (5.37).

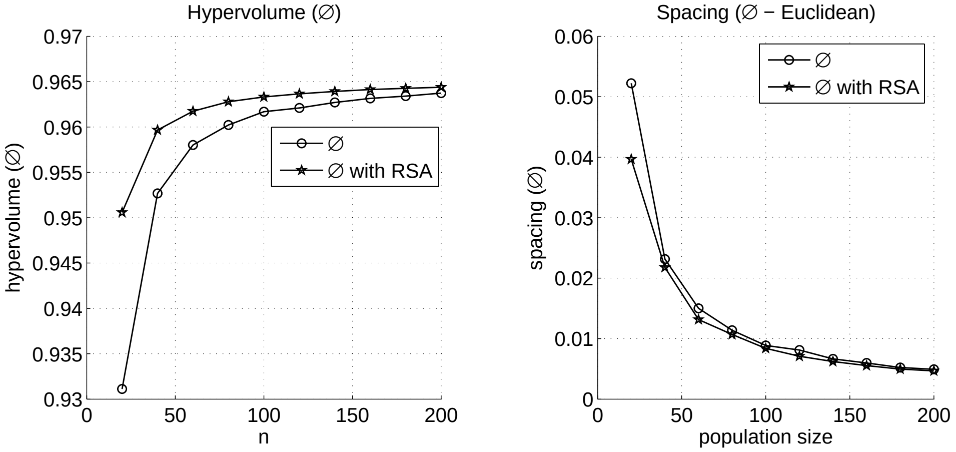


Figure 5.30: Hypervolume and spacing results for the test problem ZDT

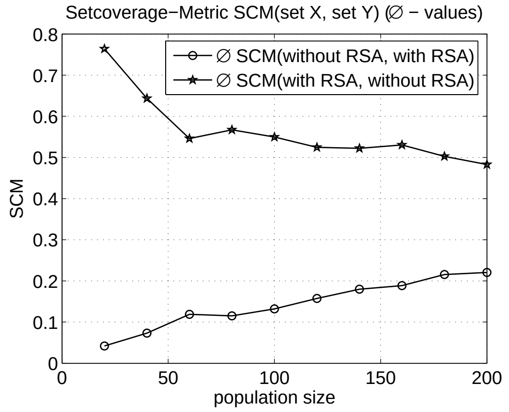


Figure 5.31: Set coverage metric results ($S\bar{M}C(\bar{\mathbf{P}}_{n_i}, \bar{\mathbf{P}}^{RSA}_{n_i})$,$S\bar{M}C(\bar{\mathbf{P}}^{RSA}_{n_i}, \bar{\mathbf{P}}_{n_i})$.) for the test problem ZDT

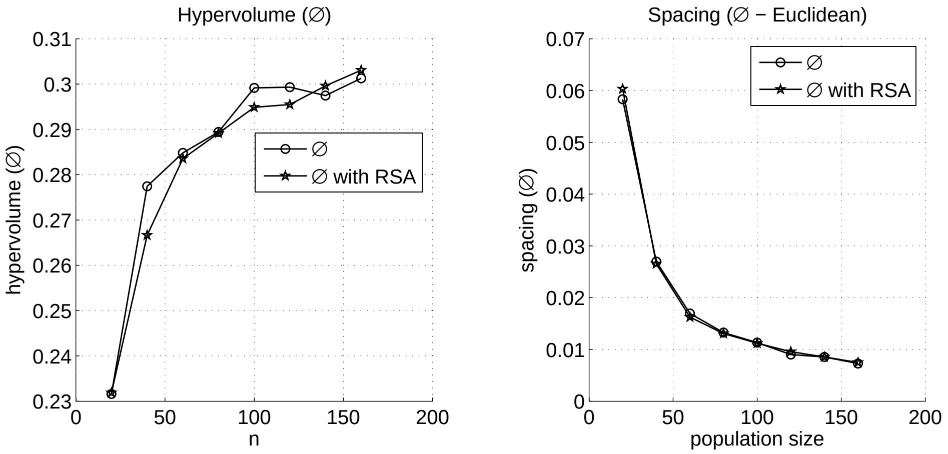


Figure 5.32: Hypervolume and spacing results for the test problem TNK

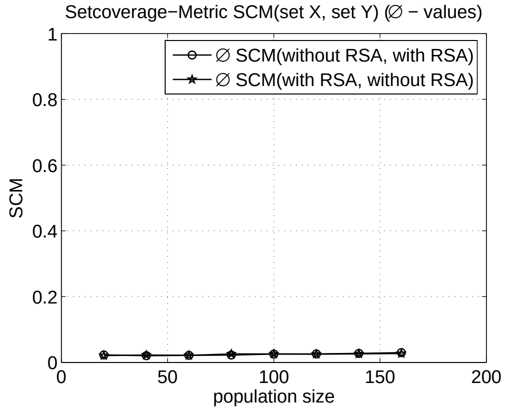


Figure 5.33: Set coverage metric results ($S\bar{M}C(\bar{\mathbf{P}}_{n_i}, \bar{\mathbf{P}}_{n_i}^{RSA})$, $S\bar{M}C(\bar{\mathbf{P}}_{n_i}^{RSA}, \bar{\mathbf{P}}_{n_i})$) for the test problem TNK

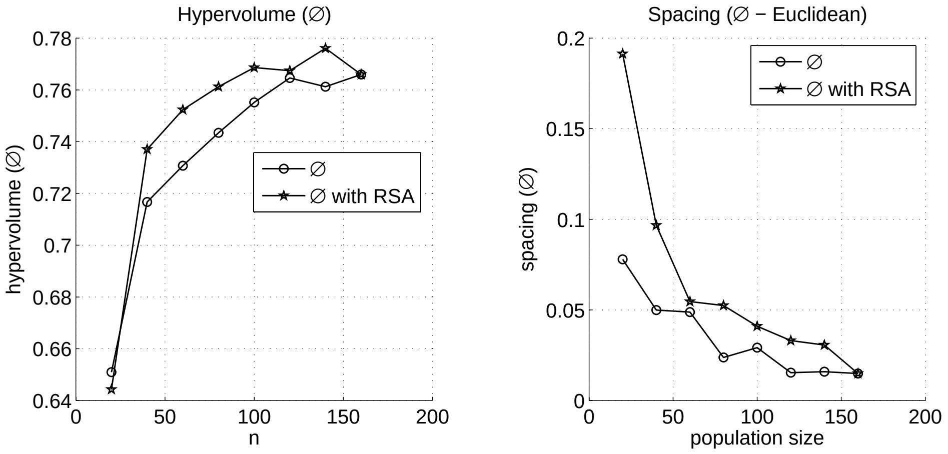


Figure 5.34: Hypervolume and spacing results for the test problem OSY

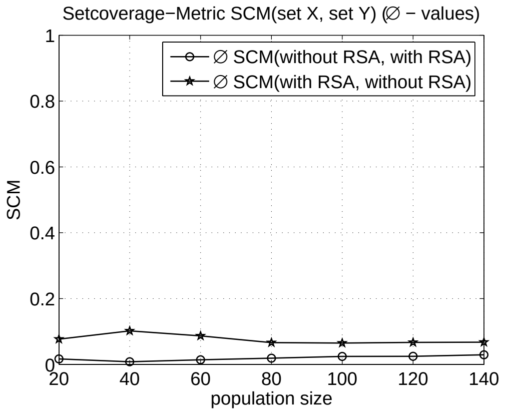


Figure 5.35: Set coverage metric results $(\bar{SMC}(\bar{\mathbf{P}}_{n_i}, \bar{\mathbf{P}}_{n_i}^{RSA})$, $\bar{SMC}(\bar{\mathbf{P}}_{n_i}^{RSA}, \bar{\mathbf{P}}_{n_i}))$ for the test problem OSY

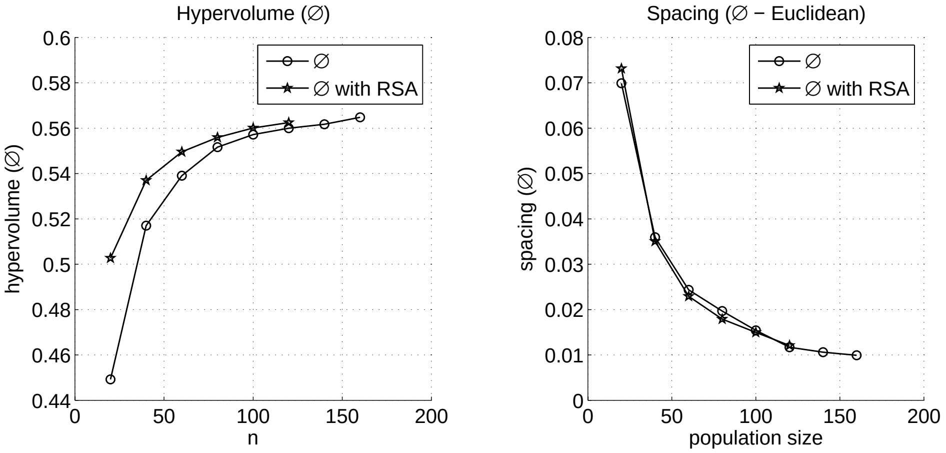


Figure 5.36: Hypervolume and spacing results for the test problem CTP1

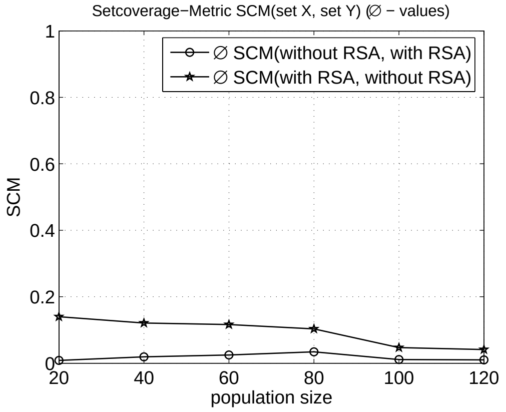


Figure 5.37: Set coverage metric results $(\bar{SMC}(\bar{\mathbf{P}}_{n_i}, \bar{\mathbf{P}}_{n_i}^{RSA})$, $\bar{SMC}(\bar{\mathbf{P}}_{n_i}^{RSA}, \bar{\mathbf{P}}_{n_i}))$ for the test problem CTP1

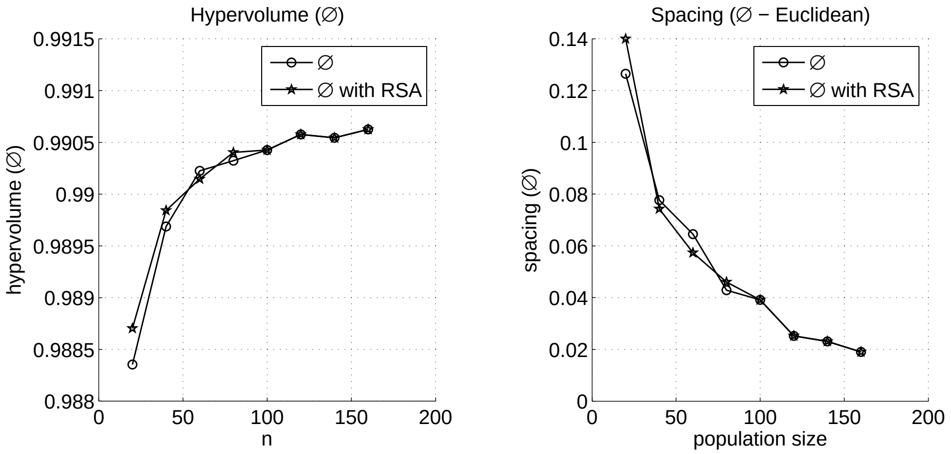


Figure 5.38: Hypervolume and spacing results for the cantilever beam test problem

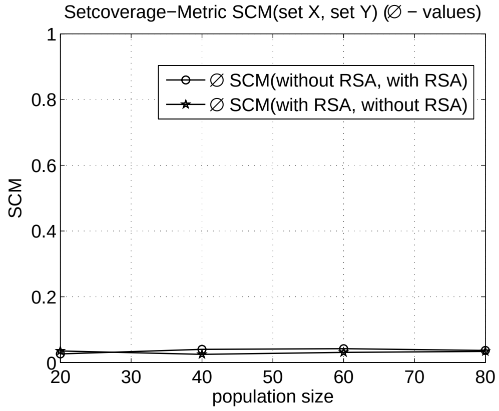


Figure 5.39: Set coverage metric results $(S\bar{M}C(\bar{\mathbf{P}}_{n_i}, \bar{\mathbf{P}}_{n_i}^{RSA})$, $S\bar{M}C(\bar{\mathbf{P}}_{n_i}^{RSA}, \bar{\mathbf{P}}_{n_i}))$ for the cantilever beam test problem

5.6.3 Discussion

The results for assessing the effects of integrating RSA methods in GAME are ambiguous. For some problems the RSA integration has led to substantial performance improvements while for other hardly any changes could be observed. The potential performance increase is obviously highly dependent on the problem character and ranges from an enormous improvement to no improvement at all.

Results for the Single-Objective Problems

Since the optimization algorithm applied for the RSA based optimization (*fmincon*, a SQP implementation in MATLAB) is a single-objective algorithm, the biggest improvements can be expected for the single-objective problems.

Reviewing the different results it can be seen that the RSA integaration improves the expected value for the best solution, it improves the overall best solution of all runs for a certain configuration, and it leads to an earlier finding of the respective optimum. But to which extent these positive effects happen varies significantly and is strongly dependent on the characteristics of the respective test problem on the one hand and on the other hand on the population size.

From the experiments varying the population size it could be seen that a growing population size leads to improved solutions, which finally converge against the true optimum. Thus the potential improvement by the RSA integration decreases with a growing population size. This is clearly reflected in the results.

Besides finding better solutions the other main objective for integrating the RSA has been to lower the computational costs. Thus most interesting are the improvements for lower population sizes. And exactly here the largest improvements can be seen.

While the test problems TNK and the cantilever beam show only small improvements, for the problems OSY and CTP1 large improvements could be achieved. In figure (5.40) the relative improvements of the expected optimum by applying the RSA is displayed in dependency of the population size.

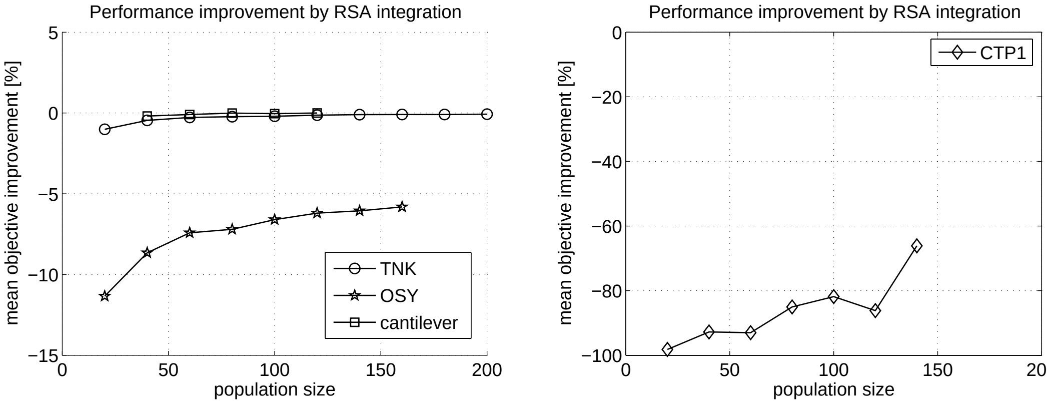


Figure 5.40: Relative improvement of the expected value for the best solution due to the RSA integration

For TNK and the cantilever beam only improvements less than five percent can be seen. But for OSY improvements larger than 10% and for CTP1 even larger than 90% are achieved. Furthermore for the test problem OSY the true optimum $(f_1 = -274)$ is reached from a population size as low as 40 on upwards not only as a single event but nearly always. It has to be noted that without employing the RSA solutions similarly close to the true optimum are not reached at all within the simulated spectrum of population sizes. Even for the highest simulated population size of 300 the average optimum without the RSA is still 5% worse than the one for a population size of 100 with the RSA. Taking into account that without employing the RSA branch the average optimum improves only slowly with an increasing population size, it can be assumed that a significantly larger population size must be applied to reach similarly good results as with the RSA. Furthermore the optima are found approximately 2 generations earlier. So very conservatively speaking at least a reduction of 88% in computational cost could be reached for the test problem OSY by applying the RSA. So the potential of the RSA integration to improve the performance and reduce the computational cost is clearly verified here.

An even greater improvement can be observed for the test problem CTP1. Here the true optimum ($f_1 = 0.001$) is reached in almost all cases even for the lowest population size of 20, from a population size from 80 on upwards the true optimum is reached in fact in all runs. And for the lower population sizes the optimum is found at least 6 generations earlier. The advantages diminish expectedly with growing population sizes. But the important point is that by employing the RSA lower population sizes can be chosen and still the same or even better solutions can be achieved as for much larger population sizes without using the RSA. In order to match the performance of GAME with the RSA for the lowest population size ($n = 20$) at least a population size of $n = 140$ is necessary for GAME without the RSA. In this case the integration of the RSA leads to a reduction of more than 85% in computational costs.

The overall improvements for the test problem TNK and the cantilever beam are much less impressive. One reason for this is that potential improvement is rather limited in both cases. This is even the case for the lowest population size because GAME already manages to perform quite well without the RSA. For $n = 20$ GAME without the RSA manages to achieve a best solution not further away from the true optimum as 1.2% for TNK or 1.7% for the cantilever beam respectively.

Nevertheless it can be observed that especially for lower population sizes the RSA helps to come closer to the true optimum of the TNK problem, the same applies for the cantilever problem. And although the absolute differences are small it has to be stated that even for the highest simulated population size $n = 200$ GAME without the RSA actually never reaches the level that GAME with the RSA already achieves for $n = 20$. So here also reduction in computational cost of at least 90% is possible. For the cantilever beam the potential savings are lower, but still about 50%. Here a population size of $n = 80$ for GAME without the RSA is required to match the performance of GAME with the RSA and $n = 40$. For this test problem with 2 truly discrete variables another effect can be observed. Due to the presence of 9 continuous subspaces, the population size $n = 20$ has not been sufficient for building up the RSAs in all those subspaces. Since there are three continuous variables statistically at least 90 individuals are necessary to build up RSA with a mixed quadratic model in all subspaces, 63 and 40 for pure quadratic or linear models respectively. In practise it can be seen that $n = 40$ are sufficient for using the RSA branch, as of course not all subspaces are present during the whole run.

Concerning the discrete design variables it can be seen, that they play a major role only in the first generations. The best possible combination is found quite soon. In the left plot of figure (5.41) the average generation, in which the best possible combination (I-profile, aluminum) has been found, is plotted in dependency of the population size. In average the best combination has been found by GAME within only 2 generations.

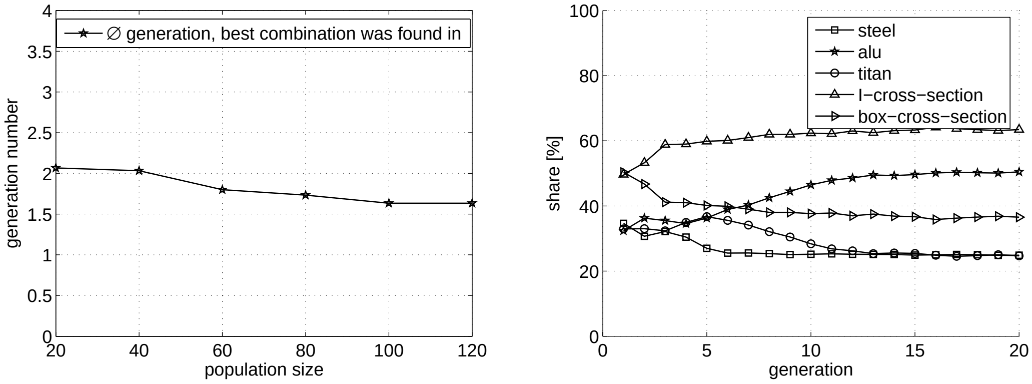


Figure 5.41: Convergence characteristic for the discrete design parameters of the cantilever beam

Additionally in the right plot of figure (5.41) the average shares of the different possible choices for the cross section and material are plotted in dependency of the generation number. It shows that even those choices for cross section and material not part of the optimal combination are present within the population until the end. So other combination can still produce competitive results. Also it can be seen that it takes much longer for GAME to reach a steady state equilibrium of shares for a discrete variable with three possible values (material) than for a discrete variable with only two ones (cross-section).

In addition to the limited potential of improvement e.g. in case of TNK, the small performance improvement by employing the RSA is also due to two other reasons: the suitability of the mixed quadratic model of the RSA to fit the actual objective- and constraint functions and the difficulty of the applied optimization algorithm with certain objective- and constraint function characteristics. The algorithm used for the RSA based optimization is *fmincon*, a MATLAB implementation of the SQP method ([Sch85b]). The test problem TNK features a nonconvex and very 'bumpy' constraint function involving a trigonometric expression, which constitutes a substantial problem for *fmincon*. Even if directly applied to the original TNK problem and not relying on possibly inaccurate approximations, *fmincon* fails in nearly 60% to converge successfully, if started from random initial design vectors. So the expected success ratio for the RSA based optimization is less than 50% anyway. An additional problem is the difficulty to built up accurate RSA for this constraint function with the employed mixed quadratic model functions. In figure (5.42) different plots show the performance of the RSA based optimization for sample runs with $n = 60$ for the test problems TNK and OSY respectively.

