Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

UNIDAD ZACATENCO

DEPARTAMENTO DE COMPUTACIÓN

Esquemas para Resolver Problemas de Optimización Multi-Objetivo a Gran Escala usando Algoritmos Evolutivos

Tesis que presenta:

Luis Miguel Antonio

para obtener el grado de:

Doctor en Ciencias

Director de tesis:

Dr. Carlos Artemio Coello Coello

México, DF **Diciembre del 2017**

Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

ZACATENCO CAMPUS

COMPUTER SCIENCE DEPARTMENT

Schemes for Solving Large Scale Multi-objective Optimization Problems Using Evolutionary Algorithms

Submitted by:

Luis Miguel Antonio

as the fulfillment of the requirement for the degree of:

Ph.D. in Computer Science

Advisor:

Dr. Carlos Artemio Coello Coello

Mexico City **December, 2017**

Resumen

Muchos problemas multiobjetivo del mundo real tienen cientos e incluso miles de variables de decisión, por lo cual la escalabilidad es un tema de gran importancia. Esto, sin embargo, contrasta con los modelos evolutivos existentes para optimización multiobjetivo los cuales suelen validarse con problemas de prueba con un número relativamente bajo de variables de decisión (normalmente no más de 30). En la actualidad la investigación existente en el área de algoritmos evolutivos multiobjetivo suele enfocar su atención a la escalabilidad en el espacio de las funciones objetivo (esta área se conoce como "many-objective optimization"), sin poner mucha atención a la escalabilidad en el espacio de las variables de decisión.

En esta tesis se proponen nuevos esquemas que permite que un algoritmo evolutivo multiobjetivo sea capaz de lidiar con problemas de alta dimensionalidad en el espacio de las variables de decisin. Para validar el algoritmo propuesto se usó un conjunto de funciones de prueba que es escalable en el número de variables de decisin y se compararon resultados con respecto a algoritmos representativos del estado del arte en al rea utilizando un nmero de variables de decisión que va de las 200 hasta las 1000.

Abstract

Many real-world multi-objective optimization problems have hundreds and even thousands of decision variables, which turns scalability into a very important topic. This, however, contrasts with the currently available evolutionary models for multi-objective optimization, which are normally validated with test problems having a relatively low number of decision variables (normally, no more than 30). Nowadays, research on multi-objective evolutionary algorithms has focused on scalability in objective function space (this area is known today as “many-objective optimization”), disregarding scalability in decision variable space.

In this thesis, we propose new schemes that allows a multi-objective evolutionary algorithm to be able to deal with problems having a large dimensionality in decision variable space. In order to validate the proposed approach, we adopted a set of test problems that are scalable in decision variable space, and we compared results with respect to those obtained by other MOEAs representative of the state-of-the-art in the area using a number of decision variables that goes from 200 up to 1000.

Agradecimientos

La realización de esta tesis marca la culminación de otra etapa más en mi vida la cual no habría sido posible sin la presencia, cariño, apoyo y guía de muchas personas. Aunque quizás algún nombre no este escrito, les doy mis más sinceros agradecimientos a todos y cada uno de ellos.

Quiero dedicar este trabajo a mi esposa, Karina C. Muñoz Salas, como agradecimiento por todo su amor y cariño, pues ha sido su presencia en mi vida la experiencia más dulce y encantadora, y es cada día un motor de persistencia, mejora y esfuerzo sin el cual habría perdido mi camino. Así también lo dedico a mi hijo Johan, pues su llegada a mi vida ha sido el más hermoso regalo y cada momento desde entonces esta dedicado a él con el más profundo sentimiento de amor y cariño, pues es él mi mayor motivo y esperanza.

Agradezco a mis padres, Dionisio Miguel López y Virginia Antonio Jimenez, por el amor y apoyo incondicional que me han brindado durante toda mi vida. Su humildad y gran esfuerzo han sido el mayor ejemplo de vida que he tenido, su amor y enseñanzas me han hecho la persona que soy.

Quiero agradecer a mi asesor el Dr. Carlos A. Coello Coello por haberme dado el honor de trabajar bajo su guía, por transmitirme su pasión por la investigación y con ello sembrar en mi un gran deseo de aprender. Así también, por todas sus enseñanzas, por su apoyo y por extralimitar sus funciones de guía compartiendo importantes consejos y experiencias de vida.

Finalmente quiero agradecer a los sinodales por sus valiosas observaciones y comentarios. También agradezco al CINVESTAV por permitirme formar parte de esta gran institución y al CONACyT por brindarme un apoyo económico, permitiendo que lograra concluir los estudios de doctorado.

Este trabajo de tesis se derivó del proyecto CONACyT "Nuevos Paradigmas Algorítmicos en Optimización Evolutiva Multi-Objetivo" (Ref. 221551), cuyo Responsable Técnico es el Dr. Carlos A. Coello Coello.

Contents

Chapter 1

Introduction

In the real word there are many problems that require the optimization of two or more objective functions at the same time. These are known as multi-objective optimization problems (MOPs), and their solution involves finding a set of solutions that represent the best possible trade-offs among the objective functions being optimized. This set of solutions is called the *Pareto optimal set*, and their corresponding objective function values form the so-called *Pareto front*.

MOPs have been solved during many years, using mathematical programming techniques [1]. However, the fact that a wide variety of MOPs in real-world applications tend to be nonlinear, and perhaps even non-differentiable, has made the use of metaheuristics increasingly popular. From the many metaheuristics in current use, Evolutionary Algorithms (EAs) are the most popular in specialized literature. Multi-objective evolutionary algorithms (MOEAs) have the advantage of being population-based, which allows them to generate several elements of the Pareto optimal set in a single run, whereas mathematical programming techniques tend to produce a single element per run.

MOEAs are stochastic search techniques, inspired on the natural evolution, where a a set of candidate solutions to the problem is consider to be our population. In a general form, MOEAs work in the next way. First they create a (randomly generated) initial population. Thereafter, variation operators (crossover and mutation) are applied to the current population to generate new solutions to the MOP and, finally, a selection procedure to choose the solutions which will be part of the next generation takes place. This process is repeated for a specific number of generations.

The current practice to validate the performance of a MOEA is by using benchmark problems such as the Zitzler-Deb-Thiele (ZDT) [2], the Deb-Thiele-Laumanns-Zitzler (DTLZ) [3] and the Walking-Fish-Group (WFG) [4] test suites, which are normally adopted with a relatively low number of decision variables (usually, up to 30). However, many real-world problems have hundreds or even thousands of decision variables.

In this thesis we are interested in the study of new schemes capable of dealing with large scale (in decision variables space) MOPs. Thus, we study here the effect of scalability in MOEAs and we investigate the improvements that these algorithms can achieve. The hypothesis under which we will work is that it is possible to propose new schemes to solve in a better way (i.e., more efficiently and more effectively) large-scale (in the space of the decision variables) multi-objective optimization problems using evolutionary algorithms.

1.1 Problem statement

The motivation of this thesis is that, although in real-world applications, many MOPs have hundreds or even thousands of decision variables, the effect of the scalability in decision variables space in modern MOEAs has not been properly addressed. In fact, scalability in decision variables space is a topic that has been only scarcely studied in the context of multi-objective optimization using MOEAs. This is perhaps motivated by the fact that most researchers assume that the currently available MOEAs should be able to work properly with a large number of decision variables. Nevertheless, there exists empirical evidence that indicates that most of the currently available MOEAs significantly decrease their efficacy as the number of decision variables of a MOP increases [5, 6].

1.2 Objectives

The overall objective of this thesis is to contribute to the advancement of the state of the art with respect to large scale evolutionary multi-objective optimization (decision variables space). We aim to find out what is the main source of difficulty for solving large scale multi-objective optimization problems using evolutionary algorithms as well as to propose new schemes to deal in a better way with this sort of problems.

1.2.1 Specific goals

- Identify the best way to tackle large scale MOPs when using an evolutionary algorithm.
- Find the main causes of difficulty when dealing with MOPs having hundreds or even thousands of decision variables.
- Design a scheme able to deal with the difficulties of the large scale MOPs in a more efficient manner.

1.2.2 Expected contributions

- An analysis of the existing algorithms which allow us to identify which approaches are more suitable to be adapted for large scale multi-objective optimization.
- An analysis of the difficulties that MOEAs have when dealing with large scale MOPs
- A new MOEA specifically created to deal with large-scale MOPs.
- A detailed analysis of the advantages and disadvantages of the proposed MOEA. This analysis will consider comparisons with well-known MOEAs, standard test functions and performance measures commonly adopted in the specialized literature.

1.3 Structure of the Document

This document is organized as follows. Chapter 2 presents a brief introduction to multi-objective optimization, including basic concepts and some methods to solve multi-objective optimization problems. In Chapter 3, we describe a general multi-objective evolutionary algorithm (MOEA) as well as the way in which the performance of MOEAs is assessed. Chapter 4 presents the state of the art in large scale multi-objective optimization. In Chapters 5, we present the basic concepts related to coevolution and some of its most representative multi-objective implementations. This review is relevant because we adopt many ideas from coevolution in most of the proposals presented in this thesis. Chapter 6 presents novel evolutionary schemes to deal with large scale MOPs. Such approaches are evaluated and compared with respect to well-known MOEAs. Finally, in Chapter 7, we present our conclusions and some possible paths for future work.

Chapter 2

Basic concepts

A large number of problems that arise in academic and industrial areas have several conflicting objectives that need to be optimized simultaneously [7]; they are called multi-objective optimization problems (MOPs).

In the case of single-objective optimization, we can determine if one solution is better than another one by comparing their obtained values when evaluating the objective function. Therefore, we usually obtain a single (global) optimal solution. However, in MOPs in which the objective functions are in conflict, the notion of optimality changes. Solving MOPs implies finding good trade-offs among all the objective functions. That is to say, one wants to obtain a set of optimal solutions instead of a single one as in the case of single-objective problems.

This chapter presents the required concepts related to multi-objective optimization to understand the work presented here as well as the most commonly used optimization methods to solve MOPs. The most important aim of this chapter is that the reader familiarizes with the basic concepts, definitions and notations used in the remainder of this document.

2.1 Multi-Objective Optimization

Formally, a multi-objective optimization problem (MOP) is defined as follows [1]:

$$\text{minimize}\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})]^T \tag{2.1}$$

subject to:

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, \ldots, m \tag{2.2}$$

$$h_i(\vec{x}) = 0 \quad i = 1, 2, \ldots, p \tag{2.3}$$

where k is the number of objective functions $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$, $g_i, h_j : \mathbb{R}^n \rightarrow \mathbb{R}, i = 1, ..., m, j = 1, ..., p$ are the constraint functions of the problem and $\vec{x} = [x_1, x_2, \ldots, x_n]^T$ is the vector of decision variables. We thus wish to determine from the set Ω (where Ω is the feasible region) of all the vectors that satisfy (2.2) and (2.3) to the vector $\vec{x}^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ of solutions that are *Pareto optimal*.

Next we give a more detailed description of each component of a MOP.

Decision variables: *Decision variables* are the numerical quantities for which values are to be chosen in an optimization problem. The vector $\vec{x}$ of n decision variables is represented by: $\vec{x} = [x_1, x_2, \cdots, x_n]^T$.

Decision variables space: The *decision variables space* is the n-dimensional space of the decision variables, in which each coordinate axis corresponds with one component of vector $\vec{x}$.

Objective functions: The *objective functions* evaluate how good a given solution is. They are usually denoted as $f_i(\vec{x}) : \mathbb{R}^n \rightarrow \mathbb{R}$. In MOPs, we want to optimize more than one objective function at the same time, so in this case

[1] Without loss of generality, we will assume only minimization problems.

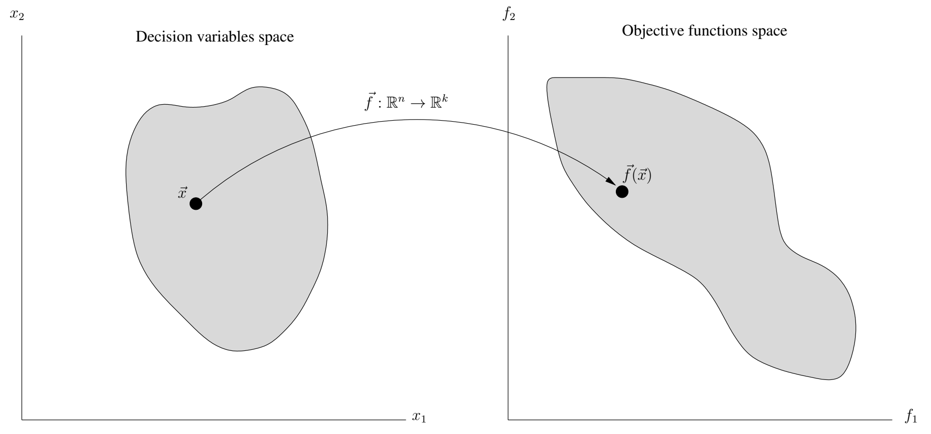
x_2
Decision variables space
$\vec{x}$
x_1
$\vec{f}:\mathbb{R}^n \rightarrow \mathbb{R}^k$
f_2
Objective functions space
$\vec{f}(\vec{x})$
f_1

Figure 2.1: Search spaces for a multi-objective optimization problem with two decision variables (x_1 and x_2) and two objective functions (f_1 and f_2). The gray area denotes the feasible region Ω.

we have an objective function vector: $\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \cdots, f_k(\vec{x})]^T$, where $\vec{f}(\vec{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^k$.

Objective functions space: *Objective functions space* is the k-dimensional space of the objective functions, in which each coordinate axis corresponds with one component of the vector $\vec{f}(\vec{x})$.

Feasible region: The *feasible region*, $\Omega \in \mathbb{R}^n$, is determined by m inequality constraint functions: $g_i(\vec{x}) \leq 0$ such that $i = 1, 2, \cdots, m$ and p equality constraint functions: $h_j(\vec{x}) = 0$ such that $j = 1, 2, \cdots, p$. We say that a vector $\vec{x}$ is *feasible*, if it satisfies all constraint functions of the problem ($\vec{x} \in \Omega$).

Figure 2.1 illustrates the above definitions for a MOP with two decision variables and two objective functions.

According to the type of the functions f, g and h, we can classify MOPs as follows:

Multi-objective linear programming: When all the objective functions and the constraint functions are linear, then the multi-objective optimization problem is called *linear*.

Nonlinear multi-objective optimization: If at least one of the objective functions or the constraint functions is nonlinear, the problem is called *nonlinear*.

Convex multi-objective optimization: The multi-objective optimization problem is *convex* if all the objective functions and the feasible region are convex.

In this thesis our main interest is to solve nonlinear unconstrained MOPs with a large number of decision variables.

2.2 Pareto Optimality

As mentioned before, it is not possible to find a single optimal solution for all the objective functions simultaneously when dealing with objectives which are in conflict with each other. In MOPs, we only produce partially ordered sets of solutions (for instance, we can say that $(0,0)^T$ is less than $(5,5)^T$, but $(0,5)^T$ and $(5,0)^T$ are incomparable).

In other words, MOPs do not have a single solution, but a set of them, representing the best possible trade-offs among all the objective functions of the problem. The notion of optimality that has been most commonly adopted in multi-objective optimization is *Pareto optimality* [8], which refers to finding the best possible trade-offs among the objective functions of a MOP. These solutions constitute the so-called *Pareto optimal set*. The image of the Pareto optimal set is called the *Pareto front*. Among the different techniques available to solve MOPs, multi-objective evolutionary algorithms (MOEAs) have become very popular, mainly due to their flexibility, capability of approximating the entire Pareto set in one single run and effectiveness in a wide variety of problems. When solving a MOP, we normally aim to minimize the distance between the approximation found and the true Pareto Front (PF), while obtaining a distribution of solutions as uniform as possible along the PF. Next, we present some important concepts related to Pareto optimality.

To describe the concept of optimality that we will adopt, we need to introduce a few additional definitions.

Definition 2.1. Pareto dominance: We say that a vector $\vec{x} = [x_1, \ldots, x_n]^T$

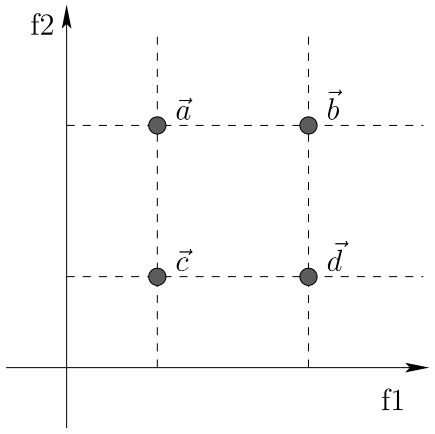

Figure 2.2: Pareto dominance relation. $\vec{c} \prec\prec \vec{b}$, $\vec{c} \prec \vec{a}$, $\vec{c} \prec \vec{d}$, $\vec{d} \prec \vec{b}$, $\vec{a} \preceq \vec{c}$, $\vec{d} \preceq \vec{c}$ and $\vec{b} \preceq \vec{d}$.

dominates vector $\vec{y} = [y_1, \ldots, y_n]^T$, denoted by $\vec{x} \prec \vec{y}$, if and only if $f_i(\vec{x}) \leq f_i(\vec{y})$ for all $i \in \{1, ..., k\}$ and there exists an $i \in \{1, \ldots, k\}$ such that $f_i(\vec{x}) < f_i(\vec{y})$.

Definition 2.2. Weak dominance: We say that a vector $\vec{x} = [x_1, \ldots, x_n]^T$ weakly dominates vector $\vec{y} = [y_1, \ldots, y_n]^T$, denoted by $\vec{x} \preceq \vec{y}$, if $\vec{x}$ is not worse than $\vec{y}$ in all objectives.

Definition 2.3. Strict dominance: We say that a vector $\vec{x} = [x_1, \ldots, x_n]^T$ scrictly dominates vector $\vec{y} = [y_1, \ldots, y_n]^T$, denoted by $\vec{x} \prec\prec \vec{y}$, if and only if $f_i(\vec{x}) < f_i(\vec{y})$ for all $i \in \{1, ..., k\}$.

Definition 2.4. Pareto optimal: A point $\vec{x}^* \in \Omega$ is Pareto optimal, if there is no other solution $\vec{x} \in \Omega$ such that $\vec{x} \prec \vec{x}^*$.

Definition 2.5. Weak Pareto optimality: A point $\vec{x}^* \in \Omega$ is weakly Pareto optimal if there is no $\vec{x} \in \Omega$ such that $\vec{x} \prec\prec \vec{x}^*$.

Figure 2.2 illustrates the Pareto dominance relation for solutions a, b, c and d of a MOP with two objective functions.

Definition 2.6. Kuhn-Tucker conditions for noninferiority: If a solution $\vec{x}$ to the general MOP is noninferior, then there exist $w_l \geq 0$, $l = 1, \cdots, k$ (w_l must be strictly positive for some $l = 1, \cdots, k$), and $\lambda_i \geq 0$, $i = 1, \cdots, m$, such that:

$$\sum_{l=1}^{k} w_l \nabla f_l(\vec{x}) - \sum_{i=1}^{m} \lambda_i \nabla g_i(\vec{x}) = 0 \quad \text{and} \tag{2.4}$$

$$\vec{x} \in \Omega$$

$$\lambda_i g_i(\vec{x}) = 0, \quad i = 1, \cdots, m$$

These conditions are **necessary** for a noninferior solution, and when all of the $f_l(\vec{x})$ are concave and Ω is a convex set, they are **sufficient** as well.

Definition 2.7. Pareto optimal set: For a given MOP, $\vec{f}(\vec{x})$, the Pareto optimal set is defined as: $\mathcal{P}^* = \{\vec{x} \in \Omega | \neg\exists \vec{y} \in \Omega : \vec{f}(\vec{y}) \prec \vec{f}(\vec{x})\}$.

Definition 2.8. Pareto front: Let $\vec{f}(\vec{x})$ be a given MOP and $\mathcal{P}^*$ the Pareto optimal set. Then, the Pareto Front is defined as: $\mathcal{PF}^* = \{\vec{f}(\vec{x}) \mid \vec{x} \in \mathcal{P}^*\}$.

Figure 2.3 shows some of the above definitions related to the Pareto dominance concept in a MOP with two decision variables and two objective functions.

Definition 2.9. Ideal objective vector: The ideal objective vector is denoted by $\vec{z}^* = [z_1^*, z_2^*, \cdots, z_k^*]^T$ and it is obtained by minimizing each of the objective functions individually subject to the constraints (i.e., $z_i^* = \min\ f_i(\vec{x})$ subject to $\vec{x} \in \Omega$).

Definition 2.10. Utopian objective vector: The utopian objective vector is denoted by $\vec{z}^{**} = [z_1^{**}, z_2^{**}, \cdots, z_k^{**}]^T$. It is an infeasible objective vector whose components are formed by $\vec{z}_i^{**} = \vec{z}_i^* - \epsilon_i$ where $\vec{z}_i^*$ is a component of the ideal objective vector and $\epsilon_i > 0$ is a relatively small but computationally significant scalar.

Definition 2.11. Nadir objective vector: The nadir objective vector is denoted by $\vec{z}^{nad} = [z_1^{nad}, z_1^{nad}, \cdots, z_k^{nad}]^T$ and its components are the upper bounds of the Pareto optimal set.

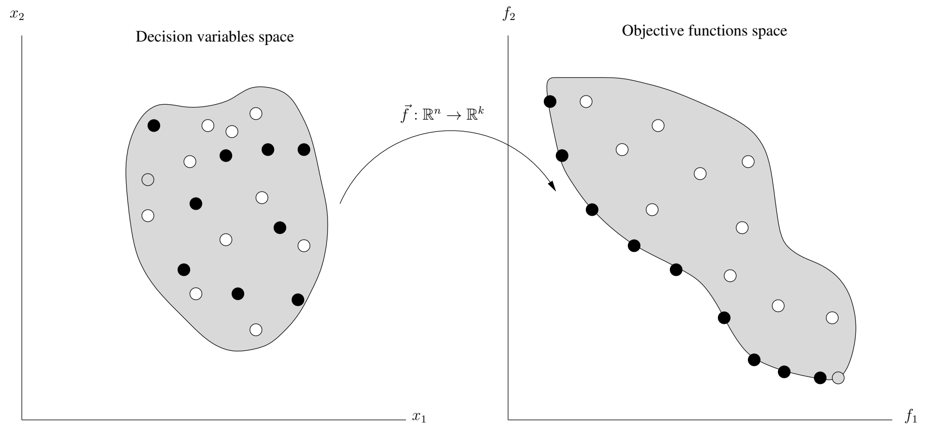
x_2
Decision variables space
x_1
$\vec{f}:\mathbb{R}^n \rightarrow \mathbb{R}^k$
f_2
Objective functions space
f_1

Figure 2.3: Pareto dominance illustration. Black points are nondominated solutions and they define the Pareto optimal set in decision variables space and the Pareto front in objective function space. White points are dominated solutions (dominated black point). The gray point is a graphic representation of a weakly Pareto optimal solution.

