INSTITUTO POLITÉCNICO NACIONAL
CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN

Multiobjective evolutionary algorithm aiming for good hausdorff approximations

TESIS

Que para obtener el grado de
Doctorado en Ciencias de la Computación

Presenta:
M. en C. Christian Horacio Domínguez Medina

Directores de Tesis
Dra. Nareli Cruz Cortés
Dr. Oliver Schütze

México D.F., Mayo 2016

SIP-14 BIS

INSTITUTO POLITÉCNICO NACIONAL
SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

ACTA DE REVISIÓN DE TESIS

En la Ciudad de México, D.F. siendo las 16:00 horas del día 25 del mes de enero de 2016 se reunieron los miembros de la Comisión Revisora de la Tesis, designada por el Colegio de Profesores de Estudios de Posgrado e Investigación del:

Centro de Investigación en Computación

para examinar la tesis titulada:

"Multiobjective evolutionary algorithm aiming for good hausdorff approximations"

Presentada por el alumno:

DOMÍNGUEZ	**MEDINA**	**CHRISTIAN HORACIO**
Apellido paterno	Apellido materno	Nombre(s)

Con registro:

B	1	1	0	8	9	7

aspirante de: **DOCTORADO EN CIENCIAS DE LA COMPUTACIÓN**

Después de intercambiar opiniones los miembros de la Comisión manifestaron ***APROBAR LA TESIS***, en virtud de que satisface los requisitos señalados por las disposiciones reglamentarias vigentes.

LA COMISIÓN REVISORA

Directores de tesis

[illegible] [illegible]

Dra. Nareli Cruz Cortés — Dr. Oliver Steffen Schütze

[illegible] [illegible]

Dr. Sergio Suárez Guerra — Dr. Carlos Fernando Aguilar Ibáñez

[illegible] [illegible]

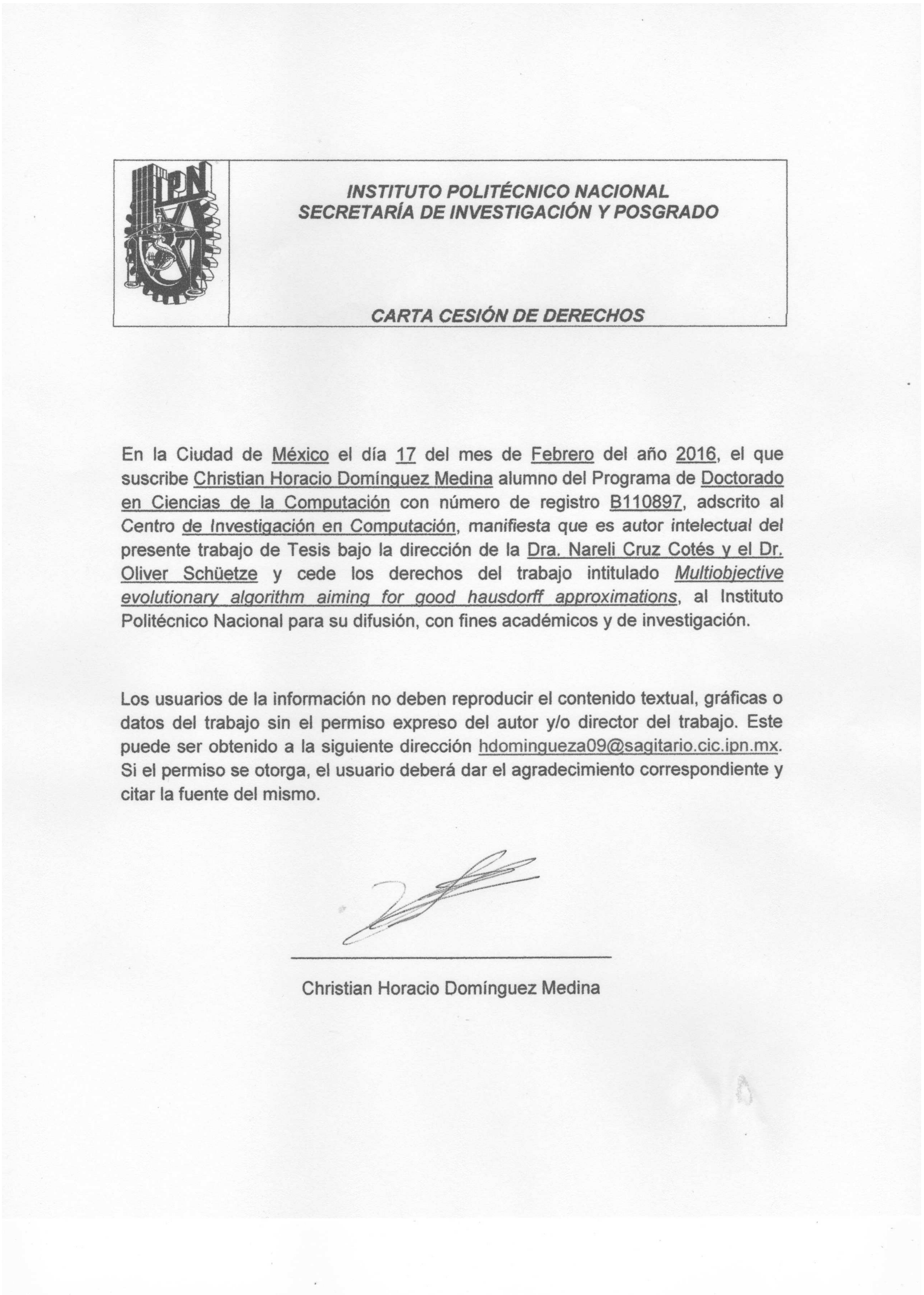
Dra. Adriana Lara López — Dr. Marco Antonio Moreno Armendáriz

PRESIDENTE DEL COLEGIO DE PROFESORES

[illegible]

Dr. Luis Alfonso Villa Vargas

ESTADOS UNIDOS MEXICANOS
INSTITUTO POLITECNICO NACIONAL
CENTRO DE INVESTIGACION EN COMPUTACION
DIRECCION



Resumen

La optimización multi-objetivo tiene como fin, el encontrar una aproximación del frente de Pareto de problemas de optimización multi-objetivo dado con la mayor precisión posible. Puesto que el frente de Pareto no puede ser, en general, encontrado completamente, es aceptado entonces que los algoritmos evolutivos de optimización multi-objetivo (EMOAs) sólo proporcionen una representación de tamaño finito del conjunto de intereses. En general, se requiere que la aproximación obtenida simultáneamente ofrezca una buena convergencia al frente de Pareto y una buena distribución de las soluciones a lo largo del frente de Pareto, dichas características pueden funcionar como indicadores de desempeño.

En esta tesis, el indicador promedio de Hausdorff (Δ_p), recientemente desarrollado y que cumple con ambos criterios de desempeño (convergencia y distribución), se utiliza para resolver el problema de encontrar aproximaciones Hausdorff de tamaño finito del frente de Pareto de problemas de optimización multi-objetivo (MOP) por medio del uso del cómputo evolutivo.

Dado que muchas aplicaciones requieren una aproximación uniformemente distribuida a lo largo del frente de Pareto y, aproximaciones que son buenas en términos de Hausdorff normalmente distribuyen las soluciones de forma uniforme a lo largo del frente de Pareto, hemos propuesto dos algoritmos basados en el indicador Δ_p con el objetivo de obtener este tipo de aproximaciones.

En primer lugar proponemos el algoritmo evolutivo de partición y selección (PSE-MOA), que se basa en el algoritmo de partición y selección (PSA) presentado recientemente. Su técnica de partición permite la selección de un conjunto finito de soluciones bien diversificado a partir de un conjunto arbitrario dado, con un bajo costo computacional independiente del número de objetivos del problema dado.

Cuando se integra el PSA dentro de un EMOA, esta estrategia mejora significativamente la explotación de la diversidad en las aproximaciones del frente de Pareto. Esta propuesta explora el uso del PSA como el mecanismo de selección de las mejores soluciones, dirigido a mejorar el valor del indicador Δ_p entre la aproximación encontrada y el frente de Pareto.

En segundo lugar, se presenta el algoritmo evolutivo Δ_p (Δ_p-EA), un nuevo algoritmo para encontrar aproximaciones Hausdorff del frente de Pareto. La idea subyacente es utilizar directamente el problema de optimización escalar que es inducido por el indicador Δ_p. Este enfoque puede ser visto como la transformación de la información de un conjunto de soluciones a un escalar y se puede abordar tanto con técnicas de programación matemática como con algoritmos evolutivos.

En este trabajo de tesis demuestra la capacidad de los nuevos enfoques propuestos en la solución de un conjunto de MOPs de prueba con dos, tres y cuatro objetivos y diferentes formas de sus frentes de Pareto.

Abstract

Evolutionary multi-objective optimization aims at approximating the Pareto front of a given multi-objective optimization problem as accurately as possible. As the exact Pareto front can in general not be computed, evolutionary multi-objective optimization algorithms (EMOAs) can only provide a finite size representation of the set of interest. In general, closeness to the Pareto front and a sufficient spread of the solutions are simultaneously required and function as performance indicators. In this thesis, the recently developed averaged Hausdorff Δ_p indicator, which explicitly aims at fulfilling both performance criteria, is used to solve the problem of computing finite size Hausdorff approximations of the Pareto front of multi-objective optimization problems (MOPs) by means of evolutionary computing.

Since many applications desire an approximation evenly spread along the Pareto front and approximations that are good in the Hausdorff sense are typically evenly spread along the Pareto front, we proposed two algorithms based on the Δ_p indicator to obtain this kind of approximations.

First we propose the part and selection evolutionary algorithm (PSEMOA), which is based on the recently presented part and selection algorithm (PSA). Its partitioning technique allows the selection of a well-diversified set out of an arbitrary given set, while maintaining low computational cost regardless of the number of objectives of the given problem. When PSA is embedded into an EMOA, this strategy has significantly enhanced the exploitation of diversity in the approximations of the Pareto front. This proposal explores the use of the PSA as an archiving selection mechanism, to improve the averaged Hausdorff distance of the found approximation of the Pareto front.

Second, we present the Δ_p evolutionary algorithm (Δ_p-EA), a novel method to compute averaged Hausdorff approximations of the Pareto front of MOPs. The underlying idea is to directly utilize the scalar optimization problem that is induced by the indicator Δ_p. This approach can be viewed as a certain set based scalarization and can be addressed both by mathematical programming techniques and evolutionary algorithms.

We demonstrate the strength of the novel approaches on some benchmark MOPs with two up to four objectives and with different shapes of their Pareto fronts.

Agradecimientos

Esta tesis está dedicada a mis padres, Elvia Medina y Rafael Domínguez, a quienes agradezco de todo corazón por su apoyo.

Agradezco a mis hermanos Karen y Oswaldo por la compañia y el apoyo que me brindan. Se que cuento con ellos siempre.

Agradezco a mis amigos por compartir conmigo sus vidas.

Agradezco a mi comité tutorial, a la Dra. Adriana, Dr. Marco, Dr. Sergio y Dr. Carlos por indicarme el camino a seguir.

Agradezco a mis asesores de tesis, la Dra. Nareli y el Dr. Oliver.

Agradezco al Instituto politécnico Nacional y al Centro de Investigación en Computación.

Agradezco al CONACyT por su gran apoyo.

Gracias a todos.

Acronyms

.

CHIM: Convex hull of individual minima.

DM: Decision maker.

EA: Evolutionary algorithm.

EMO: Evolutionary multi-objective optimization.

EMOA: Evolutionary multi-objective algorithm.

EP: Evolutionary programming.

ES: Evolutionary strategy.

GA: Genetic algorithm.

GD: Generational distance.

IBEA: Indicator based evolutionary algorithm.

IGD: Inverted generational distance.

KKT: Karush-Kuhn-Tucker.

LI: Linear interpolation.

MDS: Multi-dimensional scaling.

MOEA/D: Multi-objective evolutionary algorithm based on decomposition.

MOP: Multi-objective optimization problem.

NSGA: Non-dominated sorting genetic algorithm.

PID: Proportional-integral-derivative.

PSA: Part and selection algorithm.

PSEMOA: Part and selection evolutionary multi-objective algorithm.

R_2-EMOA: R2 Evolutionary Multi-Objective Algorithm.

SBX: Simulated binary crossover.

SMSEMOA: S-metric selection evolutionary multi-objective optimization algorithm.

SOP: Single-objective optimization problem.

SPEA: Strength Pareto evolutionary algorithm.

VEGA: Vector evaluation genetic algorithm.

Contents

List of Figures

List of Tables

List of Algorithms

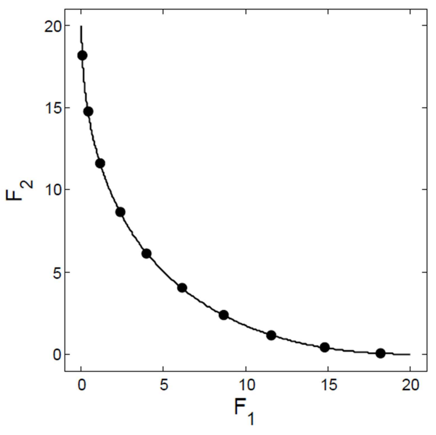
Chapter 1

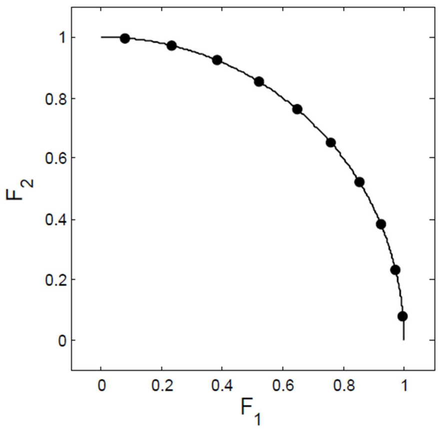
Introduction

In many real-world applications, it is required to consider several conflicting objectives concurrently leading to a multi-objective optimization problem (MOP). One important characteristic of MOPs is that the solution set, the so-called Pareto front, is typically not given by a singleton as for scalar optimization problems but forms a $(d-1)$-dimensional object, where d is the number of objectives involved in the problem. Since these sets can apart from trivial examples not be computed analytically, numerical methods are required that compute suitable finite size approximations. Evolutionary multi-objective algorithms (EMOAs) have caught the interest of many researchers in the recent past since they accomplish this task outstandingly (e.g., [1, 2, 3, 4, 5]). Due to their population based and global approach, EMOAs are capable of computing an entire set of candidate solutions A within one run of the algorithm such that the image of A (denoted by $F(A)$) is well-distributed and sufficiently close to the Pareto front which is the image of the Pareto set X^*.

One problem that remains is to measure the performance of these algorithms, i.e., the relation of $F(A)$ to the Pareto front. For this, several performance indicators have been proposed such as the Hypervolume indicator [6] and the R-indicators [7]. The indicator used within this work is Δ_p [8] which can be viewed as the averaged Hausdorff distance between $F(A)$ and the Pareto front. Optimal Δ_p approximations prefer, roughly speaking, candidate solutions whose images are evenly spread along the Pareto front (Figure 1.1 shows the best ten points approximations to three different Pareto fronts according to Δ_p in two dimensions). Thus, Δ_p aims to avoid gaps in the representation of the solution set providing the decision maker a suitable overview of the given optimal possibilities. The Δ_p value is thus a direct measure for the approximation quality of $F(A)$.

A particular example where (averaged) Hausdorff approximations are advantageous is the following approach to the the online-optimization of mechatronical systems: in a first step, the (conflicting) objectives of the underlying system are identified and an approximation A of the Pareto front of the resulting MOP is computed offline. This set A is further on used as the basis for upcoming online control by providing a repository of reference operating points: the "optimal" point $p(\lambda) \in X^*$ is determined online, i.e., while running the system, according to the current situation or demand λ of the





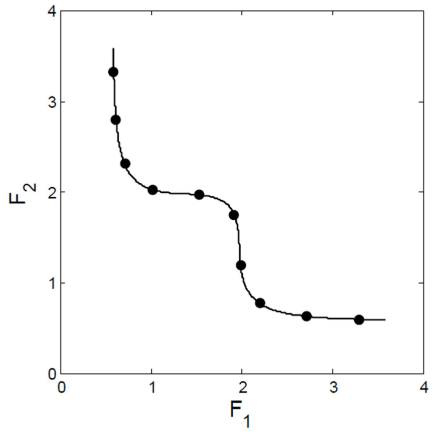


Figure 1.1: Pareto fronts and optimal Δ_1 approximations for three different MOPs with two objectives.

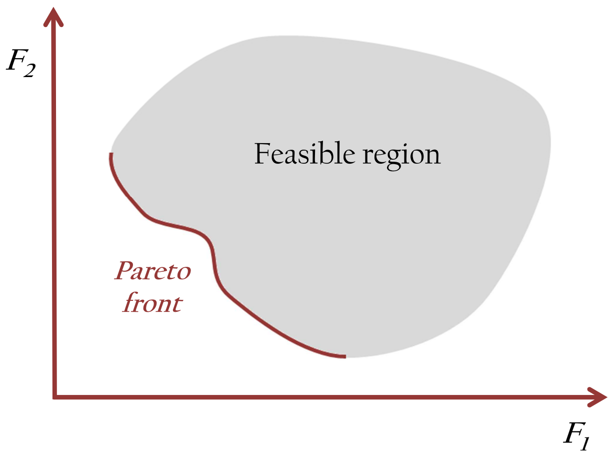


Figure 1.2: Feasible region of a bi-objective problem and its Pareto front.

system and is used as the actual *operating point*. Since $\lambda = \lambda(t)$ varies with time, this 'optimal' point has to be updated over and over again, according to the sensitivity of the system. See [9, 10] for an operating point assignment strategy of a linear drive, and [11] for an online-adjustment of an active suspension system.

Crucial for the *stability* of the system is that the switch from one point or system setting $p(\lambda_1)$ to the next one $p(\lambda_2)$ can *not* be done arbitrarily, but has to be carried out as smoothly as possible. That is, large and abrupt qualitative changes (amongst others) in terms of the changes in the influential objective values, have to be avoided. Thus, gap free representations of the Pareto front are desired. Since the values of λ_i are not known a priori, evenly spread solutions around the Pareto front seem to be most appealing which is provided by Δ_p approximations. Such approximations are particularly interesting for certain problems related to multi-objective control where the approximation of the Pareto front serves as a basis for the online control by providing a repository of reference operating points (see [12, 13, 14] for such examples).

1.1 Research Problem Area

The Pareto front of a MOP typically forms a $(d-1)$-dimensional entity, where d is the number of objectives, and these objectives are defined by the set of functions $F = (F_1, F_2, ..., F_d)^T$ involved in the problem. Figure 1.2 shows an example of a feasible region of a bi-objective (i.e., $d = 2$) problem and its Pareto front. The shape of the Pareto fronts depends of the definition of the problem and can take many different forms as it is shown in Figure 1.3 for bi-objective problems.

EMOAs generate a finite size approximation of the Pareto front, but which is the best distribution of points? Figure 1.4 shows some possible final solutions or final approximations of the Pareto front with different characteristics: approximation (b)

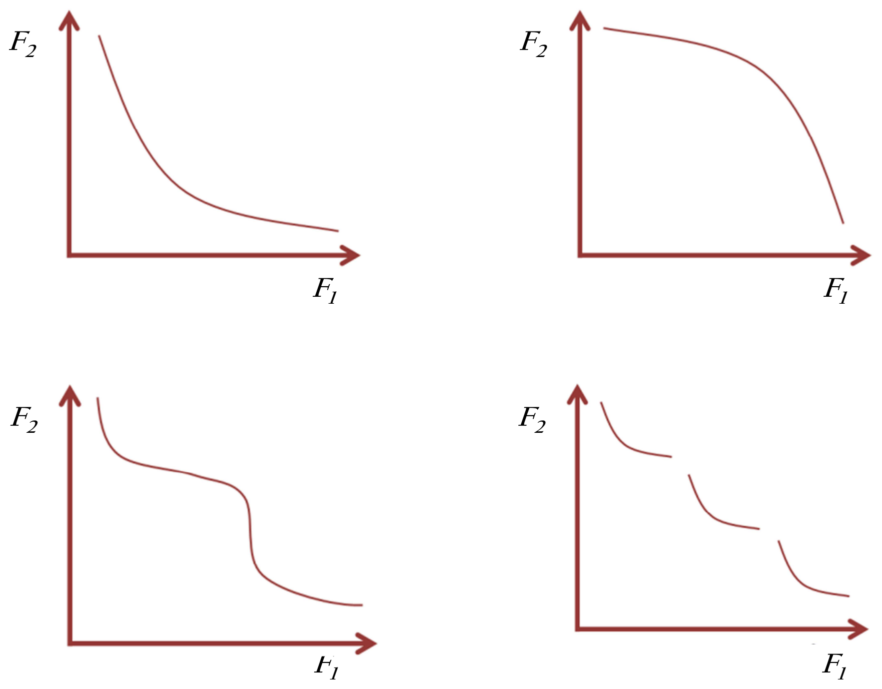


Figure 1.3: Different shapes of Pareto fronts (convex, concave, convex-concave, disconnected).

has a good proximity to the Pareto front but it does not cover it completely, meanwhile (c) has a better coverage but a very bad proximity. Approximation (d) has a very good proximity and good distribution among its points but it does not cover all the Pareto front. In this case the desirable approximation according to proximity, coverage and distribution would be approximation (a).

Since neither X^* nor the Pareto front can typically be computed analytically, the most important task is to numerically detect a finite size approximation of the Pareto front. In this thesis we are particularly interested in a small distance between the final solution A and the Pareto front and a sufficiently good spread.

For the numerical treatment of MOPs specialized EMOAs have been used in the recent past (e.g., [1, 2, 3, 4, 5]). One main reason (among others) is that algorithms of that kind are capable of delivering reliably a finite size approximation A of the Pareto front in one single run of the algorithm, and a desired candidate to guide the search during the evolution of the algorithm is the Δ_p indicator. Reasons for this are that approximations that are good in the Δ_p sense are, roughly speaking, ones those elements are evenly spread along the Pareto front, which is closely related to the terms *spread* and *convergence*. Further, a low Δ_p value gives a clear statement on the approximation quality of A to the Pareto front, see e.g. [8].

The optimal distribution of the elements of A to get a "suitable" approximation of the Pareto front is, however, not completely clear and certainly problem dependent which is e.g. reflected by the numerous performance indicators that exist in the evolu-

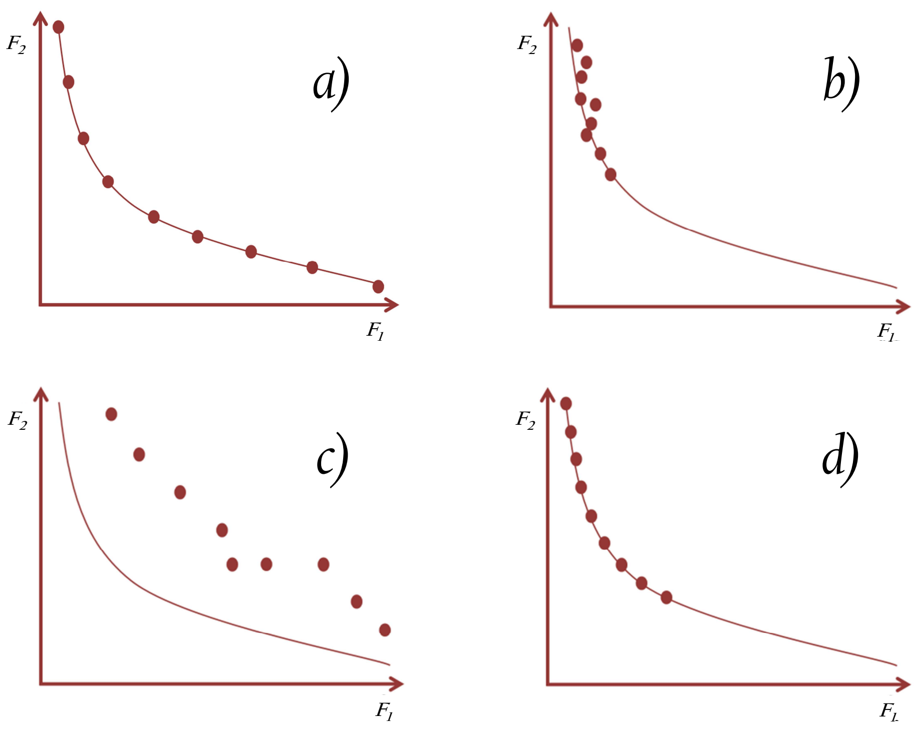


Figure 1.4: Example of final approximations of the Pareto front, represented by the dots.

tionary multi-objective optimization (EMO) community (see, e.g., [15, 16, 6, 7, 17, 18]).

In this thesis, we focus on MOPs with $d = 2, 3$ and 4 objectives since (so far) for every number of objectives involved in the problem a new strategy has to be found to obtain satisfying results.This is due to the fact that the distance of a current point or set to the Pareto front is of course not known but has to be approximated. Since further the dimension of the Pareto front depends on d, the strategies for this approximation change accordingly, at least up to now. Former studies have addressed the problem at hand for $d = 2$ and 3, see [19, 20, 21]. For more than three objectives, these methods can either not be applied (e.g., the polygon approximation of the Pareto front [19] that is restricted to bi-objective problems) or lose their efficiency (such as the multidimensional scaling approach [20]). Here we present two new EMOAs capable of selecting a well-spread subset out of a given data set and will be used for selection in the candidate population set. This together with a mechanism to get a pressure toward the Pareto front will be the basis for the proposed algorithms that aim for both spread and convergence.

1.2 Motivation

In several applications (e.g., in engineering and finance) it is desired that several objectives have to be optimized concurrently leading to a MOP. One important characteristic of a MOP is that its solution set typically forms a $(d-1)$-dimensional object.

Since these sets can in general not be computed analytically the question arises how to compute suitable finite size approximations of them that have to be presented to the decision maker (e.g., [22, 23, 24, 25, 6]).

On the other hand many applications desire an approximation evenly spread along the Pareto front, and when selection takes place for the sake of exploiting convergence and spread, proper selection criteria must be formulated, in order to achieve a balance among these two objectives. Such a balance is not easy to achieve because normally these motivations are contradicting each other. Approximations that are good in the Hausdorff sense are typically evenly spread along the Pareto front, and this is the reason that using the Δ_p indicator as guide during the search is desire to reach such approximations.

To accomplish this task, specialized EAs have caught the interest of many researchers in the recent past (e.g, [1, 5]). Reasons for this include the global approach of these methods, their relatively low assumptions on the model, their high robustness, and that they are capable of delivering a finite size approximation of the entire set of interest in one run of the algorithm.

The PSA selects a diverse subset from a given set of points. This mechanism has a low computational complexity no matter the number of dimensions of the problem is, and it is capable to select a well-spread subset, of any size, even if the original set is poorly distributed. These properties make PSA suitable as a selection mechanism within EMOAs. We proposed the PSEMOA that integrate the PSA into an EA in order to improve its ability to find a well-spread approximation of the Pareto front.

Another form to make a better selection during the evolution is using a performance indicator to do it. Among these methods, indicator based evolutionary algorithms (IBEAs, e.g., [26, 4, 19, 27]) are commonly accepted due to their high performances and since the use of an indicator allows to include user preferences into the computation of the approximation. Since an indicator assigns to each archive (or population) A a value $I(A) \in \mathbb{R}$ an IBEA hence transforms the given MOP (implicitly) into a scalar optimization problem.