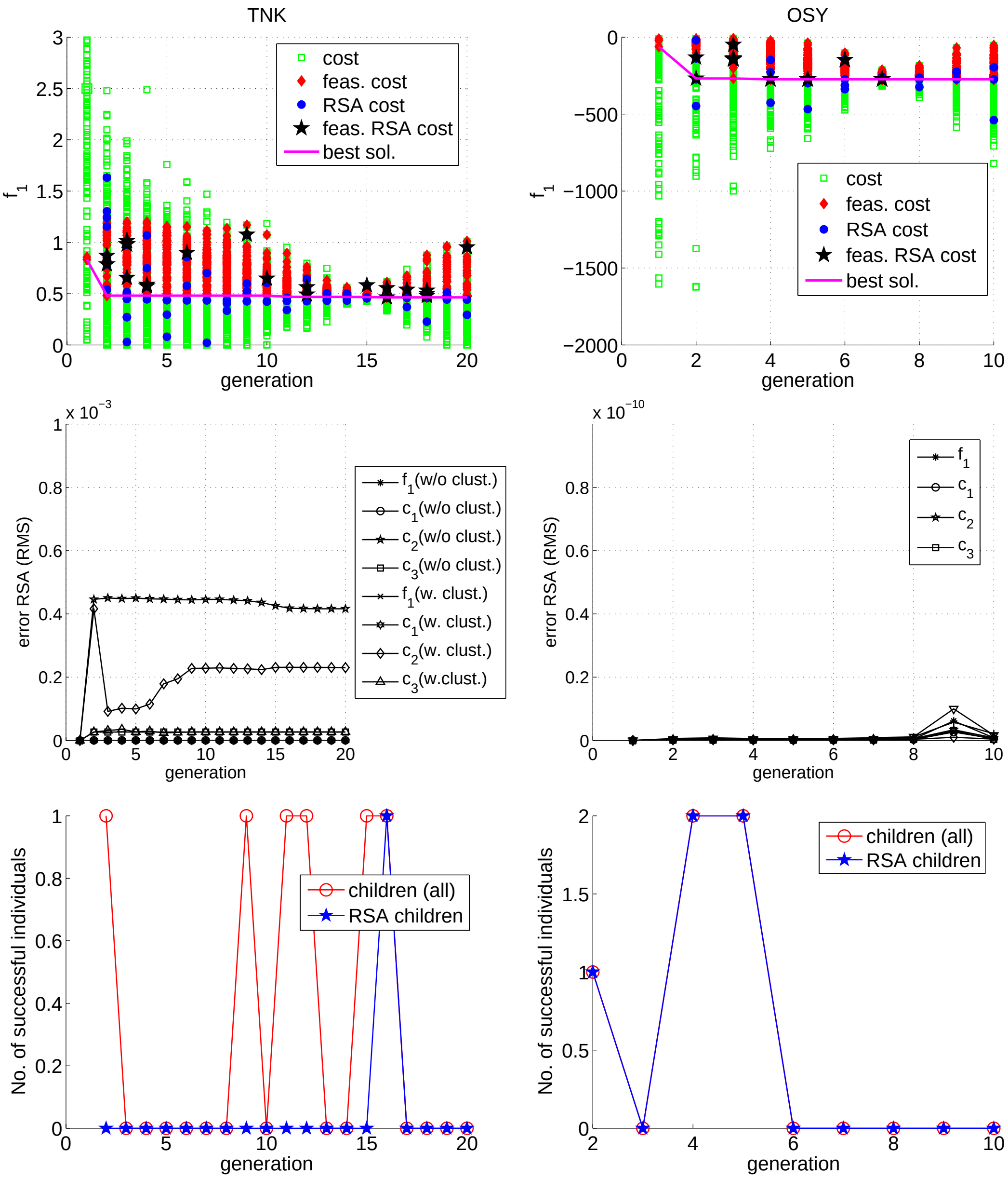


Figure 5.42: Performance assessment of the RSA integration for a sample run of the problem TNK (left) and OSY (right): convergence plot for f_1 (top), approximation error (RMS) for the different cost functions (middle), number of successful children (bottom)

On the top a convergence plot over the generations is plotted showing the distribution of the respective populations for the objective function f_1. Especially marked are the feasible individuals, the RSA based offspring (feasible and non feasible), and the currently best feasible solution. In the middle the approximation error for the respective RSA functions for the different objective and constraint functions are shown. On the bottom the number of all offspring and RSA based offspring individuals are plotted which posed a new best solution at their generation. Comparing the RMS error for the RSAs between TNK and OSY it can clearly be seen that the error for the second constraint of TNK is orders of magnitude higher than for all other functions. The employed clustering procedure is successfully able to reduce this error but not down to an comparable level. As a consequence the RSA based children are more successful for OSY than for TNK. For OSY the RSA based offspring individuals set a new best solution level three times. The first time being the most important, because there the RSA based child poses a very significant improvement. For TNK RSA based children are only once successful to set a new best solution and this is near the end of the evolutionary run with a relatively small improvement. So the TNK example shows some limitations of the chosen implementation of the RSA branch. One reason for this, the inherent difficulty of gradient based algorithms to handle nonconvex multimodal functions, cannot be changed. But for building sufficiently accurate RSA the chosen mixed quadratic models are obviously not adequate for certain problems. For compensating the limited abilities of the chosen simple model function a clustering approach subdividing the design space has been chosen (see chapter 4.8). With this approach the accuracy of the RSAs can be successfully increased, but there may still be problems, for which this strategy is not sufficient enough as can be seen for TNK.

Results for the Multiobjective Problems

Before discussing the results for the multiobjective test problems a difference to the single objective problems has to be discussed. Since the RSA based optimization uses the single objective *fmincon*-algorithm, for the multiobjective case the objectives have to be aggregated into a single objective substitute function. For this purpose the weighted sum approach is applied. But as a consequence, if the RSA based optimization run is successful, only a single point is added to the current nondominated frontier. But as the frontier consists of many points, the overall result cannot be expected to be influenced as strongly as for the single objective problems. In order to lessen this disadvantage, multiple fmincon-runs with different random weight settings are performed, so that multiple points are added to the population based on the RSA branch in each generation. For GAME the number of fmincon-runs per RSA model is chosen to be twice the number of objectives. The clustering procedure can further increase the total number. Nevertheless the ratio of the number of individuals coming from the RSA branch and the total number of children will still be relatively small, especially for higher population sizes. Typically it will be less than 10%.

However, also for the multiobjective test problems improvements can be observed. In figure (5.43) the relative improvement in terms of the hypervolume measure can be seen. The results are ambivalent again. While for the test problems ZDT, OSY and CTP1 significant improvements can be observed, basically no improvements could be achieved for the test problems TNK and the cantilever beam. With respect to the results of the single-objective experiments and considering the above reasoning this could be expected. The major reasons are in the first place the limited potential for improvements since GAME without the RSA branch already performs quite well.

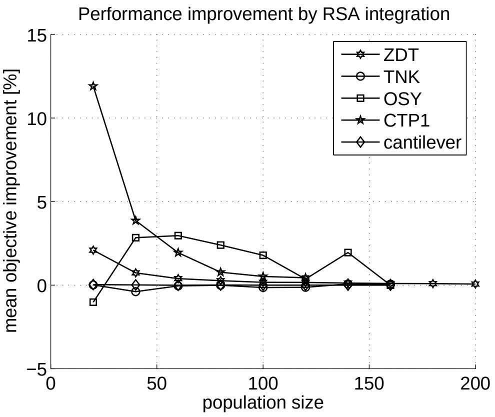


Figure 5.43: Relative improvement of the expected value for the hypervolume measure due to the RSA integration

And secondly for TNK again the problems of the applied SQP optimization algorithm with the nonconvexity of the test function are prohibitive for achieving substantial improvements. For the TNK example even degradations in terms of hypervolume can be observed. But due to the parallel implementation of the RSA branch there is no comprehensible reason why GAME with the RSA should perform worse than without it. The reason for this partial degradation can therefore rather be addressed to stochastic variation and some inherent characteristics of the hypervolume metric. For smooth continuous fronts it provides a suitable measure, while for disconnected, local, or non smooth fronts like for the test problems TNK and OSY, it can lead to large jumps if fronts are only partially covered.

Substantial improvements though can be achieved for the test problems ZDT and CTP1. The biggest improvements are found at lower population sizes: for $n = 20$ improvements of 2% for ZDT and 12% for CTP1 are achieved. The advantages decrease slowly towards higher population sizes. This behavior can be expected, because on the one hand the performance of a conventional EA without the RSA branch improves with a increasing population size (see chapter 5.3.2) as it becomes more and more a 'brute force' search. And on the other hand there is a definite upper limit for the hypervolume, the true Pareto frontier.

An improvement of only 2% in hypervolume for test problem ZDT may seem rather small but it has to be kept in mind that the hypervolume measure is highly dependent on the reference points used. In figure (5.44) two sample solution fronts are shown for a population size $n = 20$ with a hypervolume of $HV = 0.9546$ (with RSA) and $HV = 0.9251$ (without RSA) to demonstrate the difference of even low improvements in hypervolume measure for this test problem. A substantial improvement can be seen. This finding is also supported by the results for the set coverage metric (SCM). For $n = 20$ almost 80% of the solutions from optimizations without RSA branch are dominated by solutions from optimizations with the RSA branch, while vice versa only approximately 5% do so.

To a lower but still significant level this applies also for the results for CTP1. Here

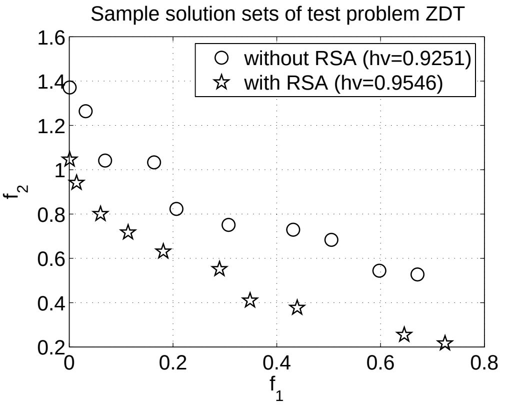


Figure 5.44: Sample solution set for the test problem ZDT for a population size $n = 20$

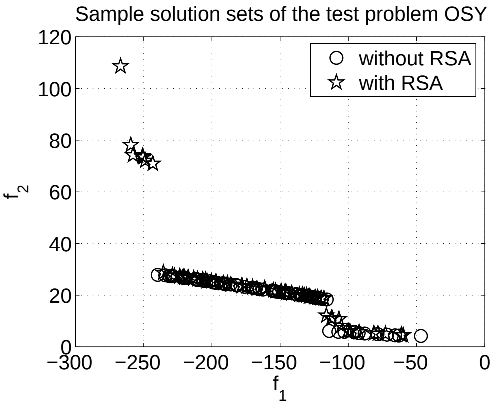


Figure 5.45: Sample solution set for the test problem OSY for a population size $n = 60$

the solutions determined with the RSA dominate up to 10% of the solution achieved without the RSA while vice versa this applies only for less than 2%.

To illustrate the performance of the RSA integration some RSA related measures as well as convergence plots are displayed for a sample run with the lowest population size $n = 20$ in figures (5.46) and (5.47). For the approximation error (RMS values) of the RSA functions only the first 4 functions are labelled due to the lack of space. Although the RSA for the second objective function shows a considerable error of up to $1E^{-3}$ the resulting RSA based optimization runs are highly successful. This can be seen in the number of successful children generated by the RSA branch, i.e. children that become part of the nondominated frontier in their respective generation. As can be seen in the right plot of figure (5.46) the RSA based children make up the largest part of the successful children. This can also be seen in the convergence plots in figure (5.47). In the right plot the situation at the end of generation 4 is plotted. Here the RSA based children not only pose the complete successful offspring, they also introduce a significant improvement with respect to the last generation. Also in the left plot, which shows all generations, the RSA based offspring makes up a considerable part of the final nondominated frontier. As a reference for judging the quality of the final nondominated front, the true Pareto frontier for the unconstrained problem is given.

The spacing results are nearly identical, so the quality of the fronts is basically the same. In conclusion for these two test problems substantial improvements are achieved by the application of the RSA branch. These improvements can be used for reducing the computational cost while keeping the the quality level of the results. E.g. for the test problem ZDT it requires only a population size of $n = 120$ with the RSA to achieve a comparable solution quality as for a population size of $n = 200$ without the RSA. So here a reduction in computational cost of 40% is possible.

For the test problem OSY the integration of the RSA also shows substantial improvements. In contrast to the single objective case here a population size of $n = 20$ is

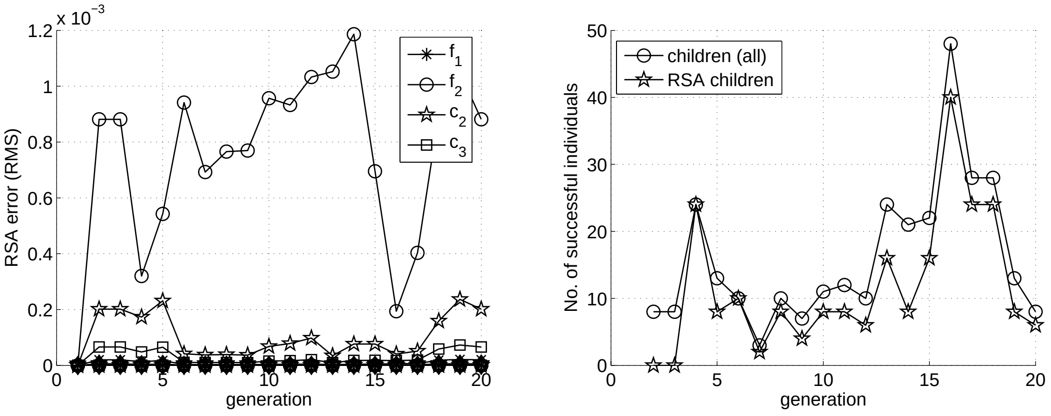


Figure 5.46: Performance assessment of the RSA integration for a sample run of the problem CTP1 (left: RSA error(RMS) of the different functions, right: number of successful children)

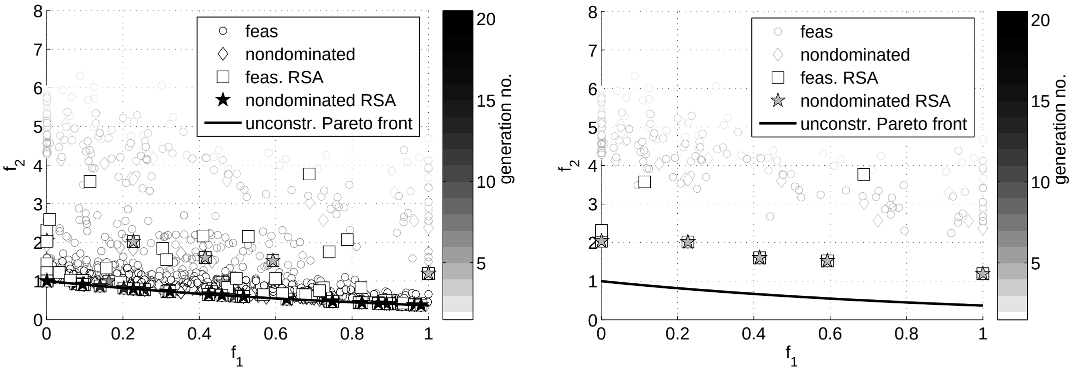


Figure 5.47: Convergence plot for a sample run of the test problem CTP1 (left: all generations, right: up to generation 4)

no sufficient basis for creating successful RSA based offspring. Only from $n = 40$ on upwards advantages are visible. The results of the set coverage metric also reflect these improvements, although there they are smaller. For $n = 40$ the solutions achieved with the RSA dominate 10% of the solutions achieved without it, while vice versa only less than 2% do so. Thus there is no dominant superiority of the solutions achieved with the RSA. The reason for this alleged contradiction between the relatively large improvement in hypervolume and the marginal superiority in terms of the set coverage metric can be seen in figure (5.45), in which two sample fronts of the test problem OSY are displayed for a population size of $n = 60$. Due to the piecewise character of the Pareto front it can happen easily that not all parts are covered by solutions, especially because some parts are significantly easier to find than others. For this sample run all solutions achieved without the RSA lie on the two lower right pieces of the front. Also most of the solution gained with the RSA do so. But with the help of the RSA a few solutions on more distant parts of the front could be determined. Because their number is relatively low this does not lead to a large advantage in the set coverage metric but increases the hypervolume considerably. However, the application of the RSA achieves the determination of solutions on hard to find parts of the frontier. Finally to illustrate the performance of the RSA integration during the search process for this problem again some performance measures of RSA as well as convergence plots are given in figure (5.48) and (5.49). As can be seen in the left plot in figure (5.48) the RSAs generally have a sufficient accuracy, only for certain generations like e.g. 5 and 12 high RMS errors can be observed. This coincides well with the fact that for those generation none of the resulting RSA based children has been successful. But for other generations the contribution of the RSA branch have been highly successful like e.g. for generation 4 where all successful children were contributed by the RSA branch. To see the improvement of the population by these respective children in the right convergence plot in figure (5.49) the situation at the end of this generation for is shown. The RSA based offspring improves the limits of the nondominated frontier significantly. Also for the last generation, shown in the left plot, one can see that the RSA based offspring contributes successfully to the nondominated frontier. In contrast to the test problems ZDT or CTP1 for OSY the advantage of applying the RSA is not so much the potential reduction of computational costs but the improvement of the achieved solutions.

Concluding it can be stated that the RSA integration in GAME successfully showed the potential to improve the solution quality as well as to increase the efficiency in comparison with conventional EAs. The chosen approach to use relatively simple model functions and built up RSAs only over smaller subsets of the design space in order to achieve sufficiently accurate approximations has also shown to work. Nevertheless the clustering approach also has its limits as the TNK example showed. Another problem with the RSA implementation that has been revealed during these experiments is due to the interaction with the chosen constraint handling in GAME. The applied SQP algorithms of course determines solutions on constraint borders.

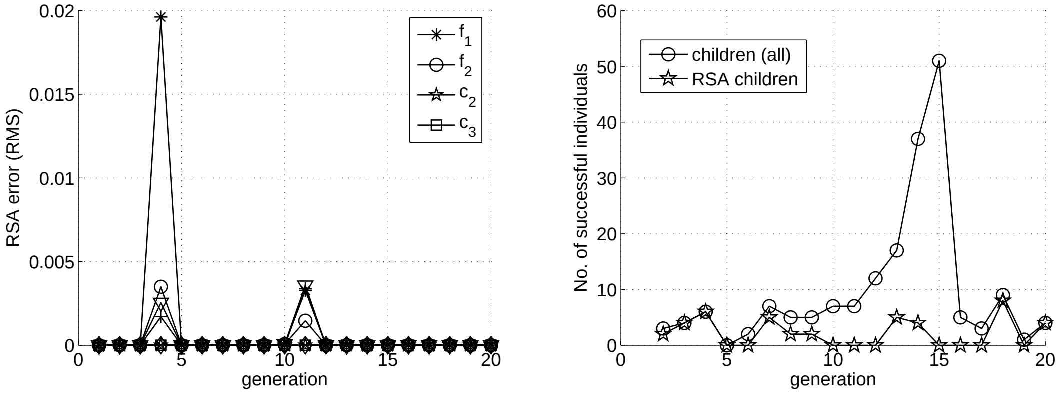


Figure 5.48: Performance assessment of the RSA integration for a sample run of the problem OSY (left: RSA error(RMS) of the different functions, right: number of successful children)

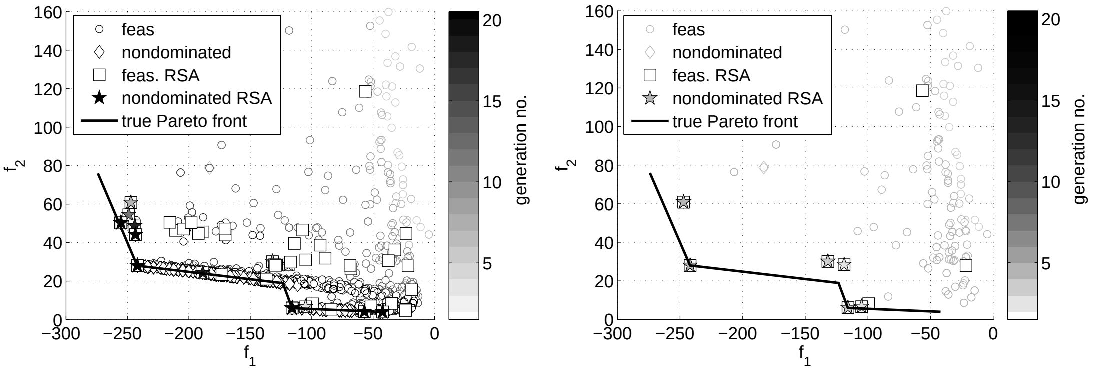


Figure 5.49: Convergence plot for a sample run of the test problem OSY (left: all generations, right: up to generation 4)

But due to the approximation errors of the RSAs the resulting optima are sometimes slightly infeasible when evaluated with the true constraint functions. The goals and priority approach discriminates sharply between feasible and infeasible solutions and thus assigns them a bad fitness. In order to solve this problem, all temporary solutions at each iteration of the SQP optimization are stored. In case the resulting optimum is infeasible a back-tracing procedure is performed on the search path and the last feasible solution is selected. The disadvantage of this procedure is that it requires additional objective function evaluations.

But summing up, the goals to improve the efficiency of conventional EAs as well as the solution quality by the integration of the RSA mehtods has been verified with these experiments.

5.7 Performance Comparison - GAME vs. NSGAII

In order to evaluate the performance of GAME with reference to other contemporary multiobjective evolutionary algorithms, it has been compared with the NSGAII algorithm ([DPAM02]). This algorithm has performed well in several benchmark comparisons and along with SPEA can be regarded as a representative of the state-of-the-art.

5.7.1 Experiment Configuration

Like before the comparison encompasses experiments with the single- as well as multiobjective configurations of the problems TNK, OSY, CTP1, ZDT, and the cantilever beam. In table (5.8) the configuration for the respective experiments are shown. For each configuration 50 runs have been performed. With respect to the implementation of NSGAII it has to be noted, that for the design variable representation and the re-

	configuration
population size n	100
ratio r_{cp}	1
number of children n_c	100
number of generations n_{gen}	50
probability for recombination p_r	0.2
probability for mutation p_m	0.9
initial standard deviation for mutation σ_0	0.2 (ub-lb)

Table 5.8: Configuration of the experiments for the performance comparison between GAME and NSGAII

production operators (recombination and mutation) the same ones as in GAME have been used, since in literature no detailed description could be found. So the NSGAII implementation used in this comparison is a real coded one. The decisive elements of the fitness assignment, selection and replacement operators have been implemented exactly according to [DAPM00]. Since in NSGAII the number of offspring matches the population size, GAME is also run with a corresponding configuration. So both algorithm are run with the same computational costs.

5.7.2 Experiment Results

In table (5.9) the results for experiments with the single-objective configurations of the respective test problems are summarized, in table (5.10) the results for experiments with the multiobjective problems.

For the comparison of nondominated solutions sets of GAME and NSGAII with the set coverage metric (SCM), the solution set of each GAME run has been compared with each NSGAII run: $SCM(\bar{\mathbf{P}}_{GAME,i}, \bar{\mathbf{P}}_{NSGAII,j})$, $SCM(\bar{\mathbf{P}}_{NSGAII,j}, \bar{\mathbf{P}}_{GAME,i})$, $i, j\epsilon[1, 50]$. The SCM results for all these combinations are displayed figure (5.50) for all test problems. The corresponding mean values are listed in table (5.10).