$\vec{z}^{nad}$ can be estimated from a payoff table which is formed by using the decision vector obtained when calculating the ideal objective vector. Row i of the payoff table displays the values of all objective functions calculated at the point where f_i obtained its minimal value. The value of the component z_j^{nad} can be estimated by using the maximum value of the column j. Figure 2.4 illustrates the ideal vector, the utopian vector and the nadir vector for a MOP with two objective functions.

2.3 Optimization Methods to Solve MOPs

In general, it is not easy to find an analytical expression of the line or surface that defines the Pareto front and in most cases, it turns out to be impossible since the size of the Pareto optimal set might be infinite. Therefore, the goal of most optimization methods is to find an approximation of the Pareto optimal front.

An approximation of the Pareto front is a subset of the objective space $\mathcal{Z}$ composed of mutually nondominated vectors (e.g., $\mathcal{A} \subset \mathcal{Z}$ such that for any two vectors $\vec{z}^1, \vec{z}^2 \in \mathcal{A}$ is true that $\vec{z}^1 \nprec \vec{z}^2$ and $\vec{z}^2 \nprec \vec{z}^1$).

Currently, it is well-accepted that the quality of an approximate Pareto front is

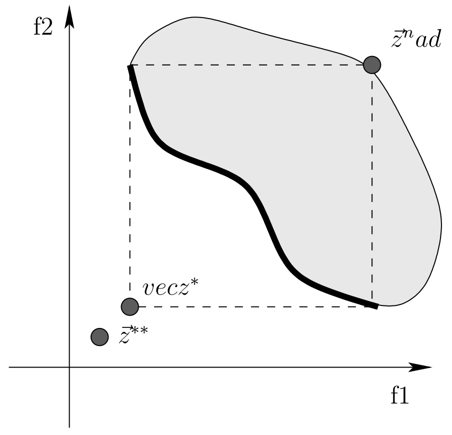

Figure 2.4: Ideal vector ($\vec{z}^*$), utopian vector ($\vec{z}^{**}$) and nadir vector ($\vec{z}^{nad}$) for a MOP with two objective functions.

determined by one of the following criteria:

- Minimizing the distance of the approximate Pareto front with respect to the true Pareto front.
- Maximizing the spread of solutions, so that one can have a distribution of vectors as smooth and uniform as possible and.
- Maximizing the number of elements of the approximate Pareto front.

The optimization methods to solve MOPs can be classified in many ways according to different criteria. Next, we present a classification of methods taken from [7].

Enumerative methods: these are the simplest search methods. They evaluate each possible solution within some finite search space. However, these methods are infeasible when the search space is too large or continuous, which is evidently the case of the large scale MOPs we are interested in solving in this thesis.

Deterministic methods: these approaches incorporate problem domain knowledge. Some examples of this type of algorithms are: *greedy methods* which make locally optimal choices and *hill-climbing methods* which use the direction of steepest ascent from the current position. Deterministic algorithms have been successfully used in solving a wide variety of problems. However, when the MOP is high-dimensional, discontinuous, multimodal, and/or NP-complete, they are often ineffective.

Stochastic methods: these methods require a function to assign fitness values to the possible solutions of the problem, as well as a mechanisms for encoding/decoding, in order to perform the mapping between the problem and the domain of the algorithm. Stochastic methods can eventually find the optimum but they cannot guarantee to find always the optimal solution. In general, they provide good solutions to a wide range of optimization problems for traditional deterministic search methods that are difficult to solve.

Since many real world multi-objective optimization problems are irregular, *stochastic search* and optimization approaches such as *simulated anneling*, *Monte Carlo methods*, *tabu search* and *evolutionary algorithms* have been adopted as alternative approaches for solving them. In this thesis, we are interested in solving MOPs using **evolutionary algorithms**.

The *operations research community* has proposed several optimization techniques (deterministic and stochastic) to solve MOPs which are known as *mathematical programming techniques.* These techniques can be *linear* or *nonlinear.* *Linear programming* is designed to solve problems in which the objective functions and all constraint relations are linear. *Nonlinear programming techniques* solve MOPs which do not meet those restrictions but usually require convex constraint functions. Finally, *stochastic programming* is used when random-valued parameters and objective functions subject to statistical perturbations are part of the problem formulation. Cohon and Marks [9] proposed one of the most popular classifications of techniques within the operations research community:

A priori methods: In this sort of techniques, a decision maker must define the preferences of the objective functions before starting the search. When the decision maker has not properly defined her/his expectations, this type of methods are not recommended.

A posteriori methods: These methods start by obtaining an approximate Pareto front, and then, this approximation is presented to the decision maker, who selects the most preferred solutions according to her/his preferences. Some disadvantages of this type of methods are that: they are computationally expensive, and it is difficult to generate well-distributed solutions along the Pareto front using them. Also, it is hard for the decision maker to select from a large set of alternatives.

Interactive methods: In these approaches, both the optimizer and the decision maker work progressively. The optimizer produces solutions and the decision maker provides preference information. These methods do not require to have any previous knowledge about the preference structure. However, the information which is presented to the decision maker should be meaningful and easy to understand.

Next, we present some instances of the aforementioned schemes.

2.3.1 A priori methods

- **Goal Programming**. The ideas of this method were originally introduced by Charles et al. [10], but the term *goal programming* was introduced by Charnes and Cooper [11]. It was one of the first methods explicitly created for multi-objective optimization. In this method, the decision maker specifies aspiration levels $\bar{z}_i (i = 1, \cdots, k)$ for each objective function and any deviations from these aspiration levels are minimized. An objective function jointly with an aspiration level forms a goal. This method has several variants (e.g., weighted and lexicographic approaches). In the weighted approach, we must solve the following problem:

$$\min \sum_{i=1}^{k} w_i |f_i(\vec{x}) - \bar{z}_i|, \quad \text{subject to:} \quad \vec{x} \in \Omega \tag{2.5}$$

In the lexicographic approach, the decision maker must specify a lexicographic order on the goals in addition to the aspiration levels. A combination of the weighted and the lexicographic approaches is quite popular. In this case, several objective functions may belong to the same class of importance in the lexicographic order. In each priority class, a weighted sum of the deviational variables is minimized.

Since goal-setting is an understandable and easy way of making decisions, goal programming is a widely used method to solve practical MOPs. However, the specification of the weighted coefficients or the lexicographic ordering may be difficult. Goal programming is not appropiate if we wish to obtain trade-offs. More details of this method can be found in [12].

- **Lexicographic Method**. In the lexicographic method [13], the objective functions are ranked in order of importance by the decision maker (from best to worst). After ranking, the most important objective function is minimized subject to the original constraints. If this problem has a unique solution, it is the solution of the whole MOP. Otherwise, the second most important objective function is minimized but a new constraint is added. The new constraint guarantees that the most important objective function preserves its optimal value. If this problem has a unique solution, it is the solution of the original MOP. Otherwise, the process goes on as above. Suppose that f_1 is the most important objective function. Then, the first problem is formulated as follows:

$$\min f_1(\vec{x}), \quad \text{subject to} \quad g_j(\vec{x}) \leq 0; \quad j = 1, 2, \cdots, m \tag{2.6}$$

Let $\vec{x}_1^*$, $f_1^* = f_1(\vec{x}_1^*)$ be the solution to the first problem (eq. 2.6) and f_2 is the second most important objective function. Then, the second problem is formulated as follows:

$$\min f_2(\vec{x}), \quad \text{subject to} \tag{2.7}$$

$$\begin{aligned} g_j(\vec{x}) \leq 0; \quad j = 1, 2, \cdots, m \\ f_1(\vec{x}) = f_1^* \end{aligned}$$

This process continues until all k objectives have been considered. The solution to the lexicographic problem is Pareto optimal. However, this method has several disadvantages. The decision maker may have difficulties in establishing an absolute order of importance among the objective functions. Also, it is very likely that the least important objective functions are not taken into consideration at all.

2.3.2 A posteriori methods

- **Weighting Method**. It was presented by Gass and Saaty [14]. This method associates to each objective function one weighting coefficient and its goal is to minimize the weighted sum of the objectives. The weighting coefficients w_i are real numbers and they are normalized. In this way, the multi-objective optimization problem is transformed into a single objective function as follows:

$$\min \sum_{i=1}^{k} w_i f_i(\vec{x}), \quad \text{subject to} \quad \vec{x} \in \Omega \tag{2.8}$$

where $w_i \geq 0$ for all $i = 1, \cdots, k$ and $\sum_{i=1}^{k} w_i = 1$. Zadeh [15] was the first to show that the third of the Kuhn-Tucker conditions for noninferior solutions implies that these noninferior solutions might be found by solving a scalar optimization problem in which the objective function is a weighted sum of the components of the original vector-valued function. Indeed, the weighting method is a simple way to generate different Pareto optimal solutions. Pareto optimality is guaranteed if the weighting coefficients are positive or the solution is unique. The weakness of the weighting method is that not all of the Pareto optimal points can be found if the problem is nonconvex. The same weakness may also occur in problems with discontinuous objective functions. More details of this method can be found in [12].

- ϵ-**Constraint Method**. This method was introduced by Haimes et al. [16]. It also follows directly from the Kuhn-Tucker conditions for noninferior solutions. The idea of this method is to minimize one (the most preferred or primary) objective function at a time, considering the other objectives as constraints bounded by some allowable levels ϵ_j. By varing theses levels ϵ_j, the nondominated solutions of the problem can be obtained. The problem to be solved is now of the form:

$$\min f_i(\vec{x}), \quad \text{subject to} \tag{2.9}$$

$$\begin{aligned} f_j(\vec{x}) \leq \epsilon_j \text{ for all } j = 1, \cdots, k, j \neq i, \\ \vec{x} \in \Omega \end{aligned}$$

  where $i = 1, \cdots, k$. Thus, every Pareto optimal solution of any MOP can be found by the ϵ-constraint method by altering the upper bounds and the function to be minimized. In fact, even the existence of duality gaps in nonconvex problems does not disturb the functioning of the ϵ-constraint method. However, computationally speaking, this method is expensive, i.e., the ϵ-constraint method needs to perform k optimizations each of the objective functions in order to generate one Pareto optimal solution.

- **Method of weighted metrics**. This method, also known as *compromise programming* [17], obtains different solutions by altering the weighted coefficients w_i in the weighted L_p- and Tchebycheff metrics. The weighted L_p-problem for minimizing distances is:

$$\min \left(\sum_{i=1}^{k} w_i |f_i(\vec{x}) - z_i^*|^p \right)^{1/p}, \quad \text{subject to} \quad \vec{x} \in \Omega \tag{2.10}$$

for $1 \leq p < \infty$. The weighted Tchebycheff problem was originally introduced by Bowman [18] and it is of the form:

$$\min \max_{i=1,\cdots,k} [w_i |f_i(\vec{x}) - z_i^*|], \quad \text{subject to} \quad \vec{x} \in \Omega \tag{2.11}$$

If $p = 1$, the problem to be solved is equal to the weighted problem except for a constant (if $\vec{z}^*$ is known globally). If $p = 2$, we have a method of least squares. As p gets larger, the minimization of the largest deviation becomes more and more important. Finally, when $p = \infty$, the only thing that matters is the weighted relative deviation of one objective function. Although the weighted Tchebycheff problem can find every Pareto optimal solution, weakly Pareto optimal solutions may also be included and they need to be filtered out. More details of this method can be found in [12].

2.3.3 Interactive methods

- **Tchebycheff Method**. This method was refined by Steuer [19] and is an interactive weighting vector space reduction method. It is easy to use and it does not require complicated information. It starts by establishing an utopian vector below the ideal vector. Then, the distance from the utopian vector to the feasible region, measured by a weighted Tchebycheff metric, is minimized. Different solutions are obtained with different weighting vectors in the metric. The solution space is reduced by working with sequences of smaller and smaller subsets of the weighting vector space. Thus, the idea is to develop a sequence of progressively smaller subsets of the Pareto optimal set until a final solution is located. At each iteration, different alternative objective vectors are presented to the decision maker and (s)he must select the most preferred of them. The feasible region is then reduced and alternatives from the reduced space are presented to the decision maker for selection. We know that some of the generated solutions may be weakly Pareto optimal. However, producing weakly Pareto optimal solutions is overcomed in the Tchebycheff method by formulating the distance problem as a lexicographic weighted Tchebycheff problem.

- **GUESS Method**. This method was proposed by Buchanan in [20]. It requires that the ideal vector $\vec{z}^*$ and the nadir vector $\vec{z}^{nad}$ are available. The general idea is to maximize the minimum weighted deviation from the nadir objective vector. In this case, the decision maker specifies a reference point (or a guess) $\vec{z}^h$ and a solution with equal proportional achievements is generated. Then, the decision maker specifies a new reference point and the iteration continues until the decision maker is satisfied with the solution produced. The scales of the objective functions are normalized:

$$\frac{z_i^{nad} - f_i(\vec{x})}{z_i^{nad} - z_i^*} \quad \text{for all} \quad i = 1, \cdots, k; \tag{2.12}$$

The weighted max-min problem to be solved is:

$$\max \quad \min_{i=1,\cdots,k} \left[\frac{1}{w_i} \frac{z_i^{nad} - f_i(\vec{x})}{z_i^{nad} - z_i^*} \right] \quad \text{subject to} \quad \vec{x} \in \Omega \tag{2.13}$$

where the weighting coefficients w_i are positive and they must not be equal to zero. If $w_i^h = \frac{z_i^{nad} - \bar{z}_i^h}{z_i^{nad} - z_i^*}$ for all $i = 1, \cdots, k$, we can write the problem to be solved in the form:

$$\max \quad \min_{i=1,\cdots,k} \left[\frac{z_i^{nad} - f_i(\vec{x})}{z_i^{nad} - z_i^h} \right] \quad \text{subject to} \quad \vec{x} \in \Omega \tag{2.14}$$

Therefore, the GUESS method is based on a trial and error procedure. The decision maker can examine what kind of an effect her or his input has on the solution obtained and then modify the input. The system does not provide any additional or supporting information about the problem to be solved. The weakness of the GUESS method is its heavy reliance on the availability of the nadir objective vector and we know that it is not easy to determine it (it is usually only an approximation). More details of this method can be found in [12].

Mathematical programming techniques have some disadvantages when solving certain types of MOPs. For instance, these techniques cannot be applicable or have a poor performance in some problems in which the objective functions are non-differentiable or obtained from a simulation model. Also, most mathematical programming techniques need to be run several times to obtain several elements of the Pareto optimal set and many of them require specific domain knowledge about the problem to be solved (e.g, the derivatives of the objectives). Additionally, some of them are very sensitive to the shape or continuity of the Pareto front. As an alternative, in recent years, a number of stochastic optimization techniques have been used to solve MOPs, such as simulated annealing, tabu search, ant colony optimization and evolutionary algorithms among many others.

Chapter 3

Evolutionary Algorithms for Multi-Objective Optimization

Evolutionary algorithms (EAs) are stochastic search techniques that operate on a population of solutions and simulate a reproduction process to generate new solutions. They adopt a selection mechanism to decide which solutions are better and use such good solutions to guide the search process. EAs are very suitable for solving MOPs, since they can find several members of the Pareto optimal set in a single run through an appropriate use of their population. Another advantage of using MOEAs is that they require very little knowledge about the problem that we want to solve and they are also less susceptible to the shape or continuity of the Pareto front than mathematical programming techniques. Furthermore, they are easy to implement, robust, and can be easily implemented in a parallel environment.

Rosenberg [21] was the first to propose the use of genetic algorithms to solve MOPs at the end of the 1960s. However, it was until 1984, when David Schaffer [22] proposed the first actual implementation of what it is now called a Multi-Objective Evolutionary Algorithm (MOEA). After that, several different algorithms have been proposed and successfully applied to a wide variety of problems [23, 24, 25, 26, 27, 28, 29, 30, 31].

In this chapter we present a description of the way in which a MOEA works, as well as the way in which we assess its performance. Also, we present the most popular MOEAs in current use, together with their advantages and disadvantages.

3.1 Multi-objective Evolutionary Algorithms

Single objective EAs and MOEAs share a similar structure. The main difference is the fitness assignment mechanism that each kind of algorithm adopts. For the case of MOEAs, they deal with more than one objective function at a time. Finding an approximate Pareto optimal front is itself a bi-objective problem with the following objectives: Minimize the distance of the generated solutions to the optimal Pareto Front and maximize the diversity among the solutions as much as possible.

Therefore, the fitness assignment mechanism of a MOEA must consider these two objectives. The general framework of a MOEA is presented in Algorithm 1. Usually, the initial population is generated in a random manner. The fitness assignment is used to rank individuals by means of a preference relation on the objective functions. Pareto optimality is in general the most common preference relation adopted in MOEAs, but it's not the only one. The selection of individuals for reproduction is normally based on the preference relation adopted for ranking the population, but a density estimator can also be adopted to generate different solutions in a single run of a MOEA.

Algorithm 1 A generic Multi-objective Evolutionary Algorithm

```
$t \leftarrow 0$
Create an initial population ($\mathcal{P}(t)$)
while Stopping criterion is not fulfilled do
    Evaluate the objective vector $\vec{f}$ for each individual in $\mathcal{P}(t)$
    Assign the fitness of each individual in $\mathcal{P}(t)$
    Select from $\mathcal{P}(t)$, a group of parents ($\mathcal{P}'(t)$), preferring the fitter ones
    Recombine individuals of $\mathcal{P}'(t)$ to obtain an offspring population ($\mathcal{P}''(t)$)
    Mutate individuals in $\mathcal{P}''(t)$
    Combine $\mathcal{P}$ and $\mathcal{P}''(t)$ and select the best individuals to get
    the new population ($P(t+1)$)
    $t \leftarrow t + 1$
end while
```

The essential elements of a MOEA are:

1. **Fitness assignment**. In a MOEA, we need an additional process to transform a fitness vector into a scalar value. Mainly, there are three schemes to carry out this process:

    - **Criterion-based**. This approach alternately chooses each of the objective functions during the selection stage, i.e., in order to select an individual or group of individuals, only one objective is considered. The Vector Evaluated Genetic Algorithm (VEGA) [22] is an example of this technique. VEGA divides the population into k subpopulations (where k is the number of objectives of the MOP) and a different objective is used to assign fitness within each subpopulation.
    - **Aggregation-based**. In this case, the objective functions are aggregated or combined into a single scalar value. During the optimization process, the parameters are systematically varied to generate different elements of the Pareto optimal set. Note that, although an aggregation-based approach can be formulated as a preference relation, the solutions are not compared in objective function space (vectors are mapped from $\mathbb{R}^k$ to $\mathbb{R}$ before the comparison).
    - **Preference-based**. In this method, a preference relation is used to induce a partial order of the population in objective function space. Then, one rank (scalar score) is assigned to each solution based on how one solution compares with respect to the other solutions. Pareto dominance is the preference relation most commonly adopted in MOEAs.
    - **Indicator-based**. This approach uses performance indicators (also called quality indicators) to assign the fitness of each individual based on how much a solution contributes to a given indicator. The hypervolume indicator has been the most popular quality indicator used for this purpose [32, 31, 33, 34, 35]. However, in recent years other quality

indicators such as $R2$ [36, 37, 38] or Δ_p [39, 40, 41] have also been adopted.

2. **Elitism**. It refers to retaining the best solutions during the optimization process since they could be lost when applying the evolutionary operators. This concept plays an important role in modern MOEAs since, along with mutation, guarantees global convergence[42]. In multi-objective optimization, the implementation of elitism is more complicated than in single-objective optimization because we have several solutions that are equally good (i.e., the elements of the Pareto optimal set) and their number can significantly grow over time. There are two main approaches to implement elitism:

    - To combine the offspring and parents populations, and then use a deterministic selection mechanism to preserve the best solutions for the next generation.
    - To mantain an external set of individuals called "archive" which stores the nondominated solutions found during the search process. It is worth emphasizing that the size of this external archive is normally bounded, because if it grows too much, it may dilute the selection pressure.

3. **Density Estimators**. As we already know, one challenge of MOEAs is to obtain a set of nondominated solutions which are well distributed along the Pareto front. For this reason, several techniques to maintain diversity in the population have been proposed. Some of them are:

    - **Fitness sharing / Niching approach**. The idea of this technique is to consider fitness as a resource that needs to be shared among individuals in the same niche. The size (or radius) of a neighborhood (or niche) is controlled through the σ_{share} value (niche radius). Then, one must count how many solutions are located within the same niche, and the fitness is decreased proportionally to the number of individuals sharing the same neighborhood [43, 44]. Formally, the shared fitness F_{Si} of individual i is defined by:

$$f_{Si} = \frac{f_i}{\sum_{j=1}^{N} \phi(d_{ij})}, \tag{3.1}$$

where f_i is the fitness of individual i, and $\phi(d_{ij})$ is the sharing function, defined by:

$$\phi(d_{ij}) = \begin{cases} 1 - \left(\frac{d_{ij}}{\sigma_{share}}\right), & d_{ij} < \sigma_{share} \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

where d_{ij} is the distance between individuals i and j. Note the following:

 - The definition of the σ_{share} parameter is critical.
 - Distances between solutions can be measured in genotypic or phenotypic space.
 - It is possible to adopt different topologies for defining neighborhoods.