One major drawback of IBEAs is that they need quite a few function evaluations to obtain good approximations of the Pareto front. Moreover, it is accepted that these methods, in their current implementations, do not have to converge to the optimal approximation. For instance, for the hypervolumen indicator ([6]) it is known that if only μ children from μ parents (where $|A| = \mu$) and $\lambda < \mu$ offspring are chosen to update the archive in each generation (as e.g. done in [4] with the SMSEMOA for $\lambda = 1$) there is no guarantee for convergence ([28]). Other studies (e.g., [29, 30]) indicate similar results for further performance indicators since the consideration of less than μ elements in one iteration is equivalent to a cyclic search.

Based on this insight, we propose here the selection of $\lambda = \mu$ offspring in each generation which means that the SOP induced by the indicator is used directly. The challenge here is that the problem is lifted into a higher dimensional search space compared to the given MOP. If the problem is addressed via evolutionary strategies, however, the transformation from MOP to SOP comes with the potential that the used

EA may converge, ideally even linearly, to the best solution ([31]).

We present a novel method to compute averaged Hausdorff (Δ_p) approximations of the Pareto fronts for bi-objective problems. It directly utilizes the scalar optimization problem that is induced by the performance indicator Δ_p. This method can be viewed as a certain set based scalarization approach and can be addressed both by mathematical programming techniques and EAs. In this method, we focus on the latter and propose a first single objective EA (Δ_p-EA) for such Δ_p approximations.

1.3 Objectives

The goals of this thesis are as follows:

To design EMOAs capable to efficiently find "good" Hausdorff approximations of the Pareto front for MOPs with 2, 3 and 4 objectives.

1. To analyze the suitability of using PSA as tool to select a well-distributed set of points during the evolution.
2. To compare the new proposal against other state-of-the-art algorithms.
3. To define possible realizations tailored to Δ_p that can be addressed both by mathematical programming techniques as well as evolutionary algorithms.
4. To expand the latter approaches into a first single objective evolutionary algorithm aiming for good Δ_p approximations of the Pareto front for bi-objective problems with different shapes of the Pareto front.
5. To analyze the suitability of this proposal for three-objective problems.
6. To apply the proposed algorithms to a real application.

1.4 Demarcations of scope and key assumptions

In this thesis, we aim for Δ_p approximations since the Δ_p indicator, which measures the averaged Hausdorff distance between the image of the archive and the Pareto front, prefers evenly spread solutions along the Pareto front which comes closest to the terms *spread* and *convergence* as used EMO community. This indicator has the potential drawback that the Pareto front is not known a priori which is needed to compute the indicator value during the run of the algorithm. The latter is a very important issue to solve since Δ_p needs a reference set to be computed. However, if this reference is not "good enough", i.e. the reference is not a "good" approximation of the Pareto front, then we cannot expect good results by using this indicator. If, on the other side, this reference set is defined "good", i.e. the reference is a "good" approximation of the Pareto front, then we can expect very good results. The complexity to define the reference during the evolution of the algorithm increases when the dimensionality of the problem increases, and then the reference set needs different strategies to be generated.

1.5 Contributions

The contributions of this thesis are as follows:

- We present a novel method to compute averaged Hausdorff (Δ_p) approximations of the Pareto fronts of MOPs, utilizing directly the scalar optimization problem that is induced by the performance indicator Δ_p. This method can be viewed as a certain set based scalarization approach and can be addressed both by mathematical programming techniques and EAs.
- It is proposed a first single objective EA (Δ_p-EA) guided directly by the Δ_p indicator, which uses a novel strategy to generate the reference set needed by this indicator for two and three-objective optimization problems.
- We define an EMOA called PSEMOA, which can be used to solve two, three, and four-objective optimization problems, and it is guided by Δ_p and uses the PSA as tool to generate the reference set needed. This algorithm has a low computational complexity even when the number of objectives is increased.
- We applied the PSEMOA to an application which arises in the design of proportional-integral-derivative (PID) controllers.

1.6 Publications

Within this thesis project the following papers where published:

O. Schütze, C. Domínguez-Medina, N. Cruz Cortés, L. G. de la Fraga, J. Q. Sun, G. Toscano-Pulido, R. Landa, **"A Scalar Optimization Approach for Hausdorff Approximations of the Pareto Front"**, in Engineering Optimization, 2015, to appear.

G. Rudolph, O. Schütze, C. Grimme, C. Domínguez-Medina, H. Trautmann, **"Finding Averaged Hausdorff Approximations of Pareto Fronts in 2D"**, in Computational Optimization and Applications, 2015, to appear.

S. Salomon, C. Domínguez-Medina, G. Avigad, A. Freitas, A. Goldvard, O. Schütze, and H. Trautmann, **"PSA Based Multi Objective Evolutionary Algorithms"**, in EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolution- ary Computation III, Eds. Springer International Publishing, 2014, vol. 500, pp. 233-259.

C. Dominguez-Medina, G. Rudolph, O. Schütze, and H. Trautmann. **"Evenly Spaced Pareto Fronts of Quad-Objective Problems Using PSA Partitioning**

Technique". In Evolutionary Computation (CEC), 2013 IEEE Congress on, pp. 3190-3197.

H. Trautmann, G. Rudolph, C. Dominguez-Medina, and O. Schütze. **"Finding Evenly Spaced Pareto Fronts for Three-Objective Optimization Problems"**. In EVOLVE II, edited by O. Schütze et al. 2013. pp. 89-105. Springer: Berlin Heidelberg.

C. Dominguez-Medina, N. Cruz Cortés, J. Q. Sun, H. Trautmann, G. Rudolph, and O. Schütze. **"Computing Evenly Spread Solutions for a Three Objective PID Control Problem"**. In EVOLVE II, edited by O. Schütze et al. 2013. Springer: Berlin Heidelberg.

1.7 Content

Including this introduction, the thesis consists of six chapters. Chapter 2 presents basic definitions and concepts as a background for the following chapters of this document. Chapter 3 is dedicated to the presentation of the state-of-the-art related to understand the rest of the thesis. Next, in Chapter 4, we present the PSEMOA which instead of an approximation of the reference front, the PSA has been used on the candidate population set. The PSA is capable of quickly selecting a "well-spread" subset out of a given data set and serves thus as an alternative to determine the reference set, in particular for problems with more than two objectives, mainly because its computational requirement does not increase significantly when the number of objectives increases. Chapter 5 presents a novel method to compute Δ_p approximations of the Pareto front, which directly utilizes the scalar optimization problem that is induced by the performance indicator Δ_p. This method can be viewed as a certain set based scalarization approach and can be addressed both by mathematical programming techniques and EAs. Here we focus on the latter and propose the Δ_p-EA for such Δ_p approximations. In Chapter 6, we present numerical results of the PSEMOA on a three objective optimization problem related to the design of proportional-integral-derivative (PID) controllers, and results of the Δ_p-EA on the bi-objective optimization problem taking into account only two objectives of the related design of PID controllers optimization problem, and we give our conclusions and possible paths for future research in Chapter 7.

Chapter 2

Background

This chapter introduces a number of background concepts related to the content of this thesis.

2.1 Single-Objective Optimization

A single-objective optimization problem (SOP) represents the minimization or maximization of a real function by searching input values from within an allowed set and computing the value of the function. More generally, optimization includes finding "best available" values of some objective function given a defined domain (and/or a set of constraints). The general mathematical problem is how to choose some variables collected in a vector $x = (x_1, x_2, x_3, ..., x_n)^T$ to minimize or maximize an objective function $F(x)$, often subject to some equation constraints for the type $g(x) \leq 0$ and/or some equality constraints $h(x) = 0$, where $g : \mathbb{R}^m \rightarrow \mathbb{R}$ and $h : \mathbb{R}^p \rightarrow \mathbb{R}$ ([5]).

Definition 1 (General SOP). *A general SOP is defining as minimizing (or maximizing) $F(x)$ subject to $g_i(x) \leq 0$, $i = \{1, ..., m\}$, $g_i : \mathbb{R}^m \rightarrow \mathbb{R}$, and $h_j(x) = 0$, $j = \{1, ..., p\}$, $h_i : \mathbb{R}^p \rightarrow \mathbb{R}$, $x \in \Omega$. A solution minimizes (or maximizes) the scalar $F(x)$ where x is a n-dimensional decision variable vector $x = (x_1, x_2, x_3, ..., x_n)^T$ from some domain Ω.*

The set Ω contains all possible decision variables x that can be used to satisfy an evaluation of $F(x)$ and its constraints. Of course, x can be a vector of continuous or discrete variables as well as F being continuous or discrete.

An optimization method tries to find the global optimum (may not be unique) of F subject to Ω. Figure 2.1 shows graph and the global minimum of a hypothetical one-dimensional unconstrained SOP.

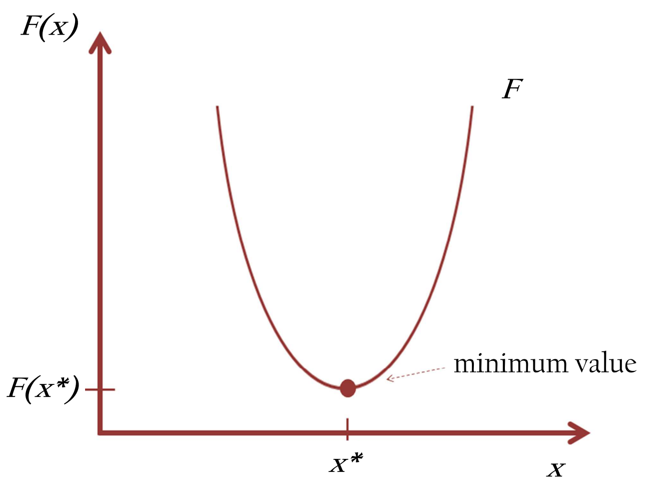


Figure 2.1: Hypothetical SOP.

2.2 Multi-Objective Optimization

Optimization problems with more than one objective are known as multi-objective optimization problems. Normally, these objectives are in conflict each other leading to uncountably many solutions in the solution set, the Pareto set. These solutions represent vectors whose components define a trade-off in the objective functions. A decision maker (DM) has the responsibility to select the "best" solution among this set according to what is desired (i.e., cheaper production, stronger engine, faster performance, etc.).

More generally, a MOP is a problem of finding a set of decision variables $x = (x_1, x_2, ..., x_n)^T$ which satisfies constraints and optimizes a set of objective functions $F = (F_1, F_2, ..., F_d)^T$.

Continuous MOPs, as we consider here, can be stated as

$$\min_{x\in\Omega}\{F(x)\}, \tag{2.1}$$

where $\Omega \subset \mathbb{R}^n$ is the domain and $F : \Omega \to \mathbb{R}^d$ is defined as the vector of the objective functions $F(x) = (F_1(x), \ldots, F_d(x))^T$. For simplicity we assume that each objective $F_i : \Omega \to \mathbb{R}$ is sufficiently smooth.

In the same way as far SOP, it is possible that a MOP has constraints which are imposed by the particular characteristics of the environment or available resources (e.g., physical limitations, time restrictions, economic founding, etc.). These restrictions must be satisfied in order to consider a certain solution acceptable under the real conditions. In general these constraints describe dependencies among decision variables and constants (or parameters) involved in the problem. These constraints are known as inequalities:

$$g_i(x) \leq 0 \qquad i = 1, ..., m, g_i : \mathbb{R}^m \to \mathbb{R}, \tag{2.2}$$

or equalities:

$$h_j(x) = 0 \qquad j = 1, ..., p, h_i : \mathbb{R}^p \to \mathbb{R}. \tag{2.3}$$

The set of objective functions F gives a value of how "good" a solution x is. In real-world problems, some functions are in conflict with others, and some must be minimized while others must be maximized. These functions may be measured in the same units or not. The multiple objectives being optimized almost always conflict, leading to a partial, rather than a total, ordering on the search space [32].

2.3 Basic Concepts of Multi-Objective Optimization

For this kind of problems it is not a trivial task to define which solutions are the "best" or "optimal", because we can have only a partial order between solutions. In order to solve this problem a comparison strategy between solutions has to be defined. The most common way to compare two different solutions into a MOP is using the concept of *Pareto dominance* [33]. It allows us to define which are the optimal solutions within the domain Ω. The Pareto dominance is defined as follows:

(a) Let $v, w \in \mathbb{R}^d$. Then the vector v is *less than* w ($v <_p w$), if $v_i < w_i$ for all $i \in \{1, \ldots, d\}$. The relation $\leq_p$ is defined analogously.

(b) A vector $y \in \Omega$ is *dominated* by a vector $x \in \Omega$ ($x \prec y$) with respect to (2.1) if $F(x) \leq_p F(y)$ and $F(x) \neq F(y)$, else y is called non-dominated by x.

(c) A point $x \in \Omega$ is called *(Pareto) optimal* or a *Pareto point* if there is no $y \in \Omega$ which dominates x.

(d) The set X^* of all Pareto optimal solutions is called the *Pareto set* and its image $F^* = F^*(X^*)$ the *Pareto front.* Figure 2.2 shows the Pareto fronts of a continuous and a discrete MOP.

Table 2.1 shows the different dominance relations between two solutions ($F(x_1)$, and $F(x_2)$) in the objective space and between two approximation sets of the Pareto front (A and B).

Figure 2.3 shows the comparison between several vectors for a bi-objective problem (assuming minimization). In this example, in the objective space, point u dominates

Table 2.1: Dominance relations between solutions and approximation sets of the Pareto front.

Relation	**Objective Solutions**	
Strict dominance	$F(x_1) \prec\prec F(x_2)$	$\forall_{i \in F}$, $F_i(x_1)$ is better than $F_i(x_2)$
Dominance	$F(x_1) \prec F(x_2)$	$F(x_1)$ is not worse than $F(x_2)$ in all objectives and better in at least one objective
Weakly Dominates	$F(x_1) \preceq F(x_2)$	$F(x_1)$ is not worse than $F(x_2)$ in all objectives
Incomparability	$F(x_1) \| \| F(x_2)$	neither $F(x_1)$ weakly dominates $F(x_2)$ nor $F(x_2)$ weakly dominates $F(x_1)$
Indifference	$F(x_1) \sim F(x_2)$	$F(x_1)$ has the same value $F(x_2)$ in each objective
Relation	**Approximation Sets**	
Strictly Dominates	A $\prec\prec$ B	Every b $\in$ B is strictly dominated by at least one a $\in$ A
Dominates	A $\prec$ B	Every b $\in$ B is dominated by at least one a $\in$ A
Weakly Dominates	A $\preceq$ B	Every b $\in$ B is weakly dominated by at least one a $\in$ A
Incomparable	A $\| \|$ B	neither A weakly dominates B nor B weakly dominates A
Indifferent	A $\sim$ B	A weakly dominates B and B weakly dominates A

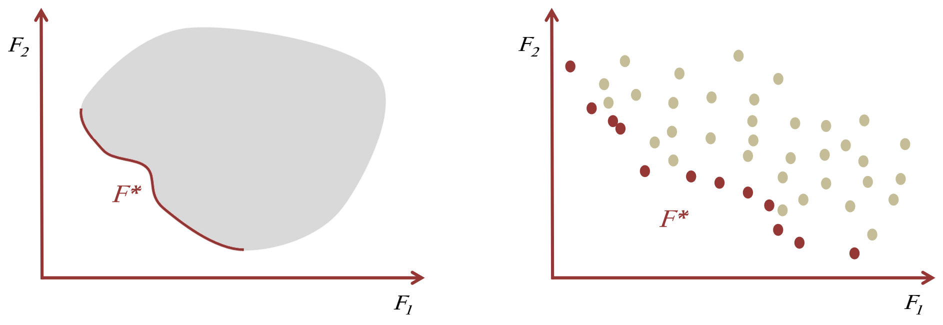


Figure 2.2: Continuous (left) and discrete (right) Pareto fronts (F^*) of hypothetical bi-objective problems.

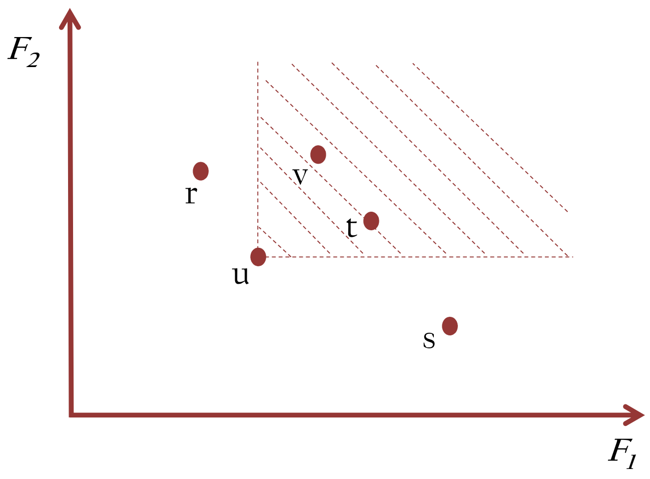


Figure 2.3: Pareto dominance.

v ($u \prec v$) because its both objective values are smaller than the values of v. If we compare points u and s we have that u is better according to F_1 but it is worst for F_2 then $u \not\prec s$. In case $u \not\prec s$ and $s \not\prec u$ then both solutions u and s are called *incomparable* between each other.

Since the complete Pareto front cannot be computed analytically (except for trivial examples), it is common to accept a finite size approximation to the Pareto front in order to select the "best" solution. The question arises how to compute a suitable finite size approximation of the Pareto front.

2.4 Evolutionary Multi-Objective Optimization Algorithms

In order to solve MOPs it is necessary to define numerical methods that compute suitable finite size approximations of the Pareto front. EMOAs have caught the interest of many researchers in the recent past since they accomplish this task outstandingly (e.g., [1, 2, 3, 4, 5]) and which are capable of computing an entire set of candidate solutions within one run of the algorithm. The EMOAs are the multi-objective approach of the evolutionary algorithms which are described as follows.

2.4.1 Evolutionary Algorithms (EAs)

EAs provide an alternative to traditional optimization techniques by using directed random searches to locate optimal solutions in complex landscapes [34].

In nature, the best individuals suited to competition for limited resources survive. Adapting to environment changes is essential for the survival of individuals of each species. While the various features that uniquely characterize an individual determine its survival capacity. the features in turn are determined by the individuals genetic content, specifically, each feature is controlled by a basic unit called a *gene*. The set of genes controlling features form the *chromosomes*, which are the "keys" to the survival of the individual in a competitive environment.

Evolution represents a succession of changes in features of the species, it is the changes in the species genetic material that form the essence of evolution. Specifically, evolution drives the joint action of natural selection and the recombination of genetic material that occurs during reproduction.

In nature the competition among individuals for limited resources such as food and/or space results in the fittest individuals dominating over weaker ones. Only the fittest individuals survive and reproduce, a natural phenomenon called "the survival of the fittest". Hence, the genes of the fittest survive to the next generation, while the genes of weaker individuals die out. Natural selection leads to the survival of the fittest individuals, but it also implicitly leads to the survival of the fittest genes.

The reproduction process generates *diversity* in the "gene pool". Evolution starts when the genetic material (*chromosomes*) from two parents recombines during reproduction. New combinations of genes are generated from previous ones and then a new gene pool is created. The exchange of genetic material among chromosomes is called *crossover*. Segments of the two parent chromosomes are exchanged during crossover, creating the possibility of a "better" combination of genes for the next individuals. Repeated selection and crossover cause the continuous evolution of the gene pool and the generation of individuals that survive better in a competitive environment.

In the early 1970s [35] an evolutionary algorithm was proposed as a computer program that mimics the evolutionary process in nature. This algorithm manipulates a population of potential solutions to an optimization (or search) problem. It operates on encoded representations of the solutions, which are equivalent to the genetic material of individuals in nature, and not directly on the solutions themselves. This algorithm encodes the solutions as strings of bits from a binary alphabet. As in nature, selection provides the necessary driving mechanism for better solutions to survive. Each solution is associated with a fitness value that reflects how good it is, compared with other solutions in the population. The higher the fitness value of an individual, the higher its chances of survival and reproduction and the larger its representation in the subsequent generation. Recombination of genetic material in evolutionary algorithms is simulated through a *crossover* mechanism that exchanges portions between strings. Another operation, called *mutation*, causes sporadic and random alteration of the bits of strings. Mutation too has a direct analogy from nature and plays the role of regenerating lost genetic material.

2.4.2 Simple Evolutionary Algorithm Structure

The simplest EA evolves a population of binary strings (solutions). Each string of 0s and 1s is the encoded version of a solution to the optimization problem. Using genetic operators *crossover* and *mutation* the algorithm creates the subsequent generation from the solutions of the current population. This generational cycle is repeated until a desired termination criterion is reached (for example. a predefined number of generations are processed).

Algorithm 1 summarizes the process of a simple EA which has the following components:

- A population of binary strings,
- control parameters,
- a fitness function,
- genetic operators (crossover and mutation),
- a selection mechanism, and
- an encoding mechanism to encode the solutions as a defined representation.

Algorithm 1 Simple EA Structure.

```
1: Initialize population $P$.
2: Evaluate all solutions $p \in P$ according to the objective function $F$.
3: while Stop criterion is fulfilled do
4:     Select best solutions for the next population.
5:     Perform crossover and mutation operators.
6:     Evaluate new solutions according to the objective function $F$.
7: end while
```

Encoding mechanism. Fundamental to the EA structure is the encoding mechanism for representing the optimization problem variables. The encoding mechanism depends on the nature of the problem variables. For example, when we are solving the optimal flows in a transportation problem, the variables (flows in different routes) assume continuous values, while the variables in a traveling sales man problem are binary quantities representing the inclusion or exclusion of an edge in the circuit.

Fitness function. The objective function, the function to be optimized, provides the mechanism to evaluate each solution. The value of the objective function is the fitness of the solution, which the selection mechanism used to evaluate the solutions of the population. In case of constrained optimization problems, the most common method is to use penalty functions that penalize infeasible solutions by reducing their fitness values in proportion to their quality or number of constraint violations ([36]).

Selection. Selection models from nature, survival of the fittest individual. Fitter solutions survive while weaker ones perish. A fitter solution receives a higher number of offspring and thus has a higher chance of surviving in the subsequent generation. In a proportionate selection scheme, a solution with a fitness value higher than the average is allocated more than one offspring, while a string with a fitness value less than the average is allocated less than one offspring. Sometimes is commonly used a random selection of parents in order to generate new offspring.

Crossover. After selection comes the crossover operation. Pairs of solutions are picked at random from the population to be subjected to crossover. A simple EA uses the simplest approach single-point crossover. Assuming that l is the solution length, it randomly chooses a crossover point that can assume values in the range 1 to l - 1. The portions of the two solutions beyond this crossover point are exchanged to form two new strings. The crossover point may assume any of the l - 1 possible values with equal probability. Further, crossover is not always effected. After choosing a pair of strings, the algorithm invokes crossover only if a randomly generated number in the range 0 to 1 is greater than a value called the crossover probability. Otherwise the solutions remain unaltered.

One example of crossover is the simulated binary crossover (SBX) operator ([1]). It has search power similar to that of a single-point binary-coded crossover operator. The search power was defined as the ability to create any arbitrary child solution from two parent solutions. Based on a derived probability distribution of creating a child solution in the single-point crossover operator, a similar probability distribution was used directly to choose a child solution in SBX. The difference in the implementations of the real-coded GAs wit SBX and binary-coded GAs with single-point crossover is that in SBX the coding of variables is eliminated and a child string is created from a probability distribution that depends on the location of the parent strings.

The SBX uses a spread factor property, it is the probability of occurrence of spread factor $\beta \approx 1$ is more likely than any other β value. If $\beta < 1$ we have a contracting crossover (i.e., the offspring points are enclosed by the parents points) as in a) in Figure 2.4, if $\beta > 1$ we have a expanding crossover (i.e., the offspring points enclose by the parent points) as in b), and if $\beta = 1$ there is a stationary crossover (i.e., the offspring

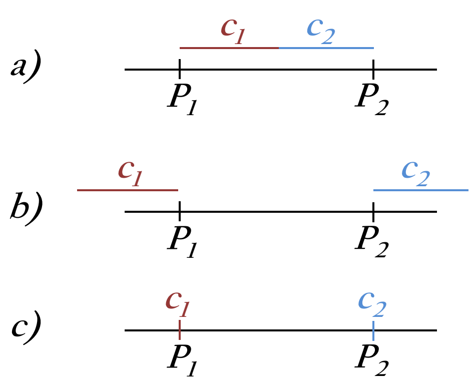


Figure 2.4: Spread factor β.

points are the same as parent points) as it is shown in c).

Mutation. After crossover the solutions are subjected to mutation. Mutation of a bit involves flipping it: changing a 0 to 1 or a 1 to 0. Just as we have a crossover probability, we need another parameter, the mutation probability. It gives the probability that a bit will be flipped. The bits of a solution are independently mutated (the mutation of a bit does not affect the probability of mutation of other bits). A simple EA treats mutation only as a secondary operator with the role of restoring lost genetic material. For example, suppose all the solutions in a population have converged to a 0 at a given position and the optimal solution has a 1 at that position. Then crossover cannot regenerate a 1 at that position. while a mutation could.

One example is the polynomial mutation operator ([37, 38]) with a user-defined index parameter (η_m). η_m induces an effect of a perturbation of $O((b-a)/\eta_m)$ in a variable, where a and b are lower and upper bounds of the variable x_i. $\eta_m \in [20, 100]$ is adequate in most problems. In this operator, a polynomial probability distribution is used to perturb a solution in a parents vicinity. The probability distribution in both left and right of a variable value is adjusted so that no value outside the specified range $[a, b]$ is created by the mutation operator. For a given parent solution $p \in [a, b]$, the mutated solution p' for a particular variable is created for a random number u created within $[0, 1]$, as follows:

$$p' = \begin{cases} p + \delta_L(p - x_i^{(L)}), & \text{for} \quad u \le 0.5, \\ p + \delta_R(x_i^{(U)} - p), & \text{for} \quad u > 0.5. \end{cases} \tag{2.4}$$

Then, either of the two parameters (δ_L or δ_R) is calculated, as follows:

$$\delta_L = (2u)^{1/(1+\eta_m)} - 1, \text{ for } \ u \le 0.5, \tag{2.5}$$

$$\delta_R = 1 - (2(1-u))^{1/(1+\eta_m)}, \text{ for } \ u > 0.5. \tag{2.6}$$

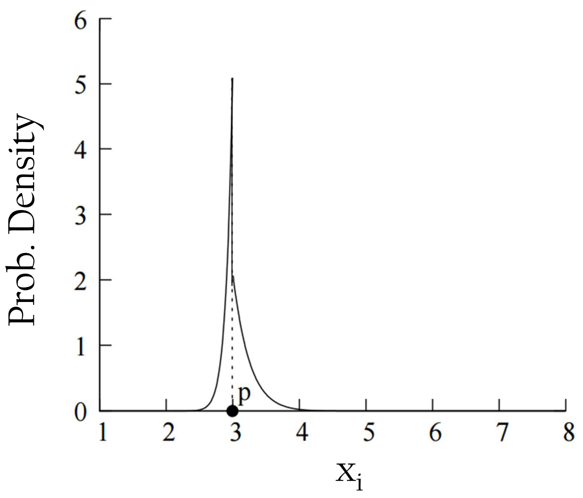


Figure 2.5: Probability density function of creating a mutated child solution using polynomial mutation operator.

Figure 2.5 shows the probability density of creating a mutated child point from a parent point $p = 3.0$ in a bounded range of $[1, 8]$ with $\eta_m = 20$.