5.7.3 Discussion

As can be seen in table (5.9) GAME performs better than NSGAII for all single objective test problems. For the test problems TNK and the cantilever beam the advantages with respect to the expected best solution are small with 1.1% and 0.4%, but for the problems OSY and CTP1 significant advantages with 6.4% and more than 90% can be observed. Furthermore with respect to the absolute best optimum found in all 50 runs, GAME proved the potential to reach the true optimum of the respective problems while for NSGAII the absolute best solutions are quite a distance away from the true optimum. This applies especially for OSY and CTP1. The reasons for the better performance of GAME can be directly addressed to the

		best solution		generation in which the best solution was found		best solution of all runs
		(Ø)	(σ)	(Ø)	(σ)	
TNK	GAME	0.4640	5.8216e-004	22.3	0.9	0.4633
	NSGAII	0.4692	0.0032	23.2	1.9	0.4647
OSY	GAME	-266.0477	15.0051	21.3	3.9	-274.0000
	NSGAII	-250.1085	6.8367	24.1	2.5	-263.2893
CTP1	GAME	0.0019	0.0038	8.3	10.8	0.0010
	NSGAII	0.0824	0.0340	47.1	2.3	0.0235
Cantilever	GAME	2.3348	0.0002	35.3	6.4	2.3346
	NSGAII	2.3416	0.0091	36.9	11.9	2.3349

Table 5.9: Results for the single-objective test problems for the GAME and NSGAII

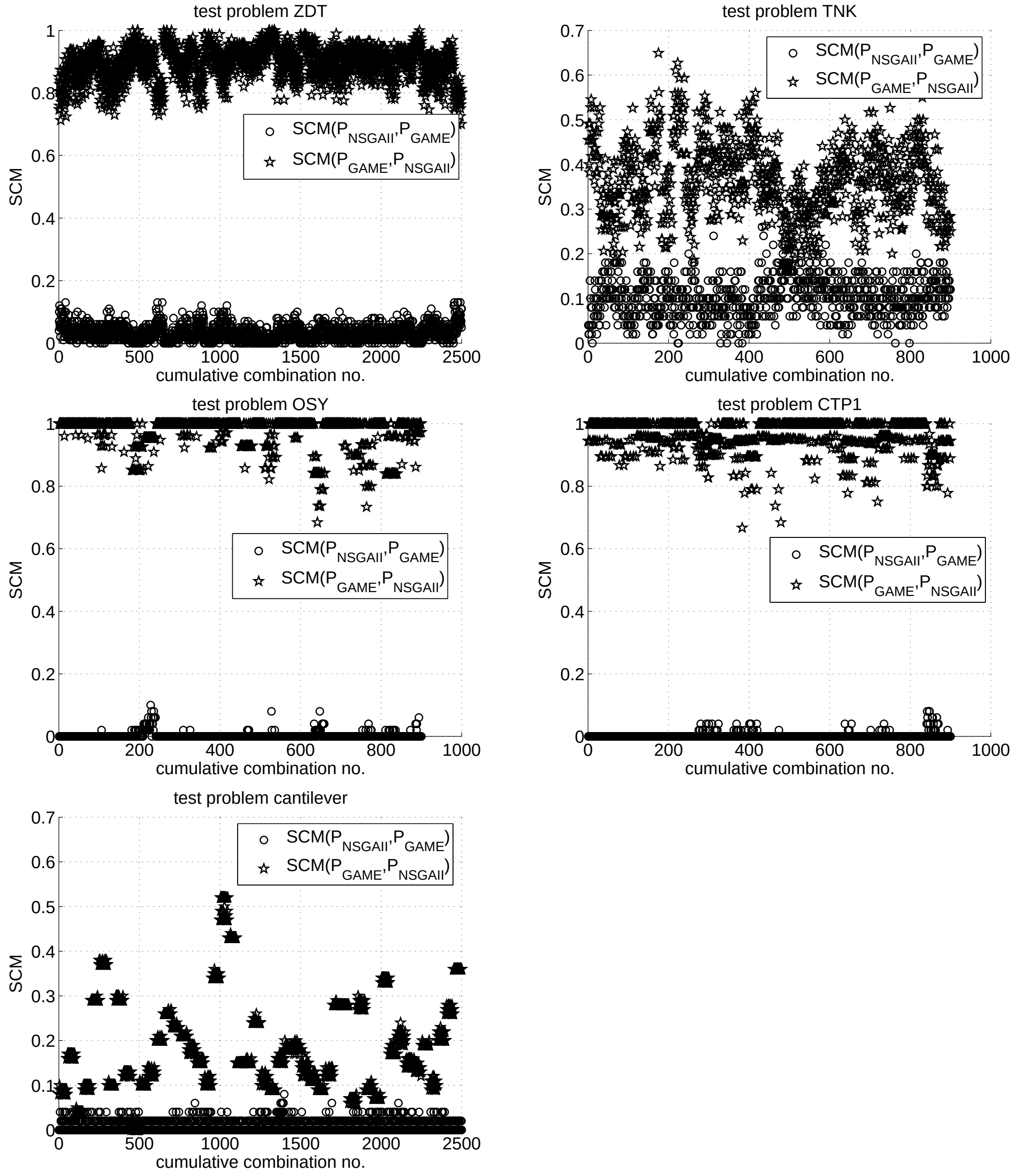
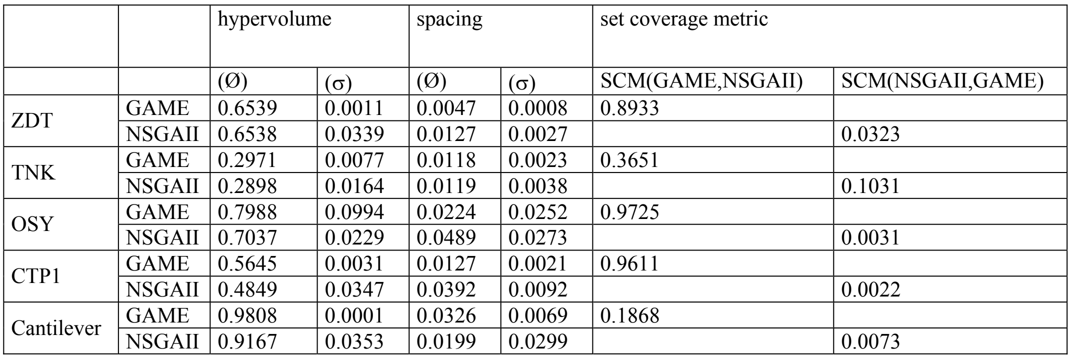


Figure 5.50: Comparison of the solution sets of GAME and NSGAII in terms of the set coverage metric for the test problems ZDT, TNK, OSY, CTP1, and the cantilever beam. The values of $SCM(\bar{\mathbf{P}}_{GAME,i}, \bar{\mathbf{P}}_{NSGAII,j}), SCM(\bar{\mathbf{P}}_{NSGAII,j}, \bar{\mathbf{P}}_{GAME,i}), i, j\epsilon[1, 50]$ are plotted versus the cumulative combination number



		hypervolume		spacing		set coverage metric	
		(Ø)	(σ)	(Ø)	(σ)	SCM(GAME,NSGAII)	SCM(NSGAII,GAME)
ZDT	GAME	0.6539	0.0011	0.0047	0.0008	0.8933	
	NSGAII	0.6538	0.0339	0.0127	0.0027		0.0323
TNK	GAME	0.2971	0.0077	0.0118	0.0023	0.3651	
	NSGAII	0.2898	0.0164	0.0119	0.0038		0.1031
OSY	GAME	0.7988	0.0994	0.0224	0.0252	0.9725	
	NSGAII	0.7037	0.0229	0.0489	0.0273		0.0031
CTP1	GAME	0.5645	0.0031	0.0127	0.0021	0.9611	
	NSGAII	0.4849	0.0347	0.0392	0.0092		0.0022
Cantilever	GAME	0.9808	0.0001	0.0326	0.0069	0.1868	
	NSGAII	0.9167	0.0353	0.0199	0.0299		0.0073

Table 5.10: Results for the multiobjective test problems for GAME and NSGAII

RSA implementation as can be seen in the previous section (5.6). For the test problem CTP1 GAME reaches its final optimum significantly earlier than NSGAII. Although showing a large standard deviation in average the best solution is found more than 30 generation earlier by GAME than by NSGAII. So for this example a reduction in computational cost of more than 80% is possible. For the other example problems the difference is less impressive, but a tendency can be observed that GAME finds its optimum approximately at least one generation earlier and thus achieves saving in computational cost of up to 5%.

The results for the multiobjective configurations of the test problems agree with the ones of the single objective ones. GAME performs better than NSGAII for all test problems. The only test example where NSGAII can almost match the performance of GAME is the unconstrained multiobjective problem ZDT. Here the mean value of the hypervolume of the solutions sets generated by GAME and NSGAII are nearly identical. But although the advantages are small, the solutions generated by GAME in average dominate more than 90% of the NSGAII solutions while only less than 4% do vice versa. In figure (5.51) this is illustrated by the plot of the final nondominated sets of sample runs of GAME and NSGAII. With respect to the results of the spacing measure GAME also achieves a better distribution of the solutions across the Pareto front in average. This finding is also reflected by the sample fronts.

The most impressive advantages can again be observed for the test problems OSY and CTP1. Here improvements in hypervolume of 13% and 16% respectively have been achieved, while simultaneously the quality of the spacing has been improved by 54% and 67% respectively. Consequently the result of the comparison in terms of the set coverage metric is unambiguous: in average more than 90% of the NSGAII solutions are dominated by GAME solutions, while only far less than 1% do vice versa. For these two problems GAME clearly outperforms NSGAII. To get an impression of how typical nondominated frontiers achieved by GAME and NSGAII look like for these two problems, in figure (5.51) the solutions sets for two sample runs are plotted. The superiority of the GAME solutions is clearly visible. Also a much better spread of the solutions is achieved by GAME. The lower number of NSGAII solutions are due to the fact that in final population of the NSGAII run still approximately 75% of the solutions are infeasible, while the final population of GAME is completely feasible. The constraint handling of GAME proves to be significantly more efficient for these problems.

For the other problems the advantages are smaller but still significant. For the test problem TNK GAME seems to be better able to manage the complex, disrupted shape of the true Pareto frontier and draws its main adavantages from being able to cover a larger part of the front in average. The same applies for the test problem of the cantilever beam for which GAME is able to cover a significantly larger spectrum of Pareto-optimal solutions than NSGAII. NSGAII continuously produces solutions only on a relatively small section of the complete Pareto front, although the most interesting part of the front. The significant differences in the size of the covered sections of the Pareto-front also explain the erratic results for the comparison of the respective solutions sets. To illustrate these differences in figure (5.51) representative solution sets for GAME and NSGAII are plotted for two sample runs. Also for these test problems GAME shows to be able to maintain a better spread of the solutions.

In conclusion GAME has proven to perform better than NSGAII for all considered test problems experiments. Only for the unconstrained example ZDT GAME and NSGAII show nearly an equal performance. This hints that not only the integration of RSA functionalities but also the constrained handling in GAME is responsible for the performance advantages. For these experiments GAME and NSGAII have been allowed exactly the same number of function evaluations. Thus the main question of this comparison has been which algorithm achieves the better results with a given budget of computational costs. According to the findings in chapter (5.3) the quality of the solutions for GAME without the RSA integration is directly dependent on the invested computational cost. So it can be deduced that for a given quality level of solutions GAME can be run with lower population sizes than NSGAII and thus achieves equivalently savings in computational costs.

So the results of these experiments indicate that at least within the spectrum of the considered test problems the development goals of GAME to increase the performance and efficiency of conventional EAs have successfully been achieved. Of course it is difficult to generalize the results of this comparison, but these experiments establish a good basis for the assumption that GAME is at least able to keep up with current state-of-the-art multiobjective evolutionary algorithms.

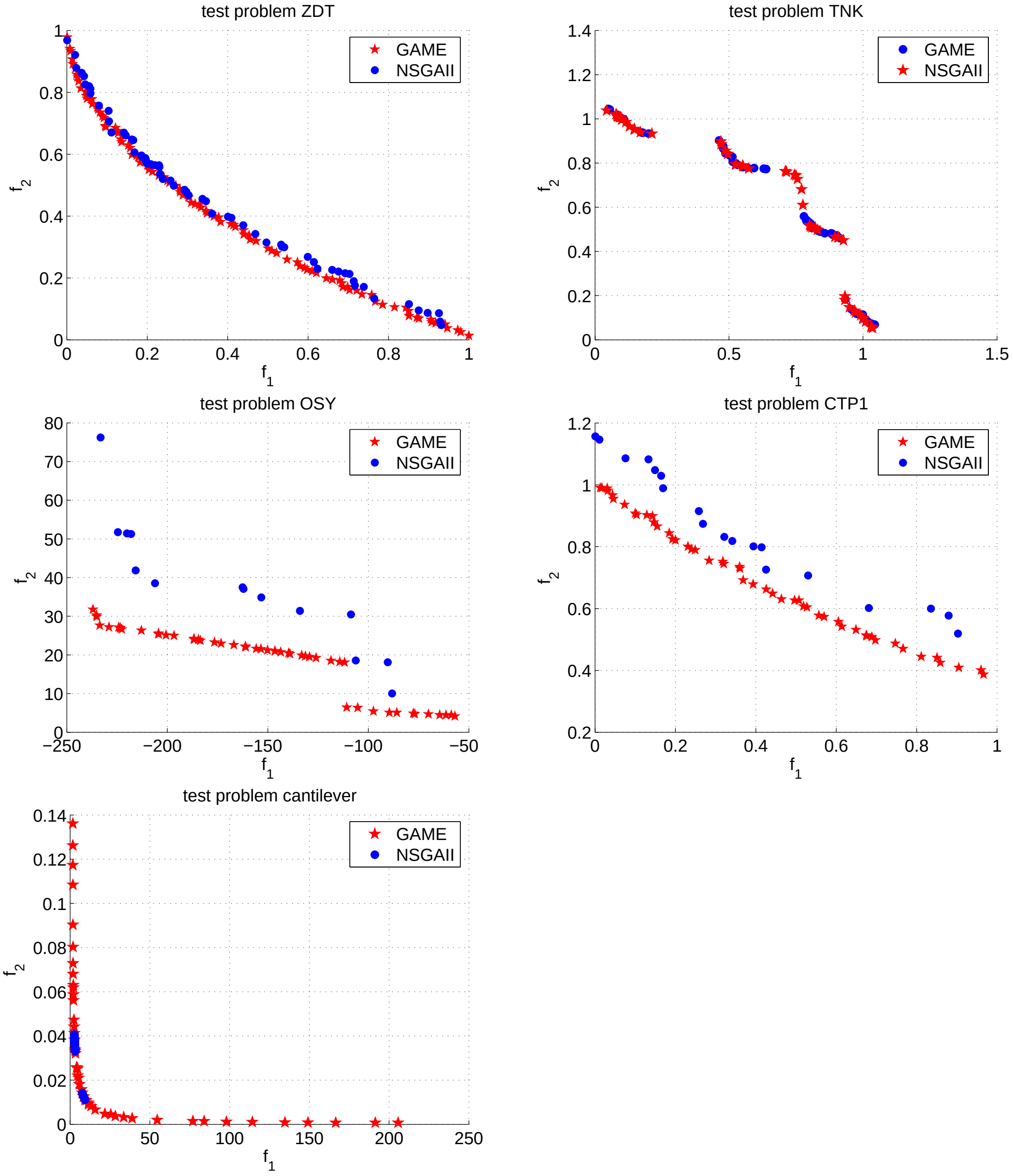


Figure 5.51: Solution sets of sample GAME and NSGAII runs for all test problems

6 Applications

During the work on this thesis GAME has been applied to several design problems. Some are presented in this chapter. First GAME is applied to the optimal design of actively damped structures. For a beam and a plate the optimal positioning and sizing of piezo actuators and simultaneously different control parameters are to be determined . The second problem is the optimal design of a stringer stiffened plate. Here the type, number, and sizing of the stringer are to be determined. The final problem is the design of a high precision CFRP beam which is subject to structural as well as thermal loads. Preliminary versions of GAME have also been successfully applied in other projects like the optimization of an electric motor in collaboration with the German Aerospace Center (DLR) ([Mül01]), the design of a space based reflect-array antenna ([Baj04]), or the configuration optimization of satellite structures ([PLB04]).

6.1 Optimization of a Beam and a Plate Structure with Active Damping by Piezo Actuators

One strategy to satisfy the steadily rising demands on lightweight structures with respect to improved static and dynamic behavior is the development of adaptive or smart structures. Adaptive structures can be defined as structures with integrated actuators and sensors as well as a controller. With suitable control laws the static and dynamic behavior of the structure can be modified and improved.

6.1.1 Problem Statement

One field of adaptive structures is the active damping of lightweight structures, in which it is tried to increase the damping and lower the vibration amplitudes. The two problems discussed here have been part of a research project in this field ([LLMB00]). The subjects of this research are a simply supported beam and a clamped plate with collocated pairs of piezoceramic patches as sensor and actuators. In figure (6.1) the two structures are shown. The dimensions of the sample structures are given in table (6.1). The sensor patches act as strain rate sensors while the actuator patches work as bending actuators. The control law is the direct velocity feedback ([Bal79]). In figure (6.2) this control strategy is illustrated by a flowchart.

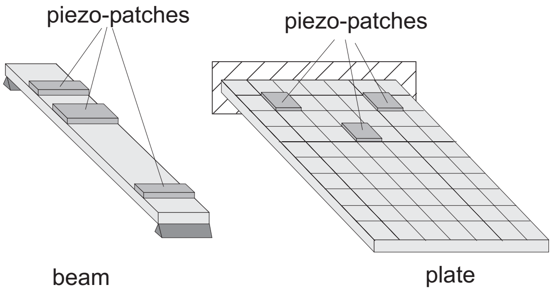


Figure 6.1: Actively damped beam and plate structure with piezoceramic patches

	beam	plate
length (l)[m]	0.3	0.42
width (w)[m]	0.033	0.48
thickness (t) [m]	0.001	0.001
thickness piezo (t_p)[m]	0.0003	0.0003

Table 6.1: Specifications of the sample structures

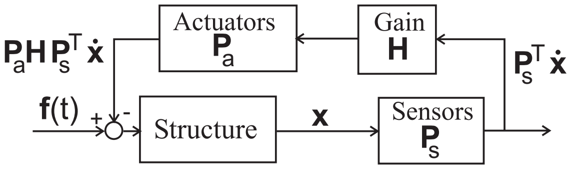

Actuators
$\mathbf{P}_a$
Gain
$\mathbf{H}$
$\mathbf{P}_a\mathbf{H}\mathbf{P}_s^T\dot{\mathbf{x}}$
$\mathbf{P}_s^T\dot{\mathbf{x}}$
f(t)
+
-
Structure
x
Sensors
$\mathbf{P}_s$

Figure 6.2: Block diagram for an actively damped structure with direct velocity feedback

$\mathbf{P}_a$ and $\mathbf{P}_s$ are the influence matrices of the actuator and sensor patches, $\mathbf{H}$ the gain matrix of the controller, $\mathbf{F}$ are external forces, and $\mathbf{x}$ the state vector. For direct velocity feedback the voltage applied to the actuator patches and thus the applied bending moment is directly proportional to the strain rate of the sensor patches. This control law resembles basically the working principle of velocity dependent viscous damping. The governing equations are given in equation (6.1):

$$\begin{aligned} &\text{Structure} &:\quad &\mathbf{M}\ddot{\mathbf{x}} + \mathbf{D}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} &&= \mathbf{f}(t) + \mathbf{P}_a\mathbf{u} \\ &\text{Sensor} &:\quad &\mathbf{y} &&= \mathbf{P}_s^T\dot{\mathbf{x}} \\ &\text{Controller} &:\quad &\mathbf{u} &&= -\mathbf{H}\mathbf{y} \end{aligned} \tag{6.1}$$

The equations for the closed loop system are:

$$\begin{aligned} \mathbf{M}\ddot{\mathbf{x}} + (\mathbf{D} + \mathbf{P}_a\mathbf{H}\mathbf{P}_s^T)\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} &= \mathbf{f}(t) + \mathbf{P}_a\mathbf{u} \\ \mathbf{y} &= \mathbf{P}_s^T\dot{\mathbf{x}} \end{aligned} \tag{6.2}$$

Each pair of sensor and actuator patch form a independent control loop and thus the employed control concept belongs to the group of decentralized low authority controller. The collocation of sensor and actor provides a theoretically guaranteed stability of the control loop.

The design task for the beam and the plate is to determine the number, positioning, size, and feedback gains for the actuators so that the damping of the first two eigenmodes is maximized while keeping the total size of all applied patches as small as possible. For the plate structure the problem has also been solved with a gradient based optimization algorithm for comparing the performance. For this purpose the fmincon algorithm, an SQP implementation provided by MATLAB, has been employed.

6.1.2 Modelling

For the beam and the plate FEM models are used with beam elements and shell elements respectively. The beam is meshed with 60 elements which provides a spatial resolution of $0.005m$ for the actuator placement and sizing. The plate is meshed with 42 elements in x-direction and 48 in y-direction resulting in a spatial resolution of $0.001m$. The chosen actuator patch size of 4x4 nodes thus correlates to an area of $0.000016m^2$. The FEM model of the plate is shown in figure (6.3), in which the mode shapes of the first two Eigenmodes are displayed. The simulation of the complete control loop has been performed in Matlab. For simulating the characteristics of the closed loop system the system equations (6.2) have been transformed into a state space representation with modal coordinates:

$$\begin{aligned} \dot{\mathbf{z}} &= \mathbf{A}\mathbf{z} + \mathbf{B}\mathbf{u} \\ \mathbf{y} &= \mathbf{C}\mathbf{z} + \mathbf{D}\mathbf{u}. \end{aligned} \tag{6.3}$$

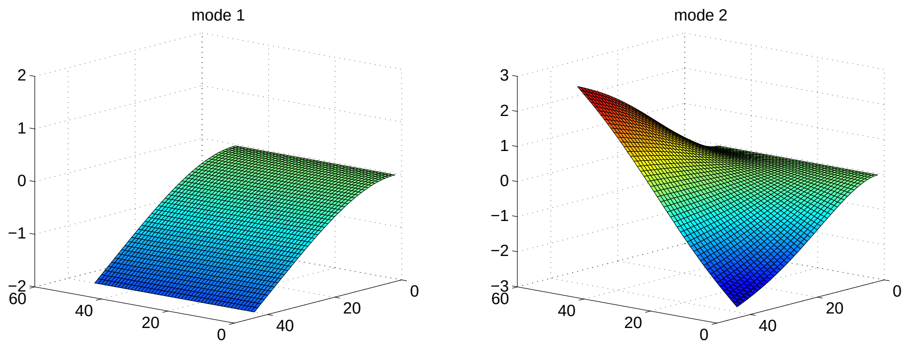


Figure 6.3: Mode shapes of the first two Eigenmodes of the plate FEM model

The state vector $\mathbf{z} = [\mathbf{q}^T, \dot{\mathbf{q}}^T]^T$ consists of the modal displacements and velocities. For the closed loop system the system-matrix $\mathbf{A}$ results to:

$$\mathbf{A} = \left[\begin{array}{cc} \mathbf{0} & \mathbf{I} \\ -\mathrm{diag}\{\omega_{s,i}^2\} & -\mathbf{\Phi}^T \mathbf{P_a} \mathbf{H} \mathbf{P}_s^T \mathbf{\Phi} - \mathrm{diag}\{2\zeta_{s,i}\omega_{s,i}\} \end{array} \right] \tag{6.4}$$

Here $\omega_{s,i}$ and $\zeta_{s,i}$ are the eigenfrequencies and the modal damping of the structure. The modal damping ζ_i of the closed loop system can be computed from the roots of the system matrix $\mathbf{A}$. For the simulation model in both cases the first 25 eigenmodes have been considered. For the modelling of the piezo patch actuators and sensors simplified concentrated force and moment models have been applied. In figure (6.4) this is illustrated for the beam. Details for the derivation of these models can be obtained from ([Loc01]).