- **Hypergrids**. A hypergrid divides objective function space in regions called hypercubes. Each nondominated solution occupies a hypercube. The idea is to accept nondominated solutions belonging to underpopulated hypercubes. Although the number of divisions in the hypergrid in each dimension is constant, the position and extension of the grid can be adapted during the search process.
- **Clustering**. A clustering technique partitions a set of points in groups (clusters). The idea is that each cluster contains points very similar to each other and, the points of one cluster are very different from the points of other clusters. In a MOEA, we use clustering as follows: First, we partition the population using a clustering technique. Second, we select the most representative individual of each cluster (i.e., the centroid). Finally, we remove all the other individuals in the cluster.
- **Relaxed forms of Pareto dominance**. Laumanns et al. [45] proposed a relaxed form of Pareto dominance called ϵ-dominance. ϵ-dominance defines a set of boxes of size ϵ and only one nondominated solution is retained for each box (e.g., the one closest to the lower left-hand corner if

all the objetives are being minimized). The use of ϵ-dominance guarantees that the retained solutions are nondominated with respect to all solutions generated during the run.

Next, we present the most representative MOEAs of the state of the art as well as the most popular MOEAs in current use according to a general classification based on the way fitness assignment is implemented and on their selection mechanism.

3.1.1 Non-Elitist Non-Pareto-based Methods

These type of MOEAs do not adopt Pareto dominance in their selection mechanism. These approaches are very simple nor elitism and computationally efficient, but they have no explicit mechanism to maintain diversity, which limits their applicability. Next, we present a representative example of this group.

VEGA: The Vector Evaluated Genetic Algorithm (VEGA) [22], is considered the first actual implementation of what it is now called a Multi-Objective Evolutionary Algorithm (MOEA). The parent selection in VEGA is performed in the following manner. First, the population is randomly split into k subpopulations (where k is the number of objective of the problems) of equal size. Each subpopulation is associated with a different objective function of the problem. Therefore, the individuals belonging to each objective function are ranked according to their performance in such objective. VEGA creates a mating pool by means of the proportionate selection method proposed in [46]. Thereafter, subpopulations are combined to apply the variation operators. VEGA suffers from some bias towards extreme points due to its association mechanism and when proportionate selection is used VEGA's selection mechanism behaves similarly to a linear aggregating function.Therefore, it cannot generate non-convex portions of the Pareto front. Also, VEGA's selection mechanism actually opposes the notion of Pareto optimality, because a solution that represents a good trade-off among all the objectives may not be the best in any of them so it won't be favored by its selection mechanism.

3.1.2 Non-Elitist Pareto-based Methods

These MOEAs rank populations using Pareto dominance, but do not incorporate elitism. In this case, all non-dominated individuals should get the highest rank so that they have the same survival probability. In order to maintain diversity, this kind of MOEAs adopt a *density estimator* in order to avoid convergence to a single solution, and to generate different elements of the Pareto optimal set in a single run.

Non-dominated sorting is the most common, but not the only scheme possible to rank solutions based on Pareto optimality. So, within this group of MOEAs we describe those that adopt a Pareto-based selection mechanism, but that do not incorporate elitism. These early Pareto-based MOEAs are easy to implement, but they are not very effective in problems having more than two or three objectives.

MOGA: The Multi-Objective Genetic Algorithm (MOGA) is an example of a non-elitist Pareto-based method. First presented in [27], MOGA ranks individuals according to the number of solutions that dominate them. According to this ranking, the fitness is computed and aims to be maximized (ranging in the interval $[0, 1]$). MOGA adopts a polynomial equation to estimate the value of the niche radius, for a fitness sharing density estimator. Parent selection is then restricted to individuals with similar fitness values.

NSGA: The Non-dominated Sorting Genetic Algorithm (NSGA) [24] is another example of this kind of approaches. NSGA adopts a large dummy fitness value assigned to the first layer of non-dominated solutions (i.e., those which are non-dominated with respect to the entire population). The first layer is then removed, so that a second set of non-dominated solutions is identified. Such solutions receive dummy fitness values which are lower than those from the first layer. This process continues until the whole population has been classified. With this idea, fitness values are shared among solutions lying in the same layer. Parents are chosen using a stochastic remainder proportionate selection based on the individuals' fitness values. NSGA adopts a fitness sharing strategy in decision variable space.

NSGA needs of specifying the parameter σ_{share} and has a high computational complexity of $O(k|P|^3)$, where $|P|$ is the population size and k is the number of objectives.

3.1.3 Elitist Pareto-based Methods

These MOEAs have a Pareto-based selection method, but also incorporate elitism The density estimators adopted in this group are also more sophisticated and, in some cases, even parameter-free. In fact, some of the MOEAs in this group are still in use today. Their main limitation is that most of them are not effective when dealing with problems having more than 3 objectives (the so-called *many-objective optimization problems*). Next, we give some examples of such approaches.

SPEA: The Strength Pareto Evolutionary Algorithm (SPEA) [25], incorporates elitism through the use of an external archive containing the non-dominated solutions, generated during the search process. This archive is pruned when it exceeds a (pre-defined) size using the average linkage method (i.e., a clustering method). Parent selection is accomplished by binary tournaments, where the union of the main population and the external archive is performed. The strenght of an individual in the external archive is computed as the number of population members that it covers divided by the size of the population, plus one. The fitness of idividuals of the main population corresponds to the accumulated strengths of the external individuals that cover it, plus one. Here, the *cover* relation relies on Pareto dominance, in the next way: given $\mathbf{x}, \mathbf{y} \in \mathcal{X}$. it is said that $\mathbf{x}$ covers $\mathbf{y}$ iff $\mathbf{x} \prec \mathbf{y}$ or $\mathbf{x} = \mathbf{y}$.

The aim of this fitness assignment mechanism is to maintain diversity in the population. Ideally, the individuals of the external archive will cover the same number of population members. However, this only works if the external archive is uniformly distributed by the clustering method.

NSGA-II: Another MOEA within this category is the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [29]. This approach alleviates the main drawbacks of the original NSGA. A more efficient non-dominated sorting algorithm is implemented and fitness sharing is replaced by *crowding distance*, which operates in objective space and can be seen as the perimeter of the cuboid formed by the nearest neighbors surrounding a particular solution. The crowding distance is computed as the average distance between the two points on either side of a solution along each objective (solutions need to be sorted with respect to one objective). In NSGA-II, instead of fitness values, a preference relation is used, favoring those individuals with lower ranks and higher crowding distances. For the mating pool selection, binary tournaments are adopted. The survival selection is elitist since the best half from the union of the parents and the offspring populations is retained.

3.1.4 Elitist Non-Pareto-based Methods

These MOEAs also incorporate elitism but their selection mechanism is not based on Pareto dominance. Next we give some examples and subgroups for this classification.

Decomposition-based Methods: These approaches adopt a MOP transformation. They decompose a MOP into several single-objective subproblems, which are solved simultaneously. Each subproblem is associated with a different target direction or weight vector in order to obtain a wide range of solutions. For this sake, they adopt a set of weight vectors uniformly distributed in the $[0, 1]^k$ space. The most famous approach of this category is the Multi-Objective Evolutionary Algorithm based on Decomposition (MOEAD), proposed by Zhang and Li [30].

MOEA/D decomposes the MOP into a set of single-objective subproblems and solves these subproblems simultaneously using an evolutionary algorithm. Such decomposition is based on a set of weights each of which corresponds to a single subproblem. Each weight vector is used as a search direction to define a scalar function. For this sake, the so called Tchebycheff decomposition is the most widely used. Given a weight vector $\lambda = [\lambda_1, \ldots, \lambda_n]^T$ the corresponding subproblem is defined as:

$$\text{minimize} \quad g^{te}(x|\lambda, z^*) = \max_{1 \leq i \leq n} \lambda_i |f_i(x) - z_i^*| \tag{3.3}$$

where z^* is the reference point chosen as the minimum of the objective function values found during the evolutionary process. The main advantage of the Tchebycheff approach is that it works regardless of the shape of the Pareto front, while other decomposition approaches (such as the weighted sum approach) only work for convex Pareto fronts. The weights are also used to define neighborhoods of the subproblems. The neighborhood relations among these subproblems are defined based on the distances between their aggregation coefficient vectors. At each generation, a new individual is generated and evaluated using its own neighborhood of weights, with the idea that any information about these closest weight vectors should be helpful for optimizing the current individual's subproblem. Once this new individual is created, it is compared to its parent and in case the offspring is better, it replaces the parent. Moreover, it is also compared to other individuals in its neighborhood and is allowed to replace some of them. Therefore, at each generation, the population is composed of the best solution found so far (i.e., since the start of the run of the algorithm) for each subproblem.

Indicator-based Methods: Performance indicators [47] were developed to evaluate the quality of an approximation to the Pareto optimal front, regarding convergence and/or diversity. Also, indicators have been mainly used to compare the effectiveness of optimizers, and recently, they have been adopted into the selection mechanism of MOEAs.

The $\mathcal{S}$ Metric Selection Evolutionary Multi-objective Algorithm (SMS-EMOA) [31] is an example of such idea. SMS-EMOA adopts NSGA-II components along an archiving strategies proposed in [48]. Parent selection is performed randomly and the survival selection mechanism relies on a $(\mu + 1)$ scheme. The core idea of this algorithm is to integrate new solutions into the population if replacing a member of the population increases the hypervolume value of the entire population. SMS-EMOA ranks the population according to NSGA-II, and the density estimator corresponds to the hypervolume contribution. At each iteration, the individual belonging to the worst rank and having the lowest hypervolume contribution is removed from the population. It is worth mentioning that the hypervolume contribution is calculated only when there is more than one solution having the worst rank.

Reference Point-based Methods: These approaches use predefined reference points as a guide for the population, with the aim of ensuring diversity. Therefore, they are usually coupled with other strategies to achieve convergence (e.g., Pareto dominance or an indicator-based mechanism). The reference points can either be supplied by the user or predefined in a structured manner by some predefined mechanism.

The most representative MOEA within this group is the Non-dominated Sorting Genetic Algorithm III (NSGA-III) [49], which is an extension of NSGA-II, modified to deal with many-objective optimization problems. NSGA-III keeps the non-dominated sorting algorithm for ranking the population but replaces the crowding distance by a niching strategy that uses a set of reference points, which are adaptively updated according to the extent of the population. In NSGA-III, the population is normalized and associated with the lines passing through the origin and the reference points. Those individuals having the closest perpendicular distance to segregated lines are chosen for the next generation.

NSGA-III's parent selection mechanism is conducted by random sampling.

3.2 Performance Assessment of Multi-objective Optimizers

The comparison of the performance of different MOEAs is an important issue in multi-objective optimization in order to measure the performance of every approach. The notion of performance includes both the **quality of the outcome and the computational cost for generating this outcome**. Regarding the latter aspect, we can monitor either the number of fitness evaluations or the overall runtime on a particular computer. In this sense, there is no difference between single and multi-objective optimization. However, in the quality aspect, there are some differences:

- In single-objective optimization, we can define quality by means of the objective function: the smaller (to minimize) or larger (to maximize) value is a better solution.
- In multi-objective optimization, we can use the concept of Pareto dominance but there is the possibility that two solutions are incomparable.

However, it gets even more complicated when we compare two sets of solutions because some solutions in either set may be dominated by solutions in the other set, while others may be incomparable. According to the goals of a MOEA, we need to consider the following aspects to design a good quality measure:

- Minimize the distance of the Pareto front approximation produced by our algorithm with respect to the true Pareto front (assuming we know its location),
- Maximize the spread of solutions found, so that we can have a distribution of vectors as smooth and uniform as possible.
- Maximize the number of elements of the approximation of the Pareto-optimal set obtained by the MOEA.

Several quality measures have been proposed. The most popular are unary quality measures. In this case, the measure assigns each approximation set a number that reflects a certain quality aspect, and usually a combination of them is used, e.g., [50, 51]. Other methods are based on binary quality measures, which assign numbers to pairs of approximation sets, e.g., [52]. Finally, we have the attainment function approach [53], which consists of estimating the probability of attaining arbitrary goals in objective space from multiple approximation sets.

For the purposes of this section it is sufficient to deal with objective function space. For each objective vector $\vec{z} = [z_1, z_2, \cdots, z_k]$, there is a decision vector $\vec{x}$ such that $\vec{z} = \vec{f}(\vec{x}) = [f_1(\vec{x}), \cdots, f_k(\vec{x})]$. Also, we consider the outcome of a MOEA (or other heuristic) as a set of incomparable solutions, which will be denoted as **approximation set**.

Definition 3.1. Let $\mathcal{A} \subseteq \Omega$ be a set of objective vectors. $\mathcal{A}$ is called an **approximation set** if any element of $\mathcal{A}$ does not weakly dominate any other objective vector in $\mathcal{A}$. The set of all approximation sets is denoted by Ω.

It is hard to evaluate an approximation set considering all aspects such as closeness to the Pareto optimal set, diversity, etc. However, we can make statements about the quality of approximation sets in comparison to other approximation sets.

Weak dominance ($\mathcal{A} \preceq \mathcal{B}$) means that any objective vector in $\mathcal{B}$ is weakly dominated by a vector in $\mathcal{A}$. In this case, we cannot say that $\mathcal{A}$ is better than $\mathcal{B}$. Instead, the relation $\rhd$ can be used. If $\mathcal{A}$ weakly dominates $\mathcal{B}$, then either $\mathcal{A}$ is better than $\mathcal{B}$ or they are equal. In the same Figure, we can see that $\mathcal{A}_1$ is better than $\mathcal{A}_2$ and $\mathcal{A}_3$ and, $\mathcal{A}_2$ is better than $\mathcal{A}_3$. *The dominance relation* ($\mathcal{A} \prec \mathcal{B}$) is a straightforward extension of Pareto dominance to approximation sets. It does not allow that two objective vectors in $\mathcal{A}$ and $\mathcal{B}$ are incomparable, and therefore is stricter than what we usually require. $\mathcal{A}_1$ and $\mathcal{A}_2$ dominate $\mathcal{A}_3$, but $\mathcal{A}_1$ does not dominate $\mathcal{A}_2$. *Strict dominance* stands for the highest level of superiority and means an approximation set is superior to another approximation set in the sense that for any objective vector in the latter there exists a vector in the former that is better in all objectives. For example, $\mathcal{A}_1$ strictly dominates $\mathcal{A}_3$, but $\mathcal{A}_2$ does not since the objective vector (11,4) is not strictly dominated by any objective in $\mathcal{A}_2$. Note that there is a natural ordering among the relations: $\mathcal{A} \prec\prec \mathcal{B} \Rightarrow \mathcal{A} \prec \mathcal{B} \Rightarrow \mathcal{A} \rhd \mathcal{B} \Rightarrow \mathcal{A} \preceq \mathcal{B}$.

3.2.1 Quality indicators

Quality measures have been introduced to compare the outcomes of multi-objective optimizers in a quantitative manner. They map approximation sets to the set of real numbers.

Definition 3.2. Quality Indicator An m-ary quality indicator I is a function $I : \Omega^m \rightarrow \mathbb{R}$, which assigns each vector $(\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_m)$ of m approximation sets a real value $I(\mathcal{A}_1, \cdots, \mathcal{A}_m)$.

Not a single indicator but rather a combination of different quality indicators is often used in order to assess approximation sets. Van Veldhuizen and Lamont [50] applied a combination of three indicators $\mathbf{I} = (I_{GD}, I_S, I_{ONVG})$, where $I_{GD}(\mathcal{A})$ denotes the average distance of objective vectors in $\mathcal{A}$ to the Pareto optimal set, $I_S(\mathcal{A})$ measures the variance of distances between neighboring objective vectors in $\mathcal{A}$, and $I_{ONVG}(\mathcal{A})$ gives the number of elements in $\mathcal{A}$.

An **interpretation function** maps vectors of real numbers to booleans, e.g., we would define $E(I_{GD}(\mathcal{A}), I_{GD}(\mathcal{B})) := (I_{GD}(\mathcal{A}) = 0 \quad \wedge \quad I_{GD}(\mathcal{B}) > 0)$. E is true if and only if $I_{GD}(\mathcal{A}) = 0$ and at the same time $I_{GD}(\mathcal{B}) > 0$ (all objective vectors in $\mathcal{A}$ have a zero distance to the Pareto optimal set P, and therefore $\mathcal{A} \subseteq \mathcal{P}$ and $\mathcal{B} \npreceq \mathcal{A}$ for any approximation set $\mathcal{B} \nsubseteq \mathcal{P}$). A combination of one or more quality indicators, $\mathbf{I}$, and an interpretation function E is also called a **comparison method** $C_{\mathbf{I},E}$. The comparison method is $C_{I_{GD},E}(\mathcal{A}, \mathcal{B}) = E(I_{GD}(\mathcal{A}), I_{GD}(\mathcal{B}))$ and the conclusion is that $C_{I_{GD},E}(\mathcal{A}, \mathcal{B}) \Leftrightarrow \mathcal{A} \subseteq \mathcal{P} \wedge \mathcal{B} \nsubseteq \mathcal{P} \wedge \mathcal{B} \npreceq \mathcal{A}$. The comparison method is formally defined as follows:

Definition 3.3. Comparison Method Let $\mathcal{A}, B \in \Omega$ be two approximation sets, $\mathbf{I} = (I_1, I_2, \cdots, I_k)$ a combination of quality indicators, and $E : \mathbb{R}^k \times \mathbb{R}^k \rightarrow$

$\{false, true\}$ an interpretation function which maps two real vectors of length k to a Boolean value. If all indicators in $\mathbf{I}$ are unary, the comparison method $C_{\mathbf{I},E}$ defined by $\mathbf{I}$ and E is a function of the form

$$C_{\mathbf{I},E}(\mathcal{A}, B) = E(\mathbf{I}(\mathcal{A}), \mathbf{I}(B)) \tag{3.4}$$

where $\mathbf{I}(\mathcal{A}') = (I_1(\mathcal{A}'), I_2(\mathcal{A}'), \cdots, I_k(\mathcal{A}'))$ for all $\mathcal{A}' \in \Omega$. If $\mathbf{I}$ contains only binary indicators, the comparison method $C_{\mathbf{I},E}$ is defined as a function of the form

$$C_{\mathbf{I},E}(\mathcal{A}, B) = E(\mathbf{I}(\mathcal{A}, B), \mathbf{I}(B, \mathcal{A})) \tag{3.5}$$

where $\mathbf{I}(\mathcal{A}', B') = (I_1(\mathcal{A}', B'), I_2(\mathcal{A}', B'), \cdots, I_k(\mathcal{A}', B'))$ for all $\mathcal{A}', B' \in \Omega$.

3.2.2 Linking comparison methods and dominance relations

It is important to know if a comparison method $C_{I,E}(\mathcal{A}, \mathcal{B})$ is a sufficient condition to say that $\mathcal{A}$ is better than $\mathcal{B}$, i.e., $C_{I,E}(\mathcal{A}, \mathcal{B}) \Rightarrow \mathcal{A} \rhd \mathcal{B}$, and, if $C_{I,E}(\mathcal{A}, \mathcal{B})$ is, in addition, a necessary condition for $\mathcal{A} \rhd \mathcal{B}$, i.e., $C_{I,E}(\mathcal{A}, \mathcal{B}) \Leftrightarrow \mathcal{A} \rhd \mathcal{B}$. The compatibility and completeness terms are used in order to characterize a comparison method.

Definition 3.4. Compatibility and Completeness Let $\blacktriangleright$ be an arbitrary binary relation on approximation sets. The comparison method $C_{\mathbf{I},E}$ is denoted as $\blacktriangleright$ $-$*compatible* if either for any $\mathcal{A}, \mathcal{B} \in \Omega$

$$C_{\mathbf{I},E}(\mathcal{A}, \mathcal{B}) \Rightarrow \mathcal{A} \blacktriangleright \mathcal{B}$$

or for any $\mathcal{A}, \mathcal{B} \in \Omega$

$$C_{\mathbf{I},E}(\mathcal{A}, \mathcal{B}) \Rightarrow \mathcal{B} \blacktriangleright \mathcal{A}$$

The comparison method $C_{\mathbf{I},E}(\mathcal{A}, \mathcal{B})$ is denoted as $\blacktriangleright$ $-$*complete* if either for any

$\mathcal{A}, \mathcal{B} \in \Omega$

$$\mathcal{A} \blacktriangleright \mathcal{B} \Rightarrow C_{\mathbf{I},E}(\mathcal{A}, \mathcal{B})$$

or for any $\mathcal{A}, \mathcal{B} \in \Omega$

$$\mathcal{B} \blacktriangleright \mathcal{A} \Rightarrow C_{\mathbf{I},E}(\mathcal{A}, \mathcal{B})$$

Let's assume that we have a comparison method that is $\rhd$ *−complete* but not *compatible* with respect to the $\rhd$ relation. If we use this comparison method to compare two sets $\mathcal{A}$ and $\mathcal{B}$ with $\mathcal{A} \rhd \mathcal{B}$ ($\mathcal{A}$ is better than $\mathcal{B}$), then our comparison method returns true. However, there are also sets $\mathcal{A}$ and $\mathcal{B}$ with $\mathcal{A} \not\rhd \mathcal{B}$ ($\mathcal{A}$ is not better than $\mathcal{B}$) for which the comparison method returns true. If we use a comparison method that is $\rhd$ *−compatible*, then the above situation is safe: if our comparison method yields true, we can be sure that $\mathcal{A}$ is better than $\mathcal{B}$. However, if the comparison method is not $\rhd$ *−complete*, there may be sets $\mathcal{A}$ and $\mathcal{B}$ where $\mathcal{A}$ is better than $\mathcal{B}$, but our comparison method returns false. Zitzler et al. [47] presented the following key results.

- Unary quality indicators are, in general, not capable of indicating whether an approximation set is better than another, even if we use several of them. This also holds, if we consider approximation sets containing a single objective vector only.

- There are unary indicators which allow to infer if an approximation set is not worse than another, e.g., the distance indicator by Czyzak and Jaszkiewicz [54], the hypervolume indicator by Zitzler and Thiele [52], or the unary ϵ-indicator presented in [47].

- Binary indicators in principle do not possess the theoretical limitations of unary indicators. The binary ϵ-indicator proposed in [47] is capable of detecting whether an approximation set is better than another. However, not all existing binary indicators have this property.