There have been three main independent implementation instances of EAs: Genetic algorithms (GA), developed by Holland in [35] and thoroughly reviewed by Goldberg in [39], evolution strategies (ESs), developed in Germany by Rechenberg in [40] and Schwefel in [41] and evolutionary programming (EP), originally developed by Fogel et al. in [42]. Each of these three algorithms has been proved capable of yielding approximately optimal solutions given complex, multi-modal, non-differential, and discontinuous search spaces. Success has also been achieved for noisy and time-dependent landscapes.

2.4.3 Genetic Algorithms

Algorithm 2 shows the GA as developed by Holland. The GA encodes the problem within binary string individuals. Evolutionary pressure is applied in Line 3, where the stochastic technique of roulette wheel parent selection is used to pick parents for the new population. The concept is illustrated in Figure 2.6, using a trivial example with a population of four individuals. Each individual is assigned a sector of a roulette wheel that is proportional to its fitness and the wheel is spun to select a parent. While selection is random and any individual has the capacity to become a parent, selection is clearly biassed towards fitter individuals. Parents are not required to be unique and, in each iteration, fit individuals may produce many offspring.

From a population of size μ, $\mu/2$ pairs of parents are chosen. These parents form a new population. With probability P_c each pair is recombined using the crossover operator to produce a pair of children. This cut and splice operator is illustrated

Algorithm 2 Simple GA Structure.

```
1: A population of $\mu$ random individuals is initialized.
2: Fitness scores are assigned to each individual.
3: Using roulette wheel parent selection $\mu/2$ pairs of parents are chosen from the
   current population to form a new population.
4: With probability $P_c$, children are formed by performing crossover on the $\mu/2$ pairs
   of parents. The children replace the parents in the new population.
5: With probability $P_m$, mutation is performed on the new population.
6: The new population becomes the current population.
7: If the termination conditions are satisfied exit, otherwise go to Line 3.
```

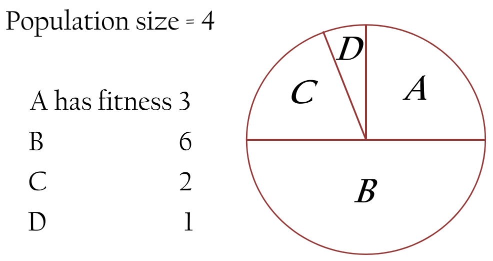


Figure 2.6: Roulette wheel parent selection.

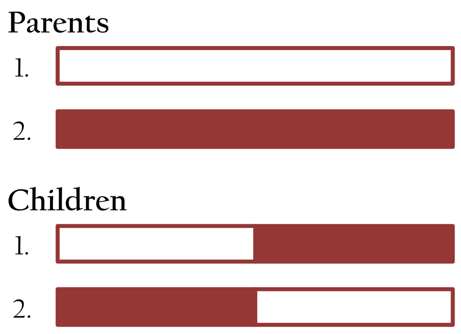


Figure 2.7: GA crossover operator.

in Figure 2.7. A cross point is selected at random. Each child is identical to one parent before the cross point and identical to the other after the cross point. The child individuals then replace their parents in the new population. Following crossover, mutation is applied to all individuals in the new population. With probability P_m, each bit on every string is inverted. The new population then becomes the current population and the cycle is repeated until some termination criteria are satisfied. The algorithm typically runs for some fixed number of iterations, or until convergence is detected within the population. The probabilities of mutation and crossover, P_m and P_c are parameters of the algorithm and must be set by the user.

Many GAs applied to real world problems bear only a passing resemblance to the GA, and GAs are best viewed as a branch for evolutionary search, rather than a specific algorithm. The binary encoding is often inappropriate for many problems and may be extended to nonbinary representations (i.e., integer string individuals or even more general representations such as tree and matrix structures).

2.4.4 Evolutionary Strategies

Algorithm 3 shows the basic structure of the ES, they were designed for parameter optimization problems. The encoding used in an individual is therefore a list of real numbers. These are called the object variables of the problem. Additionally, each individual contains a number of strategy parameters, and they are normally used to control the behavior of the mutation operator and are not required when decoding an individual.

In each iteration λ offspring are generated from a population of size μ. The recombination operator produces one child and requires two parents for each object variable and strategy parameter in the child. Historically, the same parents are used to generate all object variables in the child, then the parents are re-selected for each strategy parameter. The parents are selected randomly from the current population (i.e., there is no selection pressure at this point). A number of alternative recombination techniques are available, but the best results have been observed by setting each object variable

Algorithm 3 Simple ES Structure.

```
1: A population of $\mu$ random individuals is initialized.
2: Fitness scores are assigned to each individual.
3: $\lambda$ new offspring are generated by recombination from the current population.
4: The $\lambda$ new offspring are mutated.
5: Fitness scores are assigned to the $\lambda$ new offspring.
6: A new population of $\mu$ individuals is selected, using either $(\mu, \lambda)$ or $(\mu+\lambda)$ selection.
7: The new population becomes the current population.
8: If the termination conditions are satisfied exit, otherwise go to Line 3.
```

in the child to be the same as the object variable in one of the parents and setting each strategy parameter in the child to be the mean of the parameters values in the parents. Mutation is the main operator in the ES and acts upon strategy parameters as well as object variables. The mutation operator first perturbs the strategy parameters. The object variables are then mutated using the resulting probability distribution defined by the modified strategy parameters. This special mutation operator allows the ES to evolve good strategy parameters for the problem and has been termed self-adaptation. Selection in ESs is deterministic: the best μ individuals are taken from the λ new offspring ((μ, λ) selection) or from the union of μ parents and λ offspring ($(\mu + \lambda)$ selection). The preferred method is (μ, λ) selection, since $(\mu + \lambda)$ selection can disrupt the self-adaptation mechanism ([43]).

2.4.5 Evolutionary Programming

Algorithm 4 illustrates the form of an EP scheme. EP was originally developed for the evolution of finite state machines using a limited symbolic alphabet encoding. Subsequently the EP was extended to encode real numbers, thus providing a tool for variable optimization. Individuals in the EP comprise a string of real numbers, as in ESs. EP differs from GAs and ESs in that there is no recombination operator. Evolution is totally dependent on the mutation operator, which uses a Gaussian probability distribution to perturb each variable. The standard deviations correspond to the square root of a linear transform of the parents fitness value.

Algorithm 4 Simple EP Structure.

```
1: A population of $\mu$ random individuals is initialized.
2: Fitness scores are assigned to each individual.
3: The mutation operator is applied to each of the $\mu$ individuals in the current pop-
   ulation to produce $\mu$ offspring.
4: Fitness scores are assigned to the $\mu$ new offspring.
5: A new population of size $\mu$ is created from the $\mu$ parents and the $\mu$ offspring using
   tournament selection.
6: If the termination conditions are satisfied exit, otherwise go to Line 3.
```

Selection pressure is applied in the EP when forming a new population from parents and offspring of the mutation operator, using a mechanism called tournament selection.

Stochastic q-tournament selection is employed, where q is a parameter of the algorithm. Let U be the union of all parents and offspring. For each member $m \in U$, q opponents are selected from U at random. A count is then made of the number of opponents that have worse fitness values than m. The μ individuals with the highest tournament counts go on to form the new population. Note that as q increases the selection pressure in the algorithm increases and the selection process becomes increasingly deterministic. One side-effect of this selection process is that the best individual is always present in the new population.

2.4.6 Evolutionary Algorithm with a Multi-Objective Approach

EMOAs seem suitable to solve MOPs, because they evolve at the same time a set of possible solutions. This allows to find several members of the Pareto optimal set in a single run of the algorithm, instead of having to perform a series of separate runs as in the case of the traditional mathematical programming techniques [44, 1, 5, 32]. Additionally, EAs are less susceptible to the shape or continuity of the Pareto front (e.g., they can easily deal with discontinuous or concave Pareto fronts), whereas these two issues are serious concerns for mathematical programming techniques. EMOAs are then very attractive MOP solution techniques because they address both search and multi-objective decision making.

The first implementation of what it is now called an EMOA is credited to David Schaffer, who proposed the Vector Evaluation Genetic Algorithm (VEGA), in 1984. VEGA was mainly aimed for solving problems in machine learning [45, 46].

When we are solving a MOP we can also use a kind of modified GAs, from the EMOAs. EMOAs are based on GAs and their structure is very similar with some critical differences. An EMOAs main characteristic is the set of multiple objectives being simultaneously optimized ($F = (F_1(x), F_2(x), ..., F_d(x))^T$). If this multi-objective fitness function F is substituted for the fitness function in Algorithm 1 then this new approach is a multi-objective evolutionary algorithm.

The task decompositions of the algorithms depicted in Figure 2.8 and Figure 2.9 show a little structural difference between the EMOA and its single-objective EA counterparts. The major differences are noted as follows. By definition, the objective function evaluation task in the EMOA case computes d (where $d \geq 2$) function evaluations. In addition, because EMOAs expect a single fitness value with which to perform selection, additional processing is required to transform EMOA solutions objective vectors into a scalar ("objective transformation" task). Although the various transformation techniques vary in their algorithmic impact. The remainder of the EMOA is structurally identical to its single-objective counterpart. However, this does not imply the differences are insignificant.

In some cases it is possible to use an external archive which improves the selection of the better individuals (normally based on an indicator). These methods work with a normal internal population and add an external archive. This external archive uses a different strategy to select the new individuals and it could be used for evolving the

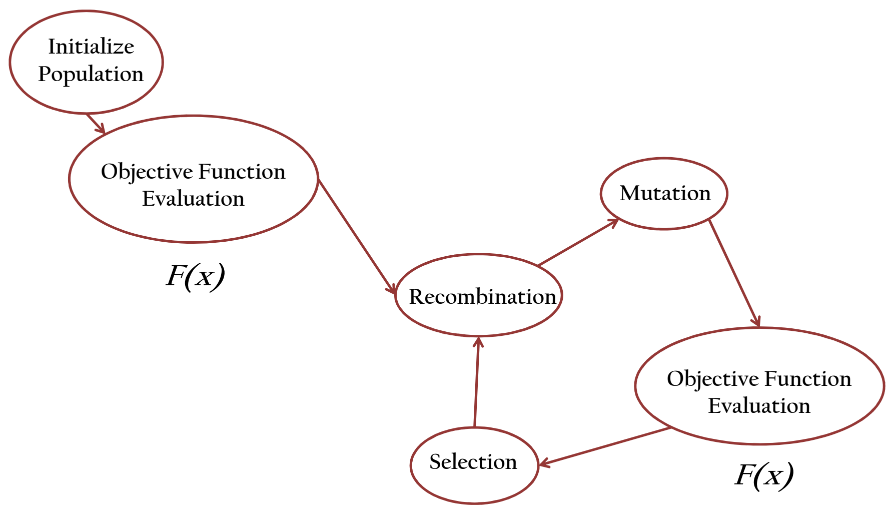


Figure 2.8: General EA task decomposition.

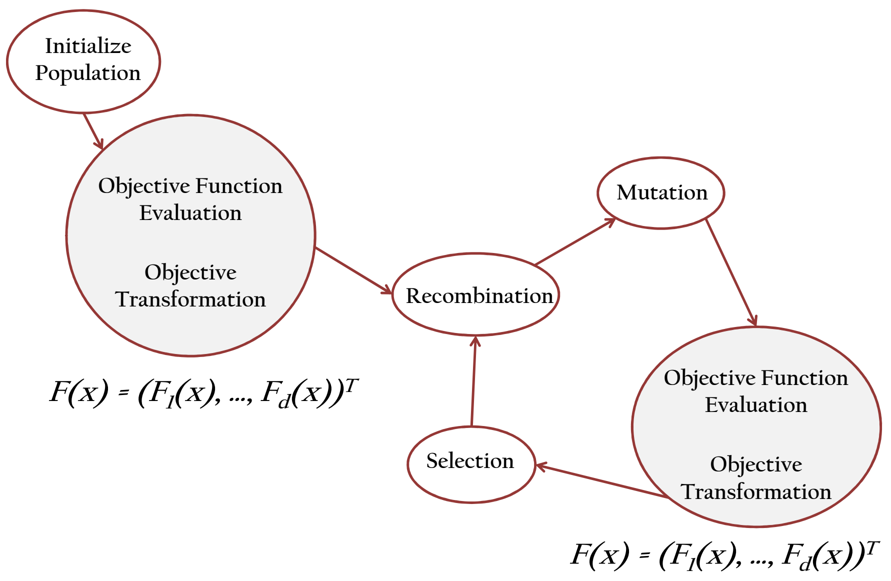

Initialize Population
Objective Function Evaluation
Objective Transformation
$F(x) = (F_1(x), ..., F_d(x))^T$
Mutation
Recombination
Objective Function Evaluation
Objective Transformation
Selection
$F(x) = (F_1(x), ..., F_d(x))^T$

Figure 2.9: General EMOA task decomposition.

internal population too. Information extracted from the external archive could be used to decide which search regions should be searched at each generation as in [47] or to generate a feasible reference set used by itself to select the new points as in [48, 49]. In such a way, the external archive strategy and the current population of the evolution can complement each other.

We have a set of tools to solve MOPs, in other words we are able to get finite size approximations of the Pareto front, but how “good” are the approximations we obtain? Several quality indicators exist into this field, which try to determine the approximation quality with one single value.

2.5 Quality Indicators

We already know a way to compare solutions between each other, the Pareto dominance, but what about the final approximations quality obtained when solving a MOP?

While visual comparisons were common in the infancy of evolutionary multi-objective optimization, quantitative performance assessment is now becoming more standard. However, no guidelines are available on how to compare the quality of the outcomes generated by several multi-objective optimizers, over several runs, to obtain quantitative and statistically sound inferences. As a consequence, most comparative studies are based on different methodologies ans assumptions and therefore the results are difficult to relate to one other.

To compare the complete set of solutions of two different approximations of the Pareto front is not straightforward. There are some studies like those of Zitzler et al. [50], Knowles et al. [51] or more recently by Schütze et al. [52], which are examples of the effort being carried out in order to provide the necessary tools for a better evaluation and comparison of multi-objective algorithms [53].

The main goal of testing is usually to compare EMOAs effectiveness over various chosen MOPs by measuring solution quality. The indicators usually fall into two performance categories [5]: **1) Efficiency** (measuring computational effort to obtain solutions, e.g., CPU time, number of function evaluations/iterations - use of spatial and temporal resources), and **2) Effectiveness** (measuring the accuracy and convergence of obtained approximations), which is the scope of this thesis.

A more formal definition of a quality indicator follows:

Definition 2 (Indicator)**.** *A quality indicator is a function $I : \Omega^{N \cdot n} \rightarrow \mathbb{R}$ which assigns to an approximation A a real value $I(A)$. N is the cardinality of A and n is the dimension of the MOP.*

Definition 3 (Pareto Compliant Indicator)**.** *An indicator $I : \Omega \rightarrow \mathbb{R}$ is Pareto compliant if for all $A, B \in \Omega : A \preceq B \Rightarrow I(A) \geq I(B)$, assuming that greater indicator values correspond to higher quality (otherwise $A \preceq B \Rightarrow I(A) \leq I(B)$).*

Using quality indicators does not intent to indicate that one EMOA is better or more robust than another one, but to describe their general experimental results. In fact, different EMOA performances are directly associated with the specific operators each EMOA employs. Some quality indicators are presented in the following.

2.5.1 Hypervolume

Since the solution of a multi-objective problem is a set of incomparable elements instead of a single value from a totally ordered set there is no straightforward approach for a comparison of solutions. A commonly accepted measure [26] for assessing the quality of an approximation is the so-called dominated hypervolume of a population. In case of two dimensions it is defined as follows:

Definition 4 (Dominated Hypervolume)**.** *Let $v^{(1)}, v^{(2)}, ..., v^{(\mu)} \in \mathbb{R}^2$ be an antichain in lexicographical order and $r \in \mathbb{R}^2$ such that $v^{(i)} \prec r$ for all $i = 1, ..., \mu$. The value*

$$H(v^{(1)}, ..., v^{(\mu)}; r) = \left[r_1 - v_1^{(1)}\right] \cdot \left[r_2 - v_2^{(1)}\right] + \sum_{i=2}^{\mu} \left[r_1 - v_1^{(i)}\right] \cdot \left[v_2^{(i-1)} - v_2^{(i)}\right] \tag{2.7}$$

is termed the dominated hypervolume with respect to r.

This indicator has a number of appealing properties [54] so that there exist some EAs which accept offspring only if the dominated hypervolume is increased by including the offspring ans discarding another individual from the population [55]. Evidently, these EMOAs are aiming at generating a population whose objective vectors maximize the dominated hypervolume.

The hypervolume looks at the multidimensional “volume” created by each set, relative to a reference point. Such volume is illustrated by the red space on Figure 2.10. it cannot really capture good hypervolume unless you have almost every point, across the full spread of objective function performance. One drawback is that the hypervolume is computationally intensive (when the number of objective increases the computationally effort increases too). Figure 2.10 shows an example of the hypervolume of a set A of points respect to the reference point r.

2.5.2 Generational Distance

The generational distance indicator [17] gives a good first approximation of the quality of an approximation set to another reference set, i.e., how “close” is the approximation set to this reference set. It is computed as follows:

$$GD(A, R) = \frac{1}{|A|} \left(\sum_{a \in A} d(a, R)^p \right)^{1/p}, \tag{2.8}$$

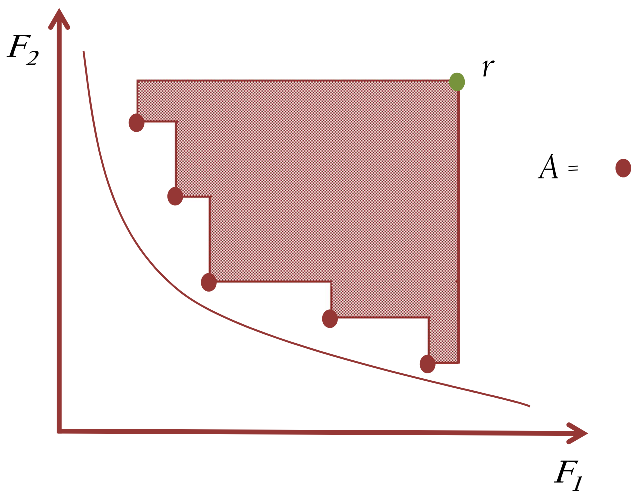


Figure 2.10: Hypervolume of a set of points A respect to the reference point r.

where $d(a, R) = \inf\{\|a - r\| : r \in R\}$, i.e., the minimum distance between a and R.

One problem with this indicator is that if A contains only one point a such as $a \in R$, then it gets almost perfect generational distance performance, forgetting the distribution and coverage of the final solution. The red lines in Figure 2.11 show a calculation of distances. For each of the red points, find the closest point in the reference approximation R. The indicator is then the "average" of these distances.

2.5.3 Inverted Generational Distance

The inverted generational distance [18] is the "average" distance from each reference point to its nearest solution from an approximation A.

$$IGD(A, R) = \frac{1}{|R|} \left(\sum_{r \in R} d(r, A)^p \right)^{1/p}, \tag{2.9}$$

where $d(r, A) = \inf\{\|r - a\| : a \in A\}$, i.e., the minimum distance between r and A.

When a set of well-distributed reference points over the entire reference set R is used, a small value of this indicator suggests the convergence of solutions of A. This indicator has the same problem than the generational distance indicator, because if R contains only one point r such as $r \in A$, then it gets almost perfect generational distance performance, forgetting the distribution and coverage of A. Figure 2.12 shows how now the points from the reference set R are thoses used to compute the minimum distances between A and R.

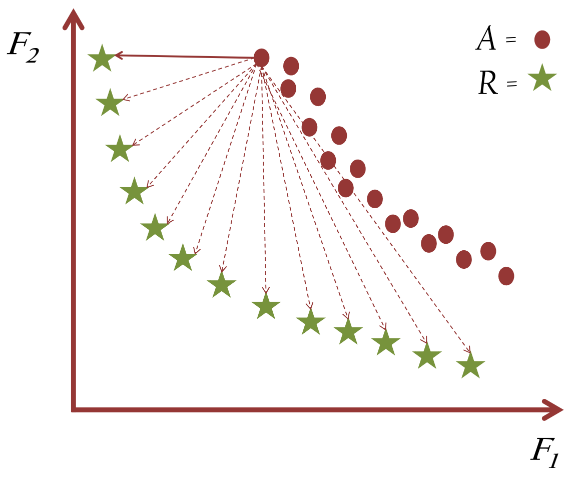


Figure 2.11: Generational distance indicator between two sets.

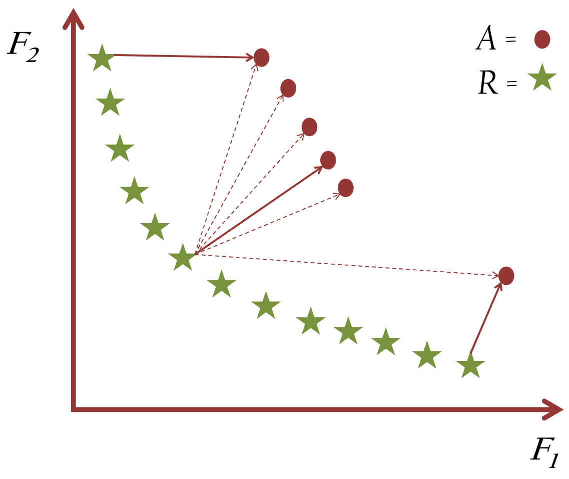


Figure 2.12: Inverted generational distance indicator between two sets.

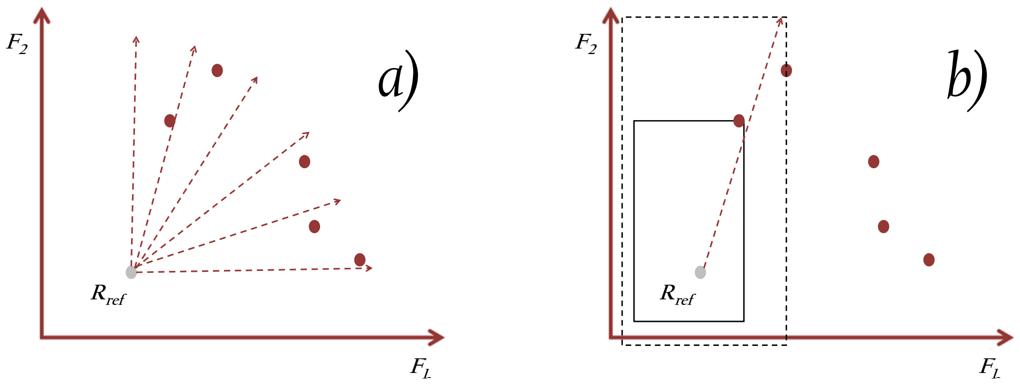


Figure 2.13: a) and b) illustrate how the functions of R$_2$ and R$_3$ are rendered. a) illustrates how the vectors are evenly spread out from the worst reference point to the best reference point. b) illustrates how the difference is calculated with respect to each vector.

2.5.4 R$_R$

The function $u(\lambda, A)$ is the minimum distance of the points in the set A to the reference point λ. The R$_2$ and R$_3$ indicators are mathematically defined as follows.

$$R_2 = \frac{\sum\limits_{\lambda \in A} u(\lambda, B) - u(\lambda, A)}{|\lambda|} \tag{2.10}$$

$$R_3 = \frac{\sum\limits_{\lambda \in A} \big[u(\lambda, B) - u(\lambda, A)\big]/u(\lambda, B)}{|\lambda|} \tag{2.11}$$

Figure 2.13 illustrates the R$_2$ and R$_3$ indicators. These functions u require a reference point and a user-specified number of scalarizing vectors, λ. Vectors are uniformly distributed across the objective space. The distance of the point (in each set) that is closest to the reference point is measured and the differences in these distances are added up. In order to obtain an indicator from these two indicators, the set, B, is replaced with a reference set containing the true Pareto front points, R. These indicator functions then effectively measure the difference in the mean distance of the attainment sets A and R from a user-defined reference point.

2.5.5 The Hausdorff Distance

To measure the distance of two sets the Hausdorff distance d_H is widely used in many fields. It is computed as follows:

$$d_H(A, R) = \max\Big[\mathrm{d}(A, R), \mathrm{d}(R, A)\Big], \tag{2.12}$$

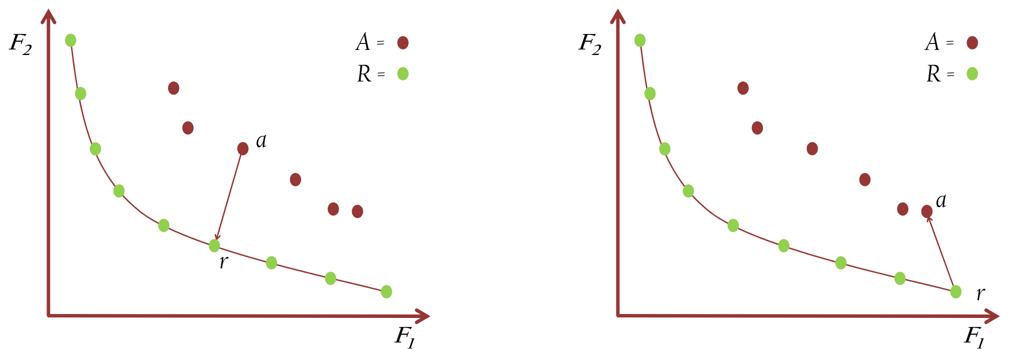


Figure 2.14: Hausdorff distance indicator between two sets.

where $\mathrm{d}(R, A) = \sup\Big\{\mathrm{d}(r, A) : r \in R\Big\}$, and $d(r, A) = \inf\Big\{\|r - a\| : a \in A\Big\}$.

Figure 2.14 shows the distances between two points $a \in A$ and $r \in R$, and the other way around.

It is, however, of limited practical use when measuring the distance of the outcome of an EMOA to the Pareto front since outliers generated by EMOAs, i.e., points far away from the remaining ones and the Pareto front, are punished too strongly by d_H. As a remedy, the *averaged* Hausdorff distance has been proposed in [8] which we use in this thesis and it is presented in detail in the following.

2.5.6 *Averaged* Hausdorff Distance (Δ_p)

Recently, it has been proposed to use the averaged Hausdorff distance Δ_p as an alternative performance indicator. Using Δ_p, the decision maker can get a clear information about the approximation quality in terms of the distance from an approximation A to a reference set R (which is typically termed as convergence in the EMO literature) as well as the distance from R to A (which is closely related to what is termed as spread in EMO literature in terms of the maximal gap in the approximation). Ideal approximations in the sense of Δ_p are non-dominated fronts which are evenly distributed along the Pareto front.

Definition 5 ([8]). *Let $A, R \subset \mathbb{R}^n$ be finite sets. The value*

$$\Delta_p(A, R) = \max(GD_p(A, R), IGD_p(A, R)), \tag{2.13}$$

where

$$GD_p(A, R) = \left(\frac{1}{|A|}\sum_{a \in A} d(a, R)^p\right)^{1/p} \text{ and } IGD_p(A, R) = \left(\frac{1}{|R|}\sum_{r \in R} d(r, A)^p\right)^{1/p}, \tag{2.14}$$

and

$$d(a, R) = \inf_{r \in R}(\max_{i=1,2,\ldots,n}|a_i - r_i|), \tag{2.15}$$

and $p \in \mathbb{N}$*, is called the* averaged Hausdorff distance *between* A *and* R*.*

The indicator Δ_p can be viewed as a composition of slight variations of the generational distance and the inverted generational distance. It is $\Delta_\infty = d_H$, but for finite values of p the indicator Δ_p averages the distances considered in d_H. Hence, as opposed to d_H, Δ_p does in particular not punish single (or few) outliers in a candidate set.