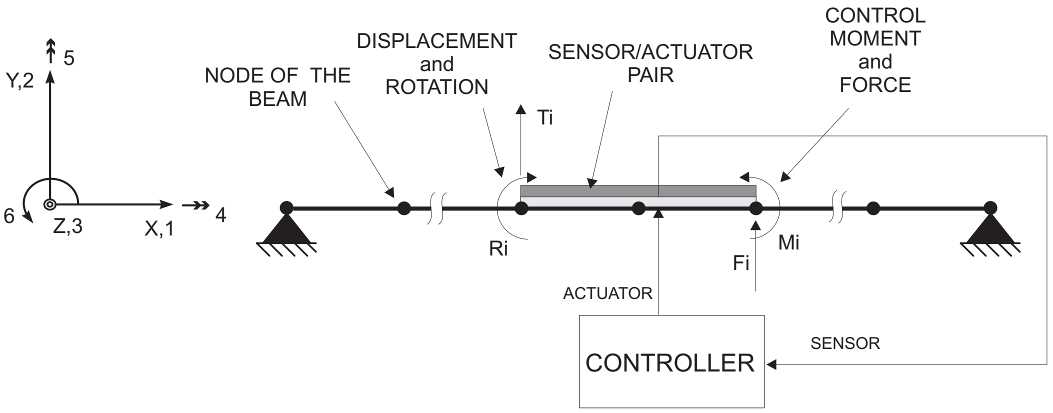


Figure 6.4: Actuator models for the piezo patches

6.1.3 Optimization Task

The objective functions are the damping of the first two eigenmodes (ζ_1, ζ_2) and the total area of all applied piezo patches (l_{total}, A_{total}). The constraints originate mainly from geometrical issues: the patches have to stay on the structure and they must not overlap. Additionally the total area of all actuator is restricted to be below a certain allowed limit.

For the beam the design variables are the number of actuators (n_{act}), their position (x_i) and size ($l_{x,i}$) as well as the feedback gains (h_i). For the plate the task is little different, here a fixed size actuator patch is given as the basis and the number of patches (n_{act}), their position ($l_{x,i}, l_{y,i}$), and the respective feedback gains (h_i) are to be determined.

The most common models in structural mechanics are FEM models. In order to develop an optimization strategy that features the generality to be applied to a broad spectrum of tasks, the beam and the plate have been modelled as FEM models. Because the positioning of the piezo patches is part of the optimization task, there is the need to modify the FEM model according to the chosen positions of the piezo patches. In principle there are two possibilities. One is to allow continuous changes for position and size of the piezo patches and remesh the FEM model each time. The other one is to allow only discrete changes of the piezo patches according to the given FEM mesh. In this case the mesh can be maintained and the piezo can be attached to the already existing nodes. Within this project ([LLMB00], [Loc01]) it has been decided to use the second approach. The direct consequence is that the design variables for positioning and sizing the actuators become discrete variables. To allow reasonable smooth changes the mesh is required to be sufficiently fine. Since the piezos can be positioned only on the existing nodes, the positioning coordinates and sizes are counted in the number of nodes from the origin or the number of nodes the patch is spanned over respectively. The number of actuators is also a discrete variable while the feedback gains are continuous.

An additional characteristic of this problem is that the number of positioning and sizing variables depends on the number of actuators, so the optimization algorithm it has to deal with a varying number of design variables. For the coding of the variables in GAME this has been addressed in the following way: Each chromosome or design vector contains the maximum possible number of variables. Additionally each set of variables describing a single actuator is augmented by a so called activating gene. This gene can be either 1 or 0 signaling whether the respective actuator is actually present or not. These activating genes are controlled by a so called steering gene, in this case the number of actuators n_{act}. The first n_{act} activating genes are set to 1, the rest to 0. In figure (6.5) this approach is illustrated. The optimization problem

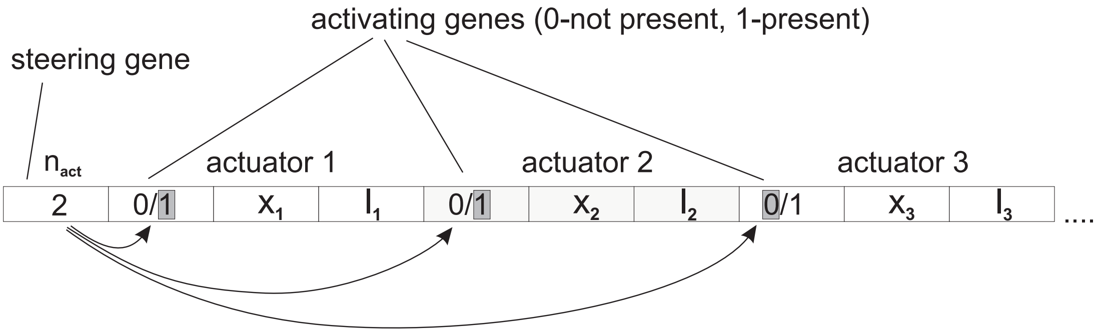


Figure 6.5: Approach for handling a varying number of design variables

can be formulated for the beam as follows:

$$
\begin{aligned}
& min\ \mathbf{z} = [-\zeta_1, -\zeta_2, l_{total}],\ l_{total} = \sum_{i=1}^{n_{act}} l_i \\
& \text{with: } \mathbf{x} = [n_{act}, x_i, l_i],\ i\epsilon[1, n_{act}] \\
& \text{subject to:} \\
& g_1(\mathbf{x}) \leq 0, g_1((\mathbf{x}) = \sum_{i=1}^{n_{act}} \sum_{j=i+1}^{n_{act}} A_{overlap}(i,j), \\
& g_2(\mathbf{x}) \leq 0,\ \leq 0, g_1((\mathbf{x}) = \sum_{i=1}^{n_{act}} A_{trespassing}(i), \\
& g_3(\mathbf{x}) = \sum_{i=1}^{n_{act}} l_i \leq l_{max}, \\
& and \\
& 1 \leq n_{act} \leq n_{act,max}, \\
& 0 \leq x_i \leq n_{nodes,x}, \\
& 0 \leq l_i \leq l_{max}, \\
& 0 \leq h_i \leq h_{max}
\end{aligned}
\tag{6.5}
$$

In equation (6.5) $A_{overlap}(i,j)$ is the overlapping are between the piezo patches i and j, $A_{trespassing}(i)$ the area of patch i trespassing the borders of the structure. The maximum allowed total actuator size is l_{max}, this is also the limit for the size of a single actuator. The x-position is limited by the maximum node number $n_{nodes,x}$ in x-direction which corresponds to the length of the beam. The number of actuators is limited to be between 1 and $n_{act,max}$. Since in this problem only the first two modes are to be damped, this number has been set to $n_{act,max} = 2$. Finally the feedback gain is also limited by an upper limit h_{max}.

So in conclusion this is a constrained multiobjective problem with three objective functions and three constraint functions. The problem features also a mixed contin-

uous/discrete design space with a varying dimension.
In equation (6.6) the optimization problem for the plate is formulated accordingly:

$$\begin{aligned}
&min\ \mathbf{z} = [-\zeta_1, -\zeta_2, A_{total}],\ A_{total} = \sum_{i=1}^{n_{act}} A_{act},\ A_{act} = const.\\
&\text{with: } \mathbf{x} = [n_{act}, x_i, y_i],\ i\epsilon[1, n_{act}]\\
&\text{subject to:}\\
&g_1(\mathbf{x}) \leq 0, g_1((\mathbf{x}) = \sum_{i=1}^{n_{act}} \sum_{j=i+1}^{n_{act}} A_{overlap}(i,j),\\
&g_2(\mathbf{x}) \leq 0,\ \leq 0, g_1((\mathbf{x}) = \sum_{i=1}^{n_{act}} A_{trespassing}(i),\\
&and\\
&1 \leq n_{act} \leq n_{act,max},\\
&0 \leq x_i \leq n_{nodes,x},\\
&0 \leq y_i \leq n_{nodes,y},\\
&0 \leq h_i \leq h_m ax
\end{aligned} \tag{6.6}$$

The major difference to the beam example is on the one hand that the positioning is a two dimensional task now and on the other hand that the actuator size is no design variable anymore but set to a fixed value ($A_{act} = 16$). The maximum number of actuators has been set to $n_{act,max} = 12$. Additionally a symmetry constraint has been imposed, so that actually the number of actuators could be chosen from the set $n_{act}\epsilon[1, 6]$.

6.1.4 Optimization Run and Results

Configuration Setup of GAME

The configuration of GAME for the two optimizations tasks is shown in table (6.2). For the plate optimization a larger population has been chosen since the two dimensional positioning task features as larger design space.

Configuration for the Optimization of the Plate with a Gradient Based Algorithm

Due to the presence of discrete design variables a gradient based algorithm cannot be applied without special adaptations. As the number of actuators is a truly discrete variable it is impossible to include it in an optimization with a gradient based algorithm. The only possibility is to run the algorithm repeatedly with different settings

	configuration beam	configuration plate
population size *n*	200	300
ratio r_{cp}	3	3
number of children n_c	600	900
number of generations n_{gen}	20	20
probability for recombination p_r	0.2	0.2
probability for mutation p_m	1	1
initial standard deviation for mutation σ_0	0.3 (ub-lb)	0.3 (ub-lb)

Table 6.2: Configuration of GAME for the optimization runs

for the number of actuators. This has been done for 2, 4, 6, and 8 actuators. By eliminating the number of actuators from the design variable vector consequently the total actuator area is eliminated as objective function, reducing the task to a two objective problem. The positioning of the actuators is basically a continuous variable, that is discrete only because of the FEM mesh. But in order to make it continuous, the damping for actuator positions that do not coincide with the nodes have to computed. For this purpose an interpolation method has been implemented, that first computes the damping for the next possible actuator positions that do coincide with the mesh and than derives the damping value for the desired position by linear interpolation. In figure (6.6) this process is illustrated. This method implies that additional objective function evaluations have to be performed. Depending on the number of actuators this can be a substantial increase of the computational costs. Since a gradient based algorithm works only with a single objective function the two objectives have been aggregated using the weighted sum approach: $f(\mathbf{x}) = w_1\zeta_1 + w_2\zeta_2$. In order to compare the performance with the results of GAME three weight settings have been chosen: $w1 = [w_1 = 1, w_2 = 0]$, $w2 = [w_1 = 0, w_2 = 1]$, and $w3 = [w_1 = 0.5, w_2 = 0.5]$. The first setting $w1$ aims solely at the maximum damping of mode 1, the second $w2$ at the maximum damping of mode 2, and the third $w3$ constitutes a compromise setting putting equal priority on both modes. As the performance of a gradient based algorithm may strongly depend on the initial vector, each setting has been run with 5 different start vectors.

Results for the Beam

In figure (6.7) the solutions for the beam optimization are displayed in a 3D trade-off plot. The x- and y-axis correspond to the achieved damping ratios ζ_1 and ζ_2 while the z-axis corresponds to the total actuator area l_{total}. In addition the three axis-parallel projections on the x-y-plane, the x-z-plane, and the y-z-plane are shown. The displayed

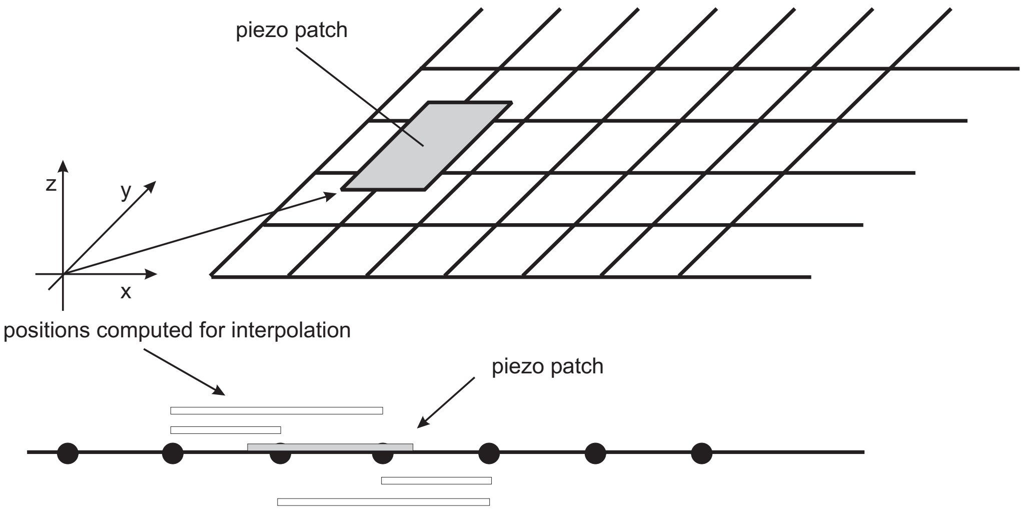


Figure 6.6: Interpolation method for computing the damping for actuator positions not coinciding with FEM mesh

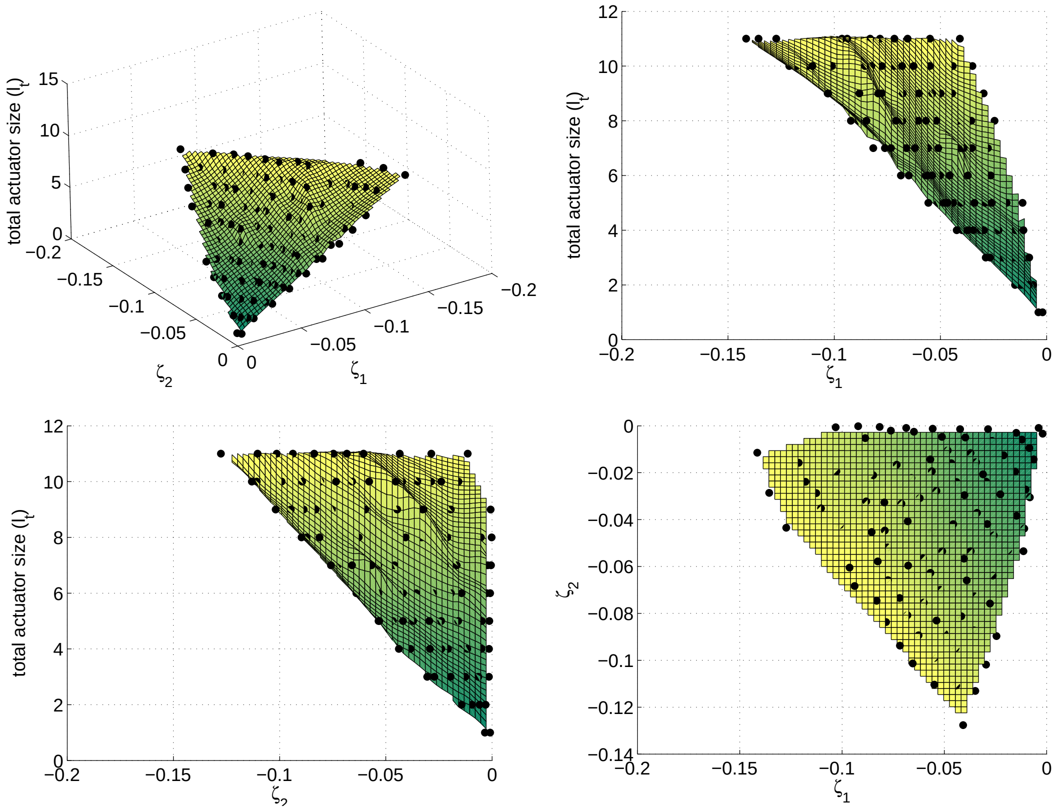


Figure 6.7: Trade-off plot of the solutions for the actively damped beam

meshed surface is only an optical help to visualize the non-dominated surface. It can be seen that a large Pareto-optimal surface has been achieved. Furthermore an almost uniform spread of the solutions across can be observed. This shows that the clustering approach used for maintaining population diversity has worked very well for this problem and the resulting nondominated surface offers a good basis for trade-off analyses.

Concerning the interpretation of the results, the results coincide with the intuitive expectation that larger actuator sizes achieve higher damping ratios. Independently from the weighting between the modes 1 and 2 the achievable damping is directly proportional to the applied actuator size. In figure (6.8) sample solutions are plotted to illustrate the actual sizing and positioning of the actuators. In the left column solutions with a large large total actuator size ($A_{total} = 11$) are plotted, in the right column for a small total actuator size ($A_{total} = 4$). The two solutions in the first row are solutions for a maximum damping of mode 1, in the middle row for an equal damping of mode 1 and 2, and in the bottom row solutions for a maximum damping of mode 2. Additionally the mode shapes of the first three eigenmodes are shown.

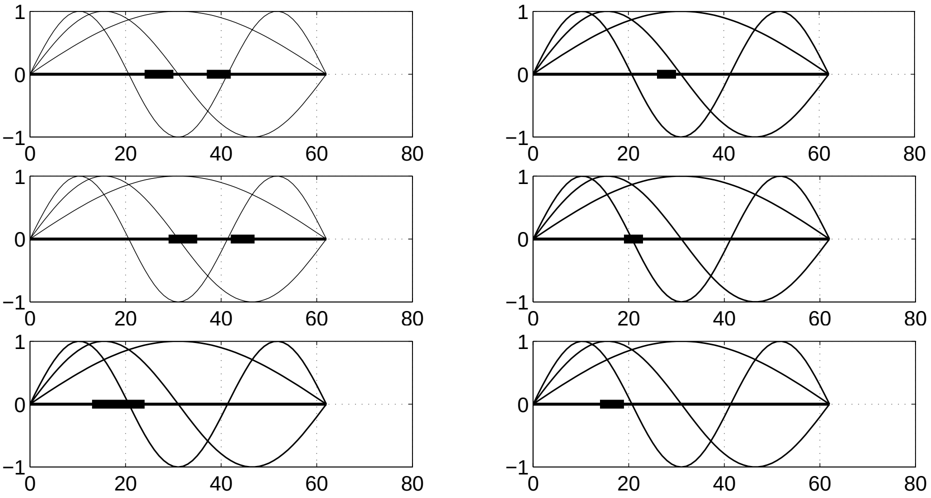


Figure 6.8: Sample nondominated solutions: solutions for a large total actuator size ($A_{total} = 11$, left column) and a small total actuator size ($A_{total} = 4$, right column): maximum damping of mode 1 (top), equal priorities for the damping of mode 1 and 2 (middle), maximum damping of mode 2 (bottom)

In general the positioning of the actuators is plausible with respect of the target mode or modes and the mode shapes. So e.g. to achieve the maximum damping for mode 2 the actuator is placed at the maximum curvature of mode 2.

In the following first the results for a large total actuator size (left column, $A_{total} = 11$) are discussed. It is interesting to see that for the maximum damping of mode 1 a solution with two actuators, that are both some distance away from the maximum curvature, is preferred to a solution with a single actuator of the same total size exactly in the middle. This could be observed for several runs. Obviously a central actuator does not offer a significant advantage over two slightly separated actuators with respect to mode 1. But a solution with two actuators additionally offers some damping for mode 2, so it is preferred. If a maximum damping of mode 2 is desired the opposite effect can be observed. Here a single actuator placed at the maximum curvature is the preferred solution. In this case the separation makes no sense because here a single actuator at this position still provides a considerable damping of mode 1. No surprise is the positioning for the compromise between the damping of mode 1 and 2. Here a two actuator solution is preferred where both actuators are placed at the respective maximum curvature of each mode.

Looking now at the solutions for a total actuator size of $A_{total} = 4$ (right column), for all cases a single actuator solution is preferred. The preferred positions are the respective positions of maximum curvature if the maximum damping for mode 1 or 2 is desired. If a compromise solution between the damping of mode 1 and 2 is desired the actuator is exactly placed at the theoretically best compromise position. So in conclusion the positioning of the actuators by GAME coincide well with common sense. Besides these 'extreme' solutions, it has to be noted that the nearly uniform spacing of the solutions across the nondominated frontier provides an excellent basis to find solutions for any priority setting.

For the 100 final nondominated solutions a total number of 11600 objective function evaluations had to be invested. This means an computational effort of 314 objective function calls per solution point.

Results for the Plate

In figure (6.9) the solutions for the plate optimization are displayed in a 3D trade-off plot in a similar way as for the beam. In addition to the solutions determined by GAME also the solutions achieved by the fmincon runs are shown. Again a large nondominated surface could be achieved, but in comparison to the beam results the spreading of the solutions is worse. One obvious reason for this is that third objective function, the total actuator area, is only dependent on the number of actuators, since the actuator size is chosen fix. So in consequence there are only six discrete possible values for this objective. But one can also notice that there are more and better distributed solutions for a lower number of actuators. This can be explained by the increased difficulties to find feasible solutions for higher number of actuators without any overlapping.

Nevertheless, the results still form a good basis for a trade-off analysis. Expectedly a larger total actuator size, i.e. a higher number of actuators, leads to higher damping ratios. But this time the gain of damping ration is not any more directly proportional to the increase in total actuator size. A convergence can be observed. This holds especially for the damping of mode 2, where a clear convergence can be seen at a

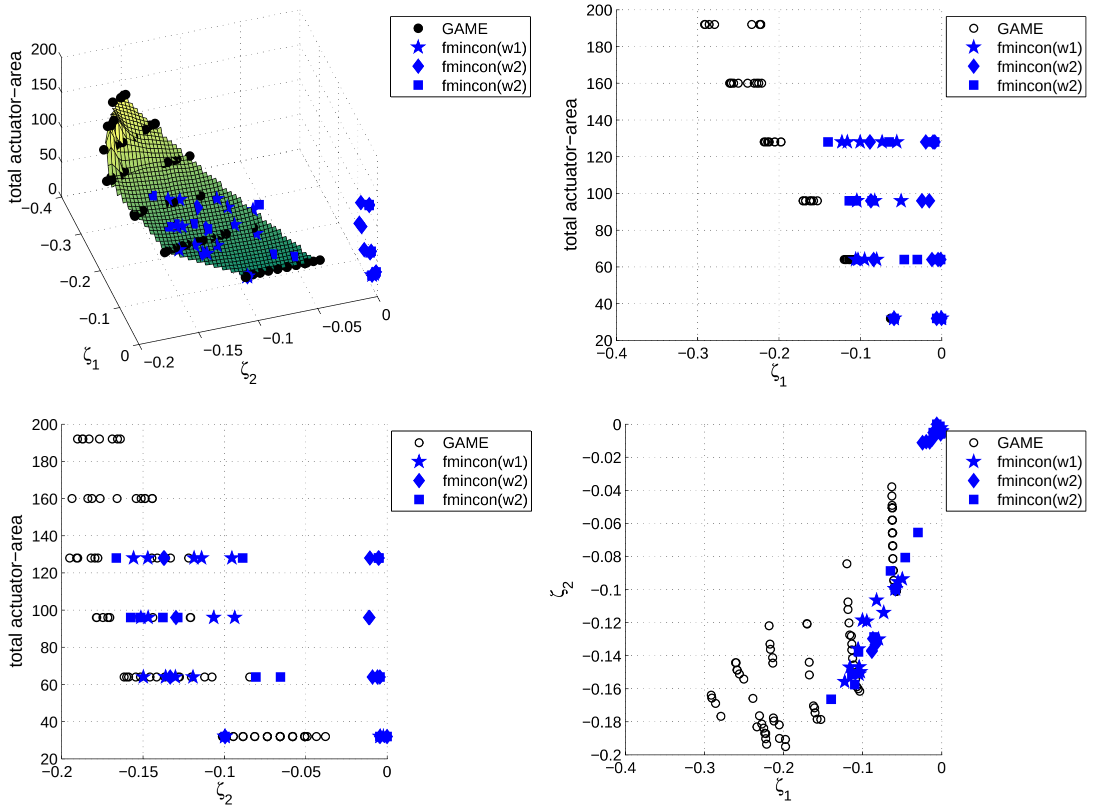


Figure 6.9: Trade-off plot of the solutions for the actively damped plate

number of $n_{act} = 8$ actuators. More actuators do not achieve a higher damping ratio. The reason for this effect can addressed to the mode shapes of mode 1 and 2. Mode 1 is the first bending mode of the single-sided clamped plate while mode 2 is the first torsion mode. For the torsion mode the best positions are at the outer borders of the plate close to the support. But there is only limited space. If this space is filled up, additional actuators can only be placed in less advantageous positions, which obviously are not effective enough for increasing the damping ratio further. These findings are illustrated in figure (6.10). In this figure sample solutions are plotted for 2, 4, 6, 8, and 10 actuators for a maximum damping of mode 2.