Knowles et al. [55] presented a similar idea on the performance assessment of stochastic multi-objective optimizers in which they recommend having quality indicators that are only Pareto dominance compliant (or Pareto compliant).

Definition 3.5. Pareto compliant An indicator $I : \Omega \rightarrow \mathbb{R}$ is Pareto compliant if for all $\mathcal{A}, \mathcal{B} \in \Omega : \mathcal{A} \preceq \mathcal{B} \Rightarrow I(\mathcal{A}) \geq I(\mathcal{B})$, assuming that greater indicator values correspond to higher quality (otherwise $\mathcal{A} \preceq \mathcal{B} \Rightarrow I(\mathcal{A}) \leq I(\mathcal{B})$). In the context of order theory, a Pareto compliant indicator I is an order-preserving function from $(\Omega, \preceq)$ to $(\mathbb{R}, \geq)$ (respectively, $(\mathbb{R}, \leq)$).

Many of the indicators that are employed in the MOEA literature are not Pareto compliant. Several popular indicators are designed to assess just one isolated aspect of an approximation sets quality, e.g., its proximity to the Pareto optimal front, or its spread in objective space. These quality indicators are known as Pareto non-compliant.

Definition 3.6. Pareto Noncompliant Any indicator that can yield for any approximation sets $\mathcal{A}, \mathcal{B} \in \Omega$ a preference for $\mathcal{A}$ over $\mathcal{B}$, when $\mathcal{B}$ is preferable to $\mathcal{A}$ with respect to weak Pareto dominance ($\mathcal{B} \preceq \mathcal{A} \wedge \mathcal{A} \preceq \mathcal{B}$, or $\mathcal{B} \lhd \mathcal{A}$ for short), is Pareto non-compliant.

3.2.3 Unary quality indicators

Let $\mathcal{PF}$ be the Pareto optimal front and $\mathcal{A}$ an approximation of the Pareto optimal set, we define some unary quality indicators as follows:

- **Error Ratio (I_{ER})**: It reports the percentage of the number of vectors in $\mathcal{A}$ that are not members of $\mathcal{PF}$ [56, 57]. I_{ER} is Pareto compliant and it is defined as follows:

$$I_{ER} = \frac{\sum_{i=1}^{|\mathcal{A}|} e_i}{\mathcal{A}} \tag{3.6}$$

  where $e_i = 0$ when the i^{th} vector of $\mathcal{A}$ is an element of $\mathcal{PF}$ and $e_i = 1$ when the i^{th} vector of $\mathcal{A}$ is not an element of $\mathcal{PF}$. If $I_{ER} = 0$ then $\mathcal{A} \subseteq \mathcal{PF}$; but when $I_{ER} = 1$ none of the points in $\mathcal{A}$ are in $\mathcal{PF}$. A lower I_{ER} is better.

- **Generational Distance (I_{GD})**: It reports how far, on average, $\mathcal{A}$ is from $\mathcal{PF}$ [42, 58, 56]. I_{GD} is Pareto non-compliant and it is defined as:

$$I_{GD} = \frac{1}{|\mathcal{A}|}\left(\sum_{i=1}^{|\mathcal{A}|} d_i^p\right)^{\frac{1}{p}} \tag{3.7}$$

where $|\mathcal{A}|$ is the number of vectors in $\mathcal{A}$, $p = 2$ and d_i is the euclidean phenotypic distance between each member, i, of $\mathcal{A}$ and the closest member in $\mathcal{PF}$ to that member, i. If $I_{GD} = 0$, $\mathcal{A} \subseteq \mathcal{PF}$.

- **Inverted Generational Distance (I_{IGD}).** It does not only indicate the proximity of the set $\mathcal{A}$ to $\mathcal{PF}$, but also gives a certain sense about its extension [59]. I_{IGD} is Pareto non-compliant and it is analogous to GD, but measured from $\mathcal{PF}$ to $\mathcal{A}$. I_{IGD} is defined by:

$$I_{IGD} = \frac{1}{|\mathcal{PF}|}\left(\sum_{i=1}^{|\mathcal{PF}|} d_i^p\right)^{\frac{1}{p}} \tag{3.8}$$

where $|\mathcal{PF}|$ is the number of vectors in $\mathcal{PF}$, $p = 2$ and d_i is the euclidean phenotypic distance between each member, i, of $\mathcal{PF}$ and the closest member in $\mathcal{A}$ to that member, i. If $I_{IGD} = 0$, $\mathcal{A} = \mathcal{PF}$.

- **Spacing (I_S)**: It describes the spread of the vectors in $\mathcal{A}$ [42, 60]. This Pareto non-compliant indicator measures the distance variance of neighboring vectors in $\mathcal{A}$. Formally, I_S is defined as follows:

$$I_S = \sqrt{\frac{1}{|\mathcal{A}|-1}\sum_{i=1}^{|\mathcal{A}|}(\bar{d} - d_i)^2} \tag{3.9}$$

where $d_i = \min_{j=1\cdots,|\mathcal{A}|, i\neq j} \sum_{m=1}^{k} |f_m^i - f_m^j|$, $i, j = 1, \cdots, |\mathcal{A}|$, k is the number of objective functions and $\bar{d}$ is the mean of all d_i. When $I_S = 0$, all members are spaced evenly apart. Note that this indicator assumes that a MOEA has already converged to the true Pareto front.

- **Hypervolume (I_H).** It was originally proposed by Zitzler and Thiele in [52], and it's defined as the size of the space covered by the approximation obtained of the Pareto optimal front. If Λ denotes the Lebesgue measure, I_H is defined as:

$$I_H(A, \vec{z}_{ref}) = \Lambda\left(\bigcup_{\vec{z}\in\mathcal{A}} \{\vec{y} \mid \vec{z} \prec \vec{y} \prec \vec{z}_{ref}\}\right) \tag{3.10}$$

where $\vec{z}_{ref} \in \mathbb{R}^k$ denotes a reference point that should be dominated by all the Pareto optimal points. A high I_H value, indicates that $\mathcal{A}$ is close to $\mathcal{PF}$ and has a good spread towards the extreme portions of the $\mathcal{PF}$. This is the only unary indicator that is known to be Pareto compliant.

- ***R*2-Indicator**. It belongs to the family of R indicators proposed by Hansen and Jaszkiewicz [61]. R-indicators use utility functions for evaluating approximation Pareto sets. The $R2$-indicator is weakly monotonic and simultaneously evaluates several desired aspects of a Pareto front approximation. Let U be a set of general utility functions, the $R2$-indicator is thus defined as [62]:

$$I_{R2}(\mathcal{A}, U) = -\frac{1}{|U|}\sum_{u\in U} \max_{\vec{a}\in\mathcal{A}}\{u(\vec{a})\} \tag{3.11}$$

Regarding the choice of the utility functions u, there are several possibilities, such as: weighted sum, least squares, weighted Tchebycheff metric, etc. These utility functions have an associated set of uniformly distributed weight vectors W and a reference point $\vec{z}^*$, in order to maintain diversity. The most common utility function is the weighted Tchebycheff metric:

$$R2(\mathcal{A} : W, \vec{z}^*) = \frac{1}{|W|}\sum_{\vec{w}\in W} \min_{\vec{a}\in\mathcal{A}} \left[\max_{i=1,\cdots,k} w_i |a_i - z_i^*|\right] \tag{3.12}$$

- **Delta p Indicator (I_{Δ_p}).** It was proposed by Schütze et al. [63]. It can be seen as an "averaged Hausdorff distance" between the approximate Pareto front and the Pareto optimal front. It is composed of slight modifications of two well-known performance indicators: generational distance (I_{GD}) and inverted generational distance (I_{IGD}). I_{Δ_p} is defined as:

$$I_{\Delta_p} = \max(I_{GD_p}, I_{IGD_p}) \tag{3.13}$$

I_{Δ_p} simultaneously evaluates proximity to the Pareto optimal front and spread of solutions along it. I_{Δ_p} is Pareto non-compliant.

3.2.4 Binary Quality Indicators

- **ϵ-Indicator (I_ϵ).** It is a Pareto compliant measure. Given two approximate sets, $\mathcal{A}$ and $\mathcal{B}$, the ϵ-indicator measures the smallest amount, ϵ, that must be used to translate the set $\mathcal{A}$, so that every point in $\mathcal{B}$ is covered [47].

  Definition 3.7. Let's assume, without loss of generality, a minimization problem with k objective functions, then, an objective vector $\vec{z}^1 = [z_1^1, \cdots, z_k^1]$ is said to **ϵ-dominate** another objective vector $\vec{z}^2 = [z_1^2, \cdots, z_k^2]$, denoted by $\vec{z}^1 \preceq_\epsilon \vec{v}^2$, if and only if $\forall 1 \leq i \leq k : u_i \leq \epsilon \cdot v_i$ for a given $\epsilon > 0$.

  Loosely speaking, a vector $\vec{z}^1$ is said to ϵ-dominate another vector $\vec{z}^2$, if we can multiply each objective value in $\vec{z}^2$ by a factor ϵ and the resulting objective vector is still weakly dominated by $\vec{z}^1$. Therefore, $\vec{z}^1 \prec\prec \vec{z}^2$ implies that there exists an $\epsilon < 1$ such that $\vec{z}^1$ ϵ-dominates $\vec{z}^2$. We define the binary ϵ-indicator I_ϵ as:

$$I_\epsilon(\mathcal{A}, \mathcal{B}) = \min\{\epsilon \in \mathbb{R} \quad | \quad \forall \vec{b} \in \mathcal{B} \quad \exists \vec{a} \in \mathcal{A} : \vec{a} \prec_\epsilon \vec{b}\} \tag{3.14}$$

  So, when $I_\epsilon(\mathcal{A}, \mathcal{B}) < 1$, all solutions in $\mathcal{B}$ are dominated by a solution in $\mathcal{A}$. If $I_\epsilon(\mathcal{A}, \mathcal{B}) = 1$ and $I_\epsilon(\mathcal{B}, \mathcal{A}) = 1$, then $\mathcal{A}$ and $\mathcal{B}$ represent the same Pareto front approximation. If $I_\epsilon(\mathcal{A}, \mathcal{B}) > 1$ and $I_\epsilon(\mathcal{B}, \mathcal{A}) > 1$, then $\mathcal{A}$ and $\mathcal{B}$ are incomparable because they both contain solutions not dominated by the other set.

- **Two Set Coverage (I_{CS}).** It was proposed by Zitzler et al. [64] and it is a Pareto compliant indicator. Let $\mathcal{A}, \mathcal{B}$ be two approximate sets, I_{SC} is defined as follows:

$$I_{SC}(\mathcal{A}, \mathcal{B}) = \frac{|\vec{b} \in \mathcal{B} \text{ such that } \exists \vec{a} \in \mathcal{A} \text{ with } \vec{a} \prec \vec{b}|}{|\mathcal{B}|} \tag{3.15}$$

  If all points in $\mathcal{A}$ dominate or are equal to all points in $\mathcal{B}$, then by definition $I_{CS} = 1$. $I_{CS} = 0$ implies that no element in $\mathcal{B}$ is dominanted by any element of $\mathcal{A}$. In general, both $I_{CS}(\mathcal{A}, \mathcal{B})$ and $I_{CS}(\mathcal{B}, \mathcal{A})$ have to be considered due to set intersections not being empty.

Chapter 4

Large Scale Multi-objective Optimization

Large-scale multi-objective optimization refers to the solution of MOPs that involve hundreds or even thousands of decision variables. As explained by Weise et al. [65], several factors make large-scale optimization problems extremely difficult. For instance, when the number of decision variables of a problem increases, the volume of the search space grows exponentially, and the complexity of the fitness landscape increases rapidly as well. However, the major difficulty comes from the interactions between decision variables, better known as *non-separability*. In a more formal way, an objective function $f(\vec{x})$ is said to be separable with respect to a decision variable x_k if the following condition is fulfilled [66]:

$$f(\vec{x}) < f(\vec{x'}) \rightarrow f(\vec{y}) < f(\vec{y'}) \tag{4.1}$$

where $\vec{x} = [x_1, ..., x_k, ..., x_n]^T, \vec{x'} = [x'_1, ..., x'_k, ..., x'_n]^T, \vec{y} = [y_1, ..., x_k, ..., y_n]^T$ and $\vec{y'} = [y'_1, ..., x'_k, ..., y'_n]^T$ are four different decision vectors. Otherwise, $f(\vec{x})$ is considered to be non-separable with respect to x_k, which means that x_k has interactions with one or more other decision variables. Specially, if the decision variables only interact with some (rather than all) of the other decision variables, the objective function $f(\vec{x})$ is known as a partially separable function. Several important observations can be made with regard to separability, non-separability and partial separability [67], which are listed below.

A decision variable x_k can be optimized independently iff the objective function $f(\vec{x})$ is separable with respect to it:

$$\underset{\vec{x}}{\operatorname{argmin}}\, f(\vec{x}) = (\underset{x_k}{\operatorname{argmin}}\, f(\vec{x}), \underset{\forall x_j, j\neq k}{\operatorname{argmin}}\, f(\vec{x})) \tag{4.2}$$

The decision variables can be optimized component-wise independently iff an objective function $f(x)$ is partially separable:

$$\underset{\vec{x}}{\operatorname{argmin}}\, f(\vec{x}) = (\underset{x_1}{\operatorname{argmin}}\, f(x_1, \dots), \dots, \underset{x_m}{\operatorname{argmin}}\, f(\dots, x_m)) \tag{4.3}$$

where $x_1, \dots, x_m$ are disjoint sub-vectors of $\vec{x}$, and the interactions only exist among decision variables inside each sub-vector, but the decision variables in different sub-vectors do not interact with each other.

In this chapter, we present the works present in the state of the art, related to the solution of large scale MOPs.

4.1 Studies on Large Scale Multi-objective Optimization

In the evolutionary multi-objective optimization community, objective function scalability has been a hot research topic in recent years [68, 69]. In contrast with this, parameter scalability has been rarely considered before. Regarding studies of decision variables scalability, the most significant ones presented in the state of the art that we are aware of are those reported by Durillo et al. [5, 6]. In this works the behavior and effect of decision variables scalability over eight state-of-the-art multi-objective metaheuristics is analyzed. Such metaheuristics include three genetic algorithms (GAs) (NSGA-II, SPEA2 and PESA-II), an evolution strategy

(PAES), a PSO algorithm (OMOPSO), a cellular GA (MOCell), an algorithm based on differential evolution (GDE3) and a Scatter Search algorithm (AbYSS). All of these approaches were among the most representative of the state-of-the-art in evolutionary multi-objective optimization by the time of these works, and were studied when solving a benchmark of parameter-wise scalable problems, the ZDT [2] test suite. The authors analyzed the behavior of these eight multi-objective metaheuristics over parameter scalability, using a number of decision variables that ranged from 8 up to 2048. The hypervolume performance indicator [70] was adopted to define a stopping criterion. The study paid particular attention to the computational effort required by each algorithm for reaching the true Pareto front of each problem, an algorithm was considered successful when the hypervolume of its current population (or archive, depending on the algorithm) was higher than the 95% of the hypervolume of the Pareto front. Authors performed 100 independent runs using 10,000,000 function evaluations in all the runs. Among all the compared metaheuristics, the study revealed that differential evolution and particle swarm optimization are the most promising approaches to deal with the scalable problems adopted by the authors, since GDE3 and OMOPSO were the ones showing the best efficiency, although the showed poor performance on multi-frontal MOPs. These papers provide the first empirical evidence of the decrease in efficacy and efficiency that multi-objective metaheuristics have when dealing with MOPs with a large number of decision variables, as it is shown in their results. Next we present the works devoted to solve large scale decision variables MOPs.

4.2 MOEA/D

One of the first works to take a look at large scale MOPs was a small study presented in [30], where ZDT1 is solved with up to 100 decision variables using MOEA/D. The *multi-objective evolutionary algorithm based on decomposition* (MOEA/D) [30] has been one of the most adopted approaches for the community, due to its simplicity and to its effectiveness when applied to a broad range of MOPs. MOEA/D decomposes the MOP into a set of single-objective subproblems and solves these subproblems simultaneously using an evolutionary algorithm. Such decomposition is based on a set of weights each of which corresponds to a single subproblem. Each weight vector is used as a search direction to define a scalar function. For this sake, the so called Tchebycheff decomposition is the most widely used. Given a weight vector $\lambda = [\lambda_1, \ldots, \lambda_n]^T$ the corresponding subproblem is defined as:

$$\text{minimize} \quad g^{te}(x|\lambda, z^*) = \max_{1\leq i\leq n} \lambda_i |f_i(x) - z_i^*| \tag{4.4}$$

where z^* is the reference point chosen as the minimum of objective function values found during the evolution. The main advantage of the Tchebycheff approach is that it works regardless of the shape of the Pareto front, while other decomposition approaches (like the weighted sum approach) only work for convex Pareto fronts. The weights are also used to define neighborhoods of the subproblems. The neighborhood relations among these subproblems are defined based on the distances between their aggregation coefficient vectors. At each generation, a new individual is generated and evaluated using its own neighborhood of weights, with the idea that any information about these closest weight vectors should be helpful for optimizing the current individual's subproblem. Once this new individual is created, it is compared to its parent and in case the offspring is better, it replaces the parent. Moreover, it is also compared to other individuals in its neighborhood and is allowed to replace some of them. Therefore, at each generation, the population is composed of the best solution found so far (i.e., since the start of the run of the algorithm) for each subproblem.

MOEA/D can be summarized as follows:

Input:

- The MOP
- N: The number of subproblems considered in MOEA/D

- S: The number of species for decision variables decomposition
- A set of N uniform spread weight vectors:
  $\lambda^1, \ldots, \lambda^N$
- T: The neighborhood size

Output:

- PS: the final solutions found during the search

Step 1) Initialization:

Step 1.1) Set the external population of final solutions $PS = \emptyset$.

Step 1.2) Find the T closest weight vectors to each weight vector. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$, where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i .

Step 1.3) Generate an initial population $x^1, \ldots x^N$ randomly or by a problem-specific method. Set $FV^i = f(x^i)$.

Step 1.4) Initialize $z = [z_1, \ldots, z_k]^T$, where z_i is the best value found so far for objective f_i.

Step 2) Update:

For $i = 1, \ldots, N$ do

Step 2.1) Crossover and Mutation:

For $j = 1, \ldots, S$ do

Step 2.1.1) Randomly select two indexes p, q from $B(i)$, and then generate a new solution y from x_p and x_q using crossover.

Step 2.1.2) Apply a problem-specific repair improvement heuristic on y to produce y'.

Step 2.2) For each $j = 1, \ldots, k$, if $z_j > f_j(y')$, then set $z_j = f_j(y')$.

Step 2.3) Update of Neighboring Solutions: For each index $j \in B(i)$, if $g^{te}(y'|\lambda^j, z^*) < g^{te}(x^j|\lambda^j, z^*)$, then $FV^j = f(y')$.

Step 2.4) Remove from the external population PS all the vectors dominated by $f(y')$. Add $f(y')$ to PS if no vectors in PS dominate it.

Step 3) Stopping Criterion: Stop if the termination criterion is satisfied. Otherwise, go to Step 2.

In their work, authors of MOEA/D analyze how the computational cost over this approach, in terms of the number of function evaluations, increases as the number of decision variable of the problem increases. This is shown using a number of decision variables that ranges from 10 up to 100 variables. They used as a performance index the average number of function evaluations used by MOEA/D for reducing the D-metric [71] and concluded that the average number of function evaluations linearly scales up, as the number of decision variables increases.

They attributed these results to two facts:

1. The number of scalar optimization sub-problems in MOEA/D is fixed to be 100, regardless of the number of decision variables of the problem.

2. The complexity of each scalar optimization could scale up linearly with the number of decision variables.

However, this study is too small to show a general behavior of MOEA/D over large scale (in decision variables space) MOPs.

4.3 Cooperative coevolutionary MOEA for large scale Multi-objective Optimization

The first algorithm created with the specific task of dealing with large scale (in decision variables space) MOPs was presented in [72]. Authors adopted the CCGA

[73] framework (adapted for multi-objective optimization) and GDE3 [74] (as its basic multi-objective optimizer) to create the Cooperative Coevolutionary GDE3 (CCGDE3), with the idea of decomposing large scale problems into subproblems which are easier to deal with MOPs decision variables scalability. CCGDE3 works as follows: at the beginning, it divides the vector of decision variables $\vec{x}$ of dimension $D \in \mathbb{N}$ into $S \in \mathbb{N}$ subcomponents of equal size. Each subcomponent is created from a random grouping of decision variables in order to increase the probability of grouping interacting variables in non-separable problems. At the same time, S subpoputaions (species) are created, each one with NP individuals, and these S subpopulations are assigned their corresponding decision variables in a random way. This means that to each subpopulations, it corresponds a subcomponent from the S which have been already done. Thus, every subpopulation will have a total of m decision variables. Once the subpopulations are created, the algorithm does a random initialization of all the individuals across all subpopulations. Then, the algorithm performs the *cycles* in which the evolution of each of the subpopulations is done for a given number of *generations*. This will continue until the stop condition is reached, and at the end, the solutions that are globally nondominated (i.e., with respect to all the subpopulations), constitute the outcome of the algorithm.

The collaboration among the subpopulations takes place in the next way: in the first generation, random collaborations are formed and evaluated, obtaining a random individual from each subpopulation and forming a complete set of solutions to be evaluated in their objective functions. Then, the results from the evaluation are assigned back to the individual under evaluation. After the first generation, the resulting child subpopulations Q_1 to Q_S will be evaluated by forming collaborations with randomly selected components from the *best* non-dominated levels in the subpopulations, P_1 to P_S, of the previous generation. The algorithm iterates until some termination condition is fulfilled (usually when a certain predefined number of cycles is reached). At the end, we apply a fast non-dominated sorting procedure as in the NSGA-II [75] to the best non-dominated levels of each subpopulations in order to obtain a final set of solutions for the problem being solved. A summary of the way in which CCGDE3 works is presented in Algorithm 9.