The principal challenge when we use Δ_p as guide to solve MOPs is that the Pareto front is not known a priori, i.e., it is necessary to generate a substitute reference set R for the Pareto front in order to be able to use Δ_p.

Chapter 3

State-of-the-Art

For the computation of the Pareto set/front of a given MOP there exist so far several EMOAs such as NSGA-II ([2]), NSGA-III ([56]), SPEA2 ([57]), and MOEA/D ([58]). Next to these "general purpose" EMOAs (i.e., they do not use an explicit rule for the search as a particular performance indicator) that aim to compute a set of converged points along the Pareto front, there exist indicator based evolutionary algorithms (IBEA) that aim to achieve good approximations with respect to a given performance indicator. There are, for instance, the SMSEMOA ([59]) and NDS-HV ([60]) that aim for hypervolume approximations of the Pareto front and DDE ([61]) and Δ_p-EMOA ([24], [20], [49]) that aim for respective Δ_p approximations.

Finally, in [62], [63] the hypervolume gradient is presented for complete archives which gives rise to set based mathematical programming techniques (as e.g. done in [64]).

In the following, NSGA-II, and the SMSEMOA are discussed in more detail. NSGA-II and SMSEMOA have been selected for more detailed discussion since they have been ones of the most popular state-of-the-art EMOAs for a long time. For this reason they have been chosen for comparison in this thesis work.

3.1 General Purpose EMOA

Some general purpose EMOAs are presented next.

3.1.1 Non-Dominated Sorting Genetic Algorithm II (NSGA-II)

The elitist non-dominated sorting genetic algorithm (NSGA-II) [2, 65] has been one of the most referenced multi-objective optimization method in the EMO literature. The working principle of NSGA-II is as follows: at each generation, a new child population is created, and it has an equal size compared to the parent population. After each generation, the parent and child populations are combined together. If the population size is NP, then the combined population has size $2NP$. This combined population is sorted using non-dominated sorting defined in [2, 65], and the best NP individuals are selected based on non-dominance ranking. The individuals from the best non-dominated class are selected first for the next generation, then from the second best non-dominated class, and so on, until the number of selected individuals is NP. If the last non-dominated class of solutions is too big to fit completely in the set of NP individuals, then this non-dominated set is reduced based on a crowding estimation among the individuals of the class. The idea is to remove the most crowded individuals until the remaining individuals fit into the selected set of NP individuals.

Crowding estimation in NSGA-II is based on a distance metric called the *crowding distance*. The crowding distance for a member of a non-dominated set tries to approximate the perimeter of a cuboid formed by using the nearest neighbors of the member. The cuboid in the case of two objectives is shown in Figure 3.1. For a member of a non-dominated set, the crowding distance is calculated by finding the objective value difference between the two nearest solutions on either side of the member along each of the objectives (distances d_1^i and d_2^i in Figure 3.1). Then the differences are normalized by dividing them by the difference between the maximum and minimum values of the corresponding objectives. Finally, these normalized distances are summed, giving a crowding distance for the corresponding member. For those members which have a maximum or minimum value for any objective, the crowding distance is assigned to have an infinite value, i.e., those members are considered as the least crowded. Finally, the members of the non-dominated set are sorted according to their crowding distances and a desired number of members having the smallest crowding distance are removed [65].

The above described selection process of individuals for the next generation is illustrated in Figure 3.2. It should be noted that pruning based on diversity is done only among the members of the last non-dominated class of solutions that is selected for the next generation.

In [66], it is claimed that with early generations there exist several different non-dominated sets/classes and the diversity preservation has little effect on the selection process. When the population starts to converge to the Pareto front, the non-dominated classes become larger and eventually it is likely that the best non-dominated class is larger than NP. Thus, only little diversity preservation is performed at the early generations but more during the late generations. This kind of strategy provides a way to balance between convergence and diversity, but unfortunately, it works only with two objectives, because the crowding distance metric used in NSGA-II does not estimate crowding well when the number of objectives is more than two. Even if there were a working diversity preservation technique, the balance between convergence and diversity changes when the number of objectives increases. When the number of objectives increases, the number of non-dominated individuals increases and diversity preservation becomes a dominating operation in the survival selection only. In the light of this behavior, it becomes evident that, NSGA-II in its original form performs well only with problems having two objectives [67].

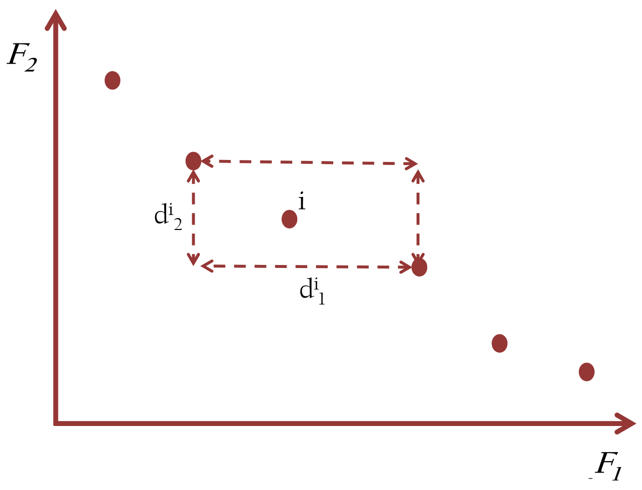


Figure 3.1: Example of the cuboid of a solution in the case of two objectives.

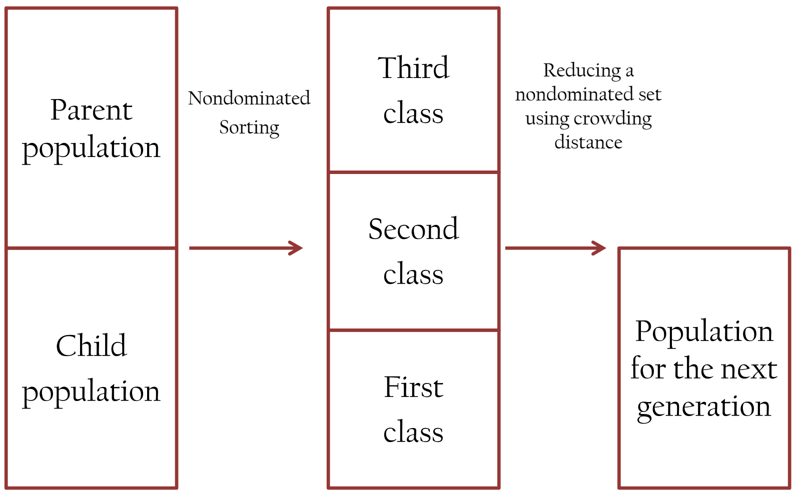


Figure 3.2: Selection of individuals for the next generation in NSGA-II.

3.1.2 Non-Dominated Sorting Genetic Algorithm III (NSGA-III)

A potential EMOA for solving many objective optimization problems (having four or more objectives) is NSGA-III, which is a reference-point-based many objective evolutionary algorithm following the NSGA-II framework, and it emphasizes population members that are non-dominated, yet close to a set of supplied reference points.

The basic framework of the proposed many objective NSGA-III very similar to the original NSGA-II algorithm with significant changes in its selection operator. But, unlike in NSGA-II, the maintenance of diversity among population members in NSGA-III is aided by supplying and adaptively updating a number of well spread reference points.

Let us consider the $t-$th generation of NSGA-II algorithm. Suppose the parent population at this generation is P_t and its size is N, while the offspring population created from P_t is Q_t having N members. The first step is to choose the best N members from the combined parent and offspring population $R_t = P_t \subset Q_t$ (of size $2N$), preserving elite members of the parent population. To achieve this, first the combined population R_t is sorted according to different non-dominated classes (F_1, F_2, and so on). Then, each non-dominated class is selected one at a time to construct a new population S_t , starting from F_1, until the size of S_t is equal to N or for the first time exceeds N. Let us say the last class included is the lth class. Thus, all solutions from class $(l + 1)$ onward are rejected from the combined population R_t. In most situations, the last accepted level (lth level) is only accepted partially. In such a case, only those solutions that will maximize the diversity of the lth front are chosen. In NSGA-II, this is achieved through a computationally efficient, yet approximate, niche-preservation operator that computes the crowding distance for every last class member as the summation of objective-wise normalized distance between two neighboring solutions. Thereafter, the solutions that have larger crowding distance values are chosen.

The above procedure of identifying non-dominated classes/fronts using the usual domination principle is also used in NSGA-III. All population members from the non-dominated front level 1 to level l are first included in S_t. If $|S_t| = N$ no further operations are needed and the next generation is started with $P_{t+1} = S_t$. For $|S_t| > N$, members from one to $(l-1)$ fronts are already selected, i.e., $P_{t+1} = \cup_{i=1}^{l-1} F_i$, and the remaining $(K = N - |P_{t+1}|)$ population members are chosen from the last front F_l.

NSGA-III uses a predefined set of reference points to ensure diversity in obtained solutions. The chosen reference points can either be predefined in a structured manner or supplied preferentially by the user.

For example, in a three-objective problem, the reference points are created on a triangle with the apex at (1, 0, 0), (0, 1, 0), and (0, 0, 1). If 15 reference points will be created (Figure 3.3). In the proposed NSGA-III, in addition to emphasizing non-dominated solutions, it emphasizes population members that are in some sense associated with each of these reference points. Since the above created reference points

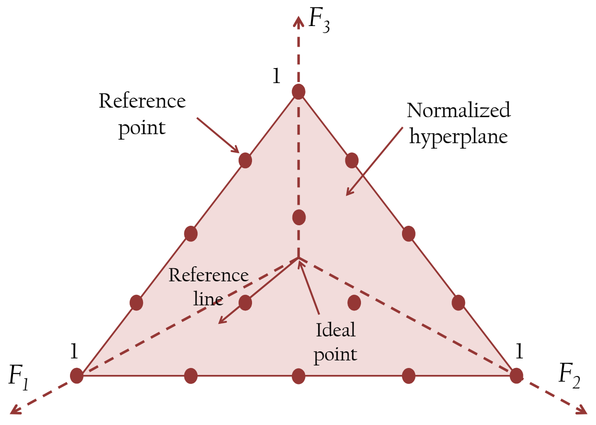


Figure 3.3: Fifteen structured reference points are shown on a normalized reference plane for a three-objective problem.

are widely distributed on the entire normalized hyperplane, the obtained solutions are also likely to be widely distributed on or near the Pareto front. In the case of a user supplied set of preferred reference points, ideally the user can mark H points on the normalized hyper-plane. The proposed algorithm is likely to find near Pareto-optimal solutions corresponding to the supplied reference points.

After normalizing the reference set according to each objective adaptively based on the extent of members of S_t in the objective space, the way to associate each population member with a reference point is defined as follows: a reference line corresponding to each reference point on the hyper-plane is defined by joining the reference point with the origin. Then, the perpendicular distance of each population member of S_t from each of the reference lines is calculated. The reference point whose reference line is closest to a population member in the normalized objective space is considered to be associated with the population member, and it is used to select the best individuals for the next generations. This is illustrated in Figure 3.4.

3.1.3 Improved Strength Pareto Evolutionary Algorithm (SPEA2)

The improved version of the strength Pareto evolutionary algorithm (SPEA2) [57] is another commonly used EMOA. The fitness assignment of SPEA2 is based on calculating strength values for individuals. The strength value of an individual x measures how many individuals x dominates. A raw fitness value for an individual y is calculated as a sum of the strength values of those individuals that dominate y. This raw fitness value is smaller for the better individuals.

Diversity preservation in SPEA2 is managed by calculating the distance of individuals to the nearest neighbor in the objective space. This distance value is transformed to a crowding measure; A small value means low crowding, and a large value means high crowding. The crowding measure is scaled between [0, 1] and added to the fitness

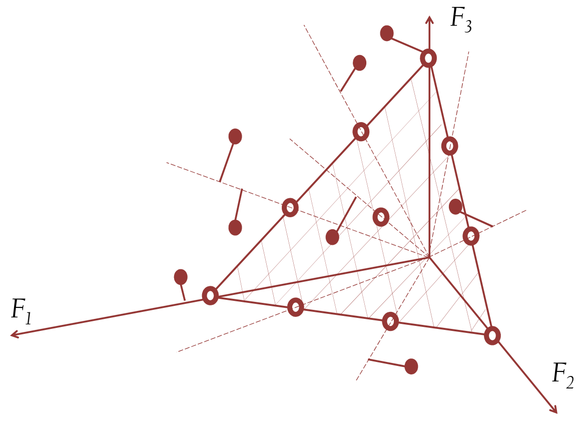


Figure 3.4: Association of population members with reference points.

value of each individual. Therefore, individuals are primarily ranked using the raw fitness values and in the case of identical raw fitness values, the less crowded individual will be preferred.

In addition to the population, SPEA2 uses an extra archive for solutions. This archive has a fixed size and is mainly reserved for non-dominated solutions. However, if there are not enough non-dominated individuals, the space in the archive is filled based on the fitness value of the individuals. If the number of non-dominated individuals is larger than the archive size, then redundant individuals are removed based on the fitness values.

SPEA2 provides a well distributed set of solutions also when the number of objectives is more than two. However, when the number of objectives increases, the search will slow down because fewer solutions dominate each other.

3.1.4 Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D)

Multi-objective evolutionary algorithm based on decomposition (MOEA/D) [58] decomposes a MOP into N different scalar optimization problems using the weighted Tchebycheff approach and solves these simultaneously using an evolutionary algorithm. This algorithm is restricted to performing reproduction using only vectors close to each other in the weight vector space. Neighboring solutions are replaced with the new solution, if the new one has a better scalarized value.

MOEA/D represents a return to classical multi-objective optimization methods. However, MOEA/D is a recent method with good results and it might be usable also in cases when the number of objectives is large and the search based on Pareto dominance stagnates.

In [68], MOEA/D with weighted sum was favorably evaluated on many objective problems with convex Pareto fronts; however, MOEA/D using the weighted Tchebycheff method [69] takes advantage in non-convex Pareto Fronts, thus, since the choice of the appropriate scalarization function to use in MOEA/D depends on the shape of the Pareto front it was proposed in [68] to automatically alternate between weighted sum and weighted Tchebycheff.

3.2 Indicator Based Evolutionary Algorithms

Some indicator based EMOAs are presented next.

3.2.1 S-Metric Selection Evolutionary Multi-Objective Optimization Algorithm (SMSEMOA)

The SMSEMOA [4] is known as a powerful state-of-the-art EMOA with particular merits in case of more than three objectives [70]. The method uses two selection methods sequentially: firstly, the population is partitioned in a hierarchy of antichains (termed non-dominated sorting) which represents a ranking of individuals w.r.t. their degree of non-dominance. Secondly, the element with the least hypervolume contribution from the worst ranked subset is discarded.

An approximation set A can be partitioned into h disjunct antichains $R_1, ..., R_h$, where h is the length of the longest chain in this set: $R_1 = M_f(A, \preceq)$ and

$$R_k = M_f\Big(P\backslash \bigcup_{i=1}^{k-1} R_i, \preceq\Big) \text{ for } k = 2, ..., h \text{ if } h \geq 2,$$

where $M_f(A, \preceq)$ is the set of non-dominated points from A.

Evidently, every element from R_j is dominated by some individual in R_i if $i < j$.

The hypervolume is defined as the area (volume in more than two objectives) of coverage with in the objective space [5, 26] for a bi-objective optimization problem. This equates to the summation of all the rectangular areas, bounded by some reference point r and the non-dominated points an approximation A. Mathematically, this is described as follows:

$$HV(A, r) = \Big\{\bigcup_i vol_{a_i}|r : a \in A\Big\}.$$

The hypervolume contribution of some element $x \in R_k$ is simply the difference $HV(R_k, r) - HV(R_k\backslash x, r)$ between the dominated hypervolume of set R_k and the dominated hypervolume of set R_k without element x. The description of the SMSEMOA is depicted in Algorithm 5.

Algorithm 5 Pseudo code of SMSEMOA.

```
Require: draw multiset $P$ with $\mu$ elements $\in \mathbb{R}^n$ at random
1: while Stop criterion is not met do
2:     generate offspring $x \in \mathbb{R}^n$ from $P$ by variation
3:     $P = P \cup \{x\}$
4:     build ranking $R_1, ..., R_h$ from $P$
5:     $\forall i = 1, ..., d : r_i = max\{F_i(x) : x \in R_h\} + 1$
6:     $\forall x \in R_h : h(x) = HV(R_h, r) - HV(R_h \backslash \{x\}, r)$
7:     $x^* = argmin\{h(x) : x \in R_h\}$
8:     $P = P \backslash \{x^*\}$
9: end while
```

3.2.2 R_2 Evolutionary Multi-Objective Algorithm (R2-EMOA)

An IBEA is introduced in [71] which incorporates the contribution to the unary R_2-indicator as the secondary selection criterion. It is based on a set of utility functions. In total, three different variants were proposed which differ in the way the utilities are evaluated and combined, the ratio of one set being better than the other (R_1), the mean difference in utilities (R_2), or the mean relative difference in utilities (R_3).

The proposed R_2-EMOA implements a steady state strategy based on the contribution to the R_2-indicator (see Algorithm 6).

Algorithm 6 Pseudo code of the R_2-EMOA.

```
Require: draw multiset $P$ with $\mu$ elements $\in \mathbb{R}^n$ at random
1: while Stop criterion is not met do
2:     generate offspring $z \in \mathbb{R}^n$ from $P$ by variation
3:     $P = P \cup \{z\}$
4:     build ranking $R_1, ..., R_h$ from $P$
5:     $\forall x \in R_h : r(x) = R_2(P \backslash \{x\}, \lambda, i)$
6:     $x^* = argmin\{r(x) : x \in R_h\}$
7:     $P = P \backslash \{x^*\}$
8: end while
```

3.2.3 Δ_p-EMOA

The first EMOA designed for the computation of Hausdorff approximations of the Pareto front was proposed in [19] for bi-objective problems. The challenge for all such algorithms is that the Pareto front is not known a priori. In [19] this was solved by creating a piecewise linear front out of the non-dominated solutions of the current archive. This together with an application of the PL metric ([72]) was the basis of the selection mechanism.

Since the averaging Hausdorff measure needs two sets as arguments and the Pareto front is unknown in general have to construct a reference set for assessing the progress towards the Pareto front which is the basis for the archiving strategy. For this purpose the current approximation of the SMSEMOA is used in each iteration since the SMSEMOA will drive its approximation towards the Pareto front by maximizing the dominated hypervolume of approximation of the Pareto front. Due to this principle Pareto optimal solutions are generated sufficiently spread along the Pareto front, however, with higher solution density at the knee regions of the front. Thus, a linear interpolation between the points of the current approximation can then be used to place the points of the reference set in evenly spaced manner. In detail: let $v^{(1)}, ..., v^{(\mu)} \in \mathbb{R}^2$ be the current approximation of the Pareto front in lexicographical order provided by the SMSEMOA. Between these points in objective space a linear interpolation is calculated, i.e., for $i = 1, 2, ..., \mu - 1$

$$g_i(x) = \frac{v_2^{(i)} - v_2^{(i+1)}}{v_1^{(i)} - v_1^{(i+1)}}(x - v_1^{(i)}) + v_2^i,$$

if $x \in [v_1^{(i)}, v_1^{(i+1)}]$. The length of each linear segment is $L_i = ||v_1^{(i)} - v_1^{(i+1)}||_2$ so that the total length of the linear interpolation model of the Pareto front is just the sum of the L_i. Now it is easy to place μ evenly spaced points on the linear interpolation. These points are collected in the current reference set $R^{(t)}$.

As for the archiving strategy let $A^{(t)}$ denote the archive set and $R^{(t)}$ the reference set at step $t \geq 0$. We assume that the archive may have at most the same cardinality as the reference set: $N_A \leq N_R < \infty$. The archiving procedure (Algorithm 7) is run for every offspring x generated by the SMSEMOA for the current reference set R. The initial archive is empty.

Algorithm 7 update($x, A; R$).

```
1: $A = M_f(A \cup \{x\}, \preceq)$
2: **if** card($A$) > $N_R$ **then**
3:     **for all** $a \in A$ **do**
4:         $h(a) = \Delta_1(A \backslash \{a\}, R)$
5:     **end for**
6:     $a^* = argmin\{h(a) : a \in A\}$
7:     $A = A \backslash \{a^*\}$
8: **end if**
```

Summarizing, the SMSEMOA and the innovative archiving strategy form a close union with the purpose of minimizing the averaged Hausdorff distance Δ_p while simultaneously obtaining good convergence properties of the algorithm. The SMSEMOA ensures the latter by its efficient selection mechanism with the aim of maximizing the dominated hypervolume. In addition, linear interpolations of the sequential approximations of the Pareto front form the basis for constructing the required reference sets for calculating the averaged Hausdorff distances. However, in principle the SMSEMOA could be replaced by any EMOA of similar performance.

Later studies dealed with tri-objective problems. In [20], d-dimensional objective vectors were mapped to two-dimensional space via the *Multi-Dimensional Scaling (MDS)* metric allowing to utilize the archiving strategy used. In [21], specialized

Algorithm 8 SMSEMOA with Δ_p-Archive.

```
Require: draw multiset $P$ with $\mu$ elements $\in \mathbb{R}^n$ at random
 1: set $A = M_f(P, \preceq)$
 2: while Stop criterion is not met do
 3:     build linear interpolation (LI) based on $M_f(P, \preceq)$
 4:     place $N_R$ points evenly on LI $\rightarrow$ yields $R$
 5:     generate offspring $x \in \mathbb{R}^n$ from $P$ by variation
 6:     update$(x, A; R)$
 7:     $P = P \cup \{x\}$
 8:     build ranking $R_1, ..., R_h$ from $P$
 9:     $\forall i = 1, ..., d : r_i = max\{F_i(x) : x \in R_h\} + 1$
10:     $\forall x \in R_h : h(x) = HV(R_h, r) - HV(R_h \backslash \{x\}, r)$
11:     $x^* = argmin\{h(x) : x \in R_h\}$
12:     $P = P \backslash \{x^*\}$
13: end while
```

triangulations and boundary detection concepts to approximate the 3D surface of the considered approximations of the Pareto front were used. Finally, in [23] MOPs with four objectives were addressed.

3.2.4 Partition and Selection Algorithm

Instead of an approximation of the reference front, the PSA ([73]) can be used to generate the reference set, because PSA is capable of quickly selecting a "well-spread" subset out of a given data set.

The PSA has been recently introduced as an algorithm for selecting m "well-spread" points from a set of $n > m$ points. Its mainly characteristic is that it has a low computational complexity $O(nmd)$, where d is the number of objectives of the MOP, thus, the cost is in particular linear in d. PSA consists of two steps: first, the set of interest is partitioned into m subsets in order to group similar members into the same subset, and second, it performs the selection of one representative member from each generated subset in order to get a diverse subset of m points of the set of interest. A more detailed description is as follows:

Partitioning a Set

This is based on the concept of the *dissimilarity* of a set:

Let $A = \{\mathbf{F}_1 = [F_{11}, \ldots, F_{1d}], \ldots, \mathbf{F}_n = [F_{n1}, \ldots, F_{nd}]\}$ (i.e., n objective vectors $\mathbf{F}_i \in \mathbb{R}^d$), denote $a_j = minF_{ij}$, $b_j = maxF_{ij}$, and $\Theta_j = b_j - a_j$, $i = 1, \ldots, n$, $j = 1, \ldots, d$, then

$$diss(A) = max\Theta_j, j = 1, \ldots, d. \tag{3.1}$$

In each step, the set with the greatest dissimilarity among its members is the one that is divided. This is repeated m times to obtain the desired number of subsets. Figure 3.5 demonstrates the steps of part the set of interest into $m = 5$ subsets.

Selection of points

Once the set A has been partitioned into the m subsets $A_1, \ldots, A_m$, the "most suitable" element from each subset must then be chosen in order to obtain a subset $A_{(r)}$ of A that contains m elements. This is of course problem dependent. Figure 3.5 (bottom right) illustrates the rule of the nearest points to the centers. The centers of the grey rectangles are marked with a cross. In each subset the member closest to the center is circled (a random member is circled in the subset with only two members). The representative set $A_{(r)} = \{a_1, a_2, a_3, a_4, a_5\}$ is the set of all yellow points.

3.2.5 Summary

Indicator-based EMOAs use a quality indicator to assign fitness to solutions. These algorithms transform the original many objective problem into a single objective one (i.e., the problem of optimizing the given indicator). An unary indicator takes one set of non-dominated solutions and returns a real number related with a given performance criteria, whereas, binary quality indicators serve to compare the relative quality of two sets of non-dominated solutions.

The principal difficulties in solving MOPs using EMOAs include in general: (i) the growing proportion of non-dominated solutions, (ii) the computational cost, and (iii) visualization of results. Methods based on preference relations as the NSGA-II or the SPEA2 focus on the first of these difficulties by providing improved ranking schemes, on the other hand, methods based on transformations of the original problem, as the IBEAs presented, also considers, with a varying extent, the other two issues.

Several EMOAs, such as the NSGA-II, are designed with diversity preservation mechanism. Because obtaining a good diversity in MOPs with a large proportion of non-dominated solutions is not a difficult task, but there is a trade-off between convergence and the extension of the covered Pareto front, then the role of these diversity preservation mechanisms in EMOAs must be evaluated.

An advantage of methods based on performance indicators is that they are relatively easy to implement into the same underlying EMOA. Also, preference relations may sample different space regions as in MOEA/D. Therefore, an alternative to obtain a good relation among convergence and diversity into a broader range of problems may be to explore combinations of preference relations, introducing perhaps more than one performance indicator taking care in the computational complexity it represents (e.g., the computational load of hypervolume calculation, in general, prevents to use this indicator in MOPs with a high number of objectives).