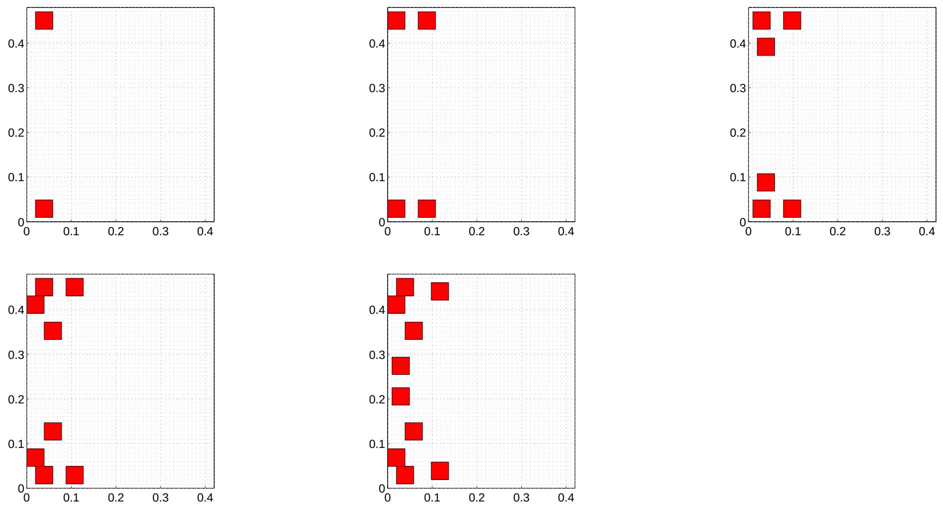


Figure 6.10: Sample solutions for placing 2, 4, 6, 8, and 10 actuators for maximum damping of mode 2

Another effect of the mode shapes is that the optimal actuator positions for mode 1 and 2 are partly identical. For mode 1 the most promising area is the complete border along the support. So positions near the outer border and close to the support are good for both modes. This can also be seen in the ζ_1-ζ_2-trade-off plot in figure (6.9), where the nondominated frontier is comparably small indicating the lack of larger conflicts between the two objectives. One consequence of this is that for a low number of actuators the optimal compromise solution for equal damping of mode 1 and 2 is identical to the optimal solution for a maximum damping of mode 2. This can be seen in figure (6.11), in which two compromise solutions for 2 and 12 actuators are shown. In the compromise solutions with 12 actuators one can notice a clear split of tasks: some actuators are mainly responsible for the damping of mode 1, while the other ones are aiming mainly at mode 2.

When looking only at the maximum damping of mode 1 a similar convergence with increasing actuator size can be observed as for mode 2. This effect is not as severe as for mode 2, since there is more space available and the effectivity of positions

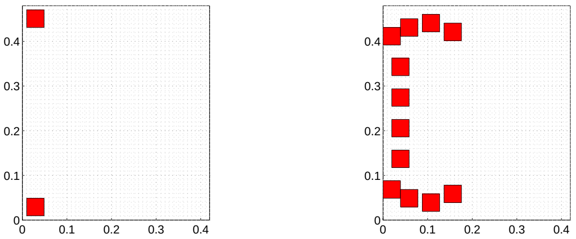


Figure 6.11: Sample solutions for placing 2 and 12 actuators if equal priority is put on mode 1 and 2

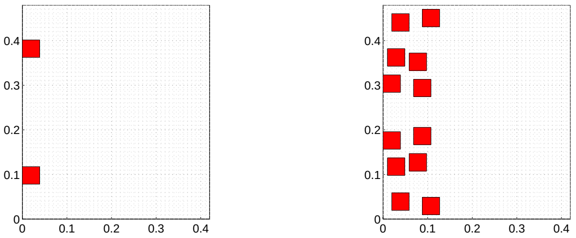


Figure 6.12: Sample solutions for placing 2 and 12 actuators for maximum damping of mode 1

slightly away from the support border does not decrease so dramatically as for the torsion mode. In figure (6.12) two sample solutions with a low number of actuators ($n_{act} = 2$) and a high number ($n_{act} = 12$) are shown for maximum damping of mode 1. Again it shows that for only two actuators the optimal solutions for mode 1 and 2 are identical. For ($n_{act} = 12$) the actuators are expectedly simply lined up close to the support.

With respect to the solutions achieved by fmincon it can be observed that they generally perform worse than the solutions of GAME. Only for the lowest total actuator size, i.e. two actuators, equally good solutions are determined. Towards higher total actuator sizes an increasing performance gap can be observed. The reason for this can directly be addressed to the search logic of the gradient based algorithm. Fmincon will wether move the actuators in the direction of the largest objective function improvement or, in case a constraint is violated, along the constraint border in the direction in which the objective improves. If the objective does not improve in any direction along the respective constraint border, the algorithm assumes that an at least local optimum has been found. But it is exactly this logic that is doomed to failure for the positioning task. In figure (6.13) this is illustrated for two sample solutions of GAME and fmincon for a maximum damping of mode 1 with 6 actuators. The GAME solutions achieve an damping ratio of $\zeta_1 = 0.17$, whereas the fmincon solution only reaches $\zeta_1 = 0.11$. The fmincon dilemma is perfectly illustrated here. A better result could be reached by moving the outer two actuators closer to the left border, so the gradient based search direction points towards the border. But in this direction the two respective actuators hit on the ones already there. And moving parallel to

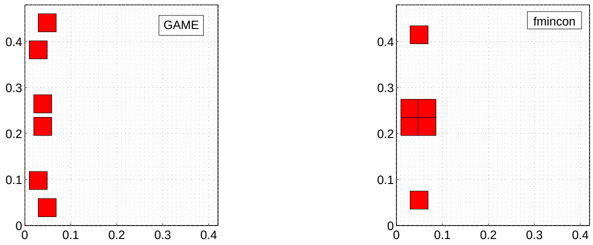


Figure 6.13: Sample solution for placing 6 actuators for maximum damping of mode 1 by GAME (left) and fmincon (right)

this constraint border, i.e. parallel to the plate's support border, also achieves no improvement. Thus the search terminates. For the search logic of fmincon it is absolutely impossible to move the actuators around the other ones. Here the stochastic mutation operator of GAME has decisive advantages. Furthermore for unknown reason fmincon is not able to move the other outer two actuators closer to the border. This may be due to the approximation technique used for making the discrete positioning variable continuous and its effect on the gradient computation, but it could not definitely be explained. However, also the GAME solutions show deficits typical for EAs. Due to its stochastic search strategy and because it is simply stopped after a certain number of generation the solutions are still a little away from the true optimum, which would be having all 6 actuators along the border. Because of this inherent characteristic of EAs, it is generally only claimed that EAs reach *near-optimal* solutions. Nevertheless, GAME achieves significantly better solutions than fmincon. It is also interesting to see that the setting of the weights according to some priority a priori does not necessarily lead to the best solution with respect to the chosen priority. So e.g. the best solutions for the damping of mode 1 and 2 are not achieved by the weight settings laying all priority on the respective mode but by the compromise weight setting $w3$. This can be explained with this setting allowing better search direction with fewer insolvable constraint conflict like in the above example. This also demonstrates again the difficulty of setting priorities a priori.

In conclusion it can be said that GAME has been successful in solving this positioning problem. The achieved Pareto optimal solutions are sufficiently well distributed to provide a good basis for a final design decision. Furthermore the solutions determined by GAME perform better than the solutions achieved by fmincon. But they also reflect a general drawback of EA that the solutions are only 'near' optimal.

6.2 Optimization of a Stringer Stiffened Plate

In this application example a stringer stiffened plate under axial compression load is to be optimized with respect to buckling load and mass. In contrast to conventional sizing optimizations a complete configuration optimization of the stiffened structure is to be performed encompassing material selection and decisions on the type and number of stringer. With this example the potential of applying GAME to configuration optimization problems of lightweight structures is demonstrated. As this is mainly an academic example the scope of load cases has been limited to a single buckling load case. Preliminary work on this problem has also been performed in ([Kip03]).

6.2.1 Optimization Task

In figure (6.14) the problem is illustrated. A stringer stiffened plate is subject to a compression load. The plate (length $p_l = 0.8m$, width $p_w = 0.6m$) is simply supported at all sides. The objective functions of this task are the buckling load F_b and the mass of the plate p_m. While the first one is to be maximized the latter one is to be minimized. The design variables are the plate thickness p_t, the material of the plate and stringer p_{mat}, the number of stringer s_n, the type of the stringer s_{type}, the height and thickness of the web of the respective stringer s_{wh} and s_{wt}, and depending on the stringer type the widths and thickness of the flange s_{fw} and s_{ft}. With respect to the material four choices are at hand: aluminum, steel, titanium, and an quasi-isotropic CFRP. For this being an academic design problem it is sufficient to use only representative data for the respective materials which is given in table (6.3). Of course there do exist alloys for each of the chosen materials which have other and maybe better properties. The only constraint in this problems is that the static stresses in plate or stringers due to the buckling force must not exceed the maximum

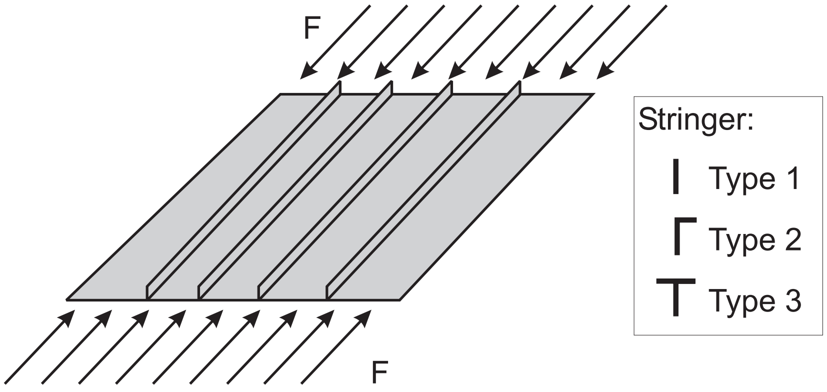


Figure 6.14: Stringer stiffened plate under compression load

property	aluminum	steel	titanium	CFRP
E [N/m^2]	70000E6	210000E6	110000E6	69000E6
ρ [kg/m^3]	2700	7800	4900	1600
σ_{allow} [N/m^2]	400E6	1200E6	1100E6	550E6

Table 6.3: Representative material properties

allowable stress of the chosen material. Concerning the type of the stringer, they can be I-, L- or T-shaped. The optimization problem can be stated as follows:

$$\begin{aligned}
&min\ \mathbf{z}(\mathbf{x}) = [p_m(\mathbf{x}), -F_b(\mathbf{x})],\\
&\text{with: } \mathbf{x} = [p_t, p_{mat}, s_n, s_{type}, s_{wh}, s_{wt}, s_{fw}, s_{ft}],\\
&\text{subject to:}\\
&g_1(\mathbf{x}) \leq 0, g_1(\mathbf{x}) = \sigma_{max,Mises}(\mathbf{x}) - \sigma_{allow}(p_{mat}),\\
&and\\
&0.0005[m] \leq p_t \leq 0.01[m],\\
&1 \leq p_{mat} \leq 4,\\
&0 \leq s_n \leq 10,\\
&1 \leq s_{type} \leq 3\\
&0.001[m] \leq s_{wh} \leq 0.05[m]\\
&0.0005[m] \leq s_{wt} \leq 0.01[m]\\
&0.001[m] \leq s_{fw} \leq 0.02[m]\\
&0.0005[m] \leq s_{ft} \leq 0.01[m]
\end{aligned} \qquad (6.7)$$

The design variables for the material p_{mat} and the stringer type s_{type} are truly discrete, the number of the stringer s_n is a quasi-continuous discrete variable, and finally the sizing parameters for the plate and stringers are continuous design variables. Furthermore the number of design variables varies in dependence on the chosen type of stringer. So in conclusion the problem can be characterized as mixed discrete/continuous constrained multiobjective problem.

6.2.2 Modelling

For the determination of the buckling load an FEM model has been developed, which is shown in figure (6.15). The buckling load is computed with the Eigenvalue buckling analysis. Since this optimization problem is primarily an academic demonstration, this

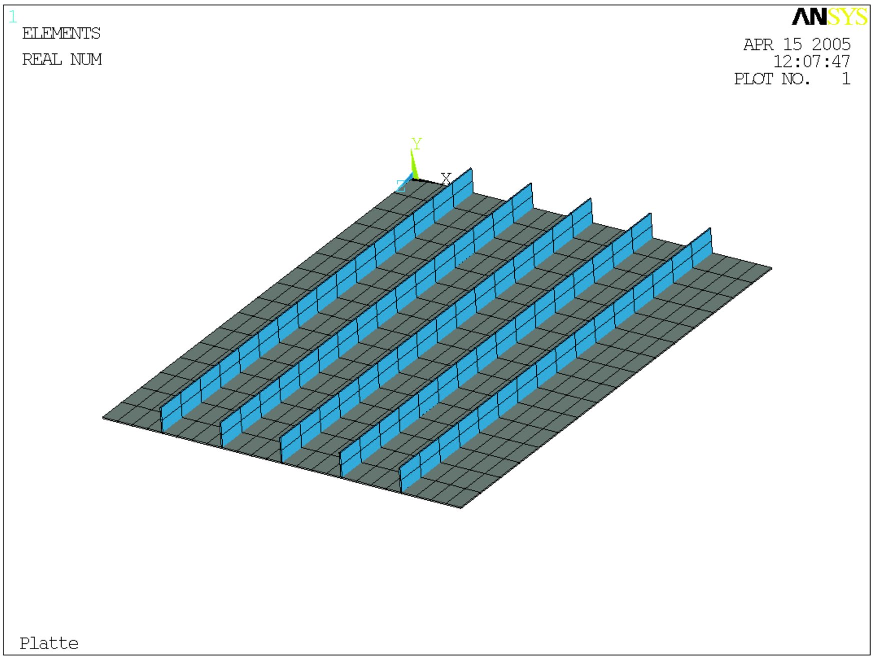


Figure 6.15: FEM model of the stringer stiffened plate

method is sufficient. A more precise determination of the buckling load via a nonlinear buckling analysis is not necessary. In a subsequent static analysis the previously determined buckling load is applied to the structure in order to check if this load leads to stresses higher than the allowed limits of the specific material.

6.2.3 Optimization Run and Results

The configuration of GAME for the optimization task is shown in table (6.4).

	GAME configuration
population size n	200
ratio r_{cp}	2
number of children n_c	400
number of generations n_{gen}	20
probability for recombination p_r	0.2
probability for mutation p_m	1
initial standard deviation for mutation σ_0	0.3 (ub-lb)

Table 6.4: Configuration of GAME

This setting results in an overall computational effort of 7800 objective function evaluation. Since the size of the final nondominated population has been set to 100 individuals, this results in an effort of 78 objective function evaluation per solution point. This effort is comparable to a gradient based optimization given that finite differences are used for the gradient computation.

The trade-off plot of buckling load vs. mass is shown in figure (6.16). The achieved buckling load is plotted in negative direction since GAME minimizes objectives. The solutions are marked according to the chosen material.

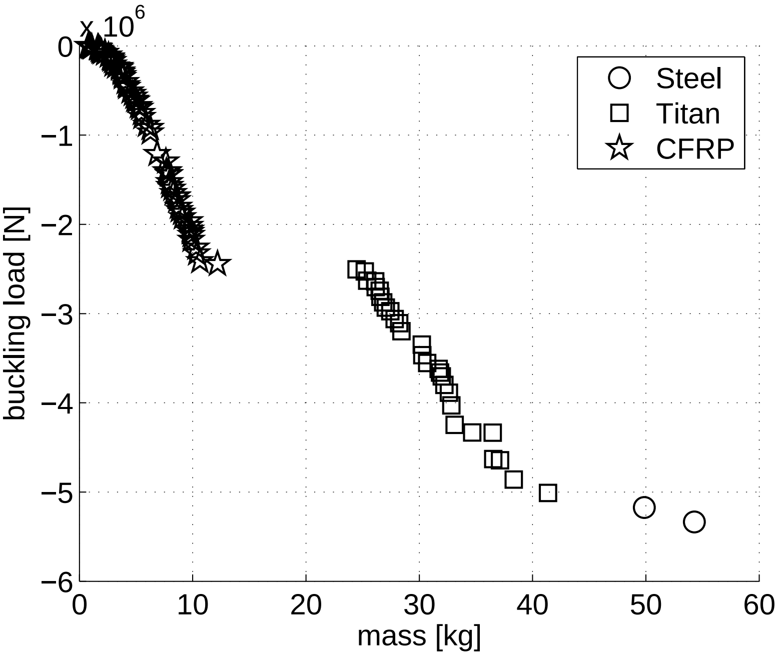


Figure 6.16: Trade-off plot of buckling load vs. mass for the stringer stiffened plate

The resulting Pareto-frontier is disconnected and consists of basically three parts. These parts correspond exactly to the material selection. The Pareto-optimal solutions for lower loads consist of CFRP. After exceeding a certain load level the nondominated solutions then all use titanium. Simultaneously a significant jump in mass can be observed. The same phenomenon can be observed again, although not so strong, for the change between titanium and steel solutions. On the first glance this seems striking, since it could be expected that the material with the best specific properties will be exclusively chosen for all Pareto-optimal and thus most efficient solutions. In order to explain the reasons for this alleged contradiction, the relation between the different design variables and the objectives is investigated. In figure (6.17) the distribution of specific settings for the design variables in dependence of the achieved buckling load is shown for all solutions of the Pareto-optimal set. Since the Pareto-optimal solutions form an ordered set an additional consideration of the dependency on mass is redundant. What characteristics can be expected from solutions achieving a high buckling load? For such solutions the first buckling mode should be the global

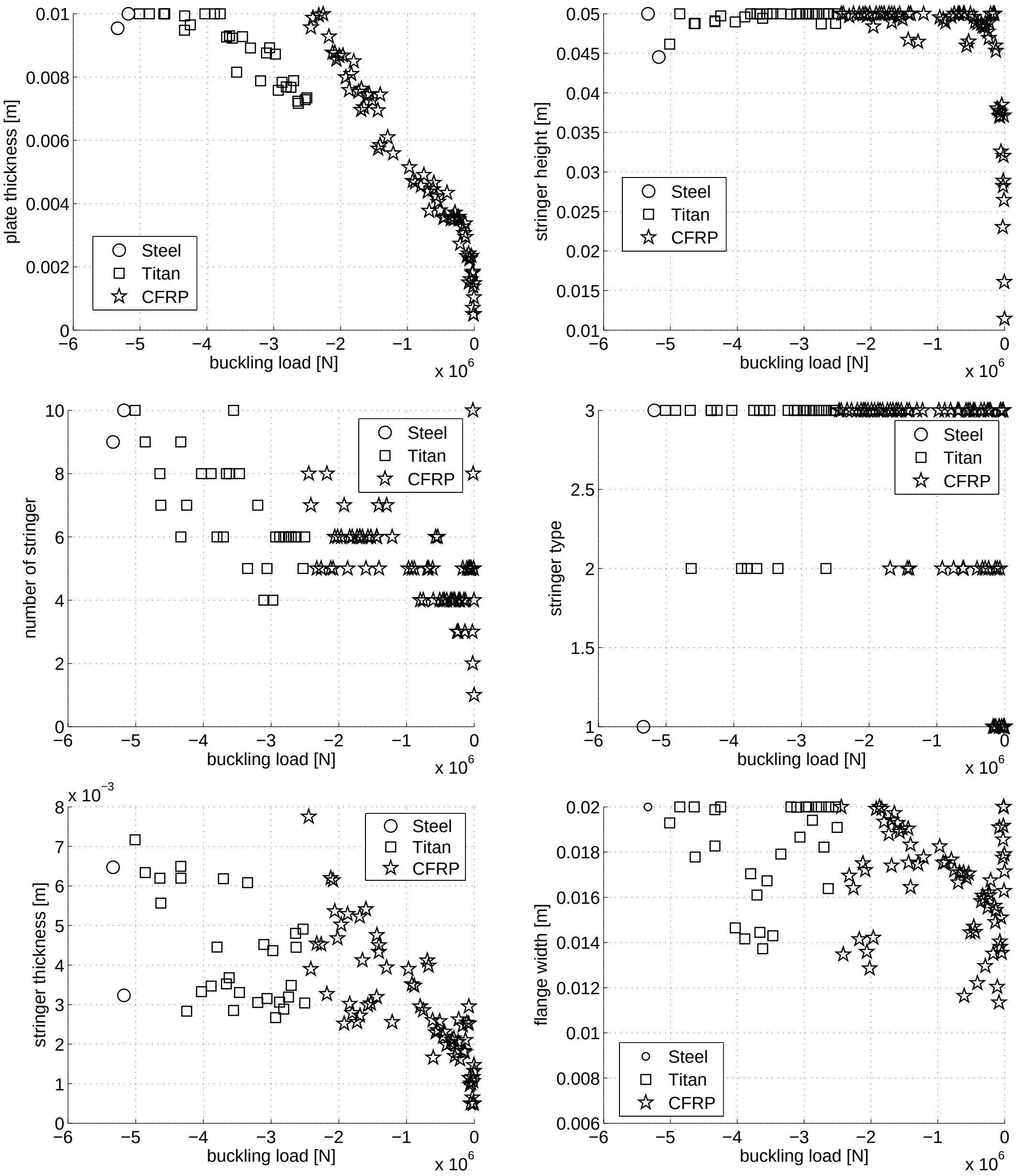


Figure 6.17: Distribution of the design variable values in dependence of the achieved buckling load for all Pareto-optimal solutions

one, no local buckling should occur prior to the first global mode. The critical load for the first global buckling mode is mainly dependent on the geometrical moment of inertia. So the solutions should reflect 'mass'-efficient ways for reaching a high geometric moment of inertia. Furthermore the structure should not fail statically prior to reaching the critical buckling load, so an appropriate cross-section area must be ensured.