Algorithm 2 Cooperative Coevolutionary Framework

```
Input: $NP$, $Cycles$, $Gmax$, $NumEsp$
Output: $SolutionSet$
  $Pobs \leftarrow Populations(NP, NumEsp)$
  $InitializeSpecies(Pobs)$
  for $j \leftarrow 1$ to $Cycles$ do
    for $i \leftarrow 1$ to $NumEsp$ do
      for $k \leftarrow 1$ to $Gmax$ do
        $MOEA(Pobs[i])$
      end for
    end for
  end for
  $SolutionSet \leftarrow ObtainNonDominatedSet(Pobs)$
  return $SolutionSet$
```

CCGDE3 was compared with respect to two MOEAs: GDE3 [74] and NSGA-II [75]. In the authors' experiments, they use a large number of decision variables that ranges from 200 up to 5000 when solving ZDT1, ZDT2, ZDT3 and ZDT6 (from the ZDT test suite [2]). Adopting the hypervolume performance indicator [76], each MOEA was run until they obtained an approximation of the Pareto front that has a hypervolume of 95% with respect to the true Pareto front or by using a maximum of 10,000,000 function evaluations. Results showed that CCGDE3 was the fastest algorithm, scaling better than GDE3 and NSGAII when the number of variables is large, not only in terms of the number of evaluations, but also in terms of CPU time.

4.4 MOEA Based on Decision Variables Analyses

An algorithm based on interdependence variable analysis and control variable analysis to deal with large scale MOPs is presented in [77]. This approach decomposes the MOPs with high dimensionality into a set of simpler sub-MOPs with low-dimensional subcomponents. Based on interdependent analysis between two variables, decision variables are decomposed into several low-dimensional subcomponents. Each sub-

MOP independently optimizes subcomponents one by one. Therefore, the approach, called MOEA/DVA is expected to have an advantage over most MOEAs which optimize all of the decision variables as a whole. Authors claim that the key issue of reducing the difficulty of a large scale MOP is to detect the variable interactions. Inspired by the fact that in MOPs some decision variables control the convergence aspect of the obtained solutions, while some others decision variables determine the spread aspect of the obtained solutions, authors work with three different classifications of the decision variables of a MOP:

Position variable: A decision variable x_i is called position variable if and only if changing x_i in $\vec{x} = [x_1, \ldots, x_n]^T$ can only cause a vector that is incomparable or equivalent to $\vec{x}$. Changing a position variable on its own never causes a dominated or dominating decision vector.

Distance variable: If changing x_i in $\vec{x} = [x_1, \ldots, x_n]^T$ can only result in a decision vector which equals $\vec{x}$, dominates $\vec{x}$, or is dominated by $\vec{x}$, then x_i is called a distance variable. That is to say, changing a distance variable on its own will never cause incomparable decision vectors.

Mixed variable: All decision variables which are neither position nor distance variables are called mixed variables. Furthermore, changing a mixed variable on its own can cause a change in distance or position.

The difference among the three kinds of variables is depicted in Fig. 4.1. In fact for continuous ZDT, DTLZ, UF, and MOP [78] problems has been defined, such that the number of position variable(s) and mix variable(s) is $k - 1$, while the number of distance variable(s) is $n - k + 1$. Where k is the number of objective functions in MOP (3) and n is the number of decision variables.

Taking this idea, authors developed a control property analysis and variable linkage analysis. Based on diverse variables (position variables and mixed variables), MOEA/DVA decomposes a complicated MOP into a set of simpler sub-MOPs. The

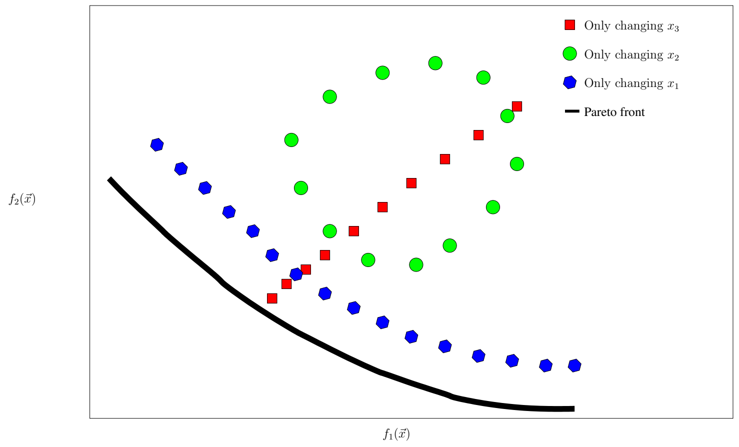

Figure 4.1: Plot of a set of sampling points generated by changing one variable at a time while the others are fixed, for a MOP with two objective functions with three decision variables $\vec{x} = [x_1, x_2, x_3]$. For any fixed x_2,x_3, changing the value of x_1 on its own in $\vec{x}$ results in a set of incomparable or equivalent solutions. Therefore, x_1 is a **position variable** for the used MOP. For any fixed x_1, x_2 , only changing x_3 in $\vec{x}$ will never result in incomparable solutions. Therefore, x_3 is a **distance variable** for the used MOP. Finally, changing the value of x_2 in $\vec{x}$ results in a set of solutions including dominated solutions and nondominated solutions. Therefore, x_2 is a **mixed variable** for the used MOP.

core of MOEA/DVA are two kinds of decompositions: decomposition of distance variables into a set of low-dimensional sub-components and MOP decomposition based on diverse variables with uniformly distributed values. The distance variables are divided into several low-dimensional subcomponents based on learned variable linkages. Each sub-MOP independently optimizes subcomponents one by one.

The process of MOEA/DVA is summarized next.

1. Decision Variable Analyses: There are two variable analyses: i) control property analysis and ii) interaction analysis. Interaction analysis provides the variable linkages for decomposition of distance variables, while control property analysis provides diverse variables (position variables and mixed variables) for MOP decomposition and offers distance variables for decomposition of distance variables.

2. Decomposition of Distance Variables: Decompose high-dimensional distance

variables into several low-dimensional subcomponents which can be optimized more easily.

3. MOP Decomposition Based on Diverse Variables: A MOP is decomposed into a set of sub-MOPs with uniformly distributed values of diverse variables (position variables and mixed variables).

4. Subcomponent Optimization: Optimize each subcomponent independently to improve the convergence speed of population.

5. Uniformity Optimization: Optimize all the decision variables including position variables and mixed variables. Its aim is to improve the uniformity of population in the objective space.

For the subcomponent optimization, authors use the evolutionary operator of MOEA/D. In their experiments ZDT, DTLZ and UF test problems are solved with up to 1000 decision variables. MOEA/DVA is benchmarked against NSGA-III, SMS-EMOA and MOEA/D. MOEA/DVA could outperformed all MOEAs when adopting the IGD-metric

4.5 A Decision Variable Clustering Based MOEA

An evolutionary algorithm for tackling large-scale MOPs based on a decision variable clustering method is presented in [79]. Following the idea of MOEA/DVA (already mention in the previous section 4.4), the proposed approach called LMEA uses a decision variable clustering method to divide the decision variables into convergence-related and diversity-related ones. Different from MOEA/DVA, instead of adopting a decision variable analysis method based on dominance based relationships, LMEA adopts a decision variable clustering method based on the k-means method with features measured by the angles between the sample solutions and the direction of convergence, where smaller angles indicate more contributions to convergence and larger angles to diversity. Consequently, a decision variable can only be divided as either convergence-related or diversity-related, thus addressing the open issue that decision variables related to both convergence and diversity can not be further distinguished.

Once the categorization has been performed, LMEA adopts two optimization strategies to deal with the two types of decision variables separately: the convergence optimization strategy and the diversity optimization strategy. Both strategies adopt non-dominated sorting as the first selection criterion but differ in the secondary selection criterion. In order to enhance convergence, this secondary selection in the convergence optimization strategy is based on the Euclidean distances from the candidate solutions to the ideal point; to man-age diversity, the secondary selection in the diversity optimization strategy is based on the angles between the candidate solutions.

In general LMEA can be summarized in the next way: first, a population of N candidate solutions is randomly initialized. Then, the developed decision variable clustering method is applied to divide the variables into two groups, convergence-related variables and diversity-related variables. Thereafter, the convergence-related variables are further divided into several subgroups based on the interaction between these variables, where the variables are interacted with each other inside one subgroup but not interacted with those in any other subgroups. The variables in each subgroup are also known as interacting variables, as they can not be optimized separately due to the interactions between each other. Once the interaction analysis is done, LMEA starts to optimize each subgroup of variables one by one using a convergence optimization strategy, while the diversity-related variables are optimized using a diversity optimization strategy.

To assess the performance of the proposed LMEA, empirical evaluations were conducted on a variety of benchmark problems (among DTLZ, WFG and UF) in comparison with several state-of-the-art MOEAs for solving large-scale MOPs, for this sake authors adopted two performance indicators, the inverted generational distance and the hypervolume. Results demonstrate that LMEA is well suited for solving large-scale MOPs having up to 5000 decision variables.

Chapter 5

Coevolution

Although parameter scalability is a topic that has been only scarcely studied in the evolutionary multi-objective optimization field, large-scale optimization has been the focus of an important amount of research in global (single-objective) optimization using evolutionary algorithms. The currently available approaches for large-scale global optimization can be roughly divided in two groups:

- Those that decompose a high-dimensional decision variables vector into small subcomponents which can then be handled by conventional EAs (see for example [66, 80]).
- Those that approach the problem by disturbing the population of the EA or by combining different evolutionary methods (see for example [81, 82, 83, 84]).

From these methods, cooperative coevolution has been found to be one of the most successful approaches for solving large and complex problems, through the use of problem decomposition. There is plenty of evidence of the success of this sort of approach in large scale global optimization [85, 86, 66]. In fact, a cooperative coevolutionary algorithm for large scale multi-objective optimization is presented in [72].

Coevolutionary algorithms (CAs) are natural extensions of traditional evolutionary algorithms (EAs). The main elements of these extensions lay in the adaptive nature of fitness evaluation for the members of coevolutionary systems where individuals are assigned fitness values based on interactions with other individuals from other species. Coevolution then refers to a reciprocal evolutionary exchange between species that interact with each other. The idea of CAs comes from the biological observation which shows that coevolving some number of species defined as collections of phenotypically similar individuals is more realistic than simply evolving a population containing representatives of one species [87]. So, instead of evolving a population (globally or spatially distributed) of similar individuals representing a global solution, it is more appropriate to coevolve subpopulations of individuals representing specific parts of the global solution.

A coevolutionary search involves the use of multiple species as the representation of a solution to an optimization problem. Each species can either compete or cooperate during the search process. Therefore, such models have been historically categorized as competitive or cooperative. In the case of cooperative algorithms, individuals are rewarded when they work well with other individuals and punished when they perform poorly together [88]. Each population represents a piece of a larger problem, and it is the task of those populations to evolve increasingly more fitter pieces for the larger problem. In the case of competitive algorithms, individuals are rewarded at the expense of those with which they interact [89]. An example of a competitive approach is the predator-prey model, in which individuals in one population represent some kind of device and individuals in another population represent some kind of input for that device. Then, the objective of the first population is to evolve better devices to handle the input, while the objective of the second population is to evolve increasingly difficult inputs for such devices.

Recent work in coevolutionary algorithms (CAs) research considers coevolution as a form of multi-objective optimization [90, 42, 91]. However, it is worth noting that most of these approaches were not created with the specific purpose of dealing with a large number of decision variables. This chapter is devoted to describe how coevolution has been used to deal with MOPs.

5.1 Coevolutionary Multi-objective Evolutionary Algorithms

In nature, coevolution is the process of reciprocal genetic change in one species, or group, in response to another. That is, coevolution refers to a reciprocal evolutionary exchange between species that interact with each other. The term *coevolution* arises from a study about the interaction between plants and butterflies conducted by Ehrlich and Raven [92] in which the coevolutionary responses of ecologically intimate organisms and community evolution in general were observed.

The relationships between the populations of two different species X and Y can be described considering all their possible types of interactions. Such interactions can be positive or negative depending on the consequences that such interaction produces on individuals of the population. Coevolution is then used as the biological process responsible for speciation, maintaining population diversity, introducing arms races and open-ended evolution. The main issue in coevolutionary algorithms is that the fitness of an individual in a population depends on the individuals of a different population. Depending of the way in which fitness is computed, there are two main classes of coevolutionary algorithms in the evolutionary computation literature:

- Those based on competition relationships: In this case, the fitness of an individual is the result of a series of encounters with other individuals [89].
- Those based on cooperation relationships: In this case, the fitness of an individual is the result of a collaboration with individuals of other species (or populations) [73, 88].

Competition and cooperation between groups species in nature has inspired researchers to incorporate coevolutionary dynamics into MOEAs. Evolutionary computation researchers have developed many coevolutionary approaches, in which two or more species (i.e., populations) that relate to each other are applied to deal with MOPs, using one of the previously indicated schemes. Also, in most cases, these

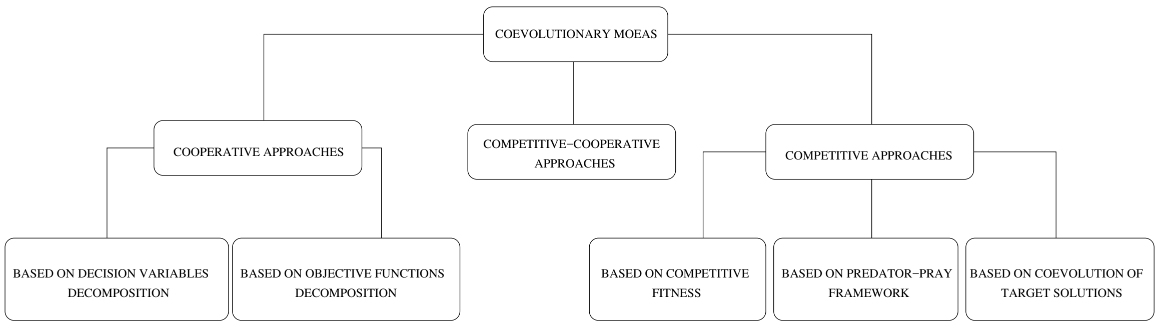
COEVOLUTIONARY MOEAS
COOPERATIVE APPROACHES
COMPETITIVE–COOPERATIVE APPROACHES
COMPETITIVE APPROACHES
BASED ON DECISION VARIABLES DECOMPOSITION
BASED ON OBJECTIVE FUNCTIONS DECOMPOSITION
BASED ON COMPETITIVE FITNESS
BASED ON PREDATOR–PRAY FRAMEWORK
BASED ON COEVOLUTION OF TARGET SOLUTIONS

Figure 5.1: Taxonomy for CMOEAs, based on the ways coevolution has been adopted.

species evolve independently by means of an evolutionary algorithm and interaction occurs when individuals need to be evaluated. Figure 5.1 presents a taxonomy for CMOEAs, based on the main ways in which coevolution has been applied to MOEAs.

In this thesis our interest is on the cooperative version of coevolutionary algorithms, since these are the ones which are normally applied in large scale single-objective optimization.

5.1.1 Cooperative Coevolutionary MOEAs

Potter and De Jong opened the door for doing research on cooperative CEAs in 1994 by developing the first framework of cooperative coevolution (CC) utilized within evolutionary algorithms [73] with their *Cooperative Coevolutionary Genetic Algorithm* (CCGA). This approach was first applied to static function optimization and later to neural network learning [88]. Potter's framework uses a divide-and-conquer approach to split the decision variables into subpopulations of smaller size, so that each of these subpopulations is optimized with a separate EA. The main idea was to decompose a high-dimensional problem into several low-dimensional subcomponents and evolve these subcomponents cooperatively for a predefined number of *cycles*. Here, a *cycle* consists of one complete evolution of all subcomponents. For the problem decomposition, Potter and De Jong took each decision variable of the problem as a subcomponent.

In Potter's model, each population contains individuals representing a component of a larger solution, and evolution of these populations occurs almost independently, in tandem with one another, interacting only to obtain fitness. Such process can be static, in the sense that the divisions for the separate components are decided *a priori* and never altered, or dynamic, in the sense that populations of components may be added or removed as the run progresses [93]. After Potter's work, there were many more *cooperative coevolutionary* approaches, most of which were aimed for large scale global optimization, since it was found that this was a good framework for solving such problems [85, 86, 66, 94, 80].

In general, the most common cooperative coevolutionary framework for global (single-objective) optimization using EAs can be summarized as follows:

1. Decompose the vector of decision variables of the problem into m low dimensional subcomponents.
2. Set $j = 1$ to start a new cycle.
3. Optimize the j^{th} subcomponent with a certain EA for a predefined number of fitness evaluations (FEs).
4. If $j < m$ then $j++$, and go to Step 3.
5. Stop if the stopping criteria are satisfied; otherwise go to Step 2 for the next cycle.

The coevolutionary effect in the CCGA is produced by a cooperation among all species.

Since the cooperative coevolutionary framework can be extended in a relatively easy way to multi-objective optimization, a number of approaches which incorporate this framework have been proposed to improve the performance of multi-objective EAs. As the nature of MOPs changes, the decomposition can be made not only in decision variable space, but also in objective function space or a mix of both. Next, we will review some cooperative coevolutionary approaches currently available in order to have a better understanding of the way this approach has been used in the context of multi-objective optimization.

5.1.2 Cooperative CMOEAs based on decision variables decomposition

In Cooperative CMOEAs based on decision variables decomposition, the MOP is decomposed along the search space of the problem. So, every decision variable of the problem is assigned to a species population and each species population optimizes one or more decision variables at the same time.

In other words, each population has individuals which represent a particular part (in decision variables space) of the MOP. Thereafter, every member from each population is needed in order to assemble a full solution to the problem. The evaluation of individuals from a particular species' population is then performed by making the individual collaborate with members from the other species. The drawback with this kind of problem decomposition approach is that information about the ideal number of components or the optimal way to assign them is, in most cases, not known *a priori*. Also, many problems present highly complex interdependencies and the decomposition becomes harder to perform. A graphical description of the decision variables decomposition is presented in Figure 6.22. Next, we present the most representative approaches within this category, but more examples can be found in [95, 96, 72, 91, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107].

Multi-objective Co-operative Co-evolutionary Genetic Algorithm (MOCCGA)

The MOCCGA [96] is a multi-objective evolutionary algorithm proposed by Keerativuttitumrong et.al. It integrates Potter and De Jong's cooperative coevolutionary framework (presented in their CCGA [73]) with Fonseca and Fleming's

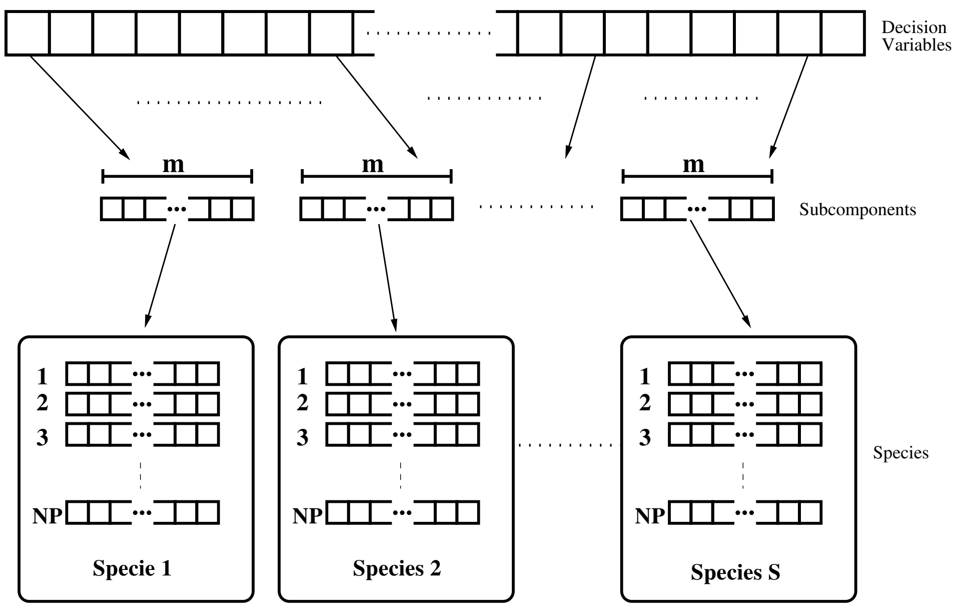

Figure 5.2: Graphical representation of the species creation.

MOGA [27]. In the same way as in CCGA, each species in the MOCCGA represents a decision variable of the problem which is needed to be optimized. Here, instead of assigning a fitness value to the individual of interest which participates in the construction of a complete possible solution, a rank value is determined first. The MOCCGA uses a dominance rank for individuals, in which a count of the number of individuals dominating others is the fitness criterion. Similar to the MOGA, the rank of each individual is obtained after comparing it with the remaining individuals of the same species and they are ranked only within the same subpopulation (species). Then a fitness value can be interpolated onto the individual. Here, a fitness sharing strategy is also used and is carried out in objective space. For further details, a comprehensive description of the MOGA can be found in [27]. In MOCCGA, the objectives are evaluated twice for each individual, both with the best ranked individuals from each subpopulation, as well as with randomly selected individuals. This follows the approach described by Potter and De Jong, which aims to decrease the premature convergence observed on some test problems adopted with the original cooperative coevolutionary framework. However, in this way the number of evaluations over the objective functions duplicates. The performance of MOCCGA was assessed using the Zitzler-Deb-Thiele (ZDT) test suite [2] and was benchmarked against MOGA. In Keerativuttitumrong et al.'s investigation, each algorithm was run only five times and the results from these runs were combined to extract the non-dominated solutions from the overall results. They concluded from these results that their approach improves the performance of MOGA in the context of the Pareto front coverage and on the closeness of non-dominated solutions to the true Pareto optimal solutions. It is important to mention that the number of evaluations adopted by the authors and some parameter settings such as the number of individuals used for each species are not reported anywhere in the paper. A summary of the way in which this approach works is presented in Algorithm 3.