The use of Δ_p as the indicator that guides the search of the evolution seems to

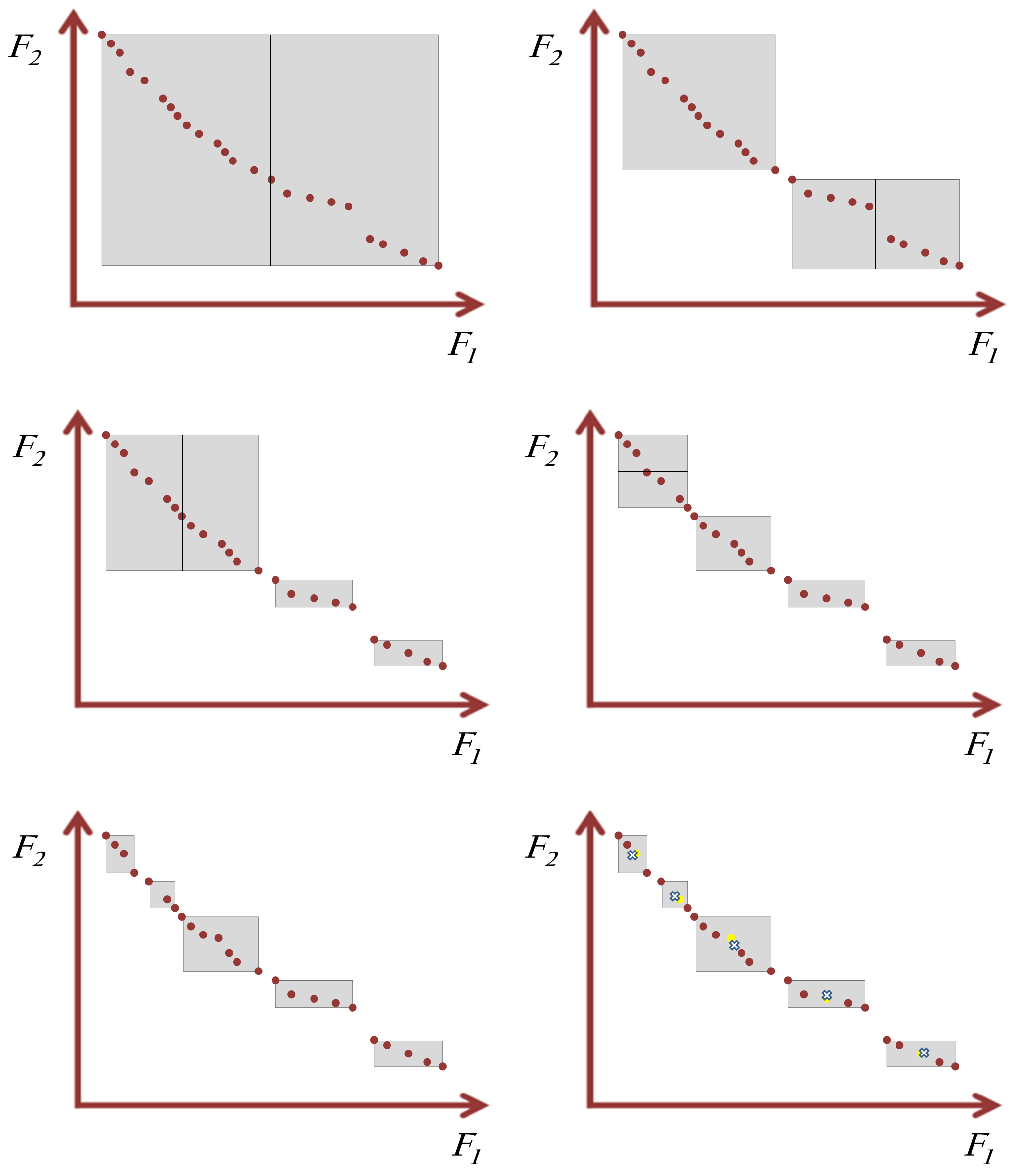


Figure 3.5: Partitioning of 24 elements in bi-objective space into 5 subsets (gray boxes) and selection of the subset $A_{(r)}$ using the center points (bottom right).

be promising, because this indicator aims at the same time for a good spread and convergence. Of course we face with the problem that we need a reference set which allows the computation of Δ_p, and a very good alternative to generate this reference set is the PSA, which computes a good reference of the Pareto front with low computational cost (no matter the number of objectives).

Chapter 4

PSEMOA

4.1 Basic Idea

The PSEMOA is an alternative algorithm aiming for Hausdorff approximations of the Pareto front that uses PSA as tool to select the best individuals during the evolution of the algorithm, those principal characteristic is that it can be used for more than two objectives without increasing significantly the computational requirement for its realization.

The steps of PSEMOA are as follows: first, a population is chosen at random. When a new offspring of out of the population is created via crossover and mutation during the evolution, the set A is defined, which is the set of non-dominated points of the union of the current population and the offspring. Since PSA preserves any outliers encountered during the search, a strategy to delete these points is needed (the maximum values per objective are deleted from A). If the size of the set of non-dominated points free of outliers A' is less than the maximum population size, then a non-dominated sorting of A is performed [2], and the next generation is selected according to this sorting. If the size of A' is bigger than the maximum population size, then PSA selects n points out of A' as the next generation. In order to increase the quality of the approximations found by PSEMOA, the Δ_p external archive strategy and the offline version defined in Section 4.2 are used. The offline version allows PSA to have much more information to generate a good reference set needed by the Δ_p external archive, which will guide the external archive to better results. In Algorithm 9 a pseudo code of PSEMOA can be found.

4.2 Offline Approach

Suppose that some EMOA has generated an approximation of the Pareto front for some MOP. Typically, this approximation does not yield a finite point set in objective space that is evenly distributed. Therefore, it is proposed the following *offline* approach:

Algorithm 9 PSEMOA Algorithm with the implementation of the Δ_p external archive strategy and the *offline* approach.

```
$maxEval$ = Maximum number of evaluations.
$popSize$ = Size of the population.
Draw multiset $P_1$ with $popSize$ elements $\in \mathbb{R}^n$ at random.
while $contEval < maxEval$ do
  Generate new $popSize$ offspring $x_k \in \mathbb{R}^n$ from $P_i$ by variation.
  Save the offspring into an external file.
  $contEval = contEval + popSize$.
  $A = M_f(P_i \cup \{x_k\}, \preceq)$.
  if $i$ is pair then
    $A' = deleteOutliers(A)$.
  else
    $A' = A$.
  end if
  if $|A'| < popSize$ then
    Build ranking $R_1, \ldots, R_h$ from $P_i \cup \{x_k\}$.
    while $|P_{i+1} \cup R_j| < popSize$ do
      $P_{i+1} = P_{i+1} \cup R_j$.
      $j = j + 1$.
    end while
    if $|P_{i+1}| < popSize$ then
      $s = popSize - |P_{i+1}|$.
      $R'_j = \{s$ elements of $R_j$ selected randomly$\}$.
      $P_{i+1} = P_{i+1} \cup R'_j$.
    end if
  else
    $P_{i+1} = PSA(A', popSize)$.
  end if
  $i = i + 1$.
end while
$A = \{$all points of the external file$\}$.
$A' = M_f(A, \preceq)$.
$A'_R = eliminateAllNondominatedOutliers(A', P_i)$.
$R = PSA(A'_R, popSize)$.
for all $a \in A$ do
  $Archive = \Delta_p\_update(a, Archive, R)$.
end for
Print $Archive$.
```

1. Run your favorite EMOA that is equipped with a tiny add-on:
   - as soon as an offspring is generated and evaluated store a copy in a file
2. After termination of your favorite EMOA:
   - construct an evenly spaced reference front from a given approximation of the Pareto front (e.g. from last population of favorite EMOA)
   - feed each stored offspring into the Δ_p-archive updater sequentially
   - output: archive A

After the reference front R has been constructed it is used in the Δ_p-archive updater to decide which point should be added to or deleted from the archive. An update operation can be realized as sketched in algorithm 10.

Algorithm 10 Δ_p-UpdatePSA

```
Require: archive set $A$, reference set $R$, new element $x$
 1: $A = M_f(A \cup \{x\}, \preceq)$
 2: if $|A| > N_R = |R|$ then
 3:     for all $a \in A$ do
 4:         $h(a) = \Delta_1(A \setminus \{a\}, R)$
 5:     end for
 6:     $A^* = \{a^* \in A : a^* = \mathsf{argmin}\{h(a) : a \in A\}\}$
 7:     if $|A^*| > 1$ then
 8:         $a^* = \mathsf{argmin}\{GD_P(A \setminus \{a\}, R) : a \in A^*\}$      {ties broken at random}
 9:     end if
10:     $A = A \setminus \{a^*\}$
11: end if
```

The most obvious order of feeding the stored pairs $(x, F(x))$ into the archive updater is the order of their generation. We call this the “forward update”. In this manner, many individuals will pass the initial dominance check, so that subsequent Δ_p-calculations are necessary. Some time saving may be achieved by feeding the stored pairs into the archive update in inverted order. We call this the “backward update”. Since points that have been generated in later iterations of the EMOA are more likely to dominate previous points, most points from the rear of the inverted sequence will probably not pass the initial dominance check, so that subsequent Δ_p-calculations can be avoided. Since the order of the points presented to the archive clearly affects the final outcome of the archive, we shall compare both approaches experimentally in the next Section.

4.3 Experiments and Results

The PSEMOA is used to solve the problem of computing finite size Hausdorff approximations of the Pareto front of four-objective optimization problems by means of evolutionary computing.

4.3.1 Construction of Benchmark

Reference Front for DTLZ1

The Pareto front of the test problem DTLZ1 is given by

$$F^* = \left\{ y \in \mathbb{R}^d_+ : \sum_{i=1}^{d} y_i = \frac{1}{2} \right\}$$

for $d \geq 2$. Therefore, a regular mesh is easily generated.

Reference Front for DTLZ2 and DTLZ3

The Pareto front of the test problems DTLZ2 as well as DTLZ3 is the intersection of the surface of the unit hypersphere with the nonnegative orthant of the standard coordinate system. As a consequence, they can be easily expressed via polar coordinates. If $d = 4$ then

$$F^* = \left\{ \begin{pmatrix} \cos\omega_1 \\ \sin\omega_1 \, \cos\omega_2 \\ \sin\omega_1 \, \sin\omega_2 \, \sin\omega_3 \\ \sin\omega_1 \, \sin\omega_2 \, \cos\omega_3 \end{pmatrix} \in \mathbb{R}^4 : \omega \in \left[0, \frac{\pi}{2}\right]^3 \right\}.$$

4.3.2 Analysis of the Results

Besides the PSEMOA described above, we also deploy three state-of-the-art EMOAs (NSGAII [2], MOEA/D [74], SMSEMOA [4]) for getting an idea how much can be gained from using a special purpose EMOA.

Experiments were conducted to compare the behavior of all considered algorithms. Three different four-dimensional test problems with different shapes of the Pareto front and multi-modality characteristics are addressed, i.e., DTLZ1 (linear, $n = 8$), DTLZ2 (concave, $n = 13$), DTLZ3 (concave, multimodal, n=13) [75].

The algorithms were independently run 30 times on each test function for 200,000 function evaluations using a population size $\mu = 400$. Standard settings were chosen for the variation operator parameters of SBX and polynomial mutation for all EMOAs but MOEA/D, i.e., $p_c = 0.9$, $\eta_c = 15$, $\eta_m = 20$ and $p_m = 1/n$. The MOEA/D uses Tchebycheff decomposition, a neighborhood parameter of 10, a subproblem size of 400 and differential evolution parameters $F = 0.5$ and $CR = 0.5$. The parameter settings of the Δ_p-EMOA coincide with the respective ones in [20]. The size of the external archive equals the population size for the Δ_p-EMOA. We chose a value of $p = 1$ within the Δ_p-Indicator in order to minimize the influence of outliers.

Figure 4.1 visualizes the distributions of the Δ_p values at the final generation of the EMOAs for the considered test problems.

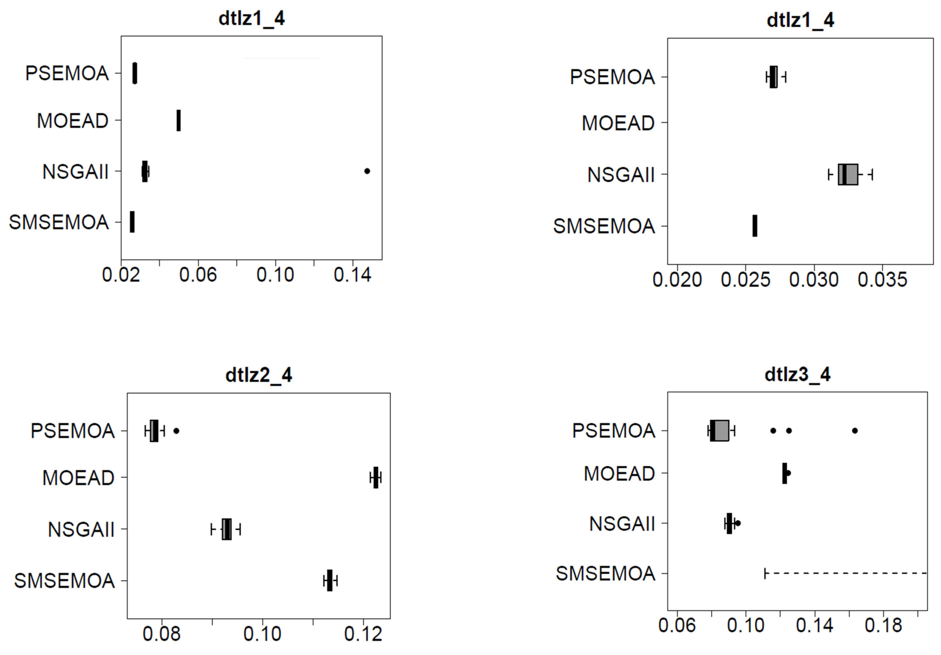


Figure 4.1: Boxplots of Δ_p at final generation for the considered test problems. Algorithms with missing boxes perform much worse.

It seems to be the case that the algorithms tend to loose diversity with increasing problem dimension so that the results of the SMSEMOA and the Δ_p-EMOA based variants are quite different.

It becomes obvious that the PSA approach is a very suitable means to generate approximations of the Pareto front with high Δ_p performance. For all test problems the PSEMOA is performing best. While for DTLZ1 the best algorithms PSEMOA and the SMSEMOA cannot be distinguished statistically, for DTLZ2 the PSAEMOA algorithm is the best. Finally, the PSEMOA is the winner for DTLZ3. It is visible that the latter test function imposes additional challenges on the optimization as the algorithm rankings show quite different characteristics. While NSGA-II was already 3rd best on DTLZ2 it is second best on DTLZ3 behind the PSEMOA. On DTLZ3 the SMSEMOA severely looses performance. This is due to the fact that the SMSEMOA is not successful in maintaining a high diversity of solutions due to the high multimodality of this test function.

Even regarding HV the SMSEMOA is not performing best. The setting of the variation operator parameters thus are not optimal in this case, probably better performance could be gained by systematic tuning approaches which was not possible here due to the extremely high SMSEMOA run-times. Regarding run-times the PSEMOA clearly is the best option as it is of very low complexity. Keeping in mind that this algorithm is high performing in all cases it is a very promising approach for higher-dimensional problems which will be confirmed in additional systematic experimental studies.

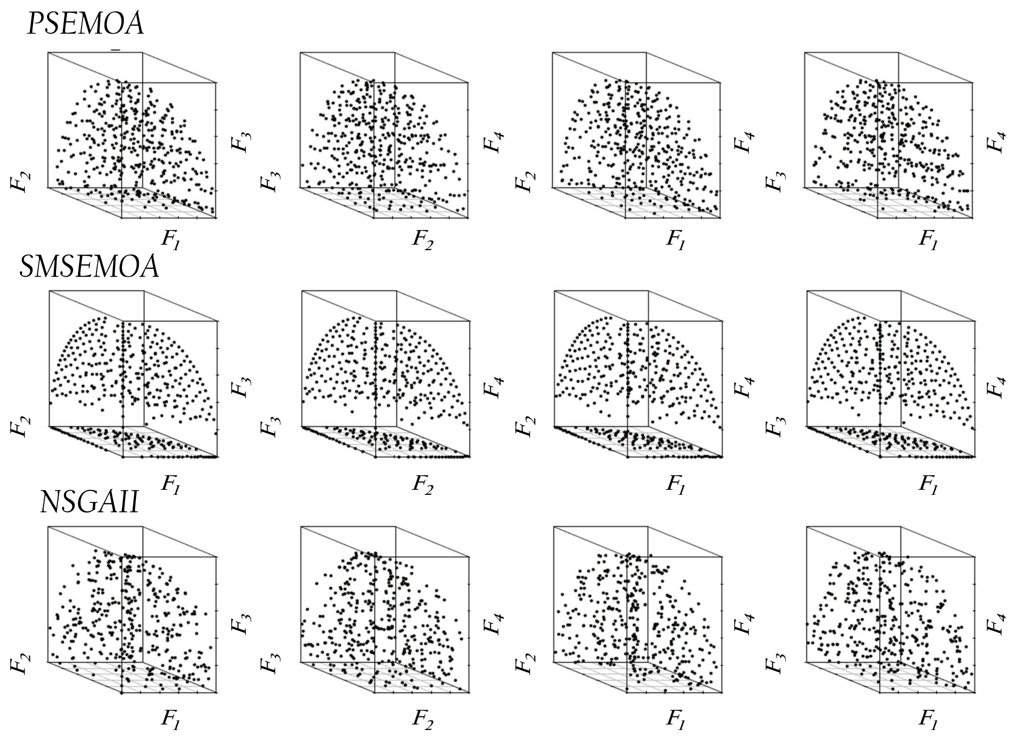


Figure 4.2: Final populations of considered EMOAs on DTLZ2 with best Δ_p value. Shown are all three-dimensional projections.

Moreover, the dimension reduction approach of the Δ_p-EMOA is not successful in four dimensions. The loss of information due to omitting two dimensions obviously is too high. A problem in this case is that the border points in 4D will not necessarily be border points in the two-dimensional mapping which is unfortunate.

Figure 4.2 shows the three-dimensional projections of the final populations on DTLZ2 for PSEMOA as well as NSGA-II and SMSEMOA. The PSEMOA successfully evolves a Pareto front approximation which is much more uniformly spread than the respective ones of the remaining EMOAs.

Chapter 5

Δ_p-EA

5.1 Basic Idea

The aim of this thesis project is to compute approximations of the Pareto front (F^*) that are optimal w.r.t. Δ_p. An optimal Δ_p archive of size μ solves the following optimization problem:

$$\min_{\substack{A \subset \Omega \\ |A| = \mu}} \Delta_p(F(A), F^*). \tag{5.1}$$

We stress that (i) since Δ_p assigns a scalar value to each archive, problem (5.1) is a SOP of dimension $n \cdot \mu$ while the dimension of the decision space of the underlying MOP (Definition 2.2) is n (that is, a n-dimensional problem if only *one* Pareto optimal solution is sought). Further, (ii) that analog SOPs are induced by any other indicator (e.g., HV and R2 [76]).

One practical problem of Δ_p is that the Pareto front of a given MOP is not known a priori which makes it impossible to evaluate the indicator value for a given archive A during the run of the algorithm. A possible remedy is to consider approximations of the Pareto front that are updated during the search process. This idea leads directly to the following dynamic SOP:

$$\min_{\substack{A \subset \Omega \\ |A| = \mu}} \xi_l(A) = \Delta_p(F(A), Z(l)). \tag{5.2}$$

Hereby, $\xi_l : \mathbb{R}^{n \cdot \mu} \to \mathbb{R}$, $l \in \mathbb{N}$, and $Z(l)$ denotes the l-th approximation of the Pareto front. Apparently, if the approximations converge toward the Pareto front, i.e., if $Z(l) \to F^*$ for $l \to \infty$, then problem (5.2) converges to the original problem (5.1). It remains hence to find ways to construct the reference fronts for which we propose two strategies in the following. For this, define $N(l) \subset \mathbb{R}^d$ as the set of candidate solutions in objective space (e.g., the image of the non-dominated solutions from a given archive), and $S(l)$ a (spline) approximation that interpolates $N(l)$. One possibility is to take $Z(l) = S(l)$ as e.g. done in [19] using linear interpolation. This approach, however, has the potential disadvantage that $S(l)$ is typically "above" the Pareto front which comes with the danger that non-optimal solutions are computed when solving (5.2). In the following we propose two remedies, namely to use (a) a penalization approach and (b) to shift the reference front to the infeasible area under the Pareto front.

5.2 Penalization Method

Define for a point $x \in \mathbb{R}^n$

$$P(x) = \min_{\alpha \in \mathbb{R}^d} \left\{ \left\| \sum_{i=1}^{d} \alpha_i \nabla F_i(x) \right\|_2^2 \; : \; \alpha_i \geq 0, \; \sum\nolimits_{i=1}^{d} \alpha_i = 1 \right\}, \tag{5.3}$$

and for an archive $A \subset \mathbb{R}^n$

$$P(A) = \sum_{a \in A} P(a). \tag{5.4}$$

Straightforward calculations show that

(i) $P(A) \geq 0$ for all $A \subset \Omega$,

(ii) P is continuous if the objectives are continuously differentiable, and

(iii) $P(A) = 0$ iff A is contained in the set of Karush-Kuhn-Tucker (KKT) points.

Thus, we can use $P(A)$ as a penalization function that operates on the set of archives and which generates a pressure toward the set of KKT points. The related dynamic SOP reads consequently as

$$\xi_{P,l}(A) = \Delta_p(F(A), S(l)) + C(l)P(A), \tag{5.5}$$

where $C(l)$ is a positive constant at step $l \in \mathbb{N}$ with $C(l) \to \infty$ for $l \to \infty$. Note that the above penalization method is formulated for unconstrained MOPs, but it may easily be adapted to constrained problems via the use of the extended KKT conditions in (5.3).

To obtain a successive approximation of the Pareto front, one can use (5.5) together with the following "bootstrapping": given $S(l)$, solve problem (5.5), define $N(l+1)$ as the obtained optimal archive, and continue the search via solving (5.5) using $S(l+1)$.

Example 5.2.1. *Figure 5.1 shows numerical results of this approach on the MOPs CONV ([19]), DTLZ2 ([75]), and DENT which have convex, concave, and convex-concave Pareto fronts, respectively, see also Table 5.1) which has a convex-concave*

Pareto front using $\mu = 10$ archive entries. The blue line represents the actual reference set $S(l)$ and the black dots the solutions of the related SOP. For $S(1)$ we have taken the convex hull of individual minima (CHIM), that is, we have first computed the minima of both objectives. The computations have been done using the solver `fmincon` *of* MATLAB[1]. *As it can be seen, an even spread of solutions along the Pareto front is obtained after three iterations (compare to Figure 5.2 which shows the optimal Δ_p approximation).*

One potential drawback of the penalization approach—in particular for the use within EAs—is that it requires gradient information. In the following we discuss a gradient free method.

5.3 Shifting Method

The underlying idea is to shift $S(l)$ such that this set is "left below" the Pareto front (i.e., in the infeasible area of the objective space) to force the algorithm to compute optimal solutions of the original MOP via solving (5.2). One such possibility is to use

$$Z_\delta(l) = S(l) - \delta(l) \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}, \tag{5.6}$$

where $\delta(l) > 0$ with $\delta(l) \to 0$ for $l \to \infty$. Other shifting directions than $(1, \ldots, 1)$ are of course possible e.g. via renormalizing the objective space based on the location of the CHIM. The resulting dynamic SOP is thus given by

$$\xi_{S,l} = \Delta_p(F(A), Z_\delta(l)). \tag{5.7}$$

By using $Z_\delta(l)$ as reference front, no gradient information is needed. On the other hand, the proper choice of $\delta(l)$ may represent a problem. It can, however, easily be checked if $\delta(l)$ has been chosen too small: if the optimal archive of the solution of (5.7) contains solutions that lie on $S(l)$, the value of δ has to be increased.

Since we are dealing here with EAs, we will use the shifting approach for our following computations.

Example 5.3.1. *Figure 5.2 shows numerical results of the shifting approach on the MOPs CONV ([19]), DTLZ2 ([75]), and DENT which have convex, concave, and convex-concave Pareto fronts, respectively. We used the same 'bootstrapping' approach but this time for $\xi_{S,l}$. For all computations we have used $\delta(1) = 1$, $\delta(2) = 0.1$, and $\delta(3) = 0.01$ and the shifted CHIM for $Z_\delta(1)$. In all three cases we obtained nearly optimal Δ_p approximations after three iterations. The figures in the last row show images of the optimal Δ_p archives for each problem.*

[1]`http://www.mathworks.com`

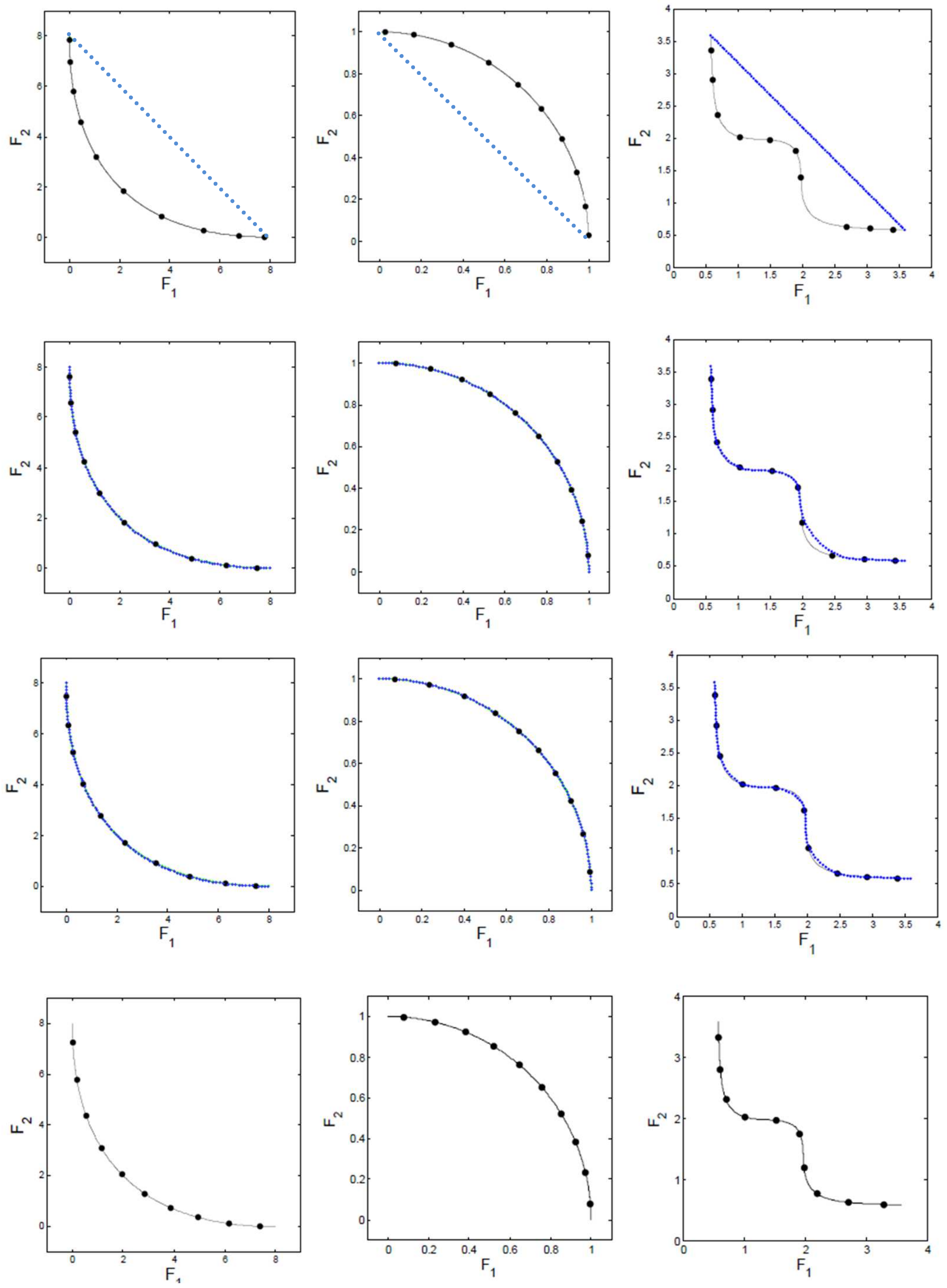


Figure 5.1: Numerical results of the “bootstrapping” method using the penalization approach on the MOPs CONV, DTLZ2, and DENT (from left to right). The first three rows show the results of the first three iterations and the last row shows the images of the optimal Δ_p archives.