For achieving a given buckling load there is definitely no one-to-one mapping to a certain setting of the design variables. The same buckling load can be reached by different combinations of the design variables. Certain deficiencies with respect to one variable can be compensated by a certain choice of another variable. Nevertheless, the sensitivity of the buckling load to certain design variables is significantly higher than to other ones.

This is reflected in the Pareto-optimal set. As can be seen in the plots the stringer height s_h is definitely the most efficient parameter to achieve a high buckling load with a minimum mass penalty. Already solutions for the lowest load level show stringer height settings at the upper boundary of 0.05m. Since the height of the stringers on the one hand contributes most to the geometrical moment of inertia of the web and on the other hand pushes the neutral axis away from the plate and thus increases the Steiner-part of of the plate, it agrees with common sense that efficient solutions necessarily feature a maximum stringer height. That the stringer height can actually be set to the upper boundary even for a small stringer thickness shows that within the given height range the stringers are relatively insensitive to local buckling.

The next important design variables for increasing the overall geometrical moment of inertia are the flange thickness s_{ft} and the plate thickness p_t, acting as a lower flange. Since the plate has a considerably higher cross-section area than the stringer flanges it will have a higher influence on both mass and buckling load. And this theoretical reasoning is clearly reflected by the solutions of the Pareto-optimal set. Although showing some small spread the stringer flange thickness of the solutions is quickly found to be close to its upper boundary with an increasing buckling load. The plate thickness increases only gradually with the rising buckling load though. Furthermore it shows a very strong, almost one-to-one coupling to the buckling load, whereas other parameters have a considerable spread. This shows that the plate thickness takes the part of the predominant trade-off parameter between mass and buckling load in this load range.

And it is also the plate thickness that the striking switch in material selection can be explained with. For a certain buckling load the plate thickness hits the upper boundary of 0.01m. At this point a complete change of the governing constraint happens. Although the buckling load could be further increased by adjusting the other parameters not being at their respective boundaries, the resulting increase of the cross-section area is not sufficient enough to have the structure statically bearing the increased load. The CFRP configurations run into the limit of the maximum allowable static stress for CFRP $\sigma_{allow,CFRP}$. This is illustrated in figure (6.18), in which the maximum Mises stress $\sigma_{max,Mises}$ for each solution is plotted in dependence of the buckling load. So now the problem is no longer driven by stability but by strength. Now not anymore the specific stiffness is governing the configuration of the solutions, but the absolute strength. Among the materials offering a higher absolute strength, steel and titanium, the latter one has a higher specific strength. So now the material titanium becomes the material of choice for efficient solutions, although imposing a significantly increase in mass. Of course, it is possible to design CFRP plates for these buckling loads with less mass, but not within the given boundaries of the design variables. Within this given boundaries configurations with a higher buckling load can only be achieved by switching to titanium. The same process can be observed at higher buckling loads for the change from titanium solutions to steel solutions. Aluminum solutions are not among the Pareto optimal set. This can be explained by the fact that in the regime, where the specific stiffness is decisive, aluminum is inferior to CFRP, and in the regime, in which absolute strength becomes decisive, it is inferior to steel and titanium. As stated before the considered material properties for this academic problem are representative values. For certain alloys with different properties there may as well be Pareto-optimal solutions with aluminum.

Concerning the other design variables no such tight coupling to the buckling load can be observed as for p_t and s_h. They all show some spread, indicating that within the set of efficient solutions the same quality can be reached by different combinations of them. Nevertheless certain trends can be observed. Expectedly most sizing variables show a tendency to rise with an increasing load level. The additional material built in corresponds to the required higher buckling stiffness and strength. Concerning the other discrete variables, the number of stringers s_n and the stringer type s_{type}, it can be observed that the number of stringers generally rises with an increasing buckling load. But most frequently the Pareto-optimal solutions have 4-6 stringer, a choice being in the middle of the allowed range. Obviously a larger number of stringer does not offer significant advantages. For lower loads all three stringer types are present, whereas for higher loads the stringer types with a flange are preferred. The reason for this is that the flange offers a higher bending stiffness and also increases the critical load for local buckling significantly. For the highest loads most frequently T-shaped stringers are chosen. For a given overall width of the flange the T-shaped stringer has a higher local buckling load for the flange than the L-shaped stringer since the free length of the flange is only half of that of the L-shaped stringer. Nevertheless these are only tendencies, respective deficiencies can be compensated by increasing other variables like the stringer thickness: Surprisingly, or not surprisingly, a solution with I-shaped stringers achieves the highest buckling load.

As mentioned before the optimization of stringer stiffened plate structures is a typical task in the field of lightweight structure design and thus has been addressed many times. In order to judge the quality of the solutions achieved by GAME, they have been compared to results by other researchers published in literature. In ([Wie96]) a thorough investigation of the optimal design of stiffened lightweight structures can

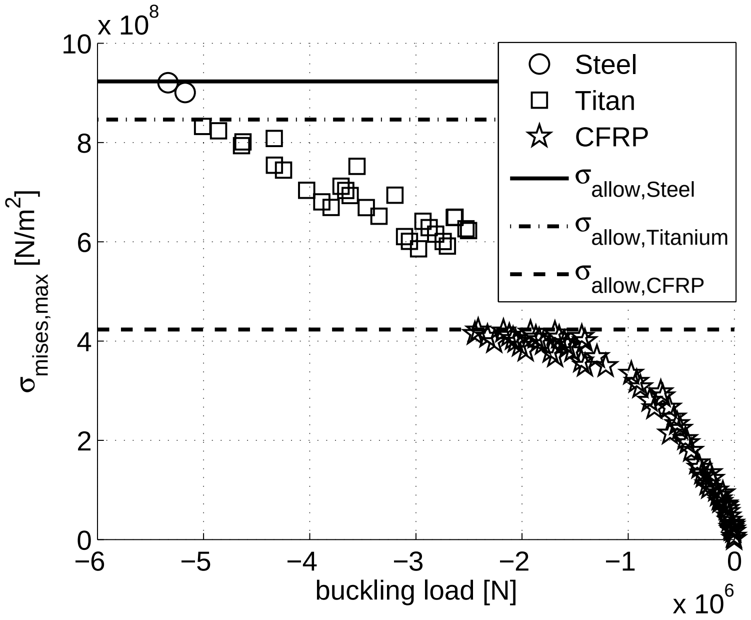


Figure 6.18: Maximum Mises stress $\sigma_{max,Mises}$ of the Pareto-optimal solutions in dependence of the buckling load

be found. Wiedemann establishes certain characteristic values with that the quality of a certain configuration can be characterized independently of load level and material properties. For the stringer stiffened plate these are $\alpha = s_t/p_t$, the quotient of stringer and plate thickness, and $\beta = s_h/(p_w/(s_n+1))$, the ratio between stringer height and the distance between the stringers. For the stinger stiffened plate with homogeneous materials and the given support condition in ([Wie96], chapter 4.3.3.3) the following optimal values for these characteristic values can be found: $\alpha \approx 0.4{-}0.5$, $\beta \approx 0.1$. For the Pareto-optimal solutions the α and β values have been computed and the results are plotted in figure (6.19) in dependence of the buckling load. It can be seen that the solutions found by GAME almost all feature β values close to 0.1 and thus comply

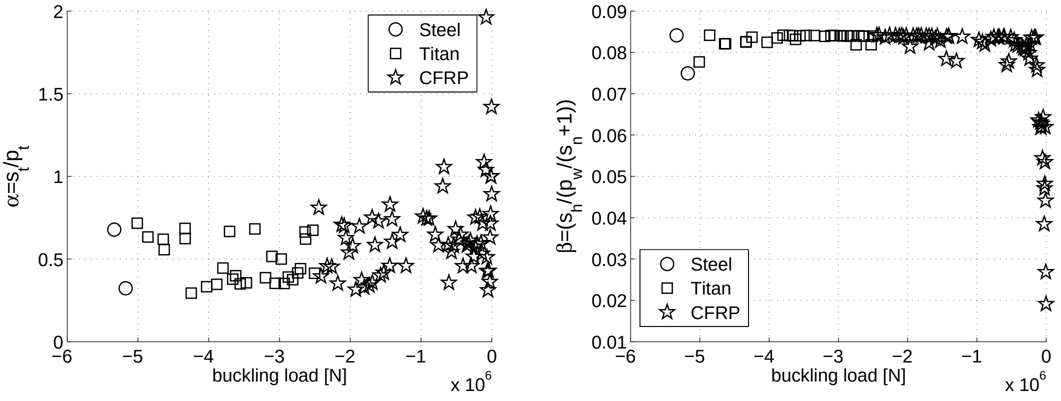


Figure 6.19: Characteristic values α and β for the Pareto-optimal set

with optimality condition claimed by Wiedemann. The α values show considerably more spread but nevertheless are mainly within the interval $[0.4, 0.6]$ and thus close to the optimal value. Wiedemann states also that the optimum is relatively insensitive to variation in α, which could be a reason for the observed spread. These findings show that GAME has successfully been able to find truly optimal configurations.

In conclusion GAME manages successfully to handle the combinatorial problem of selecting the material, the number of stringer, and the type of stringer. The resulting disconnected Pareto-frontier reveals the switch from a stiffness driven problem to a strength driven problem and thus provides good insight for the designer to the inner problem structure. One of the important findings is that the optimal material selection depends on the buckling load level aimed at. Finally in order to judge the quality of the solutions they have been compared with results from literature. The results achieved by GAME proved to be close to optimal solutions and thus verified that GAME has been successfully able to find optimal configurations.

6.3 Optimization of a High Precision Beam

For the assembly of circuit boards fast and high precision positioning systems are employed. One important aspect for the design of such systems is the dimensional stability with respect to static, dynamic and thermal loads in order to ensure the required high positioning accuracy at high feed rates. Especially the thermal stability is of interest since high displacements due to operational thermal loads require a frequent re-calibration of the system causing a stop of production.

Within a project in collaboration with an industrial partner the aluminum beam of the y-axis of such a 3-axes portal positioning system was to be replaced with an optimized design made from *Carbon Fiber Reinforced Plastic* (CFRP). The y-axis is shown in figure (6.20). In the following the y-axis is described in detail and the load cases as well as the goals and requirements are introduced. In order to judge the improvements achieved by the optimization the performance of the original aluminum design is computed as a reference. Before the actual formulation of the optimization task first some preliminary thoughts about the design task are discussed. Finally the optimization is performed and the results are discussed.

6.3.1 Description of the Y-Axis Beam

The base structure of the y-axis consists of a beam with a box profile. On this beam two rails are mounted by screws. On these rails again a carriage with the end effector moves driven by a linear motor. While the primary part of this motor is mounted on the carriage, the secondary part consisting of a segmentized magnet is screwed to the beam.

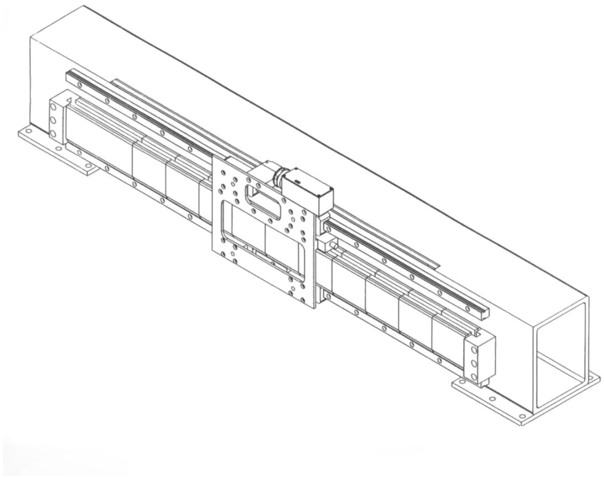


Figure 6.20: Y-axis of a 3-axes portal positioning system

6.3.2 Load Cases

The primary loads for the design are the static gravity load and a non uniform temperature load due to heating by the linear motor. Furthermore the y-axis is subject to acceleration loads due to the maneuvering during operation. The different loads are described in the following sections.

Static Load (LC1)

The static loads on the beam consist of the gravity load of its own mass as well as the gravity loads of the rails, the secondary part of the linear motor, and the carriage including the masses of the primary part, of the linear motor, and of the end-effector. The mass of the beam is dependent on its design, the respective masses of all other components are fixed and listed in table (6.5).

In figure (6.21) the positions of these loads are illustrated. The bending and torsion moments induced by the gravity loads depend on the position of the carriage. The worst case imposing the maximum leverages and thus largest moments is the center position, so this position is used as the design case.

end effector (Fz1)	5 [kg]
carriage + primary part of the linear motor (Fz2)	1.3 [kg]
rails (2*Fz3)	2*0.56 [kg]
secondary part of the linear motor (Fz4)	11.58 [kg]

Table 6.5: Component masses

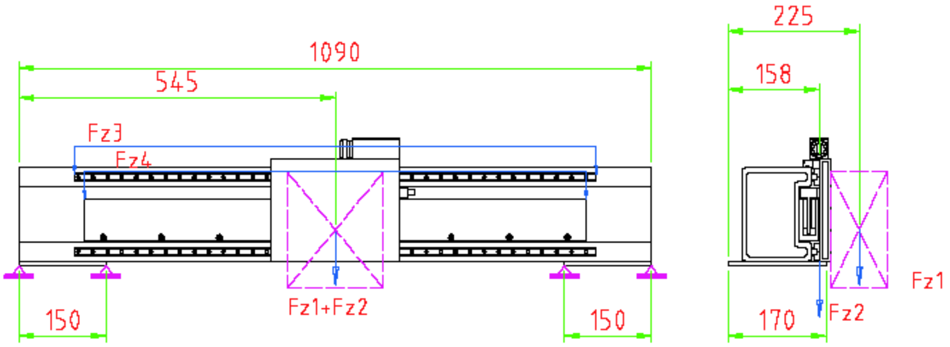


Figure 6.21: Static loads on the y-axis beam (gravity loads)

Thermal Loads (LC2)

The main source for the thermal loads is the operational temperature of the linear motor. By different heat transfer mechanism (conduction, convection, and radiation) this causes a heating up of the rails and the front side of the beam and thus a non uniform temperature load on the beam. Due to the lack of operational data only a simplified operational temperature distribution has been available, which is listed in table (6.6).

It has to be stated that this strongly simplified temperature distribution may not be particularly well suited to predict precisely the operational thermal displacements, but serves well as a comparison standard with respect to the reference design.

Dynamic Loads (LC3, LC4)

During operation the complete y-axis is accelerated in x-direction with up to $20m/s^2$. Furthermore the carriage including all attached components is accelerated along the rails in y-direction with up to $20m/s^2$. The resulting acceleration loads on the beam are dependent on the position of the carriage. But for both load cases the worst case

ambient temperature	20 [°C]
temperature of the rails	35 [°C]
temperature of the beam front side	25 [°C]
temperature of the rest of the beam	20 [°C]

Table 6.6: Temperature load case

is the center position of the carriage because it leads to the maximum leverages and longest load paths. For this reason this position is chosen as the design case.

6.3.3 Goals and Requirements

The goal of this design task was to improve the overall performance with respect to static and dynamic response behavior, thermal stability, and mass by substituting aluminum by carbon fiber composite material. Special emphasis was put on mass and thermal stability.

Detailed informations on the requirements have only been available for the static loads and are listed in table (6.7). As measures for the bending and torsion deformations the y- and z-displacement of the center line of the two rails are employed $(u_{x,r}^{(LC1)}, u_{z,r}^{(LC1)})$ as well as the twist angle $\varphi_{rp}^{(LC1)}$ of the rail plane.

Since no information on the requirements for the thermal displacements and the dynamic response behavior has been available, the respective results of the original design have been used as a reference. So the minimum requirements are that the CFRP design is not worse than the original design with respect to any of these load cases.

6.3.4 Modelling

For simulating the behavior of the y-axis a FEM model has been set up using the FEM program ANSYS. For the beam layered shell elements (shell99) have been used since they allow an explicit modelling of the laminate lay-up. The FEM model is shown in figure (6.22). The rails are modelled by beam elements (beam44). As a consequence only a substitute material and no detailed laminate can be used for the rail material. Since the secondary part of the linear motor is segmented and the segments are connected to the beam point-wise, there is no direct influence on the bending or torsional stiffness of the beam. Also due to the segmentation there is no influence on the thermal deformation of the beam. The only influence on the response behavior of the beam is based on its mass. So for the modelling of the secondary part of the linear motor point mass elements reproducing the inertia loads are sufficient. The same applies for the modelling of the carriage and the end-effector. All the point mass elements are positioned according to their center of gravity location and connected by

maximum bending displacement in z-direction $u_{z,r}^{(LC1)}$	$10.0E^{-6}$ [m]
maximum bending displacement in x-direction $u_{x,r}^{(LC1)}$	$10.0E^{-6}$ [m]
maximum torsion deformation in x-z-plane φ_{rp}^{LC1}	$25.0E^{-6}$ [rad]

Table 6.7: Maximum allowed static displacements

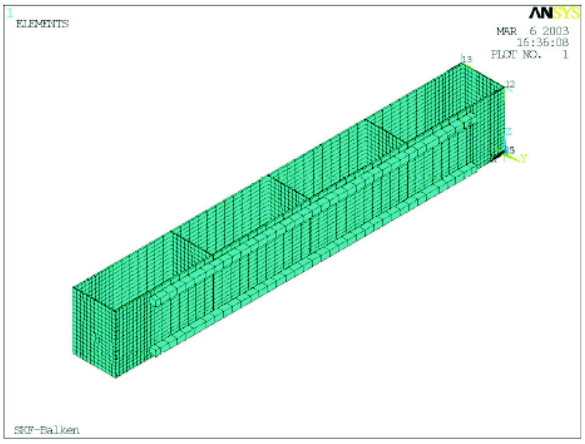


Figure 6.22: FEM model of the y-axis beam

highly stiff link elements having no mass. As stated before the carriage is positioned in the center position in order to induce the maximum possible bending and torsion loads.

The bottom of the beam is clamped on the left side over a length of $150mm$, the same applies for the right side with the exception that this end is allowed to move freely in y-direction.

6.3.5 Reference Design

The aluminum beam used as reference design has identical outer dimensions and a wall thickness of $8mm$. In table (6.8) the results of the reference design for the different load cases are shown. For the gravity load case LC1 the x- and z-deformations ($u_{x,r}^{(LC1)}$, $u_{z,r}^{(LC1)}$) are well below the requirements, while the twist angle ($\varphi_{rp}^{(LC1)}$) is close to the limit. As stated before no specific requirements have been available for the other load cases. Nevertheless, it can be stated that they lead to displacements which are considerably higher than the ones of the static load case. The thermal loads cause to the largest deformations.

	$max(u_{x,r})[1E^{-6}m]$	$max(u_{z,r})[1E^{-6}m]$	$max(\varphi_{rp})[1E^{-6}RAD]$
LC1	1.7	-2.3	25
LC2	-40	22.7	-425
LC3,LC4	20	-5	220

Table 6.8: Results for the reference design for the different load cases

6.3.6 Preliminary Design Considerations

The design problem at hand is mainly a material substitution problem. As the cross-section of the beam remains unchanged the main task is to determine the laminate lay-up. For this the layer materials, the number of layers , and their orientation angles have to be chosen. Concerning the material selection a high modulus fiber is desirable with respect to elastic displacements and a fiber with a low CTE with respect to thermal distortion. Additionally criteria with respect to availability, costs and processability have to be considered. As a compromise solution the fiber M40J has been selected. The material data for a single layer with fiber volume fraction $\nu_f = 60\%$ is given in table (6.9).

Since the results for the reference design revealed that the displacements due to the static loads are mainly caused by a shear deformations of the cross section itself, it has been decided to consider additional bulkhead walls for stiffening the cross section (see figure 6.22). These bulkheads are only subject to shear loads so their lay-up is restricted to $[\mp 45°]_S$. For the design the number of bulkheads as well as the respective number of layers remain to be determined.

Concerning the goals related to the mechanical response behavior the design task resembles the classical task of maximizing bending and torsional stiffness while simultaneously minimizing mass. The goals of minimizing thermal deformation is somewhat different. The thermal deformation has two sources: the non-uniform temperature distribution and the difference in coefficients of thermal expansion (CTE) between the beam and rail materials. The latter source resembles a kind of 'bi-metal' effect. To fight the first cause a material with a CTE as low as possible is desirable. To fight the second cause it is desirable to have the same materials or CTE in both components. Unfortunately the materials of the rails (steel) and the material of the respective substructure (CFRP $[0°/90°]_S$) could not be changed. So it remains an open question to the optimization to determine if it is more advantageous to employ a laminate with

E_1	225E9 $[N/m^2]$
E_2	7.5E9 $[N/m^2]$
G_{12}	4.7E9 $[N/m^2]$
ν_{12}	0.26
ρ	1600 $[kg/m^3]$
α_1	-0.7E-6 $[1/K]$
α_2	45E-6 [1/K]
ν_f	60 [%]

Table 6.9: Material properties for M40J

a CTE as low as possible or a CTE close the one of the rails. Also it can be expected that an optimal design for minimizing the thermal deformation will not be identical to a design for optimal stiffness. If this conflict is really significant will also be a result of the optimization task.

Concerning manufacturing it has been decided that the beam will be built by wrapping prepregs around a solid kernel made from foam. As a consequence of this manufacturing method all walls will have the same laminate lay-up. Also due to manufacturing reasons the orientation angles of the wall laminate have been restricted to $[0°, 90°, \pm 45°]$.

6.3.7 Optimization Task

For setting up the optimization task several options are possible. The most naive approach would be to minimize all three displacement measures (x- and z-displacement of the rail center line and twist angle of the rail plane) for all four load cases as well as the mass. This would result in a problem with 13 objectives. But because of the inherent and substantial difficulty of interpreting a 13 dimensional Pareto-front, it is questionable if such an approach is really reasonable. But also besides these practical objections it is questionable if such a problem formulation is really necessary. The fundamental justification for a multiobjective problem is that all objectives are conflicting. But not all of these performance measures are conflicting. E.g. although the z-displacements for the gravity load and the x-displacements for the acceleration in x-direction reflect bending stiffness with respect to different axes, the response behavior is directly coupled by the design variables. Due to the wrapping technique applied for the manufacturing all walls have the same lay-up, i.e. the bending stiffness in z-direction is directly coupled to the bending stiffness in x-direction. A design increasing the bending stiffness in z-direction will also increase the bending stiffness in x-direction, so a separate consideration makes no sense. As the bending stiffness in z-direction is already considered in the static load case, there is basically no conflict between the minimizing the displacements for the static load case and the acceleration load cases. All other deformation mechanism as torsion or shear are also induced by the gravity loads. So the deformations due to acceleration loads do not need to be considered as separate objective functions.