Algorithm 3 MOCCGA

```
Input: a multiobjective optimization problem $MOP$, the population size for the
  species $PS$, the number of generations $Gmax$, the crossover probability $cp$, the
  mutation probability $mp$
Output: $SolutionSet$
  $gen \leftarrow 0$
  $NumSpecies \leftarrow GetNumberOfVariables(MOP)$
  $Species \leftarrow CreateSpecies(PS, NumSpecies)$
  for $i = 1$ to $NumSpecies$ do
    $RandomInitializationOfPopulations(Species[i], PS)$
    $EvaluateFitnessOfIndividuals(Species[i], PS)$
  end for
  while termination condition = false do
    $gen \leftarrow gen + 1$
    for each species $S$ in $Species$ do
      $MOGA(S, Gmax, cp, mp)$
    end for
  end while
  $SolutionSet \leftarrow Species$
  return $SolutionSet$
```

The main drawback of this approach is the bad distribution of the solutions that it produces. Although its authors do not include any indicator to assess the spread of the solutions, their graphical results clearly indicate that MOCCGA can not produce solutions in some parts of the Pareto fronts of the adopted problems.

A Coevolutionary Multi-Objective Evolutionary Algorithm (CO-MOEA)

Another cooperative coevolutionary approach for multi-objective evolutionary optimization (CO-MOEA) is presented in [108]. The main idea of this approach is to concentrate the search efforts on promising regions that arise during the evolutionary process as a byproduct of a mechanism that subdivides decision variable space based on an estimate of the relative importance of each decision variable. In order to determine what regions of the search space are promising, this algorithm performs a relatively simple analysis of the current Pareto front. They divide the evolutionary process of their approach in four stages (i.e., the total number of generations is divided by four), and each stage is allocated one of these four parts. The change between each stage is controlled by a certain number of generations during which the algorithm works. The overall procedure is described next.

1) During the first stage, the algorithm explores all of the search space with a population of individuals using Fonseca and Fleming's Pareto ranking scheme [27] and the adaptive grid proposed in [109]. At the end of this first stage, the algorithm analyzes the current Pareto front stored in the adaptive grid to determine what variables of the problem are more critical. This analysis consists of looking at the values of the decision variables corresponding to the current Pareto front and is performed independently for each decision variable. With this, they try to determine if the values corresponding to a certain variable are distributed along all the allowable interval or if such values are concentrated on a narrower range. If the whole interval is being used, the algorithm keeps the entire interval for that variable, but if only a narrow portion is being used, then the algorithm tries to identify portions of the interval that can be discarded from the search process. As a result of this analysis, the algorithm determines whether is convenient or not to subdivide the interval of a certain decision variable and determines how many subdivisions to perform. Each of these different regions are then assigned to a different population (species).

2) In the second stage, the algorithm uses the species created in the first stage to search at the different regions of the search space. At each generation, the evolution

of all the populations takes place independently and then the nondominated elements from each population are sent to the adaptive grid where they *cooperate* and *compete* in order to conform a single Pareto front with respect to all the search space. After the first generation of the second stage, all the populations that do not provide any individual to the current Pareto front are automatically eliminated and the sizes of the other populations are properly adjusted giving more individual to those species that contribute more to the current Pareto front and decreasing the population size of those who contribute less (i.e, each population is assigned or removed individuals such that its final size is proportional to its contribution to the current Pareto front). Thus, populations *compete* with each other to get as many extra individuals as possible.

3) At the third stage, the algorithm checks on the current populations in order to determine how many and which of them will continue depending on the contribution of their individuals to the current Pareto front. Then, over these populations, the same process from the second stage is applied (i.e., the intervals of the decision variables will be further subdivided and more populations will be created in order to exploit these regions of the search space). In this stage, a minimum population size for each species is defined and this size is enforced for all populations at the beginning of this third stage. After the first generation of this stage, the size will be adjusted based on the same criteria as before (i.e., the size of the populations will be modified based on their contribution to the current Pareto front).

4) Finally, during the fourth stage, the same procedure of the third stage is repeated in order to allow the application of a fine-grained search is applied.

The proposed CO-MOEA was validated using three test functions taken from the specialized literature (see [108] for further details about the test functions) and was compared with respect to three representative MOEAs of the state-of-the-art at that time: the microGA for multiobjective optimization [110], the Pareto Archived Evolution Strategy (PAES) [111] and the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [75]. They used a fixed number of generations over each problem and the performance of each algorithm was assessed using four indicators: the two set coverage (SC) [112], spacing (SP) [60], generational distance (GD) [50] and the error ratio (ER) [57]. Their comparative study showed that CO-MOEA was competitive with respect to the three other algorithms. The main drawback of CO-MOEA is the number of populations that it could potentially need to handle. A summary of the way in which this approach works is presented in Algorithm 4.

Algorithm 4 CO-MOEA

```
Input: the crossover rate $p_c$, the mutation rate $p_m$, the maximum number of
    generations $Gmax$, size of initial population for the first stage $popsize_{init}$, minimum
    size of species's population for further stages $popsize_{sec}$
Output: $SolutionSet$
  $gen \leftarrow 0$
  $NumberOfPopulations \leftarrow 1$
  $populations \leftarrow CreateInitialPopulation(popsize_{init})$
  while $gen < Gmax$ do
    if $(gen = Gmax/4)$or$(Gmax/2)$or$(3 * Gmax/4)$ then
      $CheckActivePopulations(Populations)$
      $AnalysisOfDecisionVariables()$
      $ComputeNumberOfSubdivisions()$
      $CreateNewSubpopulations(Populations)$
      $UpdatePopulations(Populations, popsize_{sec})$
    end if
    for $i \leftarrow 1$ to $NumberOfPopulations$ do
      if $Populations[i]$ contributes to the current Pareto front then
        $EvolveAndCompete(Populations, i, p_c, p_m)$
      end if
    end for
    $gen \leftarrow gen + 1$
  end while
  $SolutionSet \leftarrow Populations$
  return $SolutionSet$
```

Airframe Design Using a Co-Evolutionary Multiobjective Genetic Algorithm

Parmee and Watson [113] proposed a collaborative multiobjective optimization scheme for the preliminary design of airframes. Here, they used one population to optimize each of the objective functions of the problem. The method utilizes individual genetic algorithms (GAs) for the optimization of each objective to reduce the problem to a number of concurrent co-evolutionary tasks specific to the overall design domain.

The fitness measure for individuals within each GA is adjusted at each generation by comparing the values of the variable parameters of identified solutions relating to a single objective with those of the solutions of the other GAs. The fitness for each objective is normalized relative to the maximum and minimum values found during each GA run with a constant adjustment of new upper and lower limits. A penalty relating to the degree of diversity of their variable values as compared to those of the other GA processes is applied using a generational parameter constraint map in such a way that if a variable is outside a range defined by this constraints map, it is adjusted using a penalty function. The range constraints map works as follows: Initially the map must allow each GA to produce an optimal solution based on its own specified objective by setting the value of the map to 1.0, allowing each GA to use the whole range for each variable. As the run progresses, the map increasingly reduces variable diversity (through the use of penalties) in order to draw all concurrent GA searches from their separate objectives towards a single (ideal) trade-off solution where all objectives are best satisfied. The constraint maps include a linear decrease in range constraint and a range constraint reduction based on a sine curve and allow some difference in variable values for each GA towards the end of a run to provide space within which the method can search for an overall optimal solution. This is achieved by setting a minimum value for the range constraint. The number of generations allocated to this final phase of exploration is tested using 2 values i.e., 10% and 50% of the maximum number generations.

This process is described next for the case of a MOP with two objective functions f_1 and f_2:

1. Rank the fitness of population $P1$ using f_1.

2. Rank the fitness of population $P2$ using f_2.

3. Starting with individual number 1 (the fittest), variable 1, compare the value with the equivalent variable of the best individual in $P2$. Return the difference between the two values divided by the total range defined for the variable being examined.

4. Compare the returned value against the value given by the range constraint map for the generation number.

5. If the returned value is greater than the constraint map value, apply a fitness penalty to individual 1.

6. Repeat steps 3-5 for all variables in individual 1.

7. Repeat steps 3-6 for all individuals in $P1$.

Note that the process is repeated for all individuals in population $P2$, which are compared with the best individual in $P1$.

An online sensitivity analysis which ranks the variables according to their influence upon each objective is also introduced. This design sensitivity ranking is used to adjust the fitness of each solution and to ensure that the values of the most influential variables are within the range defined by the constraint map. Solutions are assigned the highest fitness penalty where their most influential variables lie outside of the current constraint map range. With this, the authors of this approach try to ensure that subsequent populations contain high levels of feasible solutions in terms of the most influential variables and relatively redundant variables have little or even no effect on the overall solution's fitness. The authors also store solutions produced during the evolutionary process so that the user can analyze the historical paths traversed by the algorithm. In their experiments they used roulette wheel selection with one elite individual as the reproduction method. A summary of the overall way in which this approach works is presented in Algorithm 5.

There exist some other examples of the use of cooperative coevolution as a framework, but most of them work in a similar way as the ones we have described

Algorithm 5 Co-Evolutionary Multiobjective Genetic Algorithm for Airframe Design

```
Input: A multiobjective optimization problem $MOP$, population size for the species
    $PS$, crossover rate $p_c$, mutation rate $p_m$, maximum number of generations $Gmax$,
    fitness penalty $FP$
Output: $TradeOffSolution$
  $gen \leftarrow 0$
  $NumPops \leftarrow GetNumberOfObjetives(MOP)$
  $Pops \leftarrow CreateSubpopulations(PS, NumPops)$
  for $i = 1$ to $NumPops$ do
    $RandomInitializationOfPopulations(Pops[i], PS)$
    $EvaluateFitnessOfIndividuals(Pops[i], PS)$
  end for
  while $gen < Gmax$ do
    $gen \leftarrow gen + 1$
    for each subpopulation $SubPop$ in $Pops$ do
      select $SubPop(gen)$ from $SubPop(gen - 1)$ based on fitness
      $ApplyGeneticOperators(SubPop(gen), p_c, p_m)$
      $EvaluateFitnessOfIndividuals(SubPop(gen), FP)$
    end for
  end while
  $TradeOffSolution \leftarrow GetTradeOffSolution(Pops)$
  return $TradeOffSolution$
```

in this section. Besides, none of them focuses on the solution of MOPs with a high number of decision variables, which is the main motivation of this work and therefore, their discussion was omitted.

Interactive Coevolutionary Genetic Algorithm (ICGA)

In [95], Barbosa and Barreto presented the *Interactive Coevolutionary Genetic Algorithm* (ICGA) for multi-objective optimization problems applied to graph layout generation. ICGA maintains two populations: one representing the graph layout population, which contains the coordinates of all vertices in the graph, and another representing a set of weights for the vertices' connections. In ICGA, each population is evolved by an independent genetic algorithm and their evolution is based on their fitness evaluation, which involves active user intervention. For each layout, several objectives are mathematically defined such that final fitness of each solution is obtained using the current set of weights. After that, the layout population evolves for a given number of iterations, with its fitness being calculated using a fixed set of weights. Then, a sample of layouts from the current population is shown to the user for its inspection; at that point, the user ranks each solution according to his/her personal preferences. So, it could be the case where the ranking given by the user is different from the current ranking in the population. Then, the population of the set of weights evolves, while the current layout population is kept frozen, according to a fitness value. Such fitness is computed based on how well a given set of weights ranks the population of layouts as compared to the ranking periodically given by the user. The aim is to find the set of weights whose ranking is the closest to the one provided by the user. This is done in the next way: each set of weights evaluates the population of layouts and provides its own ranking. The fitness of a given set of weights is considered better when its ranking is closer to the ranking provided by the user. Also, a set of weights that ranks a layout in a very different order from that of the user is considered as a bad solution and thus has a low fitness value. After an improved set of weights is obtained, the layouts population is evolved with its fitness evaluation being now performed using the new set of weights. This process is performed for a predefined number of cycles or until a satisfactory graph layout is obtained.

In ICGA, the coevolutionary process occurs when the fitness of an element in the layout population is computed based on the current set of weights, which at the same time depends on the evolution of the population containing the set of weights, whose fitness in turn, relies on how close the current set of weights mirrors the preferences provided by the user for each of the graph layouts. That is to say, the evolution of the layout population occurs in a fitness landscape that is constantly changing. In fact, it changes every time that a new set of weights is presented by the weights population. On the other hand, the fitness landscape of the weights population changes every time the user provides a new ranking for the current layout population. Compared to previous interactive evolutionary algorithms, this interactive coevolutionary GA can use a bigger population for the application in turn since the user only has to rank a sample from the corresponding population. In this approach, a genetic algorithm is the only EA used for both tasks: searching for a good solution (a graph layout) of the MOP and learning the user's preferences. One observation about this approach is that authors do not report how expensive is the algorithm, in terms of function evaluations. Also, it is not clear what is the impact of the user intervention and if this mechanism brings any advantage to the algorithm.

Cooperative Coevolutionary GDE3 (CCGDE3)

A cooperative CMOEA for large scale multi-objective optimization is proposed by Miguel and Coello in [72], the so-called *Cooperative Coevolutionary GDE3* (CCGDE3). This Algorithm uses the CCGA [73] framework (adapted to multi-objective optimization) and GDE3 [74] as the main multi-objective optimizer. CCGDE3 includes the following processes: it starts with a division of the decision variables space where the vector $\vec{x}$ representing the set of $D \in \mathbb{N}$ decision variables is divided into $d \in \mathbb{N}$ smaller vectors of equal size. Each subcomponent is created using a random assignment of decision variables, so that the probability of grouping interacting variables into the same species in non-separable problems is increased. Also, d species of NP individual are created. Then, each of these d species are assigned a D/d number of different decision variables in a random way. In other words, each subpopulation is in charge of a subcomponent from D. After the subpopulations are created, a random initialization of all the species' populations is performed. Thereafter, CCGDE3 performs the evolution of each of the subpopulations for a given number of *generations*, throughout a certain number of *cycles*. Such process continues for a certain (pre-defined) number of cycles and generations or until a stopping criterion is fulfilled. The outcome of the algorithm is the set of solutions that are globally non-dominated, v.g., with respect to all of the non-dominated solutions from all the species' populations. The way cooperation between subpopulations is performed is described next. At the start of the algorithm, when no knowledge has been acquired, collaborations are performed in a random way, by selecting a random individual from each species and assembling a complete set of solutions to be evaluated in the set of objective functions of the MOP. Thereafter, the results from the evaluations are given back to each individual. After the first generation, collaborations will take place when the resulting offspring subpopulations need to be evaluated. Therefore, evaluation is performed by joining the individual under evaluation with randomly selected components from the set of non-dominated solutions of the other species and then applying this new assembled solutions to the objective functions. At the end, the non-dominated solutions from the set of non-dominated solutions of each species' populations is computed and given as the final result. CCGDE3 was benchmarked with respect to GDE3 [74] and NSGA-II [75]. The authors used a number of decision variables that ranged from 200 up to 5000 when solving ZDT1, ZDT2, ZDT3 and ZDT6 [64]. Adopting the hypervolume as the performance indicator [25], each MOEA was run until it obtained an approximation of the Pareto front that had a hypervolume value of 95% with respect to the true Pareto front. Results showed that CCGDE3 was the fastest algorithm, scaling better than GDE3 and NSGA-II when the number of variables was large. A summary of the overall way in which this approach works is presented in Algorithm 6.

Algorithm 6 Cooperative Coevolutionary GDE3

```
Input: $NP$, $Cycles$, $Gmax$, $NumEsp$
Output: $SolutionSet$
  $Pobs \leftarrow Populations(NP, NumEsp)$
  $InitializeSpecies(Pobs)$
  for $j \leftarrow 1$ to $Cycles$ do
    for $i \leftarrow 1$ to $NumEsp$ do
      for $k \leftarrow 1$ to $Gmax$ do
        $MOEA(Pobs[i])$
      end for
    end for
  end for
  $SolutionSet \leftarrow ObtainNonDominatedSet(Pobs)$
  return $SolutionSet$
```

The main drawback of this approach is that it is not able to deal with multi-frontal MOPs, such as ZDT4. Since the problem is divided along decision variables space, it makes it harder for the framework to give solutions in this kind of problems. In fact it was observed that CCGDE3 tends to have premature converge in these cases.

5.1.3 Cooperative CMOEAS based on objective functions decomposition

In this case, the MOP is decomposed along the objective functions of the problem. Each objective function of the problem is assigned to a certain species' population and all populations cooperate to approximate the whole Pareto Front. Each species has a population formed by individuals that represent a solution to the MOP. Individuals from each species compute all the objective functions evaluations in the same way as traditional MOEAs. However, the main difference is that the fitness value of an individual in a certain species is given only by its corresponding objective function. Hence, individuals are guided by their specific objective function, in order to search in different regions of the Pareto Front. Figure 5.3 shows the way collaboration is

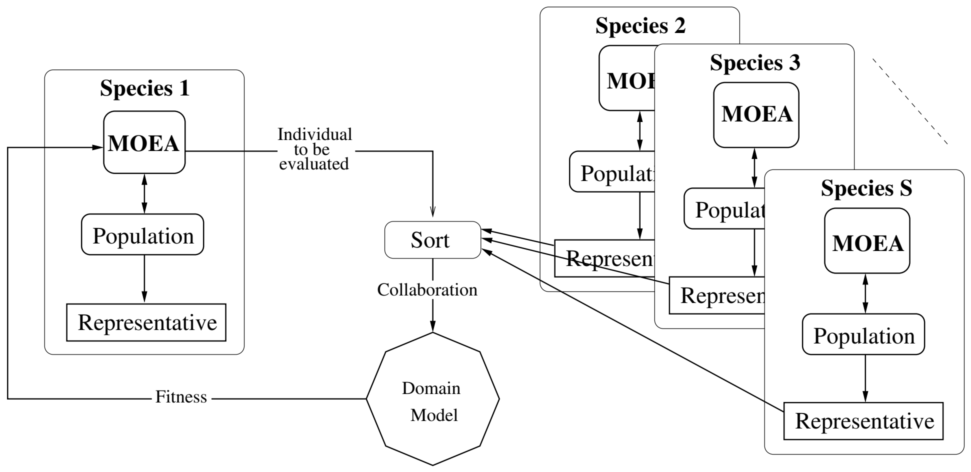

Figure 5.3: Cooperative collaboration based on an objective function decomposition architecture. Here, each species is representing a single objective function. In order to obtain its fitness value, a solution from species 1 is joined with the representatives of each species and then is evaluated under the domain model of each species, i.e., over their corresponding objective function.

performed in order to create the coevolutionary effect. Next, we present the most representative approaches within this category, but more examples can be found in [114, 115].

Multiple Populations for Multiple Objectives (MPMO)

A cooperative CMOEA which uses *Multiple Populations for Multiple Objectives* (MPMO) was presented in [116]. MPMO creates multiple species' populations to deal with the several objectives of the MOP, in such way that each species represents only one objective function and all species cooperate to approximate the whole Pareto Front. MPMO is considered a cooperative coevolutionary approach because it uses multiple species (populations) to solve problems in a cooperative way. At each generation, individuals from each species are evaluated using all the objective functions as done in traditional MOEAs. Nevertheless, once evaluated, the fitness value of an individual from the s^{th} species is assigned by the s^{th} objective function of the MOP, where $1 \leq s \leq S$ and S is the number of species and objective functions of the MOP. This way, individuals search along their respective objective function, assigned to their corresponding species, so that making use of all the S species MPMO will search different regions of the Pareto Front at the same time. Since every species is assigned a different objective function, an information-sharing mechanism is used to share their search knowledge and communicate it among each other.

The authors of this approach adopt PSO as their optimizer for each species' population and create a coevolutionary multi-swarm PSO (CMPSO) making use of the MPMO scheme to solve MOPs. CMPSO adopts an external archive as the instance of the information-sharing mechanism from MPMO, with two extra techniques to enhance the approach's performance. One is used to modify the way in which velocity of the particles is updated. Such modification consists on the use of the information obtained from the external archive. The shared archive stores the non-dominated solutions found so far by different species and is updated at each cycle of CMPSO. The velocity and position of a particle from each species' swarm are updated by taking into account its personal experience as well as the swarm's global experience in addition to the experience taken from the external archive of non-dominated solutions. In this way, all the species can share their knowledge to speed up convergence towards the whole Pareto Front. The second approach is to adopt an elitist learning strategy (ELS), which updates the archive in order to produce an appropriate diversity so that CMPSO is able to deal with MOPs having multiple local Pareto Fronts or with complicated Pareto sets.

The main advantages of the CMPSO algorithm are the following:

- To prevent the difficulty of performing fitness assignment and at the same time to obtain a benefit from having each swarm with a conventional PSO, or an improved PSO algorithm for optimizing a single objective, each swarm in CMPASO is in charge of only one objective.

- As an external archive is adopted to store the non-dominated solutions found so far in the process, each swarm can communicate with the others through this mechanism and, therefore, they can use the knowledge obtained by the other species.

- As an elitist strategy is used in the archive update mechanism, the algorithm is able to deal with MOPs having multiple local fronts.

CMPSO is benchmarked against some state-of-the-art MOEAs and MOPSOs (see [116] for further details about the test functions adopted) and results are compared using the Inverted Generational Distance (IGD) [117] indicator. Experimental results show how CMPSO outperforms, in most cases, the other algorithms on the ZDT test problems [64]. Also, it shows a remarkably better performance on the DTLZ [3] and WFG [4] test problems. When dealing with the UF [118] test problems, which has complicated Pareto sets, CMPSO is also one of the most outstanding approaches.

The main drawback of CMPSO is that, since each species optimizes only one objective function, this might contribute to the creation of more solutions on the extreme values of each objective, resulting in a poor approximation of the whole Pareto Front. Also, it has a poor performance in multi-frontal problems. Another observation has to be made over the selection method that its authors adopt when using archive information, since a random selection is misleading and can lead to undesired waste of resources (by means of function evaluations).

Preference-inspired co-evolutionary algorithm using weight vectors

A preference-inspired co-evolutionary algorithm using weight vectors (PICEA-w) is presented in [119]. PICEA-w was created with the objective of alleviating the difficulties that decomposition-based approaches have when dealing with MOPs having complex Pareto front geometries. This approach adopts a decomposition of the problems methodology that decomposes a MOP into a set of single objective subproblems defined by means of several scalarizing functions with different weights, where each weight vector is used as a search direction to define a scalar function. In PICEA-w, weights are adaptively modified by co-evolving them with candidate solutions along the search process, in order to construct suitable weights in an adaptively manner during the optimization process and with this guide candidate solutions towards the Pareto optimal front effectively. In PICEA-w, candidate solutions are ranked by each of the weighted scalarizing functions, therefore, a ranking matrix is created. Then, the fitness of candidate solutions is computed based on such ranking matrix. Thereafter, weight vectors are coevolved with the candidate solutions to create an optimal distribution, and these at the same time work to create a balance between exploration and exploitation. For each selected solution, a weight which ranks this solution as the best is selected. Such weight must maintain the convergence and exploitation as well as its distant from the solution.