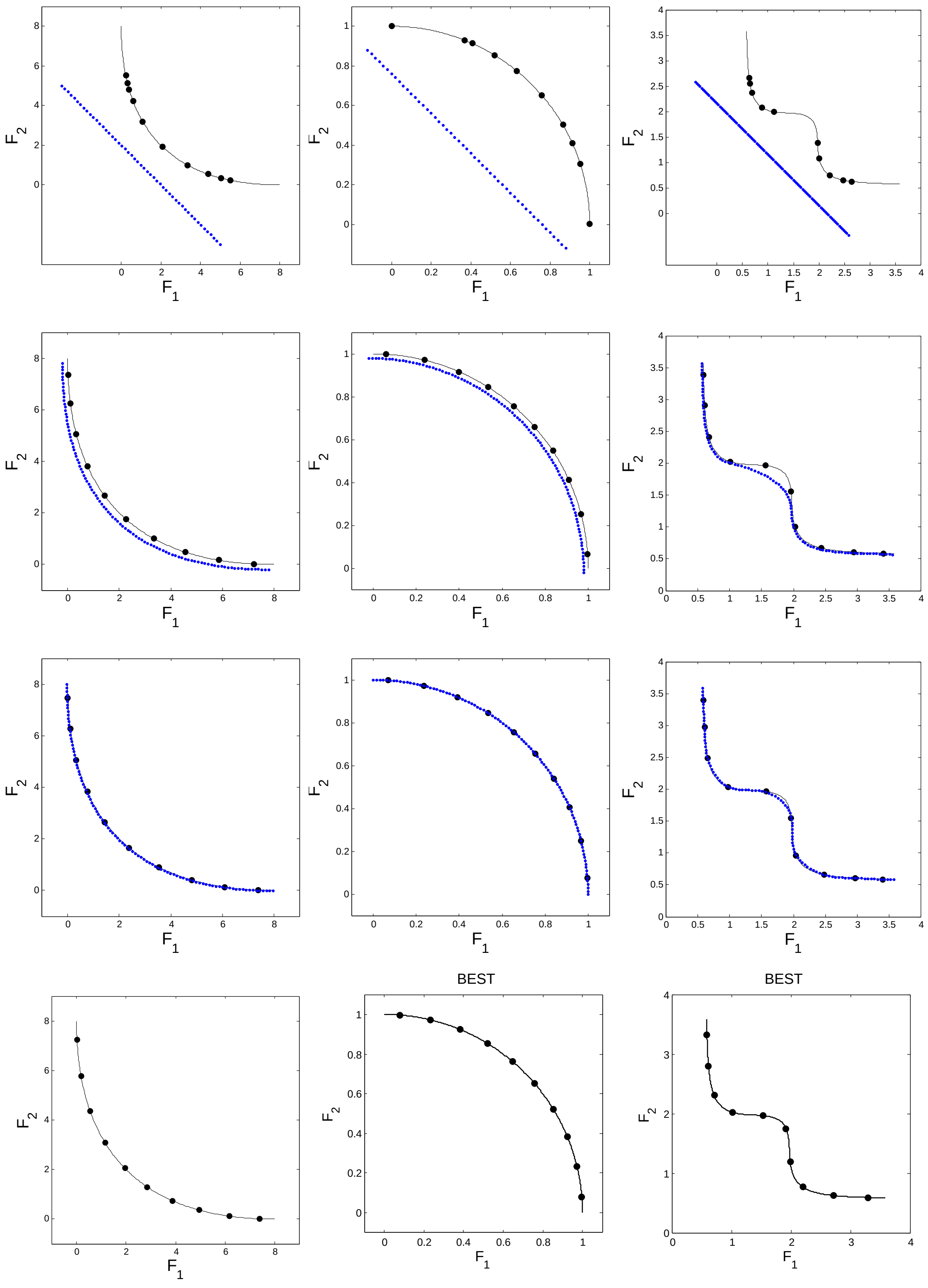


Figure 5.2: Numerical results of the “bootstrapping” method using the shifting approach on the MOPs CONV, DTLZ2, and DENT (from left to right). The first three rows show the results of the first three iterations and the last row shows the images of the optimal Δ_p archives.

So far, all Pareto fronts considered here are connected. To handle disconnected fronts, we simply used the "non-dominated part" of $S(l)$ in our computations (i.e., we omitted all dominated points) which already lead to satisfying results. Further, more sophisticated, approaches are subject of ongoing research.

5.4 Description of the Algorithm

Here we propose a first evolutionary strategy, Δ_p-EA, that aims for Δ_p approximations of the Pareto front via directly utilizing the SOP induced by Δ_p.

In this algorithm, the current population $P(l)$ is chosen from a pool of candidate solutions given by the former population and a newly generated candidate solution by considering the Δ_p contributions of each point. In addition to the population, an external archive $E(l)$ is maintained which serves as the 'best approximation' of the problem and will be returned after the run of the algorithm.

To be more precise, the l-th step of the algorithm is as follows (compare to Algorithm 11): first, the shifting distance $\delta(l)$ and the reference set $Z_\delta(l)$ are computed. For $Z_\delta(l)$, the actual candidate set $N(l)$ as well as the extreme points EP of the Pareto front are considered, where EP has to be handed to the algorithm. Second, an offspring x is generated from the actual population $P(l)$. Third, both the candidate set $N(l)$ and the external archive $E(l)$ are updated. Finally, fourth, $P(l)$ is updated by x using $Z_\delta(l)$ to estimate the Δ_p values.

To realize the algorithm we have used the following.

(i) **Shifting parameter**. At the beginning of the evolution, the reference set is shifted by the maximal distance δ_{max}. During the run of the algorithm, the distance δ is then reduced linearly to a minimal value δ_{min}. To be more precise, we have used the following formula for the shifting distance

$$\delta = \begin{cases} \delta_{max} - \left(\frac{\delta_{max} - \delta_{min}}{\delta_{max} \cdot \sigma} \cdot currentEval\right) & \text{if } currentEval < (maxEval/\sigma) \\ \delta_{min} & \text{otherwise} \end{cases} \tag{5.8}$$

where $currentEval$ and $maxEval$ denote the current and maximal number of function evaluations, respectively. Based on some studies, we have used the values $\sigma = 6$, $\delta_{max} = 1$ and $\delta_{min} = 0.01$.

(ii) **Update of the archive.** For the update of $E(l)$ we have used Δ_p-Update from [19] (see also Algorithm 12), i.e.,

$$E(l+1) = \Delta_p\text{-Update}(x, E(l), Z_\delta(l), \mu) \tag{5.9}$$

to select the best Δ_p archive w.r.t. $Z_\delta(l)$. To update $N(l)$ it is not recommended to store all non-dominated solutions gathered during the search since the magnitude of $N(l)$ will eventually go beyond every given threshold. Instead, it seems to be wise to select a suitable subset. In our computations we have used the archiver ArchiveUpdateTight2 from [22] which aims for a gap-free covering of the Pareto front in the sense of ϵ-dominance. Such coverings have a strong relation to Δ_p approximations as discussed in [8].

(iii) **Generational operators**. As generational operators we have chosen to use SBX crossover with $p_x = 0.9$ and polynomial mutation with $p_{mut} = 1/(n \cdot \mu)$. We refer to [1] for a thorough discussion of these operators.

(iv) **Reference front**. To compute the interpolant $S(l)$ which serves as the reference front during the run of the algorithm we have calculated not-a-knot cubic splines [77] from $N(l)$ to build two cubic polynomials (one for each objective) using the distance between two consecutive points as parametrization.

(v) **Initial seed**. To get an initial seed for Δ_p-EA we have applied a short run of NSGA-II (for the computations in this study, we have spent a budget of $10,000$ evaluations of the original MOP) since it is known that this algorithm already very quickly identifies a rough location of the Pareto front. As a side-effect, we received approximations of the extreme points of the Pareto front which we used for EP. During the run of the algorithm, EP got updated whenever better solutions were found.

Doing so, Δ_p-EA can thus be seen as a post-processing step of NSGA-II. Needless to say that any other state-of-the-art EMOA can be taken instead.

Note that the dominance relation is still used in the update process, however, that this is basically outside the EA (those core is the generation of x and the update of P). Thus, the use of the dominance within Δ_p-EA has no influence on its convergence rate.

5.5 Experiments and Results

In order to test and compare our method we have chosen to use the algorithms NSGA-II, MOEA/D and Δ_p-EMOA. NSGA-II and MOEA/D are "general purpose" EMOAs which are state-of-the-art for bi-objective problems as we consider here. Δ_p-EMOA is an IBEA that aims for Δ_p approximations of the Pareto fronts. As BOPs we have chosen to use the DTLZ ([75]) and ZDT ([78]) benchmark suites plus CONV and DENT which are widely used in literature since they represent problems with different characteristics. The definitions of all problems can be found in Table 5.1. For all problems

Algorithm 11 Δp-EA

```
Require: Extreme points $EP$ of the Pareto front ($F^*$)
 1: Choose initial population $P(1) \subset \mathbb{R}^{n \times \mu}$
 2: Choose $E(1)$ and $N(1)$ out of $P(1)$
 3: $l = 1$
 4: while Stop criterion is not met do
 5:    Compute shift distance $\delta(l)$
 6:    Compute reference front $Z_\delta(l)$ from $\{N(l) \cup EP\}$
 7:    for all $a \in P(l)$ do
 8:       Generate offspring $x \in \mathbb{R}^{n \times \mu}$ from $P(l)$ using evolutionary operators
 9:       Update $N(l)$ by $x$ leading to $N(l+1)$
10:       Update $E(l)$ by $x$ leading to $E(l+1)$
11:       if $\Delta_p(x, Z_\delta(l)) < \Delta_p(a, Z_\delta(l))$ then
12:          $P(l+1) = P(l) \cup \{x\} \backslash \{a\}$
13:       else
14:          $P(l+1) = P(l)$
15:       end if
16:    end for
17:    $l = l + 1$
18: end while
19: Return $E(l+1)$.
```

Algorithm 12 Δ_p-Update$(x, E, Z, popSize)$

```
1: $E$ = non-dominated solutions of $E \cup \{x\}$
2: if $card(E) > popSize$ then
3:    for all $e \in E$ do
4:       $h(e) = \Delta_p(E \backslash \{e\}, Z)$
5:    end for
6:    $e^* = argmin\{h(e) : e \in E\}$
7:    $E = E \backslash \{e^*\}$
8: end if
```

we have used $n = 10$ as dimension of the decision space and have used $p = 1$ for the Δ_p indicator.

For sake of a better graphical illustration we first consider the BOPs CONV, DTLZ2, DENT, and ZDT3 whose Pareto fronts are convex, concave, convex-concave, and disconnected, respectively, and have set the archive size to $\mu = 10$. For all cases, we have executed 30 independent runs and have set a limit of 4,000 function evaluations of ξ (ξ calls), which corresponds to 40,000 function evaluations of the original problem (F calls). Figure 5.3 shows the averaged performances (ξ calls vs. Δ_p value) for the different algorithms and Figures 5.4 to 5.7 show representative results of the final archives. Since $\mu = 10$ we counted 10 F calls as one ξ call. Further, since Δ_p was fed by NSGA-II we have shifted the results of NSGA-II and Δ_p-EMOA accordingly. In all four cases Δ_p-EA outperforms the other algorithms and comes close to the optimal solution while the indicator values stagnate earlier or oscillate for the 'conventional' algorithms.

For ZDT3, Δ_p-EA reveals also an oscillating behavior which is probably due to the fact that its Pareto front is disconnected and which motivates to use more advanced strategies for such problems in the future. Figure 5.7, however, shows that the novel algorithm though delivers the best approximation which is indeed quite close to the optimum.

Further, we have considered all BOPs, and now for archive size $\mu = 20$. Figure 5.8 shows the box plots for all computations and Tables 5.2 and 5.3 the related statistics for 30 independent runs using a budget of 2,000 ξ calls. In Figure 5.8 the values for NSGA-II and MOEA/D are not displayed since its values are much worse. Here, Δ_p-EA wins in all cases compared to NSGA-II, MOEA/D and Δ_p-EMOA, and the difference is significant in 13 out of 14 cases.

5.6 Proof of Concept in Three-Objectives

So far we have considered MOPs with two objectives. Though the approaches described above are in principle applicable to MOPs with any number of objectives, the computation of the interpolant $S(l)$ needs careful attention, in particular for more than two objectives. As a proof of concept that the method is also successfully applicable to MOPs with three objectives we consider here the three-objective problem DTLZ2 (Table 5.4). We have computed the interpolant $S(l)$ out of $N(l)$ via triangulation as follows: as $N(l)$ is a set three-dimensional points, we have projected all the points of $N(l)$ into a plane at a given direction. The resulting two-dimensional points were triangulated using Delaunay triangulation ([77]). This triangulation was returned to the original coordinate system (in 3D). New points (and new directions) are generated from the origin of the coordinate system to those points by projecting a grid of points

Table 5.1: BOPs considered in to test the Δ_p-EA.

MOP	Definition
CONV (convex)	$-3 \leq x_i \leq 3$ $a = \{1, -1\}$ $F_1 = \sum_{i=1}^{n}(x_i - a_0)^2, \quad F_2 = \sum_{i=1}^{n}(x_i - a_1)^2$
DENT (convex-concave)	$-1.5 \leq x_i \leq 1.5$ $F_1 = \frac{1}{2}\left[\sqrt{1+s(x)} + \sqrt{1+d(x)} + x_1 - x_2\right] + g(x)$ $F_2 = \frac{1}{2}\left[\sqrt{1+s(x)} + \sqrt{1+d(x)} - x_1 + x_2\right] + g(x),$ $s(x) = (x_1 + x_2)^2,$ quad $d(x) = (x_1 - x_2)^2,\ g(x) = \frac{17}{20}e^{-d(x)}$
DTLZ1 (linear)	$0 \leq x_i \leq 1$ $F_1 = \frac{1}{2}x_1x_2...x_{n-1}(1 + g(x_n))$ $F_2 = \frac{1}{2}x_1x_2...(1 - x_{n-1})(1 + g(x_n))$ with $g(x_n) = 100\left[\lvert x_n\rvert + \sum_{x_i \in X_n}(x_i - 0.5)^2 - cos(20\pi(x_i - 0.5))\right]$
DTLZ2 (concave)	$0 \leq x_i \leq 1$ $F_1 = (1 + g(x_n))cos(x_1\frac{\pi}{2})cos(x_2\frac{\pi}{2})...cos(x_{n-2}\frac{\pi}{2})cos(x_{n-1}\frac{\pi}{2})$ $F_2 = (1 + g(x_n))cos(x_1\frac{\pi}{2})cos(x_2\frac{\pi}{2})...cos(x_{n-2}\frac{\pi}{2})sin(x_{n-1}\frac{\pi}{2})$ with $g(x_n) = \sum_{x_i \in X_n}(x_i - 0.5)^2$
DTLZ3 (concave)	$0 \leq x_i \leq 1$ $F_1 = (1 + g(x_n))cos(x_1\frac{\pi}{2})cos(x_2\frac{\pi}{2})...cos(x_{n-2}\frac{\pi}{2})cos(x_{n-1}\frac{\pi}{2})$ $F_2 = (1 + g(x_n))cos(x_1\frac{\pi}{2})cos(x_2\frac{\pi}{2})...cos(x_{n-2}\frac{\pi}{2})sin(x_{n-1}\frac{\pi}{2})$ with $g(x_n) = 100\left[\lvert x_n\rvert + \sum_{x_i \in X_n}(x_i - 0.5)^2 - cos(20\pi(x_i - 0.5))\right]$
DTLZ4 (concave)	$0 \leq x_i \leq 1$ $F_1 = (1 + g(x_n))cos(x_1^{\pi}\frac{\pi}{2})cos(x_2^{\pi}\frac{\pi}{2})...cos(x_{n-2}^{\pi}\frac{\pi}{2})cos(x_{n-1}^{\pi}\frac{\pi}{2})$ $F_2 = (1 + g(x_n))cos(x_1^{\pi}\frac{\pi}{2})cos(x_2^{\pi}\frac{\pi}{2})...cos(x_{n-2}^{\pi}\frac{\pi}{2})sin(x_{n-1}^{\pi}\frac{\pi}{2})$ with $g(x_n) = \sum_{x_i \in X_n}(x_i - 0.5)^2$
DTLZ5 (concave)	$0 \leq x_i \leq 1$ $F_1 = (1 + g(x_n))cos(\theta_1\frac{\pi}{2})cos(\theta_2\frac{\pi}{2})...cos(\theta_{n-2}\frac{\pi}{2})cos(\theta_{n-1}\frac{\pi}{2})$ $F_2 = (1 + g(x_n))cos(\theta_1\frac{\pi}{2})cos(\theta_2\frac{\pi}{2})...cos(\theta_{n-2}\frac{\pi}{2})sin(\theta_{n-1}\frac{\pi}{2})$ with $g(x_n) = \sum_{x_i \in X_n}(x_i - 0.5)^2$ $\theta_i = \frac{\pi}{4(1+g(x_n))}(1 + 2g(x_n)x_i)$, for $i = 2, 3, ..., (n-1)$
DTLZ6 (concave)	$0 \leq x_i \leq 1$ $F_1 = (1 + g(x_n))cos(\theta_1\frac{\pi}{2})cos(\theta_2\frac{\pi}{2})...cos(\theta_{n-2}\frac{\pi}{2})cos(\theta_{n-1}\frac{\pi}{2})$ $F_2 = (1 + g(x_n))cos(\theta_1\frac{\pi}{2})cos(\theta_2\frac{\pi}{2})...cos(\theta_{n-2}\frac{\pi}{2})sin(\theta_{n-1}\frac{\pi}{2})$ with $g(x_n) = \sum_{x_i \in X_n}(x_i)^{0.1}$ $\theta_i = \frac{\pi}{4(1+g(x_n))}(1 + 2g(x_n)x_i)$, for $i = 2, 3, ..., (n-1)$
DTLZ7 (disconnected)	$0 \leq x_i \leq 1$ $F_1 = x_1$ $F_2 = x_2$ with $g(x_n) = 1 + \frac{9}{\lvert x_n\rvert}\sum_{x_i \in X_n} x_i$
ZDT1 (convex)	$0 \leq x_i \leq 1$ $F_1 = x_1$ $g(x_2, ..., x_n) = 1 + 9\sum_{i=2}^{n} x_i/(n-1)$ $h(F_1, g) = 1 - \sqrt{F_1/g}$ $F_2 = g(x_2, ..., x_n)h(F_1, g)$
ZDT2 (concave)	$0 \leq x_i \leq 1$ $F_1 = x_1$ $g(x_2, ..., x_n) = 1 + 9\sum_{i=2}^{n} x_i/(n-1)$ $h(F_1, g) = 1 - (F_1/g)^2$ $F_2 = g(x_2, ..., x_n)h(F_1, g)$
ZDT3 (disconnected)	$0 \leq x_i \leq 1$ $F_1 = x_1$ $g(x_2, ..., x_n) = 1 + 9\sum_{i=2}^{n} x_i/(n-1)$ $h(F_1, g) = 1 - \sqrt{F_1/g} - (F_1/g)sin(10\pi F_1)$ $F_2 = g(x_2, ..., x_n)h(F_1, g)$
ZDT4 (convex)	$0 \leq x_1 \leq 1$, and $x_2, ..., x_n \in [-5, 5]$ $F_1 = x_1$ $g(x_2, ..., x_n) = 1 + 10(n-1) + \sum_{i=2}^{n}(x_i^2 - 10cos(4\pi x_i))$ $h(F_1, g) = 1 - \sqrt{F_1/g}$ $F_2 = g(x_2, ..., x_n)h(F_1, g)$
ZDT6 (concave)	$0 \leq x_i \leq 1$ $F_1 = 1 - exp(-4x_1)sin^6(6\pi x_1)$ $g(x_2, ..., x_n) = 1 + 9((\sum_{i=2}^{n})/(n-1))^{0.25}$ $h(F_1, g) = 1 - (F_1/g)^2$ $F_2 = g(x_2, ..., x_n)h(F_1, g)$

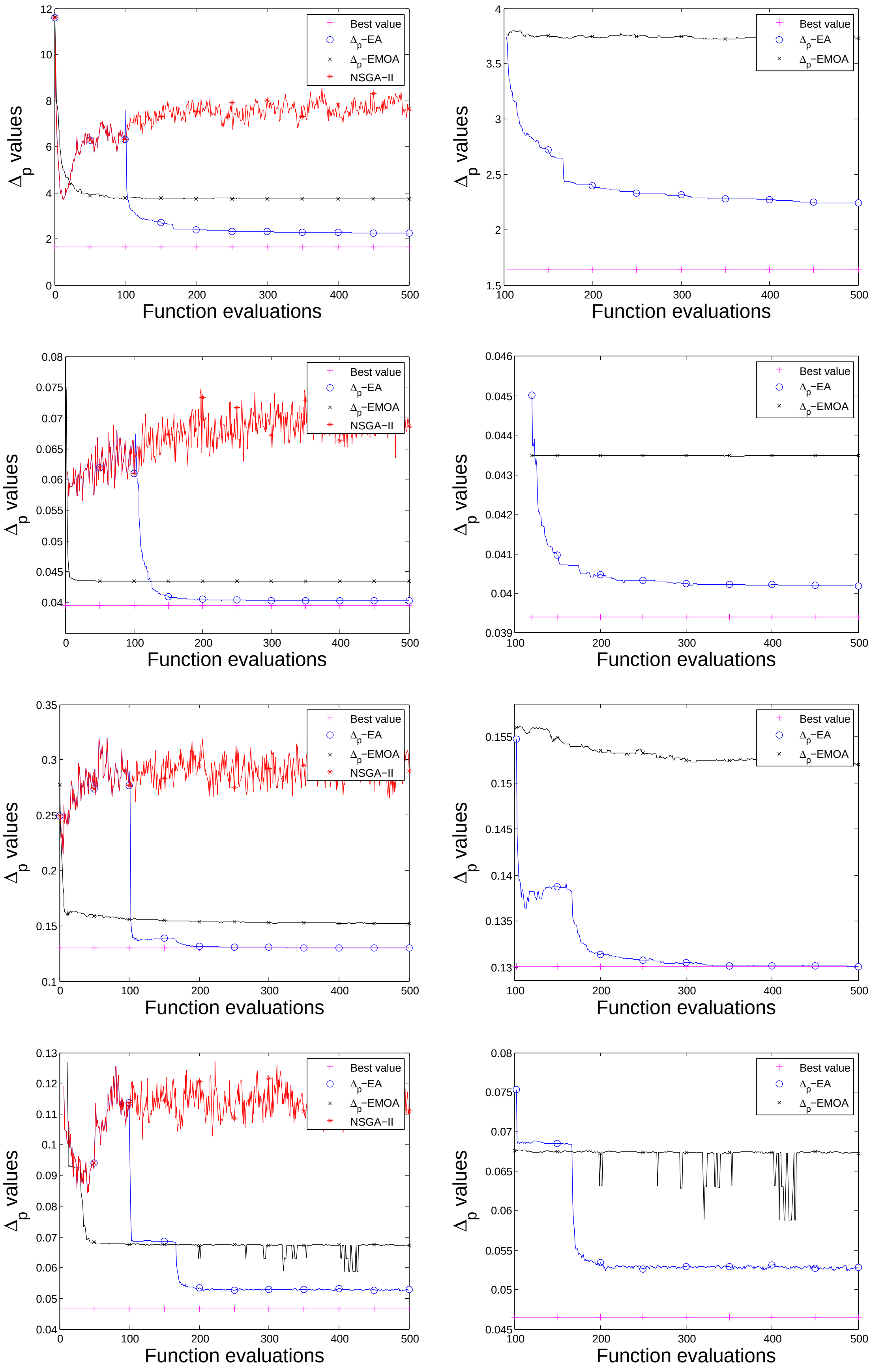


Figure 5.3: Performances (averaged) of the different algorithm on CONV, DTLZ2, DENT, and ZDT3 (from top to bottom).

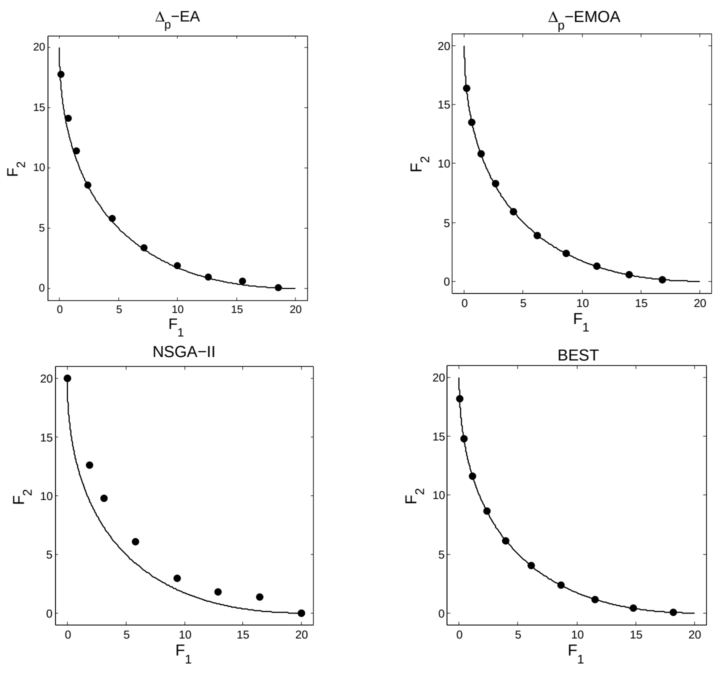


Figure 5.4: Numerical results of Δ_p-EA, Δ_p-EMOA, and NSGA-II on CONV. The best Δ_p approximation is shown right down.

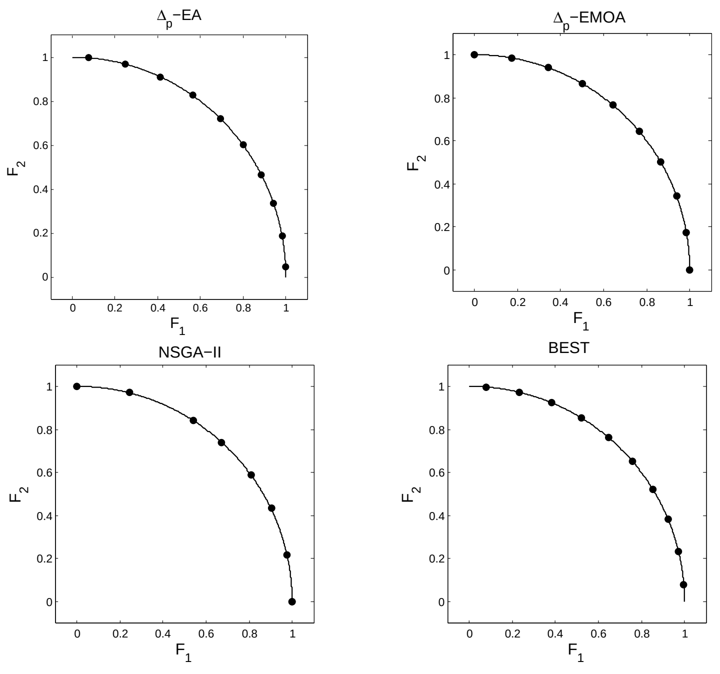


Figure 5.5: Numerical results of Δ_p-EA, Δ_p-EMOA, and NSGA-II on DTLZ2. The best Δ_p approximation is shown right down.