As stated before it is unknown if there is significant conflict between a design for minimum static deformations and minimum thermal deformations. So based on these considerations it has been decided to formulate the objective functions based on the static and thermal load cases skipping the acceleration load cases as they are inherently included.

The performance requirements have been formulated as separate x- z-, and φ deformations. In contrast to this it has been decided to use the overall displacement of the end effector (primary part) as the basis for the formulations of the objective functions. The overall displacement of the end effector is an aggregate measure integrating all deformation mechanisms. Since the essential design goal for the y-axis is the accuracy of the positioning of the end effector, its overall displacement is also the most relevant measure. So based on this reasoning there are three objective functions: the overall displacement of the end effector due to the static loads $u_{t,ee}^{(LC1)}$, the overall displacement due to the thermal loads $u_{t,ee}^{(LC2)}$, and the mass m.

To ensure that only solutions are considered that will at least meet the performance of the reference design with respect to all load cases, the respective response measures of the reference design have been applied as constraints. So the optimization task reads as follows:

$$
\begin{aligned}
&min\ \mathbf{z}(\mathbf{x}) = [m, u_{t,ee}^{(LC1)}(\mathbf{x}), u_{t,ee}^{(LC2)}(\mathbf{x})] \\
&\text{with: } \mathbf{x} = [n_{0^\circ}, n_{90^\circ}, n_{\pm 45^\circ}, n_{bh}, n_{\pm 45^\circ(bh)}] \\
&\text{subject to:} \\
&g_1(\mathbf{x}) = max(u_{x,r}^{(LC1)}) - max(u_{x,r}^{(LC1,ref)}) \leq 0, \\
&g_2(\mathbf{x}) = max(u_{z,r}^{(LC1)}) - max(u_{z,r}^{(LC1,ref)}) \leq 0, \\
&g_3(\mathbf{x}) = max(\varphi_{rp}^{(LC1)}) - max(\varphi_{rp}^{(LC1,ref)}) \leq 0, \\
&g_4(\mathbf{x}) = max(u_{x,r}^{(LC2)}) - max(u_{x,r}^{(LC2,ref)}) \leq 0, \\
&g_5(\mathbf{x}) = max(u_{z,r}^{(LC2)}) - max(u_{z,r}^{(LC2,ref)}) \leq 0, \\
&g_6(\mathbf{x}) = max(\varphi_{rp}^{(LC2)}) - max(\varphi_{rp}^{(LC2,ref)}) \leq 0, \\
&g_7(\mathbf{x}) = max(u_{x,r}^{(LC3)}) - max(u_{x,r}^{(LC3,ref)}) \leq 0, \\
&g_8(\mathbf{x}) = max(u_{z,r}^{(LC3)}) - max(u_{z,r}^{(LC3,ref)}) \leq 0, \\
&g_9(\mathbf{x}) = max(\varphi_{rp}^{(LC3)}) - max(\varphi_{rp}^{(LC3,ref)}) \leq 0, \\
&g_{10}(\mathbf{x}) = max(u_{x,r}^{(LC4)}) - max(u_{x,r}^{(LC4,ref)}) \leq 0, \\
&g_{11}(\mathbf{x}) = max(u_{z,r}^{(LC4)}) - max(u_{z,r}^{(LC4,ref)}) \leq 0, \\
&g_{12}(\mathbf{x}) = max(\varphi_{rp}^{(LC4)}) - max(\varphi_{rp}^{(LC4,ref)}) \leq 0, \\
&and \\
&1 \leq n_{0^\circ} \leq 20, \\
&1 \leq n_{90^\circ} \leq 20, \\
&1 \leq n_{\pm 45^\circ} \leq 10, \\
&1 \leq n_{bh} \leq 10, \\
&1 \leq n_{\pm 45^\circ(bh)} \leq 20,
\end{aligned}
\tag{6.8}
$$

The design variables are the number of layers in 0°-direction (n_{0°), in 90°-direction (n_{90°), in $\pm 45^\circ$-direction ($n_{\pm 45^\circ}$), the number of bulkhead walls (n_{bh}), and the number of $\pm 45^\circ$-layers for the bulkhead walls ($n_{\pm 45^\circ(bh)}$). All design variables are discrete. The optimization task is thus a discrete constrained multiobjective problem with three objectives functions.

6.3.8 Optimization Run and Results

Configuration of GAME

The configuration of GAME for this task is shown in table (6.10). These settings cause an overall computational effort of 3900 objective function evaluation. Since the size of the final nondominated population has been set to 60 individuals, this results in an effort of 65 objective function evaluation per solution point. Again the effort per solution point is in the same range as for a gradient based optimization given that finite differences are used for the gradient computation.

Results

The trade-off plot of the nondominated solution set for the three objective functions is shown in figure (6.23). The crescent like form of the Pareto-optimal surface shows two things. First it reflects the classical conflict between static displacements and mass. In the m-$u_{t,ee}^{(LC1)}$-plane a perfectly shaped frontier is formed. The second conclusion is that the conflict between the thermal deformation on the ones side and the static deformation and mass on the other side is obviously nearly insignificant. The extension of the Pareto-surface in the $u_{t,ee}^{(LC2)}$-direction is only small, the solutions gather almost only in the m-$u_{t,ee}^{(LC1)}$-plane. So it can be deduced that there is no general conflict between designing a laminate for low static deformations and designing one for low thermal deformations. This can also be seen in the $u_{t,ee}^{(LC1)}$-$u_{t,ee}^{(LC2)}$-projection, in which no classical front is built up. Here solutions with the lowest static deformations are almost identical to the ones with the lowest thermal deformations. For the design this is advantageous because now only a trade-off decision between mass and static deformations has to be drawn.

To see the relevance of the different design variables with respect to the performance, the distribution of their values has been plotted in dependence of the static and thermal deformation in figure (6.24) for all nondominated solutions. The only significant correlation can be observed between the number of $\pm 45°$ layers and the static deformation. For low static deformations a high number of $\pm 45°$ layers is obviously necessary. This reconfirms that the predominant deformation mechanisms are the torsion and shear deformation of the cross section. A higher number of $0°$- or $90°$-layers for the wall laminate or $\pm 45°$-layers for the bulkhead laminate is advantageous to achieve small static deformations, as the increased relative frequency shows. But since there exist solutions with a relatively low number of layers achieving equivalently small deformations, a higher number of layers with these orientations is not

	configuration CFRP beam
population size n	100
ratio r_{cp}	2
number of children n_c	200
number of generations n_{gen}	20
probability for recombination p_r	0.2
probability for mutation p_m	1
initial standard deviation for mutation σ_0	0.3 (ub-lb)

Table 6.10: Configuration of GAME

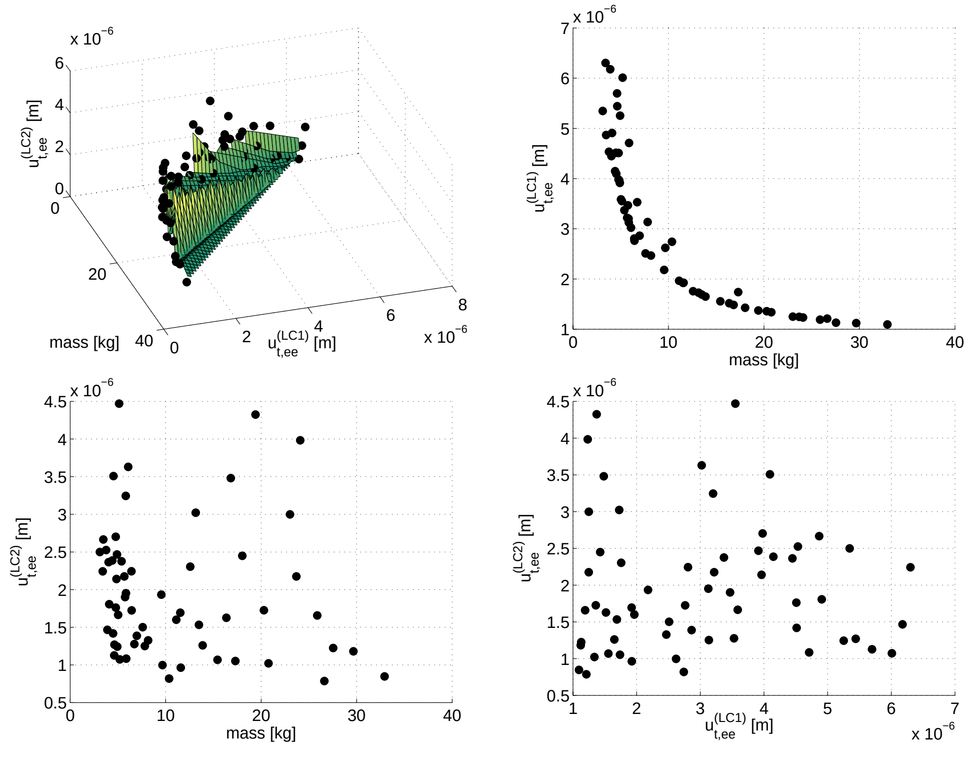


Figure 6.23: Trade-off plot m vs. $u_{t,ee}^{(LC1)}$ vs. $u_{t,ee}^{(LC2)}$ for the y-axis

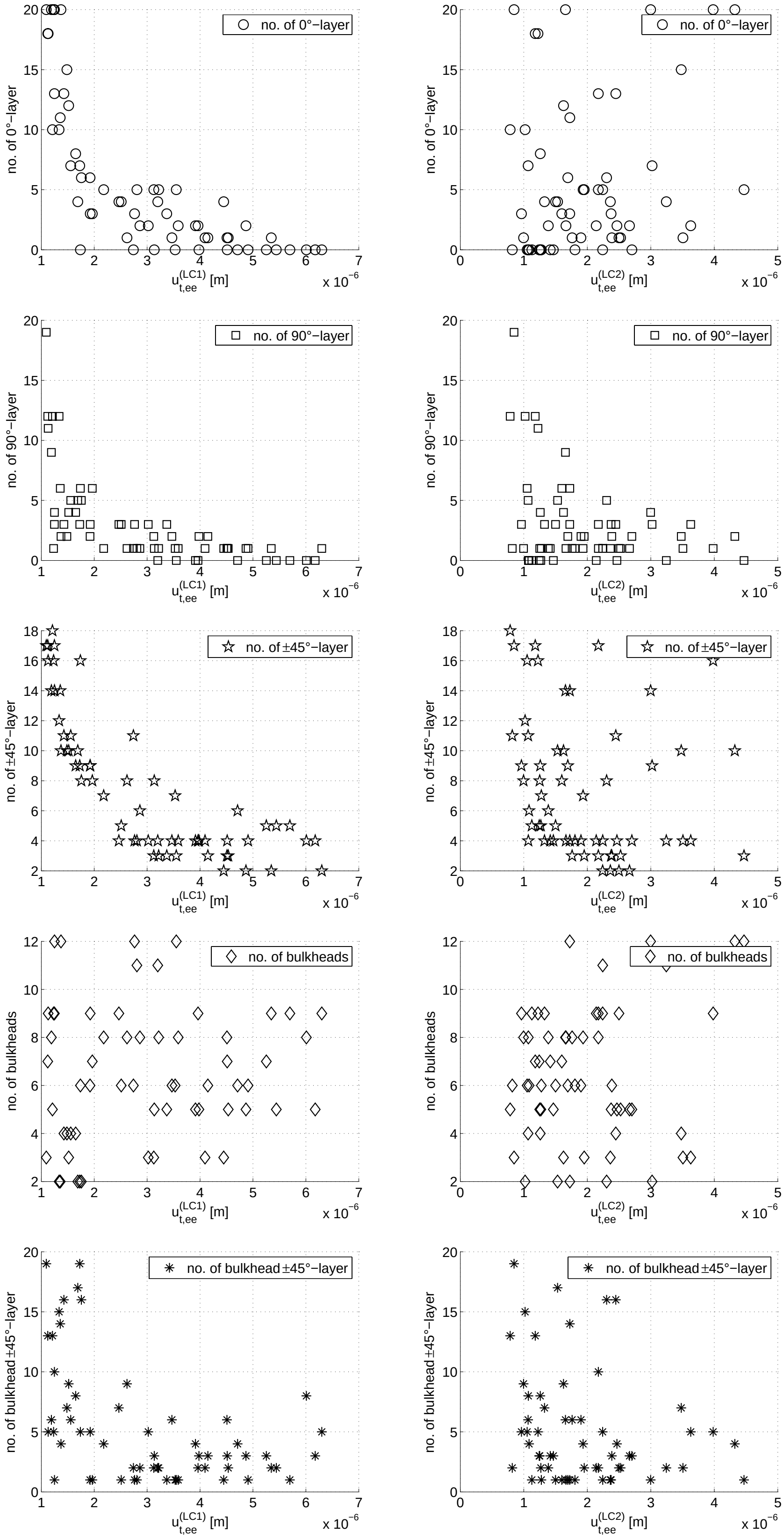


Figure 6.24: Distribution of the design variable values for all Pareto-optimal solutions in dependence of $u_{t,ee}^{(LC1)}$ and $u_{t,ee}^{(LC2)}$

necessarily needed.

The plots reveal that most of the design variables are of alternative nature: deficiencies due to the setting of one variable can be compensated by the certain settings of another one. So e.g. a lower number of bulkhead walls can be compensated by thicker bulkheads walls or more $\pm 45°$ layers in the wall laminate.

Significant correlation between a single variable and the thermal deformations cannot be observed. Again here the overall composition of all design variables seems to be decisive. To see the relevance of the resulting CTE values of the wall laminate the α_1 and α_2 values are plotted in dependence of the corresponding thermal deformation in figure (6.25). In this figure also the dependency of α_1 on α_2 is shown.

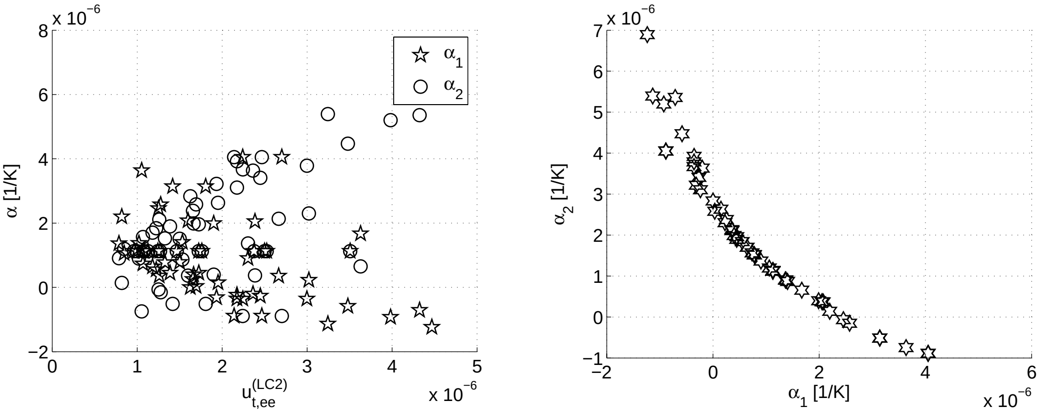


Figure 6.25: Dependency of the thermal deformation $u_{t,ee}^{(LC2)}$ on the CTE values α_1 and α_2 of the wall laminate and the dependency of α_1 on α_2

As could be expected it is not possible to have low values for α_1 and α_2 simultaneously. A low α_1 will result in a high α_2 and vice versa. Concerning the correlation between α_1 or α_2 and the resulting thermal deformation $u_{t,ee}^{(LC2)}$ it can be stated that very low or even negative α_1 does not lead to a low thermal displacement of the end effector. More promising are α_1 values close to $1E-6[1/K]$ and consequently α_2 values in the same region. Alternatively low settings of α_2 do also achieve low thermal displacements. Values of α_1 close to $1E-6[1/K]$ make sense because the effective CTE of the rails and their substructure is $1.23E-6[1/K]$. So the 'bi-metal' effect between the rails and the beam is minimized.

Final Design Decision

Due to the lack of a significant conflict between thermal and static displacements the only trade-off decision has to be drawn between mass and static deformation. In figure (6.26) again the Pareto optimal solutions are plotted in two projections (m-$u_{t,ee}^{(LC1)}$) and ($u_{t,ee}^{(LC1)}$-$u_{t,ee}^{(LC2)}$). Since the reference aluminum solutions has a weight of 13.4kg only solutions up to 20kg are displayed. Additionally a color coding of the mass has been introduced to see the mass dependency also in the second projection.

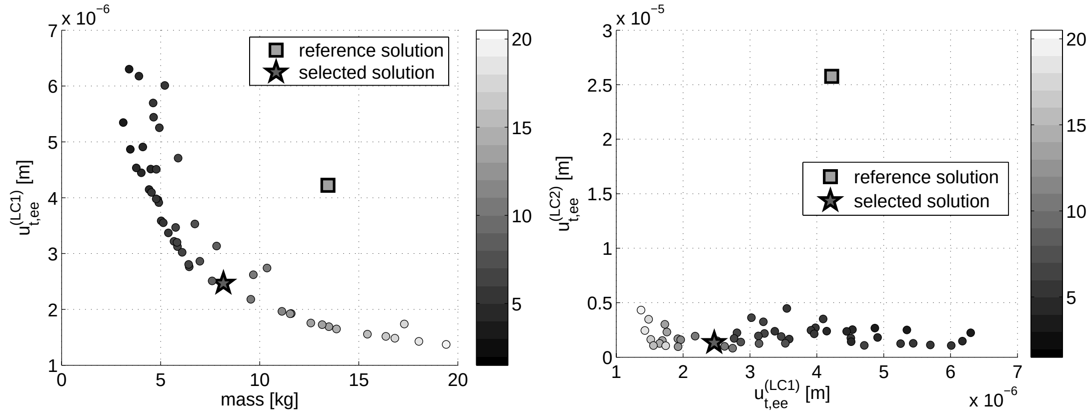


Figure 6.26: Trade-off plots: m vs. $u_{t,ee}^{(LC1)}$ and $u_{t,ee}^{(LC1)}$ vs. $u_{t,ee}^{(LC2)}$)

In both plots the reference solution and the finally chosen design are marked. When looking at the trade-off plots with respect to the thermal deformation it is obvious that all solutions present a substantial improvement in comparison with the reference solution. With respect to the trade-off between mass and static deformation a compromise solution has been selected that offers approximately equal relative improvements in terms of mass and displacement. The reason why exactly this solution has been selected and not one of the neighboring ones, is that this solution features an almost isotropic laminate lay-up. This has the advantage of being more robust against other, possible unforeseen, loads not considered in the original specifications. The configuration of the chosen design is presented in table (6.11). In figure (6.27) a contour plot of the total nodal displacements of the finally chosen CFRP design is shown for the static load case.

In table (6.12) all performance measures for the chosen CFRP design and the reference design as well as the achieved improvements are listed. The overall improvements with respect to the reference design are impressive.

design variable	value
$n_{0°}$	4
$n_{90°}$	3
$n_{\pm 45°}$	4
n_{bh}	9
$n_{\pm 45°(bh)}$	7

Table 6.11: Laminate lay-up of the final design

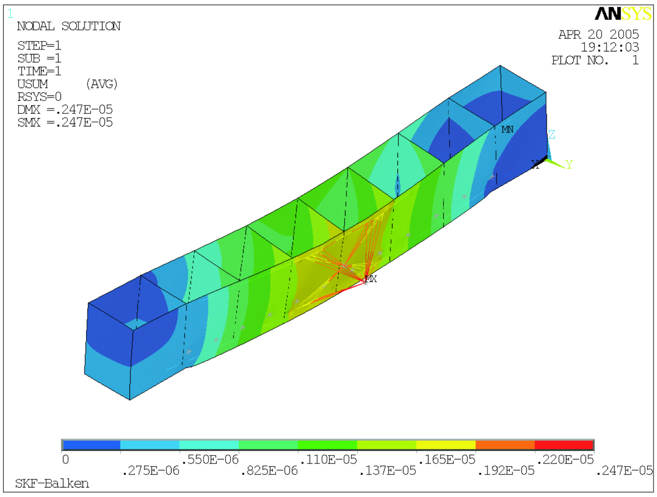


Figure 6.27: Contour plot of the total nodal displacements of the final CFRP design for the load case LC1

performance measure	reference design	CFRP design	improvement
mass [kg]	13.44	8.17	-39.2%
$u_{t,ee}^{(LC1)}$ $[1E^{-6}m]$	4.223	2.466	-41.6%
$u_{t,ee}^{(LC2)}$ $[1E^{-6}m]$	25.769	1.327	-94,8%
$max(u_{x,r}^{(LC1)})$ $[1E^{-6}m]$	1.172	0.160	-86.3%
$max(u_{z,r}^{(LC1)})$ $[1E^{-6}m]$	2.326	1.812	-22.1%
$max(\varphi_{rp}^{(LC1)})$ $[1E^{-6}RAD]$	25.065	8.924	-64,4%
$max(u_{x,r}^{(LC2)})$ $[1E^{-6}m]$	48.634	3.991	-91.8%
$max(u_{z,r}^{(LC2)})$ $[1E^{-6}m]$	22.744	0.630	-97.2%
$max(\varphi_{rp}^{(LC2)})$ $[1E^{-6}RAD]$	427.050	4.767	-98.9%
$max(u_{x,r}^{(LC3)})$ $[1E^{-6}m]$	18.650	4.356	-76.6%
$max(u_{z,r}^{(LC3)})$ $[1E^{-6}m]$	4.860	1.997	-58.9%
$max(\varphi_{rp}^{(LC3)})$ $[1E^{-6}RAD]$	222.340	9.659	-95.6%
$max(u_{x,r}^{(LC4)})$ $[1E^{-6}m]$	2.020	0.509	-74,8%
$max(u_{x,r}^{(LC4)})$ $[1E^{-6}m]$	2.468	1.954	-20,8%
$max(\varphi_{rp}^{(LC4)})$ $[1E^{-6}RAD]$	28.027	8.990	-67,9%

Table 6.12: Comparison between the chosen CFRP design and the aluminum reference design

7 Summary and Conclusions

In this thesis an optimization algorithm has been developed. The central goals for this algorithm have been that it is general enough to be applicable to an utmost range of optimization problems in the field of structural design while simultaneously being efficient enough not to be ruled out for complex problems due to excessive computational costs. The applicability of the algorithm encompasses nonlinear constrained problems, multiobjective problems as well as combinatorial problems with both discrete and continuous design variables.

The algorithm that has been named GAME (Genetic Algorithm for Multicriteria Engineering) is based on the concept of evolutionary algorithms (EA). EAs have been selected as the basis because they provide the required generality as they are applicable to virtually any kind of problem. Especially the handling of discrete variables is possible. Based on a thorough review of existing evolutionary optimization techniques on the one hand selected and well proven methods and techniques are recombined in a new and advantageous way in GAME. On the other hand GAME features new techniques. Here the essential innovation is the integration of response surface methods in the standard evolutionary algorithm for increasing performance and efficiency.