PICEA-w is implemented within a $(\mu + \lambda)$ elitist framework. Populations of candidate solutions and weight vectors, S and W (of size N and N_w) respectively, are evolved for a fixed number of generations. At each generation, parents are subjected to genetic variation operators to produce N offspring (S_c). At the same time, N_w new weight vectors (W_c), are randomly generated. Thereafter, $S \bigcup S_c$ and $W \bigcup W_c$ are sorted according to fitness and a truncation selection procedure is applied to select the best N solutions and N_w weight vectors are the new elements of PICEA-w for the next generation. Additionally, an offline archive is adopted to store all the non-dominated solutions found during the search. In order to obtain a good distribution of solutions, SPEA2's [120] clustering technique is applied after the optimization process has been performed. PICEA-w was benchmarked against 4 variations of its framework (see [119] for further details) when solving 8 test problems constructed by applying different shape functions provided in the WFG toolkit to the standard WFG4 benchmark problem [4]. The WFG parameters k (position parameter) and l (distance parameter) were set to 18 and 14 respectively, creating MOPs with $n = k + l = 32$ decision variables. These problems were adopted with 2, 4 and 7

Table 5.1: Summary of Cooperative CMOEAs

Reference	**Algorithm**	**Type of decomposition**	**Subpopulations**	**Representation**	**Optimizer**
[96]	MOCCGA	Decision variables space	One per decision variable	Real	MOGA
[72]	CCGDE3	Decision variables space	Groups of decision variables	Real	GDE3
[95]	ICGA	Decision variables space	Two populations	Graph layout	Genetic Algorithm
[116]	MPMO	Objective space	One per objective function	Real	PSO
[119]	PICEA-w	Objective space	Two populations	Real	Evolution Strategies

objective functions. PICEA-w showed to be less sensitive to the problem geometry, and outperformed other leading decomposition-based algorithms on problems with more than 4 objectives (many-objective instances). Moreover, it showed that when guiding candidate solution towards the Pareto optimal front, weights also evolve towards the optimal distribution when adopting a coevolutionary strategy. One of the observations about this approach is that their study does not include additional MOP difficulties other than geometrical ones. Also, a comparison against state of the art MOEAs is needed to have a more general idea about the efficiency of this approach.

Table 5.1 summarizes the cooperative CMOEAs we have just described.

5.2 Pathologies of The Coevolutionary Framework

Since evaluation in CMOEAs is based on coevolving individuals, coevolution settings may cause inaccurate evaluation, leading to problems such as *over-specialization*, *red queen effect*, *disengagement*, etc [121, 122, 123]. Next we briefly describe some of these pathologies in order to understand some of the issues that may arise when adopting CMOEAs.

5.2.1 Instransitivity

One problem feature that has received particular interest in the past is that of intransitivity [124]. As described by De Jong, a relation R is transitive if $aRb \wedge bRc$ implies aRc; if this cannot be guaranteed, the relation is then intransitive. The existence of such intransitive relations in a coevolution problem can lead to *cycling*, i.e., the recurrence of previously visited states of the species. Intransitivity has been viewed as an inherent feature of coevolution that can make CMOEAs unreliable. Indeed, the resulting problem of cycling has been thought of as an obstacle that could prevent coevolution from becoming a reliable problem solving technique, which is believed that, like the local minima problem in gradient-descent methods, is an intrinsic problem that cannot be completely eliminated[125].

5.2.2 Disengagement

Coevolutionary disengagement takes place when one population outperforms another to the extent that individuals from the same species become indistinguishable from one another (in terms of fitness) [126]. When this occurs, coevolved subpopulations become disassociated and selection acts with no specific direction, causing the coevolutionary process to drift and, in many cases, without any possibility of generating acceptable results.

5.2.3 Red Queen effect

Coevolutionary algorithms can suffer from the so-called *red queen effect*. This happens when, through their interaction, species alter each other's fitness landscapes [127]. Such effect significantly affects the performance of the coevolutionary process, creating fitness ambiguities that cause improvements in the performance of individuals to be considered as undesired changes and vice versa.

Chapter 6

Large Scale Optimization Schemes

As discussed in the previous chapter, cooperative coevolution is an approach for evolving solutions from different populations which are evaluated based on how well they perform together. The advantage of cooperative coevolutionary algorithms is the decomposition of the problem which allows us to learn different parts of the problem instead of the whole problem at once. Also, cooperative coevolution has already shown to be a very good alternative to deal with large scale MOPs [72]. However, previous research within the field of global optimization has shown that cooperative coevolutionary algorithms are biased towards equilibrium states. Since studies concerning cooperative coevolutionary algorithms used for solving multi-objective optimization problems were initiated, no attention had been paid to this issue. Next, we provide empirical evidence of the existence of these problems within the multi-objective optimization field and present a novel cooperative coevolution framework which, through the use of the concept of Nash equilibrium, alleviates some of those optimization-related pathologies present in cooperative coevolutionary algorithms.

6.1 Selection scheme for a faster converge in cooperative coevolution

In order to describe and perform experiments over a new model selection scheme for cooperative coevolution, with the aim of being fitter for large scale MOPs, we adopt the cooperative coevolutionary model proposed by Potter [73], where each population contains individuals that represent a particular component (in decision varibles space) of the problem, so that one member from each population is needed in order to assemble a complete solution. Evaluation of an individual from a particular population is performed by joining the individual with collaborating partners from other populations. Aside from evaluation, the populations are evolved independently. An abstract mathematical model for this system comes from evolutionary game theory (EGT). EGT is seen as a way of thinking about evolution at the phenotypic level when the fitness of particular individuals depend on their frequencies in the population [128]. As used by NSCCGA and MOCCGA, the common way to perform the evaluation of each individual is by taking one representative from the other populations that belongs to the best set of solutions found so far. We believe that this way of doing collaboration is the main cause for the aforementioned pathologies, because each species is choosing representatives from other populations having in mind only the performance of the species alone and not as a whole team among all of the species. This kind of interaction makes coevolution to explore only narrow regions of the collaboration space, which suggests that evolution is strongly attracted to certain regions of the search space. However, these regions do not necessarily correspond to (fitness-based) optimal solutions, and coevolution often converges to sub-optimal equilibria. Since in multi-objective optimization, objectives are in conflict, we believe that making use of Nash equilibrium for finding better collaborations is the best option, since it gives a solution of a non-cooperative game. According to Nash, each participant of the game has his own strategy set and objective function. Then, during the game, each player searches for the optimal strategy while other players' strategies are fixed. The game is conducted in this frame and when no player can further improve his criterion, the system is regarded as having reached a state of equilibrium, known as *Nash equilibrium*. This is exactly the way MOPs are managed by EAs, so this idea can be applied to develop a novel CC framework, which, instead of looking for best collaborations from the point of view of an specific species, will look for ideal collaborations which take into account all of the objectives in order to perform better interactions among species. Our proposed approach is described next.

6.1.1 Description of the proposed approach

Our approach works as follows: at the beginning, it divides the vector of decision variables $\vec{x}$ of dimension $D \in \mathbb{N}$ into $S \in \mathbb{N}$ subcomponents, where S is equal to the number of objective functions in the problem. Each subcomponent is created from a random grouping of decision variables in order to increase the probability of grouping interacting variables in non-separable problems. At the same time, S subpopulations (species) are created, each one with NP individuals, and these S subpopulations are assigned their corresponding decision variables in a random way. This means that to each subpopulation, it corresponds a subcomponent from S which had been already created. Thus, every subpopulation will have a total of m decision variables. At the same time, each species will be assigned with an specific objective function, so that there will be as many species as objective functions and each of them will do the fitness assignment of their individuals according to that, as will be described next. Once the subpopulations are created, the algorithm does a random initialization of all the individuals across all subpopulations. Then, the algorithm performs the *cycles* in which the evolution of each of the subpopulations is done for a given number of *generations*. This will continue until the stopping condition is reached, and at the end, the solutions that are globally non-dominated (i.e., with respect to all the subpopulations), constitute the outcome of the algorithm. The collaboration among the subpopulations takes place in the next way: in the first generation, random collaborations are formed and evaluated, obtaining a random individual from each subpopulation and forming a complete set of solutions to be evaluated in their objective functions. Then, the results from the evaluation are assigned back to the individual under evaluation. After the first generation, the resulting child subpopulations Q_1 to Q_S will be evaluated by forming collaborations with individuals from the other species which are in a Nash equilibrium according to their fitness values. As mentioned before, the usual way to perform collaborations is by selecting representatives from other species which perform the best according to their respective subpopulation. However, we believe that this is the main cause for the tendency of CCAs to fall in ESS. So, we propose to find an ideal collaboration by making use of Nash equilibrium. Next, we give a brief description of such concept.

6.1.2 Equilibriums for selection of strategies

The formal description of Nash equilibrium is presented next.

Definition 6.1. Normal form game

Let $P = \{1, \ldots, n\}$ be the set of players, $i \in P$, $a^i_x \in \Sigma^i$ be an element of the set of simple plays, and s^i_x be a strategy of player i, $s^i_x \in S_i$; let $G = (S_1, \ldots, S_n; u_1, \ldots, u_n)$ be the game in normal form [129] where:

- A strategy is a sequence of actions $s^i_x = a^i_1 \ldots a^i_n$.
- A strategy profile is an n-tuple of strategies $(s_1, \ldots, s_n)$; one strategy per player.
- S_i is the set of strategies for the i_{th} player.
- $\{S_1, \ldots, S_n\}$ is the set of all the S_i strategies.
- $\{u_1, \ldots, u_n\}$ is the set of all payoff functions; one per player.
- $u_i(s_1, \ldots, s_n) = r$, where $(s_1, \ldots, s_n) \in S_1 \times \cdots \times S_n, r \in \mathbb{R}$.

6.1.3 Nash equilibrium

The Nash equilibrium [129] is a widely used mathematical concept, especially in the modelling of non-cooperative games. To identify the strategy profiles that fit the condition of Nash equilibrium, every strategy profile is evaluated with the payoff functions of the players. Then, the chosen profiles are those that, for every player, are the options that produce the lowest loss for each one, individually, in a non-cooperative way. In a more formal way, let $s_1^*, \ldots, s_n^*$ and s_i^* be the non-cooperative player's strategies from i to the $n-1$ other players' strategies. So $(s_1^*, \ldots, s_i^*, \ldots, s_n^*)$ fits the Nash equilibrium condition, if and only if it maximizes the corresponding payoff function:

$$\begin{aligned} u_i(s_1^*, \ldots, s_i^*, \ldots, s_n^*) \geq u_i(s_1^*, \ldots, s_i, \ldots, s_n^*) \\ \forall i \in P, s_i \in S_i. \end{aligned} \tag{6.1}$$

Every strategy profile is a payoff function valued and is compared with all the others, to determine whether or not it is dominated. Given a strategy profile x_1 for each player i, the strategy profile is modified by altering the player's current strategy while keeping the strategies of the other $n-1$ players unchanged; if any deviation from x_1, evaluated by u_i, dominates it, that means that player i's profit is higher by $u_i(x_2)$. So, x_1 is dominated by x_2's profile and, therefore, x_1 is discarded. All the dominated profiles are discarded and the non-dominated profiles are the ones that fit the Nash equilibrium. Any game in (finite) normal form has at least one strategy profile that fits the Nash equilibrium [129]. Observe that in Nash equilibrium, every player is applying a non-cooperative perspective which turns out to be not as bad for him as the other players' strategies.

With all this in mind, we propose to make collaborations by finding the best unions according to the previous objective values found so far. For this purpose, we create all the combinations from the function of the individual being evaluated and the posible values given by the rest of the other populations. The way combinations are formed is depicted in Figure 6.1, in which we show an example of a three-dimensional MOP using 2 individuals for each population. Once we have this, we consider that for our purpose, each objective function value can be used as an strategy. With this, we can build a utility function from each species point of view, in order to create a non-cooperative game where each player is an species, and its interest is focused on its specific objective function. For this purpose, we adopted SPEA2's fitness assignment strategy [120] within each species. Fitness assignment operates as follows: having an individual i which belongs to the Q_j species being evolved, we will create all the combinations of the objectives from the current species with the objectives of the other species. Having this composed objective vectors we will evaluate each of these combinations according to the current species population. Thus, at the end we will have a set of strategies from which we will find the Nash equilibrium according to the fitness values that will be used as the utility functions values. So, the individuals which bring a Nash equilibrium will be selected from the subpopulations, P_1 to P_S, of the previous generation in order to collaborate. This will allow us to evaluate the new individual. The collaboration procedure is shown in Figure 6.2. The way Nash strategy profiles are obtained is shown in Algorithm 7.

Algorithm 7 Nash equilibrium algorithm for selection of strategies for CC collaborations.

```
Input: Input each strategy profile and its payoff value
 1: for all $x = (x_1, \ldots, x_m)$ strategy profiles do
 2:   for all player $i = (1, \ldots, n)$ do
 3:     if $x$ is labeled as non-dominated then
 4:       Do the derivations in $x$ for player $i$
 5:       if $x$ is dominated by at least one derivation of $i$ then
 6:         label $x$ as dominated {move to the next strategy profile}
 7:       end if
 8:     else
 9:       {move to the next strategy profile}
10:     end if
11:   end for
12: end for
```

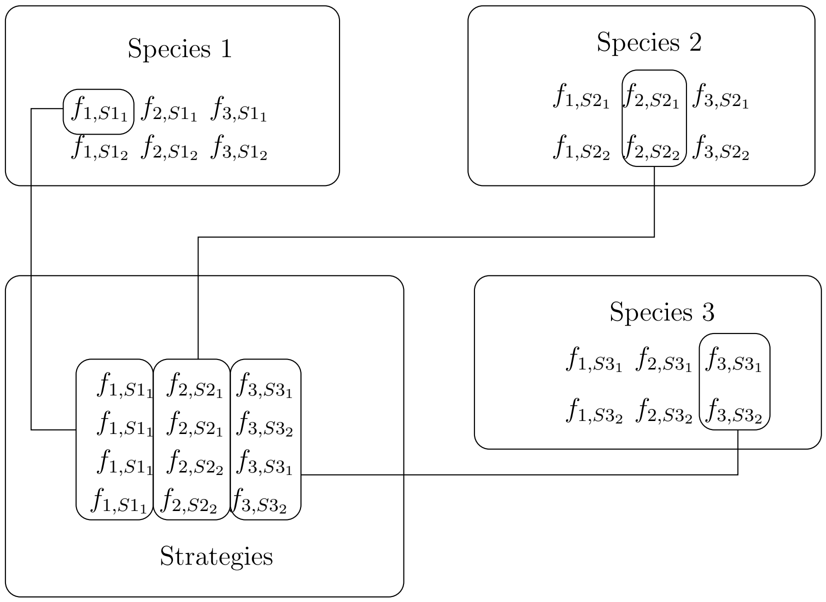
Species 1
$f_{1,S1_1}$ $f_{2,S1_1}$ $f_{3,S1_1}$
$f_{1,S1_2}$ $f_{2,S1_2}$ $f_{3,S1_2}$
Species 2
$f_{1,S2_1}$ $f_{2,S2_1}$ $f_{3,S2_1}$
$f_{1,S2_2}$ $f_{2,S2_2}$ $f_{3,S2_2}$
Species 3
$f_{1,S3_1}$ $f_{2,S3_1}$ $f_{3,S3_1}$
$f_{1,S3_2}$ $f_{2,S3_2}$ $f_{3,S3_2}$
$f_{1,S1_1}$ $f_{2,S2_1}$ $f_{3,S3_1}$
$f_{1,S1_1}$ $f_{2,S2_1}$ $f_{3,S3_2}$
$f_{1,S1_1}$ $f_{2,S2_2}$ $f_{3,S3_1}$
$f_{1,S1_1}$ $f_{2,S2_2}$ $f_{3,S3_2}$
Strategies

Figure 6.1: Strategies for the creation of one individual in a CCA with 3 Species (objective functions) and 2 individuals in each subpopulation.

The algorithm iterates until some termination condition is fulfilled (usually when a certain predefined number of cycles is reached). At the end, we get the non-dominated solutions from the non-dominated individuals of each subpopulations, in order to obtain a final set of solutions for the problem being solved. A summary of the way in which our approach works is presented in Algorithm 9.

6.1.4 Experimental Studies

For the purposes of this study, we adopted the Zitzler-Deb-Thiele (ZDT) [2] and the Deb-Thiele-Laumanns-Zitzler (DTLZ) test suites [3] with two and three objectives, respectively. Since the main objective is to evaluate the performance of our approach in terms of efficiency when solving MOPs, we will analyze our results with respect to those of the NSCCGA [97] and GCEA [130]. For this sake, we established a predefined number of function evaluations that the algorithms can use, to analyze

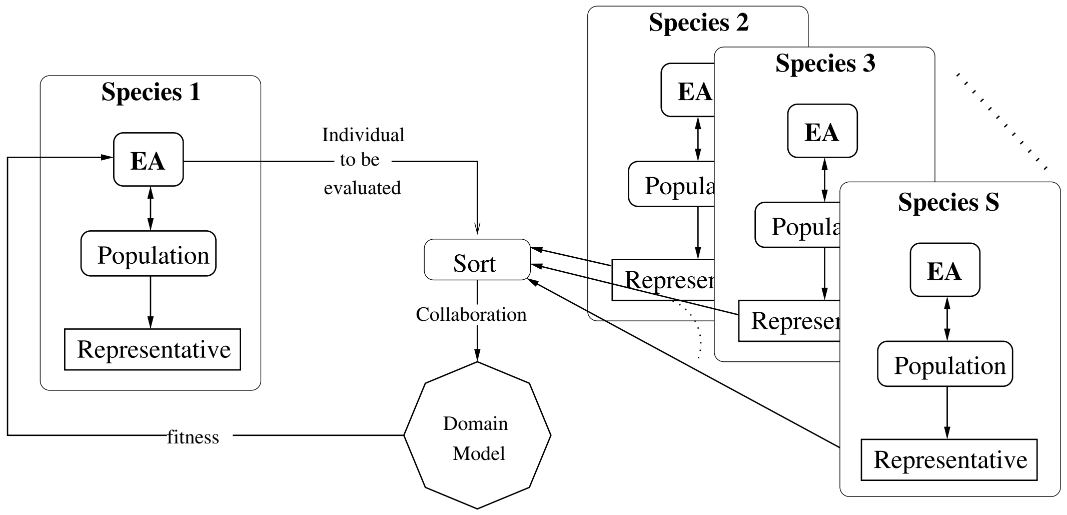

Figure 6.2: Cooperative coevolutionary collaboration architecture from the perspective of species number 1. Here, we assume that we have S species, where the representative of each species for collaboration is the one that fits a Nash equilibrium.

how they behave with the same resources. For assessing our results we adopted the hypervolume performance indicator [76]. The hypervolume is obtained by computing the volume (in objective function space) of the non-dominated set of solutions Q that minimize a MOP. For every solution $i \in Q$, a hypercube v_i is generated with a reference point W and the solution i as its diagonal corner of the hypercube. The reference point W can be generated by building a vector of worst possible objective function values. Then, the hypervolume (HV) is computed as the union of all the found hypercubes as follows:

$$HV = volume\left(\bigcup_{i=1}^{|Q|} v_i\right) \tag{6.2}$$

The aim of this study is to identify which of the MOEAs being compared is able to get closer to the true Pareto front using the same number of objective function evaluations.

Algorithm 8 Cooperative Coevolutionary Framework

```
Input: $NP$, $Cycles$, $Gmax$, $NumEsp$
Output: $SolutionSet$
  $Pobs \leftarrow Populations(NP, NumEsp)$
  $InitializeSpecies(Pobs)$
  for $j \leftarrow 1$ to $Cycles$ do
    for $i \leftarrow 1$ to $NumEsp$ do
      for $k \leftarrow 1$ to $Gmax$ do
        $MOEA(Pobs[i])$
      end for
    end for
  end for
  $SolutionSet \leftarrow ObtainNonDominatedSet(Pobs)$
  return $SolutionSet$
```

6.1.5 Parameterization

The parameters of each MOEA used in our study were chosen in such a way that we could do a fair comparison among them. Although all the algorithms used for our comparative study are CCAs, the specific nature of each of them requires different parameters. Thus, for our approach and for GCEA there will be as many species as objective functions, whereas for NSCCGA there are as many species as decision variables. For all the algorithms we used a small populations size of 16 individuals for each subpopulation (species). This was done, in order to set an environment where the aforementioned pathologies arise. For the ZDT test suite, we used 67 cycles and the number of species for NSCCGA was set to 30, since that is the number of decision variables for these problems. The exception is ZDT4 where the number of decision variables is 10. For this problem, we used 200 cycles since this was enough to bring a good converge to the Pareto front. For our approach (NashCC) and GCEA, the number of species was set to 2, because the ZDT problems have 2 objectives. The number of cycles was set to 1000. For the DTLZ test suite, we could only compare results against CCNSGA since GCEA is not able to scale to more than two objectives. The number of species for NSCCGA was set to 12, since the DTLZ problems use by default that number of decision variables. In this case, 1250 cycles were used. For our proposed NashCC, the number of species was set to 3, since these problems have 3 objectives. The number of cycles was set to 5000. For all algorithms and problems, the distribution indexes for the SBX and polynomial-based mutation operators [75], were set as: $\eta_c = 20$ and $\eta_m = 20$, respectively. The crossover probability was set to $p_c = 0.9$ and the mutation probability was set to $p_m = 0.01$. Finally, we used just one generation for each species per cycle for all approaches. All of this in order to use the same number of function evaluations in all CCAs and to allow for a fair comparison of results.

6.1.6 Analysis of results

In our experiments, we obtained the hypervolume value over the 25 independents runs performed. Tables 6.1 and 6.3 show results of the hypervolume measure for the ZDT and DTLZ test suites, respectively, as well as the reference points used for each of the problems. To ease the analysis of the results in these tables, the cells containing the best hypervolume value for each problem have a grey colored background.