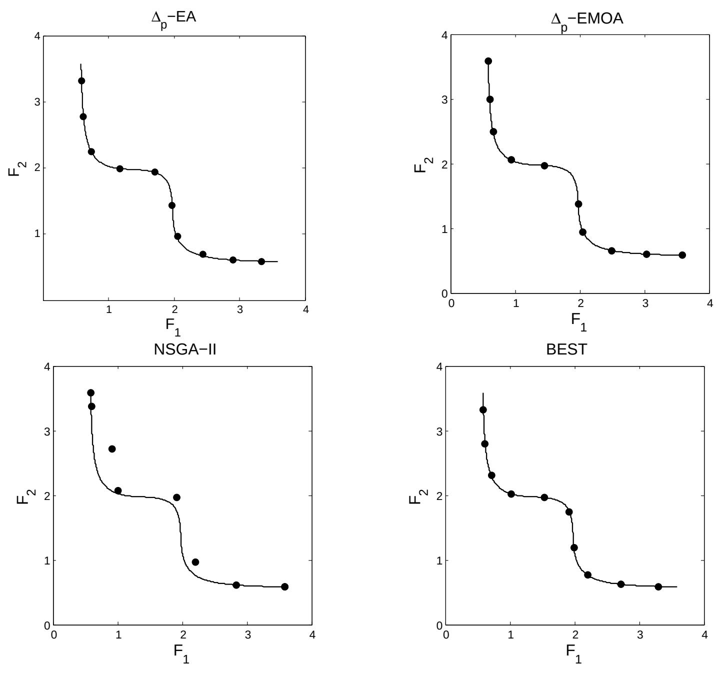


Figure 5.6: Numerical results of Δ_p-EA, Δ_p-EMOA, and NSGA-II on DENT. The best Δ_p approximation is shown right down.

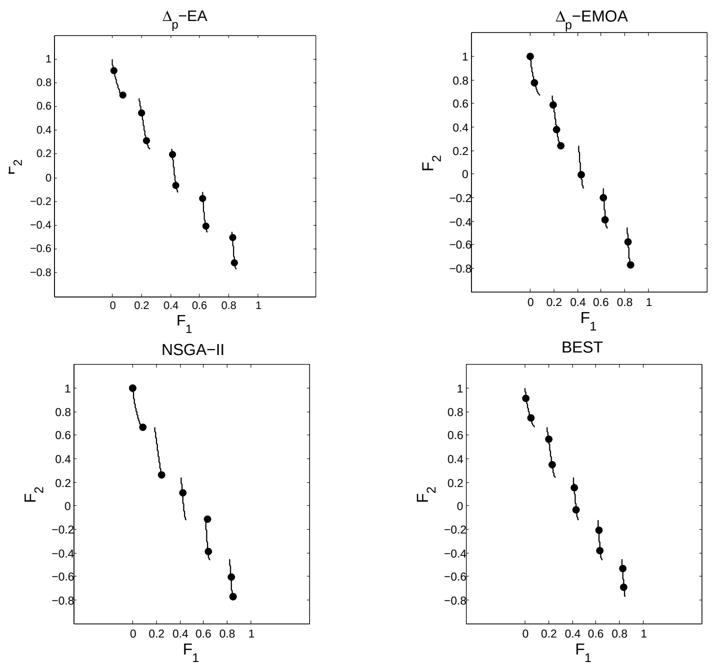


Figure 5.7: Numerical result of Δ_p-EA, Δ_p-EMOA, and NSGA-II on ZDT3. The best Δ_p approximation is shown right down.

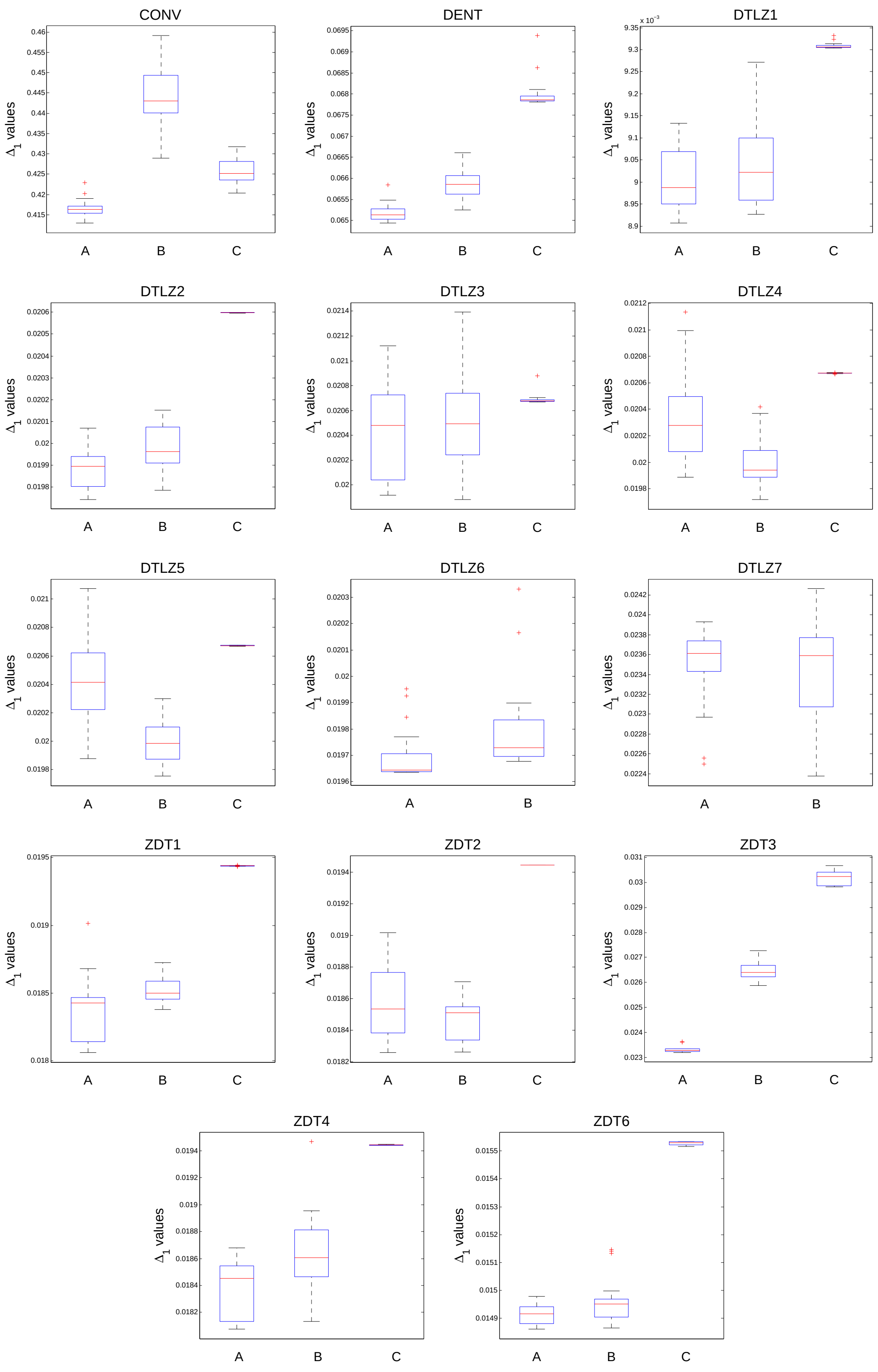


Figure 5.8: Box plots for the Δ_p approximations of Δ_p-EA (A), Δ_p-EA$_{penalization}$ (B) and Δ_p-EMOA (B). The box plots for NSGA-II and MOEA/D are out of the range of the figures.

Table 5.2: Statistics of the performance of the three algorithms on the first seven BOPs.

		CONV	**DENT**	**DTLZ1**	**DTLZ2**	**DTLZ3**	**DTLZ4**	**DTLZ5**
	Shape	convex	convex-concave	Linear	concave	concave	concave	concave
NSGA-II	Δ_p Average	0.76494636	0.10534625	0.01382910	0.02898940	0.02905807	0.02842425	0.02913877
	Δ_p S.Deviation	0.06215110	0.00531256	0.00138231	0.00175548	0.00222611	0.00302687	0.00214766
	Worst	0.85721167	0.11525818	0.01593446	0.03255413	0.03328171	0.03587941	0.03496229
	Best	0.66048310	0.09536741	0.01114975	0.02554244	0.02571242	0.02404012	0.02622696
MOEA/D	Δ_p Average	0.97260177	0.15821419	11.61402372	0.02189789	27.54896489	0.02426047	0.02207624
	Δ_p S.Deviation	0.07236135	0.00092475	3.44202546	0.00021701	6.95221611	0.00213783	0.00029538
	Worst	1.06480821	0.15987260	16.54265440	0.02240019	43.27900963	0.02836435	0.02272399
	Best	0.79801211	0.15690144	4.18500588	0.02162943	15.40691232	0.02214127	0.02167224
Δ_p-EMOA	Δ_p Average	0.43006335	0.06798755	0.00935200	0.02059835	0.02176189	0.02067266	0.02067353
	Δ_p S.Deviation	0.00660041	0.00021467	8.90104713E-05	9.69648870E-07	0.00156730	3.73585405E-06	2.70927745E-06
	Worst	0.44437900	0.06883894	0.00970879	0.02060053	0.02675508	0.02067840	0.02067947
	Best	0.42170681	0.06784579	0.00929931	0.02059716	0.02071394	0.02066283	0.02066919
Δ_p-EA	Δ_p Average	0.41657175	0.06518396	0.00900513	0.01988772	0.02044669	0.02035343	0.02042442
	Δ_p S.Deviation	0.00218202	0.00006511	7.36817886E-05	0.00009100	0.00040942	0.00035639	0.00032719
	Worst	0.42289248	0.06524919	0.00913270	0.02007071	0.02111957	0.02113673	0.02107581
	Best	**0.41291886**	**0.06513796**	**0.00890649**	**0.01974302**	0.01991761	0.01988747	0.01987420
Δ_p-EA$_{penalization}$	Δ_p Average	0.44441524	0.06585873	0.00904557	0.01998280	0.02049662	0.01999288	0.01998945
	Δ_p S.Deviation	0.00773119	0.00036606	0.00010941	0.00010558	0.00035974	0.00017775	0.00014808
	Worst	0.45918363	0.06660953	0.00927135	0.02015347	0.02139431	0.02041868	0.02029774
	Best	0.42895868	0.06525623	0.00892711	0.01978711	**0.01988137**	**0.01971656**	**0.01975173**
	BEST	0.40877654	0.06510490	0.00886243	0.01963749	0.01963749	0.01963749	0.01963749

Table 5.3: Statistics of the performance of the three algorithms on the last seven BOPs.

		DTLZ6	**DTLZ7**	**ZDT1**	**ZDT2**	**ZDT3**	**ZDT4**	**ZDT6**
	Shape	concave	disconnected	convex	concave	disconnected	convex	concave
NSGA-II	Δ_p Average	0.04276443	0.15735330	0.03023419	0.02978398	0.04422835	0.02730743	0.02686874
	Δ_p S.Deviation	0.00985763	0.19454436	0.00283671	0.00297267	0.00402699	0.00353413	0.00289392
	Worst	0.06248820	0.44734924	0.03623120	0.03677675	0.05506291	0.03620902	0.03209044
	Best	0.02976803	0.03000688	0.02598834	0.02570487	0.03947632	0.0230301953	0.02269237
MOEA/D	Δ_p Average	0.02157473	0.24020556	0.02055831	0.02041170	0.06574269	4.01820036	0.16192585
	Δ_p S.Deviation	0.00002578	0.21094751	0.00023426	0.00044651	0.06374971	1.62021290	0.11635637
	Worst	0.02161704	0.44676385	0.02117625	0.02196340	0.33644111	6.84244992	0.44384617
	Best	0.02152256	0.03414987	0.02023843	0.01994467	0.04879839	1.47255127	0.01610884
Δ_p-EMOA	Δ_p Average	0.03721674	0.23540989	0.01943981	0.01944446	0.03020516	0.01944958	0.01552745
	Δ_p S.Deviation	0.01771535	0.21528360	1.38643402E-06	9.64896030E-07	0.00032059	1.10773224E-05	6.73518960E-06
	Worst	0.07145799	0.44524852	0.01944302	0.01944790	0.03071947	0.01948630	0.01553516
	Best	0.02066646	0.02556183	0.01943732	0.01944296	0.02983031	0.01943911	0.01551512
Δ_p-EA	Δ_p Average	0.01969427	0.02350938	0.01837553	0.01857631	0.02330811	0.01837317	0.01491193
	Δ_p S.Deviation	9.96491741E-05	0.00040877	0.00024335	0.00023839	0.00011812	0.00021278	3.53063064E-06
	Worst	0.01995194	0.02393291	0.01901424	0.01901809	0.02362114	0.01867810	0.01492823
	Best	0.01973633	0.02249757	**0.01805994**	**0.01825765**	**0.02318838**	**0.01807465**	0.01491000
Δ_p-EA$_{penalization}$	Δ_p Average	0.01979564	0.02342878	0.01852716	0.01847549	0.02643472	0.01865600	0.01496129
	Δ_p S.Deviation	0.00016976	0.00057072	0.00010212	0.00012919	0.00032546	0.00028848	0.00008443
	Worst	0.02033200	0.02426751	0.01872399	0.01870573	0.02727022	0.01946957	0.01514600
	Best	**0.01967701**	**0.02237509**	0.01837827	0.01826081	0.02588671	0.01813336	**0.01486460**
	BEST	0.01963749	0.02154362	0.01753023	0.01822682	0.02300838	0.01753023	0.01480167

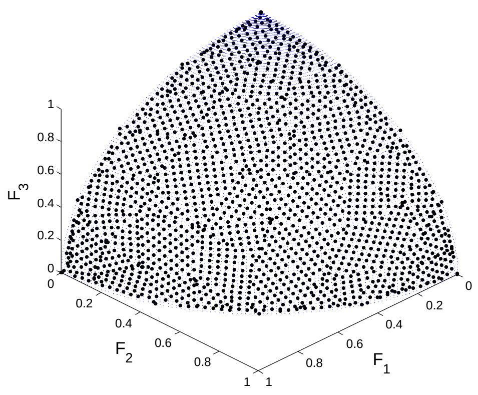


Figure 5.9: Pareto front and reference set for the three-objective DTLZ2.

to each triangle. The grid was oriented according to the direction of the greatest triangle side, and it was checked if each new grid point was inside of the triangle. The size N_G of the grid is a number given by the user. Figure 5.9 shows an example of a reference front with $N_G = 2,000$ as we have used for our computations.

As the construction of the reference set is relatively expensive, we have not done an update of $N(l)$ in every step (line 6 of Algorithm 1) but have done this update every 2,500 iterations.

Figure 5.11 shows the result of the optimization process using Δ_p-EA and the modifications described above after 2,500, 5,000, and 7,500 iterations together with the optimal archive for $\nu = 20$. Though the result is very promising, more thorough investigations and comparisons are needed which we leave for future work.

Table 5.4: Three-objective Optimization Problem considered in this example.

MOP	Definition
DTLZ2 (concave)	$0 \leq x_i \leq 1$ $F_1 = (1+g(x_n))cos(x_1\frac{\pi}{2})cos(x_2\frac{\pi}{2})...cos(x_{n-3}\frac{\pi}{2})cos(x_{n-2}\frac{\pi}{2})cos(x_{n-1}\frac{\pi}{2})$ $F_2 = (1+g(x_n))cos(x_1\frac{\pi}{2})cos(x_2\frac{\pi}{2})...cos(x_{n-3}\frac{\pi}{2})cos(x_{n-2}\frac{\pi}{2})sin(x_{n-1}\frac{\pi}{2})$ $F_3 = (1+g(x_n))cos(x_1\frac{\pi}{2})cos(x_2\frac{\pi}{2})...cos(x_{n-3}\frac{\pi}{2})sin(x_{n-2}\frac{\pi}{2})$ with $g(x_n) = \sum_{x_i \in X_n}(x_i - 0.5)^2$

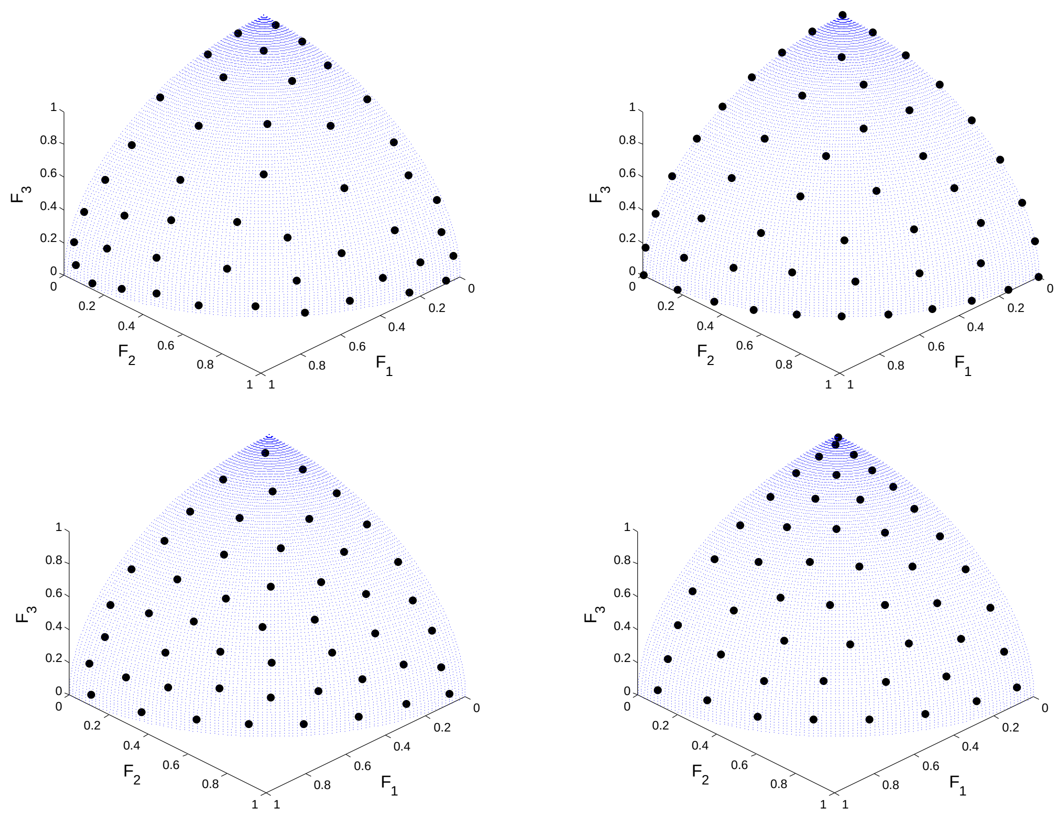


Figure 5.10: Numerical result of Δ_p-EA on the three-objective DTLZ2. 2,500 function calls (left up), 5,000 function calls (right up), 7,500 function calls (left down), and the best found solution (right down).

Table 5.5: Δ_p values for the approximations obtained for the three-objective Optimization Problem considered in this example.

	Δ_p
NSGA-II	0.08857510
MOEA/D	0.09808637
Δ_p-EA	0.06944972
Δ_p-EA$_{penalization}$	0.06913851
Best Found	0.06383316

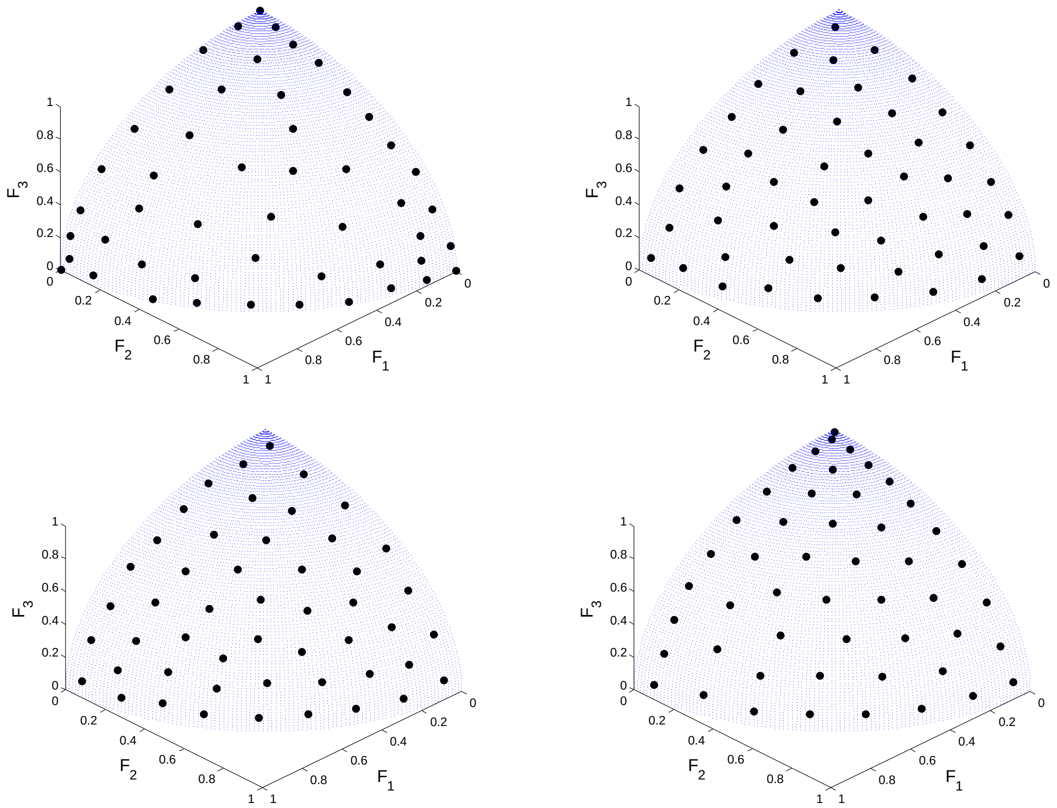


Figure 5.11: Numerical result of Δ_p-EA$_{penalization}$ on the three-objective DTLZ2. 2,500 function calls (left up), 5,000 function calls (right up), 7,500 function calls (left down), and the best found solution (right down).

Chapter 6

Applications to PID Controllers

Many real-world applications can be expressed as MOPs. Here we address a particular three objective optimization problem that is related to the design of PID controllers and where such an even distribution of the solutions along the Pareto front is desired. To this end, we use the PSEMOA and the Δ_p-EA that aim for averaged Hausdorff approximations of the Pareto front of a given MOP. Averaged Hausdorff approximations come very close to such even distributions for general MOPs, and even yield such approximations in case the Pareto front is linear. Visual observations and a comparison to other state-of-the-art EMOAs indicate that PSEMOA and Δ_p-EA are indeed good choices for the problem at hand.

We will present numerical results of the PSEMOA on a three objective optimization problem related to the design of proportional-integral-derivative (PID) controllers, and results of the Δ_p-EA on the bi-objective optimization problem taking into account only two objectives of the related design of PID controllers optimization problem.

6.1 Experiments

Here we consider a second order oscillator subject to a proportional-integral-derivative (PID) control.

$$\ddot{x} + 2\zeta\omega_n\dot{x} + \omega_n^2 x = \omega_n^2 u(t), \tag{6.1}$$

where $\omega_n = 5$, $\zeta = 0.01$,

$$u(t) = k_p\left[r(t) - x(t)\right] + k_i\int_0^t \left[r(\hat{t}) - x(\hat{t})\right] d\hat{t} - k_d\dot{x}(t), \tag{6.2}$$

$r(t)$ is a step input, k_p, k_i and k_d are the PID control gains. We consider the MOP with the control gains $\mathbf{k} = [k_p, k_i, k_d]^T$ as design parameters. The design space for the parameters is chosen as follows,

$$Q = \{\mathbf{k} \in [10, 50] \times [1, 30] \times [1, 2] \subset \mathbf{R}^3\}. \tag{6.3}$$

Peak time and overshoot are common in time domain control design objectives [79, 80, 81]. We consider the MOP to design the control gain $\mathbf{k}$,

$$\min_{\mathbf{k} \in Q} \{t_p, M_p, e_{IAE}\}, \tag{6.4}$$

where M_p stands for the overshoot of the response to a step reference input, t_p is the corresponding peak time and e_{IAE} is the integrated absolute tracking error

$$e_{IAE} = \int_0^{T_{ss}} \left| r(\hat{t}) - x(\hat{t}) \right| d\hat{t}. \tag{6.5}$$

where $r(t)$ is a reference input and T_{ss} is the time when the response is close to be in the steady state. The closed-loop response of the system for each design trial is computed with the help of closed form solutions. The integrated absolute tracking error e_{IAE} is calculated over time with $T_{ss} = 20s$.

Here, we will for sake of a comparison consider next to PSEMOA three state-of-the-art EMOAs, NSGA-II, MOEA/D, and SMSEMOA.

Experiments were conducted to compare the behavior of all considered algorithms at solving the PID problem. The algorithms were independently run 20 times on each test function for 6,000 function evaluations using population size $\mu = 100$. Standard settings were chosen for the variation operator parameters of SBX and polynomial mutation for all EMOA but MOEA/D, i.e. $p_c = 0.9$, $\eta_c = 15$, $\eta_m = 20$ and $p_m = 1/n$. MOEA/D uses Tchebycheff decomposition, a neighborhood parameter of 10, a subproblem size of 100 and differential evolution parameters $F = 0.5$ and $CR = 0.5$. The size of the external archive equals the population size for the PSEMOA. We have chosen the value of $p = 1$ within the Δ_p-Indicator [8] that measures the averaged Hausdorff distance to the Pareto front in order to minimize the influence of outliers. Note that the true Pareto front is not known analytically. In order to get the distance to this set we have computed a reference front (see Figure 6.1) that is the result of several time-consuming computations (note also that this reference front is *not* used during the run of the EMOAs).

6.2 PSEMOA Results

Figure 6.2 shows the best final approximations to the Pareto front obtained by the different algorithms. Figure 6.3 shows the boxplots of the Δ_P values at the final generation for the considered algorithms. Apparently, PSEMOA is able to obtain the best solution set, both visually and via the indicator values, giving the decision maker evenly spread solutions needed for his/her decision making process in a relatively small

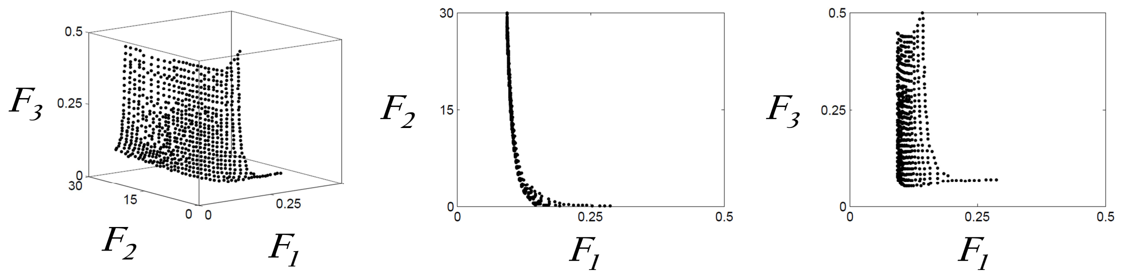


Figure 6.1: Reference front of the three-objective PID control problem.

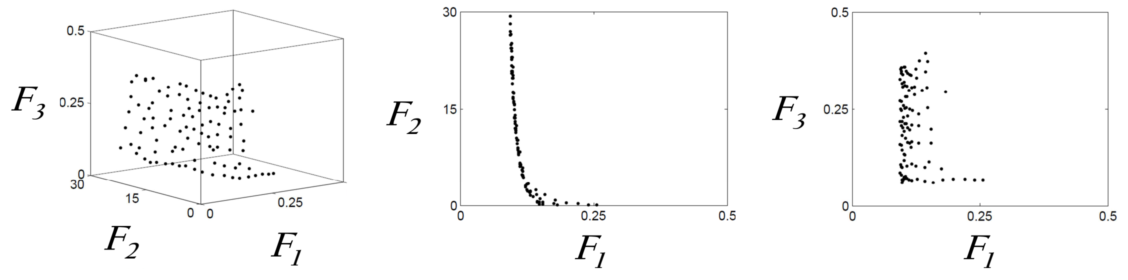
amount of time (a budget of 6,000 function calls corresponds to a total of 70 minutes running time on the standard PC we have used).