For the adaptation to multiobjective problems a specialized fitness assignment scheme is implemented in GAME. The fundamental idea of this fitness assignment scheme is to have the population converge to the Pareto-frontier. The implemented scheme directly employs the concept of Pareto-dominance for the fitness assignment and belongs to the class of Pareto-ranking methods first proposed by Goldberg. Among existing representatives of these methods it has been decided to implement the goals and priorities based approach proposed by Fonseca and Fleming in GAME. The essential advantage of this approach is that it simultaneously includes the handling of constraints. So this generalized ranking scheme not only addresses the handling of multiple objectives but also another crucial aspect in EAs, the handling of constraints, in a very elegant, parameterless way.

It is shown by numerical experiments with different test problems that this integrated constraint handling method is superior to commonly applied penalty function methods or the simple rejection of all infeasible solutions. It is also shown by numerical experiments that the implemented ranking scheme successfully achieves the convergence of the population to the Pareto-frontier or the close vicinity of the Pareto frontier.

Furthermore employing such a scheme offers the advantage of inherently increasing the efficiency of GAME, or EAs in general, for multiobjective problems. Applied to multiobjective problems conventional mathematical algorithms are forced to use aggregating methods like the weighted sum approach and thus produce only a single solution per run. In order to determine the same number of Pareto-optimal solutions as GAME does in a single run, multiple runs would have to be performed. So especially if gradients have to be computed by finite difference methods the overall computational cost can reach approximately the same level.

The review on existing algorithms has revealed a clear superiority of algorithms with elitism operators. In GAME elitism is realized by an external archive of finite size in which all nondominated solutions are stored. This external pool also serves a second important task, the preservation of diversity among the nondominated solutions. If the number of nondominated solutions exceeds a given size an intelligent replacement procedure is applied eliminating the most similar individuals and thus promoting diversity among the nondominated solutions. This replacement procedure is based on a clustering method similar to the one proposed by Zitzler. The most attractive characteristic in comparison to conventional niching techniques is that it is a parameterless method. The successful functioning of this approach is validated by the results of numerical experiments.

GAME uses a real representation for continuous variables and an integer one for discrete variables. In order to improve the search characteristics continuous and discrete variables are also treated differently by the reproduction operators. The separate treatment allows continuous recombination and mutation operators although discrete variables are present. So the mutation operator of the evolutionary strategies is applied in GAME for the continuous variables, which is substantially more effective than comparable binary operators. It allows for reduced population sizes and thus reduced computational costs.

The most important aspect for the practical applicability of GAME has been the reduction of the high computational cost of conventional EAs. For this purpose two key elements are implemented in GAME. The first one is the grain-parallelization of the objective- and constraint function evaluation for all individuals. This parallelization strategy is based on the EA specific characteristic that the evaluation of each individual is independent. This approach does not reduce computational costs directly, but, maybe even more important, the overall computation time. The grain-parallelization is realized on a PC-cluster. The potential of this approach to cut down total computation time is based on the assumption that the evaluation of the individuals takes orders of magnitude more time than internal computations of the algorithm. For the intended field of application in structural optimization this assumption can generally be considered as given. Tests with example problems involving typical FEM computations of several minutes show that the overall time reduction is nearly proportional to the number of employed computers in the cluster.

The second essential element to increase the efficiency is augmentation of the con-

ventional EA by response surface approximation methods (RSA). The basic idea is to further exploit the knowledge about the design space that has already been gathered by the current and past generations. This idea has been realized in GAME by setting up an RSA branch in parallel to the standard reproduction operators. On this branch first RSAs are built for all objective- and constraint functions using the individuals of the current and past generations. Subsequently a gradient based optimization (SQP-algorithm) is run on these RSAs and the resulting solutions are fed back to the population. As RSAs are continuous methods the presence of discrete variables in GAME poses a special challenge. So prior to the RSA built-up the population is split into continuous subsets with consistent discrete variable settings. For the RSAs only low order model function are used. In order to still achieve a sufficient accuracy of the RSAs, the RSAs are not spanned over the complete design space, but over smaller, more homogeneous parts of the design space. The technique for this subdividing is again based on special clustering methods.
The performance of the RSA integration has been verified by numerical experiments including single- and multiobjective test problems. The results confirm significant performance increases and thus show the potential of this method. Improvements for single-objective problems are higher than for multi-objective problems. It could be shown that in order to reach a similar solution quality the RSA integration allows for a reduction of the population size of up to 80%. So it could be verified that the RSA integration in EAs is a strategy for increasing the efficiency. The experiments also revealed that the success of the RSA branch is strongly dependent on the achieved accuracy of the RSAs. The implemented subdivision technique by clustering proved its ability to increase the accuracy of the RSAs. But this approach has its natural limits. So subdivision can only work down to a certain minimum cluster size including the minimum number of required points for the RSA built-up. But for certain function characteristics this may still not provide the required accuracy. Further work may concentrate on the optimal interaction between the RSA based optimization and the evolutionary process. Here the optimal selection of design points for building the RSAs in order to increase the accuracy of the RSAs may be a topic of interest.

Numerical experiments have also been performed to investigate the general characteristics of GAME. One aspect has been the optimal choice of certain algorithm parameters like the population size, the number of children, and the number of generations. The results reveal that the required computational effort for reaching a certain solution quality depends on the dimension of the design space as well as on the characteristics of the objective functions. More design variables or a more difficult objective function require a higher population size and thus lead to higher computational costs. It could be shown that this unfavorable characteristic of all EAs can be compensated partly by the RSA integration, although not for all problems. Concerning the optimal choice of population size, number of children, and number of generations the experiments indicate that the performance of GAME is relatively robust with respect to certain choices for the respective parameters. The quality of the achieved solutions is primarily depended on the total computational cost, independent of wether the number of function evaluations is increased by a larger population, more children, or a more generations.

GAME is also compared to another state-of-the-art multiobjective EA, the Nondominated Sorting Genetic Algorithm II published by Deb. For the spectrum of the chosen test problems GAME has been able to achieve better results. Although it is difficult to generalize these findings the results indicate that GAME is at least competitive in comparison to other state-of-the-art algorithms in multiobjective evolutionary optimization.

Finally by the successful application of GAME to different optimization tasks in the field of structural design its applicability to actual engineering problems is demonstrated.

Bibliography

[Bai02a] H. Baier. Faserverbundwerkstoffe. Skriptum zur Vorlesung und Übung, Lehrstuhl für Leichtbau, Technische Universität München, 2002.

[Bai02b] H. Baier. Multidisciplinary Design Optimization. Skriptum zur Vorlesung und Übung, Lehrstuhl für Leichtbau, Technische Universität München, 2002.

[Baj04] D. Bajusz. Design and Optimization of a Reflect Array. Semesterarbeit, TUM–MW 65/0401–SA, Lehrstuhl für Leichtbau, Technische Universität München, 2004.

[Bal79] M.J. Balas. Direct velocity feedback control of large space structures. *AIAA Journal of Guidance*, 2(3):252–253, 1979.

[BD87] G. E. P. Box and N. R. Draper. *Empirical Model-Building and Response Surfaces*. Wiley series in probability and mathematical statistics. John Wiley & Sons, Inc., New York, 1987.

[BGLL99] H. Baier, R. Gleichmar, H. Langer, and G. Locatelli. Eine Maschine für multidisziplinäre Strukturoptimierung. In *Proceedings zum 17. CAD-FEM Users' Meeting*, 1999.

[BGLP01] H. Baier, R. Gleichmar, H. Langer, and T. Pühlhofer. A grain parallelized structural MDO engine which includes shape and topology variables. In *CEAS Conference on Multidisciplinary Aircraft Design und Optimization, Köln*, 2001.

[BHS97] T. Bäck, U. Hammel, and H.-P. Schwefel. Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation*, 1(1):3–17, April 1997.

[BK97] T.T. Binh and U. Korn. Mobes: A multiobjective evolutionary strategy for constrained optimization problems. In Mendel, editor, *The Third International Conference on Genetic Algorithms*, pages 176–182, 1997.

[BL04] H. Baier and H. Langer. Integrated design optimization of structural and actuator parameters in reflector shape control. In *Proceedings SPIE -*

Astronomical Telescopes and Instrumentation, Glasgow, Scotland, UK, June 2004. SPIE, SPIE.

[BS91] I. N. Bronstein and K. A. Semendjajew. *Taschenbuch der Mathematik*. B. G. Teubner Verlagsgesellschaft, Stuttgart-Leipzig, 25 edition, 1991.

[BSS94] H. Baier, C. Seeßelberg, and B. Specht. *Optimierung in der Strukturmechanik*. Friedrich Vieweg & Sohn Verlagsgesellschaft mbH, Braunschweig/Wiesbaden, 1994.

[Coe] C.A. Coello Coello. Web repository of information on evolutionary multi-objective optimization (EMOO: http://www.lania.mx/∼ccoello/emoo/). Internet.

[Coe99a] C.A. Coello Coello. A comprehensive survey of evolutionary-based multi-objective optimization techniques. *Knowledge and Information Systems. An International Journal*, 1(3):269–308, August 1999.

[Coe99b] C.A. Coello Coello. A survey of constraint handling techniques used with evolutionary algorithms. Technical Report Lania-RI-99-04, Laboratorio Nacional de Informática Avanzada, Xalapa, Veracruz, México, 1999. Disponible en: http://www.lania.mx/ ccoello/constraint.html.

[Coe02] C.A. Coello Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1245–1287, January 2002.

[Coe03] C.A. Coello Coello. Evolutionary multiobjective optimization: Current and future challenges. In J.M. Benitez, O. Cordon, F. Hoffmann, and R. Roy, editors, *Advances in Soft Computing - Engineering, Design and Manufacturing*, pages 243–256. Springer-Verlag, 2003.

[DAPM00] K. Deb, S. Agrawal, A. Pratab, and 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, editors, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 849–858, Paris, France, 2000. Springer. Lecture Notes in Computer Science No. 1917.

[Deb98] K. Deb. Technical report ci-49/98: Multi-objective genetic algorithms: Problem difficulties and construction of test problems. Technical report, Department of Computer Science/LS11, University of Dortmund, Dortmund, Germany, 1998.

[Deb99] K. Deb. Multi-objective genetic algorithm: Problem difficulties and construction of test problems. *Evolutionary Computation Journal*, 3(7):205–230, 1999.

[Deb00] K. Deb. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186:311–338, 2000.

[Deb02] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. Interscience Series in Systems and Optimization. John Wiley & Sons LTD, New York, USA, 2002.

[DeJ75] K.A. DeJong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, Ann Arbor, MI, USA, 1975.

[DPAM02] K. Deb, A. Pratab, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[ea82] A.J. Morris et al. *Foundations of Structural Optimization: A Unified Approach*. Whiley Series in Numerical Methods in Engineering. John Wiley and Sons, 1982.

[EKL98] K. Ehrlenspiel, A. Kiewert, and U. Lindemann. *Kostengünstig entwickeln und konstruieren - Kostenmanagement bei der integrierten Produktentwicklung*. Springer-Verlag, Berlin, Heidelberg, NewYork, 2nd edition, 1998.

[EMH01] M. Erickson, A. Mayer, and J. Horn. The niched pareto genetic algorithm 2 applied to the design of groundwater remediation systems. In E. Zitzler, K. Deb, L. Thiele, C.A. Coello Coello, and D. Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, number 1993 in Lecture Notes in Computation Science, pages 681–695. Springer, 2001.

[FF93] C.M. Fonseca and P.J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, CA, July 1993. Morgan Kaufmann.

[FF95] C.M. Fonseca and P.J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.

[FF96] C.M. Fonseca and P.J. Fleming. On the performance assessment and comparison of stochastic multiobjective optimizers. In H.M. Voigt, W. Ebeling, I. Rechenberg, and H.P. Schwefel, editors, *Parallel Problem Solving from Nature–PPSN IV*, Lecture Notes in Computer Science, pages 584–593, Berlin, Germany, September 1996. Springer-Verlag.

[FF98a] P. J. Fleming and C. M. Fonseca. Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part I: A unified formulation. In *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, volume 28, pages 26–37, 1998.

[FF98b] P. J. Fleming and C. M. Fonseca. Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part II: A unified formulation. In *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, volume 28, pages 26–37, 1998.

[Fie99] John P. Fielding. *Introduction to Aircraft Design*. Cambridge Aeospace Series. Cambridge University Press, Cambridge, UK, 1999.

[Fou85] M.P. Fourman. Compaction of symbolic layout using genetic algorithms. In J.J. Grefenstette, editor, *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, pages 141–153, Pittsburgh, PA, 1985.

[FOW66] L.J. Fogel, A.J. Owens, and M.J. Walsh. *Artificial Intelligence Through Simulated Evolution*. John Wiley, Chichester, UK, 1966.

[Fri03] M. Friedrich. Anwendung der Response Surface Methode beim Entwurf von schnell rotierenden Chopperscheiben aus CFK. Semesterarbeit, TUM–MW 65/0330–SA, Lehrstuhl für Leichtbau, Technische Universität München, 2003.

[GC97] Mitsuo Gen and Runwei Cheng. *Genetic Algorithms and Engineering Design*. Wiley Series in Engineering Design and Automation. John Wiley and Sons, 1997.

[Gle04] R. Gleichmar. *Approximationen und paralleles Rechnen bei der multidisziplinären Strukturoptimierung*. PhD thesis, Technical University of Munich, Institute of Lightweight Structures, 2004.

[Gol89] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading,MA, 1989.

[GR87] D.E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In J.J. Grefenstette, editor, *Genetic Algorithms and Their Application: Proceedings of the Second International Conference of Genetic Algorithms*, pages 41–49, Hillsdale, NJ, USA, 1987. Lawrence Erlbaum.

[HL92] P. Hajela and C.Y. Lin. Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4:99–107, 1992.

[HM79] C.L. Hwang and A.S.M. Masud. *Multiple Objectives Decision Making - Methods and Applications*. Springer, Berlin, 1979.

[HN93] J. Horn and N. Nafpliotis. Multiobjective optimization using the niched pareto genetic algorithm. IlliGAL Report 93005, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1993.

[HNG94] J. Horn, N. Nafpliotis, and D. Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, volume 1, pages 82–87, Piscattaway, N.J., USA, June 1994. IEEE Service Center.

[Hol75] J.H. Holland. *Adaption in Natural and Artificial Systems*. University of Michigan Press, Ann-Arbor, 1975.

[Hor97] J. Horn. *Handbook of Evolutionary Computation*, chapter F1.9. Institute of Physics Publishing, Bristol, 1997.

[JH94] J. Joines and C. Houck. On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs. In D. Fogel, editor, *Proceedings of the first IEEE Conference on Evolutionary Computation*, pages 579–584, Orlando, Florida, 1994. IEEE Press.

[KC00] J.D. Knowles and D.W. Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

[Kip03] T. Kipp. Optimierung einer Stringer-versteiften Platte mittels Evolutionärer Algorithmen. Semesterarbeit, TUM–MW 65/0306–SA, Lehrstuhl für Leichtbau, Technische Universität München, 2003.

[Kir94] U. Kirsch. *Structural Optimization*. Springer-Verlag, Berlin, Heidelberg, New-York, 1994.

[Kuc91] H. Kuchling. *Taschenbuch der Physik*. Verlag Harri Deutsch, Thun und Frankfurt/Main, 13. edition, 1991.

[Kur91] F. Kursawe. A variant of evolution strategies for vector optimization. In H.P. Schwefel and R. Männer, editors, *Parallel Problem Solving from Nature - Proceedings 1st Workshop*, pages 193–197, Berlin, 1991. Springer.

[LLMB00] G. Locatelli, H. Langer, M. Müller, and H. Baier. Simultaneous optimization of actuator placement and structural parameters by mathematical and genetic optimization algorithms. In *Proc. of IUTAM Conference Smart Structures and Structronic Systems*, Magdeburg, September 2000. IUTAM, Kluwer.

[Loc01] G. Locatelli. *Piezo-Actuated Adaptive Structures for Vibration Damping and Shape Control - Modeling and Testing*. PhD thesis, Technical University of Munich, Institute of Lightweight Structures, 2001.

[LPB02] H. Langer, T. Pühlhofer, and H. Baier. An approach for shape and topology optimization integrating CAD parameterization and evolutionary algorithms. In *Proceedings of the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Atlanta, Georgia, USA, September 2002. AIAA/ISSMO.

[LPB03] H. Langer, T. Pühlhofer, and H. Baier. Towards a configuration design optimization method. In *Proceedings of the 5th World Congress on Structural and Multidisciplinary Optimization*, Lido di Jesolo, Venice, Italy, May 2003. ISSMO.

[LPB04] H. Langer, T. Pühlhofer, and H. Baier. A multiobjective evolutionary algorithm with integrated response surface functionalities for configuration optimization with discrete variables. In *Proceedings 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference 2004*, Albany, NewYork, USA, September 2004. AIAA/ISSMO.

[LRS98] M.G. Laumanns, G. Rudolph, and H.P. Schwefel. A spatial predator-prey approach to multi-objective optimization: A preliminary study. In A.E. Eigen, T. Bäck, M. Schoenauer, and H.P. Schwefel, editors, *Fifth International Conference on Parallel Problem Solving from Nature (PPSN-V)*, pages 241–249, Berlin, Germany, 1998. Springer.

[Mat04] The MathWorks, Inc., Natick, MA, USA. *Matlab, Release 14, SP 1, Online Documentation*, 2004.

[Mic99] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, Berlin, Heidelberg, New-York, 3 edition, 1999.

[Mie99] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluver, Boston, 1999.

[MM95] R. H. Myers and D. C. Montgomery. *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. Wiley series in probability and statistics. John Wiley & Sons, Inc., New York, 1995.

[Mül01] U.C. Müller. Polformoptimierung eines elektrischen Leichtbaumotors mit Hilfe eines genetischen Algorithmus. Diplomarbeit, TUM–MW 65/0127–DA, Lehrstuhl für Leichtbau, Technische Universität München, 2001.

[Mye71] R. H. Myers. *Response Surface Methodology*. Allyn and Bacon, Boston, 1971.

[N.N] N.N. Openpbs homepage. www.openpbs.org.

[OGC91] C.K. Oei, D.E. Goldberg, and S.J. Chang. Tournament selection, niching, and the preservation of diversity - IlliGal report no. 91011. Technical

Report 91011, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1991.

[OK95] A. Osyczka and S. Kundu. A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. *Structural Optimization*, 10:94–99, 1995.

[Pfi98] G. Pfister. *In Search of Clusters*. Prentice Hall PTR, NJ, 2 edition, 1998.

[PGL$^+$02] T. Pühlhofer, R. Gleichmar, H. Langer, M. Müller, U. Müller, and H. Baier. Multidisciplinary optimization and statistical evaluations of structures and electro-mechanical components with ANSYS on a PC-cluster. In *Proceedings of the 20th CAD-FEM Users' Meeting*, 2002.

[PLB04] T. Pühlhofer, H. Langer, and H. Baier. Multicriteria and discrete configuration and design optimization with applications for satellites. In *Proceedings 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference 2004*, Albany, NewYork, USA, September 2004. AIAA/ISSMO.

[Pol95] C. Poloni. Hybrid GA for multi-objective aerodynamic shape optimization. In G. Winter, J. Periaux, M. Galan, and P. Cuesta, editors, *Genetic Algorithms in Engineering and Computer Science*, pages 397–416. Wiley & Sons, Chichester, 1995.

[Pow81] M. J. D. Powell. Optimization algorithms in 1979. *Committee on Algorithms Newsletter*, 5:2–16, February 1981.

[PR01] M. Poda-Razumtsev. An overview of evolutionary algorithms for multiobjective optimization. Semesterarbeit, TUM–MW 65/0016–SA, Lehrstuhl für Leichtbau, Institut für Luft- und Raumfahrttechnik, TU München, May 2001.

[Rec73] I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog Verlag, Stuttgart, 1973.

[RSH96] W. J. Roux, N. Stander, and R. T. Hafkta. Response surface approximations for structural optimization. In *6th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA 96-4050-CP*, pages 642–654, Bellevue, WA, 1996.

[Sch85a] J. David Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In Lawrence Earlbaum, editor, *Genetic Algorithms and Their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100, Hillsdale,N.J.,USA, 1985.

[Sch85b] K. Schittkowski. NLQPL: A fortran-subroutine solving constrained nonlinear programming problems. In *Annals of Operations Research, Vol. 5, pp 485-500*, 1985.

[Sch95] J.R. Schott. Fault tolerant design using single and multi-criteria genetic algorithms. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, MA, Boston, MA, USA, 1995.

[SD94] N. Srivinas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–48, Fall 1994.

[SHF94] E. Schöneburg, F. Heinzmann, and S. Feddersen. *Genetische Algorithmen und Evolutionsstrategien*. Addison-Wesley, 1994.

[SSH96] J. Sobieszczanski-Sobieski and R. T. Haftka. Multidisciplinary aerospace optimization: Survey of recent developments. *34th AIAA Aerospace Sciences Meeting and Exhibit, AIAA 96-0711*, 1996.

[Tan95] M. Tanaka. GA-based decision support system for multi-criteria optimization. In *Proceedings of the International Conference on Systems, Man and Cybernetics*, volume 2, pages 1556–1561, 1995.

[Tha00] J. Thalhauser. Anwendung der Response Surface Methode zur Berechnung strukturmechanischer Größen für ein Laminat unter Temperaturbelastung. Semesterarbeit, TUM–MW 65/0018–SA, Lehrstuhl für Leichtbau, Technische Universität München, April 2000.

[Van84] G. N. Vanderplaats. *Numerical Optimization Techniques For Engineering Design*. McGraw-Hill in mechanical engineering. McGraw-Hill, Inc., New York, 1984.

[Vel99] D.A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.

[Ven98] G. Venter. *Non-Dimensional Response Surfaces for Structural Optimization with Uncertainty*. PhD thesis, Graduate School of the University of Florida, 1998.

[VHS96] G. Venter, R. T. Haftka, and J. H. Starnes. Construction of response surfaces for design optimization applications. In *6th AIAA/USAF/-NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA 96-4040-CP*, pages 548–564, Bellevue, WA, 1996.

[Wie96] J. Wiedemann. *Leichtbau 2: Konstruktion*. Springer, 2 edition, 1996.

[ZDT99] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results, technical report 70. Technical report, Computer Engineering and Networks Laboratory (TIK), ETH Zürich, Switzerland, December 1999.

[Zit99] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, ETH Zürich, Schweiz, November 1999.

[ZLT$^+$02] E. Zitzler, M. Laumanns, L. Thiele, C.M. Fonseca, and V. Grunert da Fonseca. Why quality assessment of multiobjective optimizers is difficult. In E.Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A.C. Schultz, J. F. Miller, E. Burke, and N.Jonoska, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002)*, pages 666–673, San Francisco, California, USA, July 2002. Morgan Kaufmann Publishers.

[ZT98] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms - a comparative case study. In *Parallel Problem Solving From Nature V*, pages 292–301, 1998.

[ZT99] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.

[ZTL$^+$02] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, and V. Grunert da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review - tik-report no. 139. Technical report, Institut für technische Informatik und Kommunikationsnetze, ETH Zürich, Switzerland, June 2002.