6.3 Δ_p-EA Results

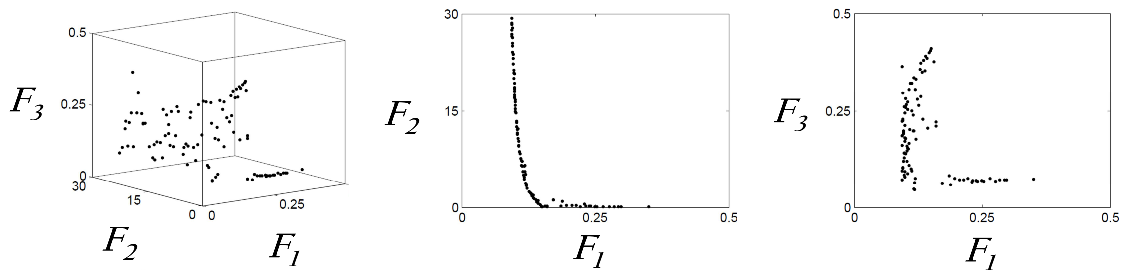
Next to the three objective PID problem we solve the bi-objective problem induced by the use of only two objectives of the second order oscillator subject to a PID control, the peak time t_p and overshoot M_p, which are common objectives in time domain control design (e.g., [79, 81]). The bi-objective that we consider here reads thus as

$$\min_{\mathbf{k}\in Q}\{t_p, M_p\}. \tag{6.6}$$

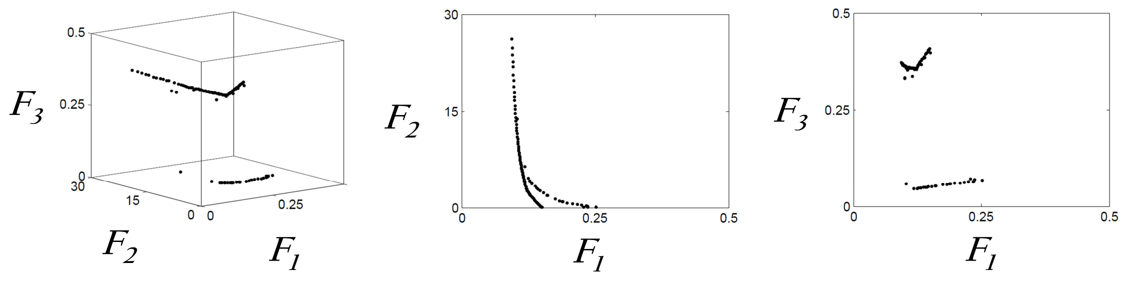
Also in this application, it is desired to present the decision maker a set of evenly distributed solutions along the Pareto front in order to give him/her an unbiased overview of optimal possibilities of the problem at hand. Figure 6.4 shows a result of the three methods on a budget of 5,000 F calls which indicates that again Δ_p-EA delivers the best results which is confirmed by the box plots shown in Figure 6.5.



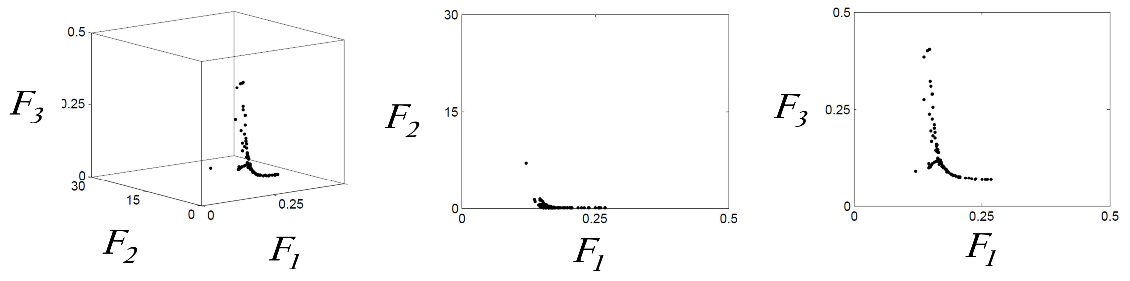
(a) PSEMOA



(b) NSGA-II



(c) SMSEMOA



(d) MOEA/D

Figure 6.2: Best approximations.

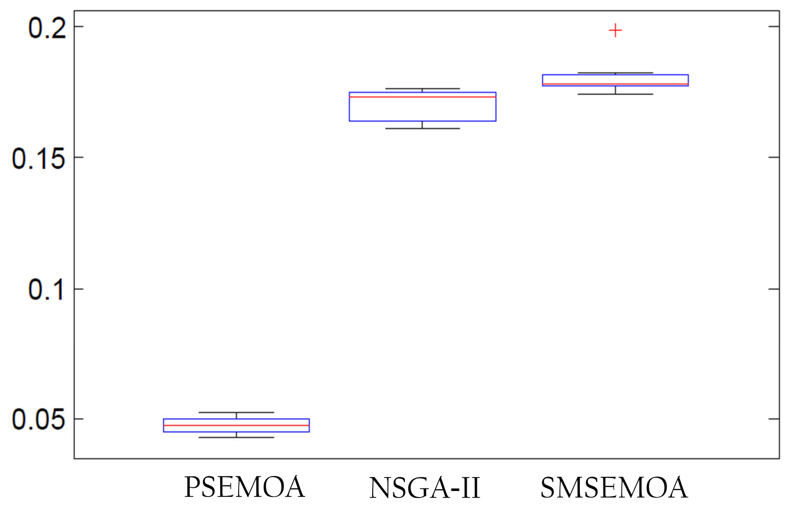


Figure 6.3: Boxplots of Δ_P values at the final generation for the considered algorithms. MOEA/D performed much worse so it does not appear in the boxplot

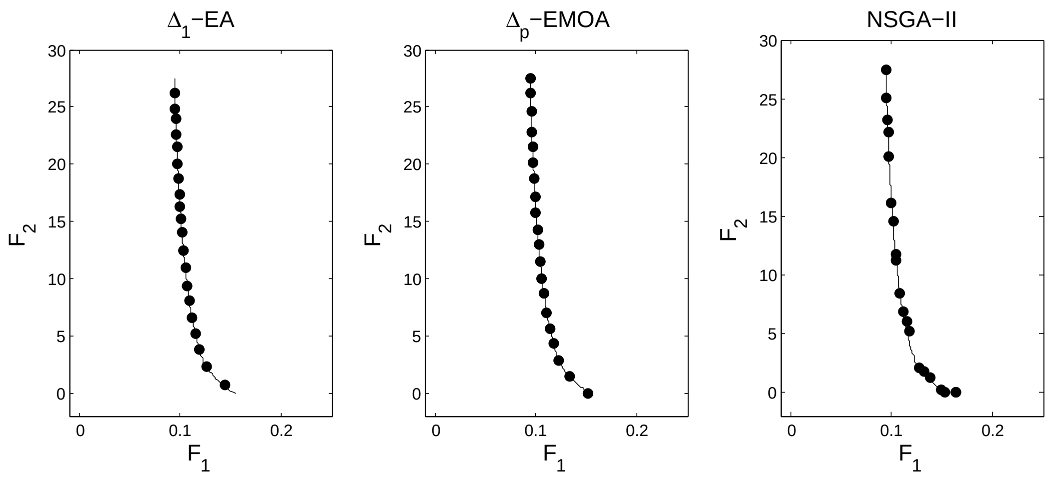


Figure 6.4: Numerical results of Δ_p-EA, Δ_p-EMOA, and NSGA-II on BOP (6.6).

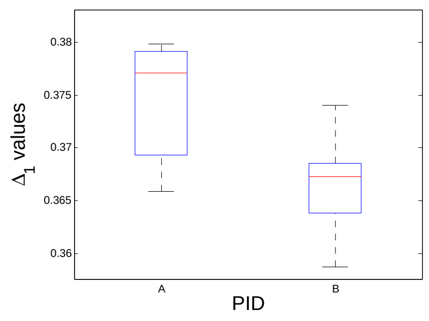


Figure 6.5: Box plots for the Δ_p approximations of Δ_p-EA (B) and Δ_p-EMOA (A) for BOP (6.6). The box plots for NSGA-II are out of the range of the figures.

Chapter 7

Conclusions and Future Work

In this thesis, we have proposed two new methods to compute Δ_p approximations of the Pareto front of a given MOP. First, an evolutionary strategy which is based on the PSA that is capable of quickly detecting a "well-spread" subset of any cardinality from a given data set. We have tested the novel strategies on three benchmark functions against other state-of-the-art (but general purpose) algorithms. The results indicate that the PSA based method indeed delivers evenly spread solutions along the Pareto front and is better than the other algorithms at least with respect to the Δ_p indicator. Second, we have proposed an algorithm based on the insight that the manipulation of $\lambda < \mu$ elements (μ=archive size) in each generation is not sufficient to obtain convergence we propose here to utilize the SOP that is induced by the given performance indicator directly. We have proposed two possible realizations tailored to Δ_p that can be addressed both by mathematical programming techniques as well as evolutionary algorithms and have further on proposed a first single objective EA for such Pareto front approximations. Numerical results on bi-objective problems with different shapes of the Pareto front and comparisons to three state-of-the-art EMOAs showed the potential of the novel approach.

Though we have focused here on Δ_p approximations, we think that the novel approach represents a promising research direction also for other performance indicators. The reason for this is the potential to construct specialized single objective EAs that, though lifted to higher dimensional space, may yield linear convergence to the set of interest.

In our computations we have restricted ourselves to relatively small archive sizes ($\mu \leq 20$). This was in order to demonstrate the effect of the method but also due to limitations of existing EAs on high-dimensional problems. We conjecture that the design of specialized algorithms for the treatment of these problems is a fruitful research field which is. Further, the treatment of MOPs with more than two objectives as SOPs has mainly been left out in this study. Though the approach presented in this paper will in principle remain the same, this does not hold for the computation of the reference sets which will need careful attention.

A promising stream of research will be to hybridize the above global approach with mathematical programming techniques in order to obtain a fast and reliable algorithm. Most promising for this purpose seem to be multi-objective continuation techniques (e.g., [82]) since they are capable of quickly finding further solutions along the set of interest from a given approximate solution.

Finally, we intend to focus on the algorithm PSEMOA since this one is not using any hypervolume calculation which is quite costly when considering more than two objectives. In particular, we conjecture that a hybridization of this algorithm with local search techniques is promising in order to increase the pressure toward the set of interest. Further, the performance of the algorithm has to be tested on problems with different objective dimensions. This, however, might not be an easy task: for less objectives the previously developed EMOAs may be beneficial. And problems with higher numbers of objectives have to be handled with care due to the dimensionality of the solution set. Nevertheless, it would be desirable to have one algorithm that copes well with all number of objectives in a certain range.

Bibliography

[1] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms.* Wiley, 2001.

[2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, April 2002.

[3] Q. Zhang and H. Li, “MOEA/D: A multi-objective evolutionary algorithm based on decomposition,” *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[4] N. Beume, B. Naujoks, and M. Emmerich, “SMS-EMOA: Multiobjective selection based on dominated hypervolume,” *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653–1669, 2007.

[5] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd ed. New York: Springer, September 2007, ISBN 978-0-387-33254-3.

[6] E. Zitzler and L. Thiele, “Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[7] M. P. Hansen and A. Jaszkiewicz, “Evaluating the quality of approximations of the non-dominated set,” 1998, tech. rep., Institute of Mathematical Modeling, Technical University of Denmark.

[8] O. Schütze, X. Esquivel, A. Lara, and C. A. Coello Coello, “Using the averaged Hausdorff distance as a performance measure in evolutionary multi-objective optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 4, pp. 504–522, 2012.

[9] A. Pottharst, K. Baptist, O. Schütze, J. Böcker, N. Fröhlecke, and M. Dellnitz, “Operating point assignment of a linear motor driven vehicle using multiobjective optimization methods,” 2004, proceedings of the 11th International Conference EPE-PEMC 2004, Riga, Latvia.

[10] K. Witting, B. Schulz, A. Pottharst, M. Dellnitz, J. Böcker, and N. Fröhleke, “A new approach for online multiobjective optimization of mechatronical systems,”

International Journal on Software Tools for Technology Transfer, vol. 10, pp. 223–231, 2008.

[11] T. Hestermeyer and O. Oberschelp, "Selbstoptimierende Fahrzeugregelung - Verhaltensbasierte Adaption," in *Intelligente mechatronische Systeme*, ser. HNI-Verlagsschriftenreihe, vol. 122. Heinz Nixdorf Institut, 2003.

[12] G. P. Liu, J. B. Yang, and J. F. Whidborne, *Multiobjective Optimisation and Control.* UK: Research Studies Press Ltd., Baldock, 2003.

[13] A. Pottharst, k. Baptist, O. Schütze, J. Böcker, N. Fröhlecke, and M. Dellnitz, *Operating point assignment of a linear motor driven vehicle using multiobjective optimization methods.* Riga Latvia: Proceedings of the 11th International Conference EPE-PEMC 2004, 2004.

[14] K. Witting, B. Schulz, M. Dellnitz, J. Böcker, and N. Fröhlecke, "A new approach for online multiobjective optimization of mechatronic systems," *International Journal on Software Tools for Technology Transfer*, vol. 10, no. 3, pp. 223–231, 2008.

[15] J. Knowles and D. Corne, "On metrics for comparing nondominated sets," in *Congress on Evolutionary Computation (CEC'2002), Vol. 1.* Piscataway (NJ): IEEE Press, 2002, pp. 711–716.

[16] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[17] D. A. V. Veldhuizen, "Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations," Ph.D. dissertation, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 1999.

[18] C. A. Coello Coello and N. Cruz Cortés, "Solving Multiobjective Optimization Problems using an Artificial Immune System," *Genetic Programming and Evolvable Machines*, vol. 6, no. 2, pp. 163–190, June 2005.

[19] K. Gerstl, G. Rudolph, O. Schütze, and H. Trautmann, "Finding evenly spaced fronts for multiobjective control via averaging Hausdorff-measure," in *Proceedings of 8th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE).* IEEE Press, 2011, pp. 1–6.

[20] H. Trautmann, G. Rudolph, C. Dominguez-Medina, and O. Schütze, "Finding evenly spaced Pareto fronts for three-objective optimization problems," in *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation II (Proceedings)*, O. Schütze *et al.*, Eds. Springer: Berlin Heidelberg, 2013, pp. 89–105.

[21] G. Rudolph, H. Trautmann, S. Sengupta, and O. Schütze, "Evenly spaced Pareto front approximations for tricriteria problems based on triangulation," in *Proceedings of 7th International Conference on Evolutionary Multi-Criterion Optimization*

(EMO 2013), E. C. Purshouse *et al.*, Eds. Springer: Berlin Heidelberg, 2013, pp. 443–458.

[22] O. Schütze, M. Laumanns, E. Tantar, C. A. C. Coello, and E.-G. Talbi, "Computing gap free Pareto front approximations with stochastic search algorithms," *Evolutionary Computation*, vol. 18, no. 1, pp. 65–96, 2010.

[23] C. Dominguez-Medina, G. Rudolph, O. Schütze, and H. Trautmann, "Evenly spaced Pareto fronts of quad-objective problems using PSA partitioning technique," in *Proceedings of 2013 IEEE Congress on Evolutionary Computation (CEC 2013)*. Piscataway (NJ): IEEE Press, 2013, pp. 3190–3197.

[24] G. Rudolph, H. Trautmann, and O. Schütze, "Homogene Approximation der Paretofront bei mehrkriteriellen Kontrollproblemen," in *Automatisierungstechnik*, vol. 60, 2012, pp. 612–621.

[25] O. Schütze, L. Laumanns, E. Tantar, C. A. Coello Coello, and E. G. Talbi, "Computing gap free Pareto front approximations with stochastic search algorithms," *Evolutionary Computation*, vol. 18, no. 1, pp. 65–96, 2010.

[26] E. Zitzler and L. Thiele, "Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study," in *Conference on Parallel Problem Solving from Nature (PPSN V)*. Berlin: Springer, 1998, pp. 292–301.

[27] H. Trautmann, T. Wagner, and D. Brockhoff, "R2-EMOA: Focused multiobjective search using R2-indicator-based selection." in *LION*, G. Nicosia and P. M. Pardalos, Eds., vol. 7997, 2013, pp. 70–74.

[28] K. Bringmann and T. Friedrich, "Convergence of hypervolume-based archiving algorithms," *Evolutionary Computation, IEEE Transactions on*, vol. 18, no. 5, pp. 643–657, Oct 2014.

[29] M. J. D. Powell, "On search directions for minimization algorithms," *Mathematical Programming*, vol. 4, no. 1, pp. 193–201, 1973.

[30] J. Nocedal and S. Wright, *Numerical Optimization*, ser. Springer Series in Operations Research and Financial Engineering. Springer, 2006.

[31] J. Decock and O. Teytaud, "Linear Convergence of Evolution Strategies with Derandomized Sampling Beyond Quasi-Convex Functions," in *EA - 11th Biennal International Conference on Artificial Evolution - 2013*, ser. Lecture Notes in Computer Science, 2013.

[32] T. Bäck, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms.* Oxford, UK: Oxford University Press, 1996.

[33] V. Pareto, *Manual of Political Economy.* The MacMillan Press, 1971.

[34] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *Computer*, vol. 27, no. 6, pp. 17–26, Jun. 1994.

[35] J. H. Holland, *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: University of Michigan Press, 1975, second edition, 1992.

[36] O. Yeniay, “Penalty function methods for constrained optimization with genetic algorithms,” *Mathematical and Computational Applications*, vol. 10, pp. 45–56, 2005.

[37] K. Deb and S. Agrawal, “A niched-penalty approach for constraint handling in genetic algorithms,” pp. 235–243, 1999.

[38] K. Deb and D. Deb, “Analysing mutation schemes for real-parameter genetic algorithms,” *Int. J. Artif. Intell. Soft Comput.*, vol. 4, no. 1, pp. 1–28, Feb. 2014. [Online]. Available: http://dx.doi.org/10.1504/IJAISC.2014.059280

[39] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[40] I. Rechenberg, *Evolutionsstrategie: optimierung technischer systeme nach prinzipien der biologischen evolution.* Frommann-Holzboog, 1973.

[41] Fogel and J. Lawrence, *Artificial Intelligence through Simulated Evolution.* New York, NY, USA: Lawrence J. Fogel [et Al.]. New York: Wiley, 1966.

[42] H.-P. Schwefel, *Numerical Optimization of Computer Models.* New York, NY, USA: John Wiley & Sons, Inc., 1981.

[43] T. Bäck and H. Schwefel, “Evolutionary computation,” vol. 1, no. 1, pp. 1–23, 1993.

[44] C. Coello Coello, “A comprehensive survey of evolutionary-based multiobjective optimization techniques,” *Knowledge and Information Systems*, vol. 1, no. 3, pp. 269–308, 1999.

[45] J. D. Schaffer, “Multiple objective optimization with vector evaluated genetic algorithms,” Ph.D. dissertation, Vanderbilt University, Nashville, Tennessee, 1985.

[46] J. D. Schaffer and J. J. Grefenstette, “Multi-objective learning via genetic algorithms,” in *Proceedings of the 9th International Joint Conference on Artificial Intelligence - Volume 1*, ser. IJCAI’85, 1985, pp. 593–595.

[47] X. Cai, Y. Li, Z. Fan, and Q. Zhang, “An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization,” *Evolutionary Computation, IEEE Transactions on*, vol. 19, no. 4, pp. 508–523, Aug 2015.

[48] S. Salomon, C. Domínguez-Medina, G. Avigad, A. Freitas, A. Goldvard, O. Schütze, and H. Trautmann, “PSA Based multi objective evolutionary algorithms,” in *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation III*, ser. Studies in Computational Intelligence, O. Schuetze, C. A. Coello Coello, A.-A. Tantar, E. Tantar, P. Bouvry, P. D.

Moral, and P. Legrand, Eds. Springer International Publishing, 2014, vol. 500, pp. 233–259.

[49] C. Dominguez-Medina, G. Rudolph, O. Schütze, and H. Trautmann, "Evenly spaced Pareto fronts of quad-objective problems using PSA partitioning technique," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*, June 2013, pp. 3190–3197.

[50] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[51] J. Knowles, L. Thiele, and E. Zitzler, "A tutorial on the performance assessment of stochastic multiobjective optimizers," Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland., Algorithm Engineering Report 214, 2006.

[52] O. Schütze, X. Esquivel, A. Lara, and C. A. Coello Coello, "Using the averaged Hausdorff distance as a performance measure in evolutionary multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 4, pp. 504–522, 2012.

[53] G. Minella, R. Ruiz, and M. Ciavotta, "A review and evaluation of multiobjective algorithms for the flowshop scheduling problem," *INFORMS J. on Computing*, vol. 20, no. 3, pp. 451–471, Jul. 2008.

[54] K. Bringmann and T. Friedrich, "Tight bounds for the approximation ratio of the hypervolume indicator," in *Parallel Problem Solving from Nature, PPSN XI*, ser. Lecture Notes in Computer Science, R. Schaefer, C. Cotta, J. Koodziej, and G. Rudolph, Eds. Springer Berlin Heidelberg, 2010, vol. 6238, pp. 607–616.

[55] N. Beume, B. Naujoks, and M. Emmerich, "Sms-emoa: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653 – 1669, 2007.

[56] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *Evolutionary Computation, IEEE Transactions on*, vol. 18, no. 4, pp. 577–601, Aug 2014.

[57] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization," in *Evolutionary Methods for Design, Optimisation, and Control.* CIMNE, Barcelona, Spain, 2002, pp. 95–100.

[58] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 6, pp. 712–731, Dec 2007.

[59] M. Emmerich, N. Beume, and B. Naujoks, "An EMO algorithm using the hypervolume measure as selection criterion," in *Evolutionary Multi-Criterion Optimization*, ser. Lecture Notes in Computer Science, C. Coello Coello, A. Hernández Aguirre, and E. Zitzler, Eds. Springer Berlin Heidelberg, 2005, vol. 3410, pp. 62–76.

[60] B. Naujoks, N. Beume, and M. Emmerich, "Multi-objective optimisation using s-metric selection: Application to three-dimensional solution spaces," in *CEC2005*, 2005, pp. 1282–1289.

[61] C. A. Rodríguez Villalobos and C. A. Coello Coello, "A new multi-objective evolutionary algorithm based on a performance assessment indicator," in *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '12. New York, NY, USA: ACM, 2012, pp. 505–512.

[62] M. Emmerich, A. Deutz, and N. Beume, "Gradient-based/evolutionary relay hybrid for computing Pareto front approximations maximizing the s-metric," in *Lecture Notes in Computer Science*, vol. 4771, 2007, pp. 140–156.

[63] M. Emmerich and A. Deutz, "Time complexity and zeros of the hypervolume indicator gradient field," in *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation III*, ser. Studies in Computational Intelligence, O. Schuetze, C. A. Coello Coello, A.-A. Tantar, E. Tantar, P. Bouvry, P. D. Moral, and P. Legrand, Eds., 2014, vol. 500, pp. 169–193.

[64] V. Sosa Hernández, O. Schutze, and M. Emmerich, "Hypervolume maximization via set based newtons method," in *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V*, ser. Advances in Intelligent Systems and Computing, A.-A. Tantar, E. Tantar, J.-Q. Sun, W. Zhang, Q. Ding, O. Schtze, M. Emmerich, P. Legrand, P. Del Moral, and C. A. Coello Coello, Eds. Springer International Publishing, 2014, vol. 288, pp. 15–28.

[65] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II." Springer, 2000, pp. 849–858.

[66] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms.* New York, NY, USA: John Wiley & Sons, Inc., 2001.

[67] S. Kukkonen and J. Lampinen, "An extension of generalized differential evolution for multi-objective optimization with constraints," in *Parallel Problem Solving from Nature - PPSN VIII*, ser. Lecture Notes in Computer Science, X. Yao, E. Burke, J. Lozano, J. Smith, J. Merelo-Guervs, J. Bullinaria, J. Rowe, P. Tio, A. Kabn, and H.-P. Schwefel, Eds. Springer Berlin Heidelberg, 2004, vol. 3242, pp. 752–761.

[68] I. Hisao, S. Yuji, T. Noritaka, and N. Yusuke, "Adaptation of scalarizing functions in moea/d: An adaptive scalarizing function-based multiobjective evolutionary algorithm," in *Evolutionary Multi-Criterion Optimization*, ser. Lecture Notes in Computer Science, M. Ehrgott, C. Fonseca, X. Gandibleux, J.-K. Hao, and M. Sevaux, Eds. Springer Berlin Heidelberg, 2009, vol. 5467, pp. 438–452.

[69] K. Miettinen, *Nonlinear Multiobjective Optimization.* New York: Springer, 1999.

[70] T. Wagner, N. Beume, and B. Naujoks, “Pareto-, aggregation-, and indicator-based methods in many-objective optimization,” in *Proceedings of the 4th International Conference on Evolutionary Multi-criterion Optimization*, ser. EMO’07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 742–756.

[71] H. Trautmann, T. Wagner, and D. Brockhoff, “R2-emoa: Focused multiobjective search using r2-indicator-based selection,” pp. 70–74, 2013.

[72] J. Mehnen, T. Wagner, and G. Rudolph, “Evolutionary optimization of dynamic multi-objective test functions,” in *Proceedings of the Second Italian Workshop on Evolutionary Computation (GSICE2)*, 2006, published on CD-ROM.

[73] S. Salomon, G. Avigad, A. Goldvard, and O. Schütze, “PSA a new scalable space partition based selection algorithm for moeas,” in *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation II*, ser. Advances in Intelligent Systems and Computing, O. Schütze, Ed. Springer Berlin Heidelberg, 2013, vol. 175, pp. 137–151.

[74] Q. Zhang and H. Li, “MOEA/D: A multiobjective evolutionary algorithm based on decomposition,” *IEEE Trans. Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[75] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, “Scalable multi-objective optimization test problems,” in *Proc. Int’l. Congress on Evolutionary Computation (CEC 2002)*, D. Fogel *et al.*, Eds., vol. 1. IEEE press, 2002, pp. 825–830.

[76] D. Brockhoff, T. Wagner, and H. Trautmann, “On the properties of the R2 indicator,” in *Proc. 14th Int’l. Genetic and Evolutionary Computation Conference (GECCO ’12)*, T. Soule *et al.*, Eds. ACM, 2012, pp. 465–472.

[77] J. Foley, A. van Dam, S. Fiener, and J. Hughes, *Computer Graphics, Principles and Practice. 2nd ed.* Addison-Wesley, 1996.

[78] E. Zitzler, K. Deb, and L. Thiele, “Comparison of multiobjective evolutionary algorithms: empirical results,” *Evol. Comput., vol. 8, no. 2, pp.173195*, 2000.

[79] Liu, G. P. and Daley, S., “Optimal-tuning PID controller design in the frequency domain with application to a rotary hydraulic system,” *Control Engineering Practice*, vol. 7, no. 7, pp. 821–830, 1999.

[80] G. P. Liu and S. Daley, “Optimal-tuning nonlinear PID control of hydraulic systems,” *Control Engineering Practice*, vol. 8, no. 9, pp. 1045–1053, 2000.

[81] Panda, S., “Multi-objective PID controller tuning for a FACTS-based damping stabilizer using Non-dominated Sorting Genetic Algorithm-II,” *International Journal of Electrical Power and Energy Systems*, vol. 33, no. 7, pp. 1296–1308, 2011.

[82] H. Wang, “Zigzag search for continuous multiobjective optimization,” *INFORMS Journal on Computing*, vol. 25, no. 4, pp. 654–665, 2013.