From Figures 6.3 to 6.29, we plot the results of the median of the 25 runs. These plots are shown for the ZDT and the DTLZ test problems, respectively. We can observe that, using the same number of function evaluations, our proposed NashCC is able to get closer than NSCCGA and GCEA to the true Pareto front in all the problems. It is clear that NashCC is much faster than the other two algorithms in terms of number of evaluations. As can be observed, NSCCGA and GCEA have premature converge in most problems, which makes them fall in false fronts in the case of multi-frontal problems such as ZDT4. This confirms that the tendency of CCAs to fall into ESS is also present when dealing with MOPs and that looking for good collaborations is an effective way to alleviate this problem.

Table 6.1: Hypervolume values for the ZDT test suite

	ZDT1			ZDT2			ZDT3			ZDT4			ZDT6		
Algorithm	NashCC	CCNSGA	CCGT	NashCC	CCNSGA	CCGT	NashCC	CCNSGA	CCGT	NashCC	CCNSGA	CCGT	NashCC	CCNSGA	CCGT
Ref. Points	(1 8)			(1 9)			(1 9)			(1 900)			(1 10)		
Best	7.6285	7.0870	5.4413	8.3108	7.8108	2.5263	9.0247	8.4213	4.2544	899.6179	890.5287	532.1361	6.7281	3.7310	0.7373
	7.6269	7.0843	4.0366	8.3106	7.7170	2.5197	9.0188	8.4140	3.9618	899.6039	889.9144	517.7009	6.7246	3.7002	0.4183
	7.6268	7.0667	3.3556	8.3089	7.7149	2.4991	9.0185	8.3796	3.7482	899.5833	886.6908	484.3817	6.7196	3.4643	0.4033
	7.6264	7.0632	3.0911	8.3082	7.6785	2.4911	9.0146	8.3700	3.7173	899.4977	885.5299	454.5236	6.7187	3.3439	0.3968
	7.6256	7.0305	3.0564	8.3070	7.6688	2.4911	9.0141	8.3097	3.7161	899.4590	884.8965	443.1219	6.7186	3.3223	0.3579
	7.6253	6.9974	2.9899	8.3053	7.6135	2.4712	9.0137	8.2885	3.6335	899.4424	884.8232	425.7738	6.7181	3.3147	0.3444
	7.6244	6.9758	2.9129	8.3012	7.6082	2.0796	9.0119	8.2882	3.5062	899.4239	884.8077	420.1604	6.7163	3.3110	0.3347
	7.6241	6.9534	2.9070	8.3010	7.5968	2.0732	9.0112	8.2664	3.4547	899.2720	884.7901	418.4893	6.7094	3.2766	0.3298
	7.6239	6.9454	2.6500	8.2991	7.5676	1.8883	9.0047	8.2413	3.4528	899.2685	884.3186	415.4923	6.7057	3.2499	0.3197
	7.6233	6.9338	2.5859	8.2987	7.5648	1.8712	9.0040	8.1980	3.4151	899.2629	880.8902	413.5572	6.7055	3.2174	0.3132
	7.6228	6.9297	2.5426	8.2982	7.5565	1.7314	9.0035	8.1663	3.2693	899.2607	879.9972	411.7965	6.6999	3.1605	0.2874
	7.6226	6.9118	2.5214	8.2964	7.5375	1.7263	9.0001	8.1254	3.2510	899.2540	878.8118	410.1775	6.6987	3.0051	0.2680
Median	7.6225	6.9105	2.4754	8.2944	7.5193	1.6522	8.9982	8.0892	3.2009	899.2539	878.0266	375.7467	6.6984	2.9893	0.2642
	7.6212	6.9095	2.2863	8.2943	7.4892	1.6327	8.9924	8.0647	3.0842	899.2509	877.0857	366.9912	6.6962	2.9829	0.2334
	7.6210	6.8809	2.1792	8.2940	7.4726	1.6086	8.9717	8.0465	2.9459	899.2400	877.0416	366.0214	6.6891	2.8825	0.2130
	7.6198	6.8771	2.0903	8.2934	7.4721	1.5961	8.9706	7.9992	2.9135	899.2394	876.6987	338.2662	6.6840	2.8581	0.1778
	7.6188	6.8707	2.0473	8.2934	7.4207	1.5661	8.9697	7.9949	2.8526	899.2325	876.6007	336.7565	6.6812	2.7485	0.1693
	7.6185	6.8668	2.0357	8.2921	7.3748	1.5278	8.9682	7.9943	2.8100	899.0523	876.0462	333.4452	6.6667	1.9087	0.0928
	7.6183	6.8500	2.0352	8.2920	7.3586	1.4419	8.9649	7.9875	2.7137	899.0446	873.8957	330.1884	6.6563	1.9012	0.0798
	7.6158	6.8195	2.0239	8.2908	7.3559	1.4396	8.9584	7.9737	2.6660	899.0429	873.8497	326.3380	6.6543	1.8504	0.0503
	7.6153	6.8105	2.0183	8.2904	7.3497	1.3578	8.9512	7.9686	2.6425	899.0254	872.5329	325.2272	6.6456	1.7674	0.0367
	7.6152	6.7614	1.9923	8.2893	7.2987	1.3367	8.9507	7.9513	2.5738	898.8857	871.4517	322.9623	6.6377	1.7088	0.0343
	7.6138	6.7308	1.9235	8.2888	7.2484	1.1220	8.9482	7.9195	2.5155	898.8794	870.6939	306.6790	6.6339	0.9597	0.0243
	7.6112	6.6191	1.8730	8.2830	7.2318	1.0963	8.9120	7.8683	2.4955	898.8418	866.3119	290.3387	6.6194	0.9265	0.0150
Worst	7.5758	6.6184	1.4873	8.2790	7.0012	1.0383	8.2848	7.8366	2.4719	898.2626	857.3100	281.6815	6.5977	0.8791	0.0059
Average	7.6195	6.9002	2.5823	8.2968	7.4891	1.7914	8.9592	8.1265	3.1707	899.2079	878.5418	385.9181	6.6849	2.6584	0.2363
STD	0.0102	0.1273	0.8232	0.0084	0.1833	0.4819	0.1435	0.1794	0.5050	0.2949	7.6306	68.3190	0.0363	0.8878	0.1727

Table 6.2: Hypervolume values for the DTLZ test suite

	DTLZ1		DTLZ2		DTLZ3		DTLZ4		DTLZ5		DTLZ6		DTLZ7	
Algorithm	NashCC	CCNSGA	NashCC	CCNSGA	NashCC	CCNSGA	NashCC	CCNSGA	NashCC	CCNSGA	NashCC	CCNSGA	NashCC	CCNSGA
Ref. Points	(35 35 35)		(1.2 1.2 1.2)		(50 50 50)		(1.2 1.2 1.2)		(1.2 1.2 1.2)		(7 7 7)		(7 7 7)	
Best	42876.2811	42861.9638	1.0987	1.0976	124999.3152	124975.9658	1.0628	0.7717	0.7180	0.7027	334.8787	323.5990	498.3214	469.9119
	42874.8800	42847.0050	1.0971	1.0928	124999.1670	124865.7086	1.0294	0.7716	0.7172	0.7019	334.8272	319.1156	498.0460	449.7765
	42874.8800	42819.1787	1.0963	1.0926	124997.6336	124715.7241	0.9992	0.7691	0.7169	0.7018	334.7990	315.1843	497.8648	430.2499
	42874.8431	42804.0798	1.0958	1.0925	124994.5279	124655.8189	0.9992	0.7685	0.7169	0.6986	334.7064	311.0600	497.5445	417.8822
	42874.8248	42742.8616	1.0956	1.0922	124994.3927	124623.6335	0.9992	0.7681	0.7169	0.6977	334.6832	310.9483	497.3881	415.6720
	42874.7272	42707.8119	1.0924	1.0896	124993.7485	124565.9743	0.9991	0.7671	0.7165	0.6969	334.6208	309.8232	497.3302	400.1097
	42874.5948	42596.9558	1.0912	1.0886	124991.6194	124355.8553	0.9991	0.7646	0.7164	0.6967	334.5886	304.8691	497.1576	398.6594
	42874.5795	42525.9315	1.0897	1.0877	124990.9612	123498.8561	0.9591	0.2880	0.7164	0.6962	334.5622	302.2213	496.9787	392.3261
	42873.8784	42501.7135	1.0890	1.0864	124990.5698	123367.9406	0.7627	0.2880	0.7161	0.6962	334.4871	294.6850	496.4218	390.9044
	42873.8354	42490.1736	1.0885	1.0858	124990.2125	122602.1798	0.7627	0.2880	0.7155	0.6944	334.4834	286.5837	496.2432	388.2298
	42872.9553	42440.4895	1.0858	1.0857	124989.1850	122528.6792	0.7627	0.2880	0.7153	0.6934	334.4430	286.5616	496.2147	383.3258
	42872.1874	42395.1893	1.0855	1.0817	124988.6998	122354.9181	0.7621	0.2880	0.7149	0.6929	334.3872	278.2876	494.8938	382.6247
Median	42872.1758	42252.9449	1.0850	1.0815	124988.2743	122249.1144	0.7621	0.2880	0.7148	0.6917	334.3602	271.4169	494.8744	382.3463
	42872.0777	42251.2821	1.0843	1.0801	124986.7070	121018.5017	0.7621	0.2880	0.7146	0.6912	334.3265	260.2275	494.8650	381.6034
	42872.0042	42240.1745	1.0841	1.0798	124984.5723	120204.4051	0.7584	0.2880	0.7138	0.6911	334.1917	256.5397	494.8056	379.4380
	42871.5896	42136.6736	1.0829	1.0772	124981.2659	120090.9469	0.7584	0.2880	0.7133	0.6907	334.1842	255.3988	494.7458	378.7837
	42871.4296	42041.5760	1.0821	1.0759	124979.7506	118446.6267	0.7584	0.2880	0.7130	0.6903	334.1492	252.3305	494.7171	373.3755
	42870.9092	42000.2356	1.0812	1.0747	124978.5341	111578.2296	0.7578	0.2880	0.7126	0.6892	334.0760	249.2532	494.6616	369.2247
	42870.6630	41934.0698	1.0807	1.0702	124977.9124	111355.0424	0.7578	0.2880	0.7121	0.6884	334.0601	243.9649	494.6347	368.6373
	42867.1791	41635.6971	1.0795	1.0674	124964.3519	110351.6186	0.7578	0.2880	0.7121	0.6876	334.0141	243.1744	494.5831	364.3375
	42859.6049	41503.9476	1.0768	1.0611	124959.8377	110256.5113	0.7461	0.2880	0.7121	0.6842	334.0050	241.0774	494.5474	361.4942
	42858.2116	41479.6057	1.0746	1.0605	124950.1114	95709.7965	0.7461	0.2880	0.7111	0.6827	333.9971	235.3766	494.4941	353.6972
	42853.2824	40617.4194	1.0725	1.0592	124943.0481	89891.5783	0.7461	0.2880	0.7108	0.6773	333.9491	221.5874	494.2884	352.4184
	42845.1465	38400.6563	1.0718	1.0562	124913.3725	73378.8013	0.7461	0.2880	0.7092	0.6765	333.9080	202.3386	494.0993	335.0071
Worst	42780.3090	32070.8457	1.0703	1.0483	124894.8008	55886.0385	0.7451	0.2843	0.7077	0.6760	333.0167	128.6456	494.0689	327.5270
Average	42866.28	41691.94	1.09	1.08	124976.90	113901.14	0.84	0.42	0.71	0.69	334.31	268.17	495.75	385.90
STD	19.4845	2210.2745	0.0083	0.0133	26.4582	17546.1981	0.1202	0.2204	0.0027	0.0076	0.4015	44.3675	1.4275	32.6220

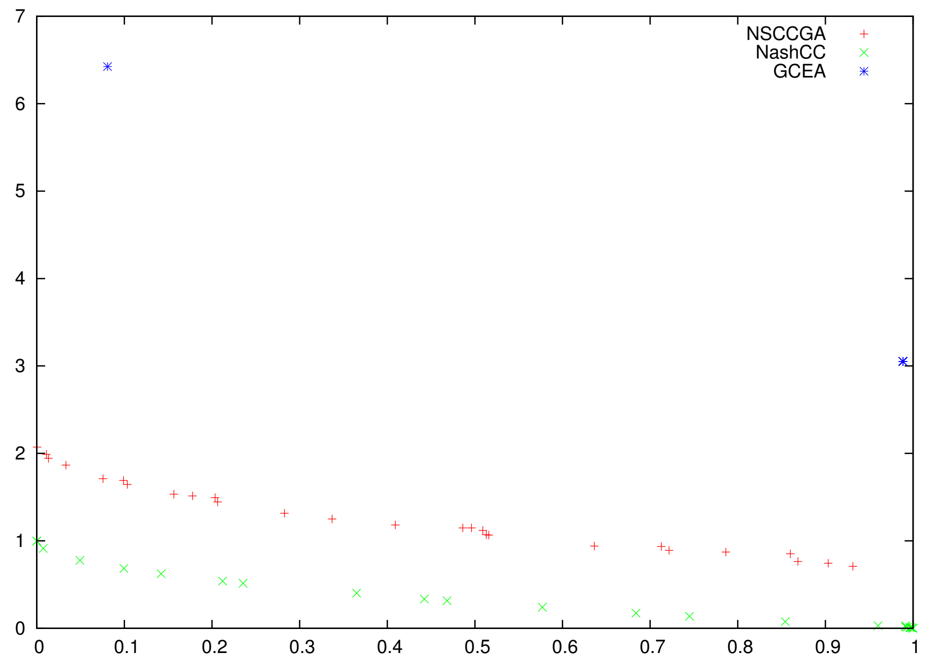

Figure 6.3: Pareto front approximations generated by NSCCGA, NashCC and GCEA of CCAs for ZDT1.

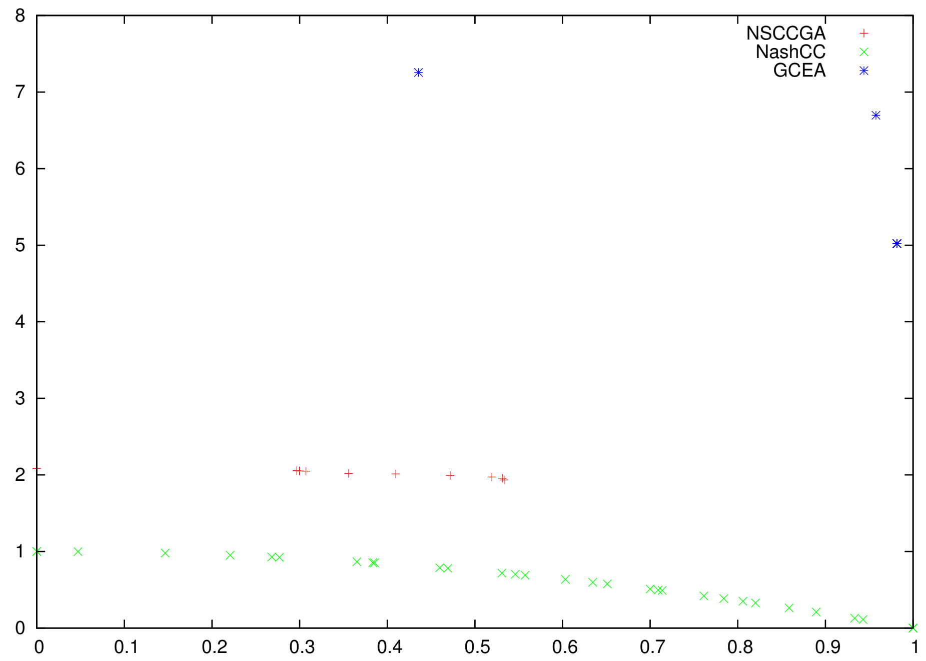

Figure 6.4: Pareto front approximations generated by NSCCGA, NashCC and GCEA for ZDT2.

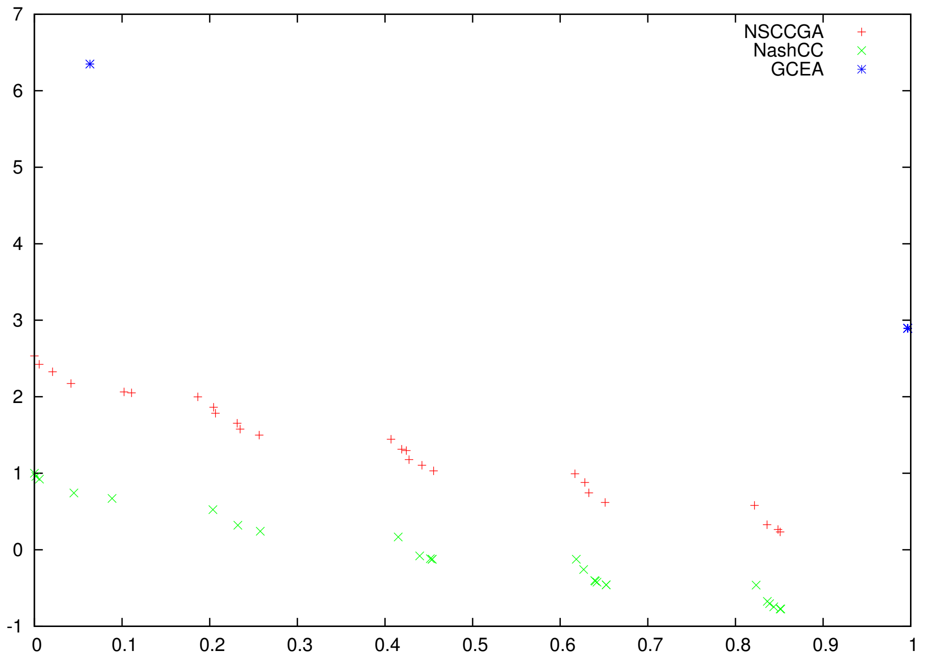

Figure 6.5: Pareto front approximations generated by NSCCGA, NashCC and GCEA for ZDT3.

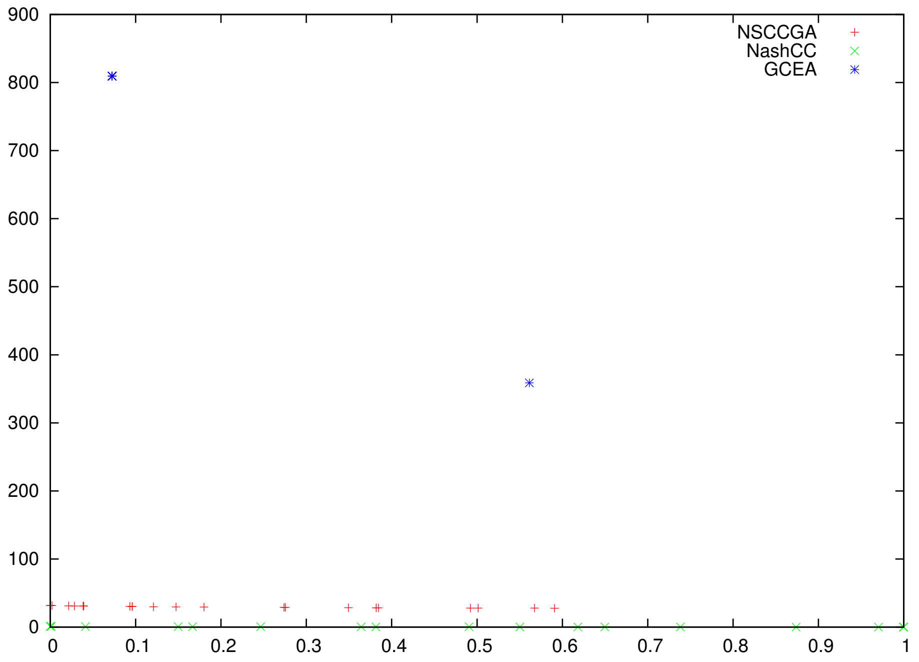

Figure 6.6: Pareto front approximations generated by NSCCGA, NashCC and GCEA for ZDT4.

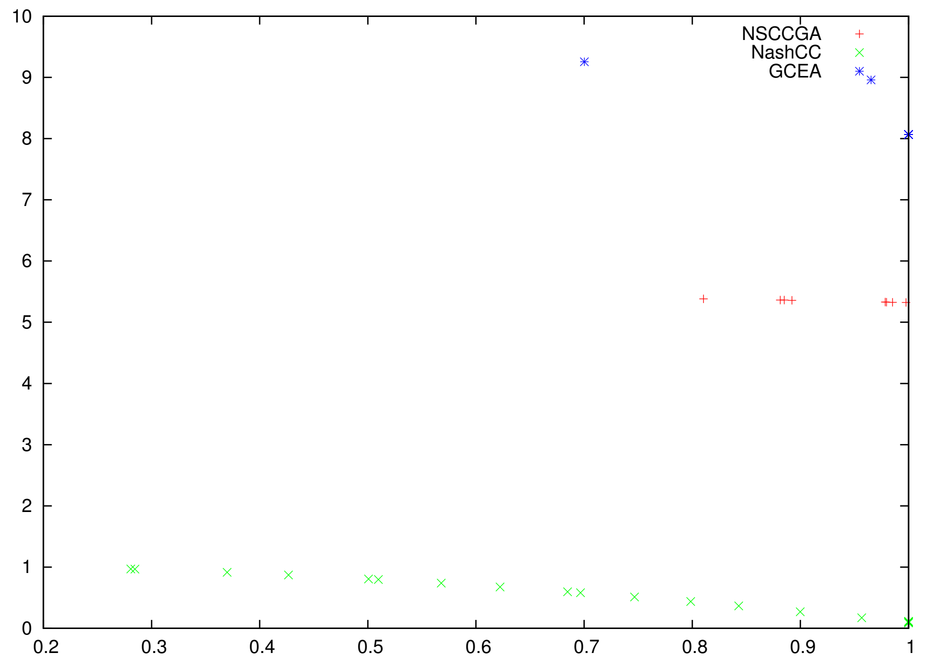

Figure 6.7: Pareto front approximations generated by NSCCGA, NashCC and GCEA for ZDT6.

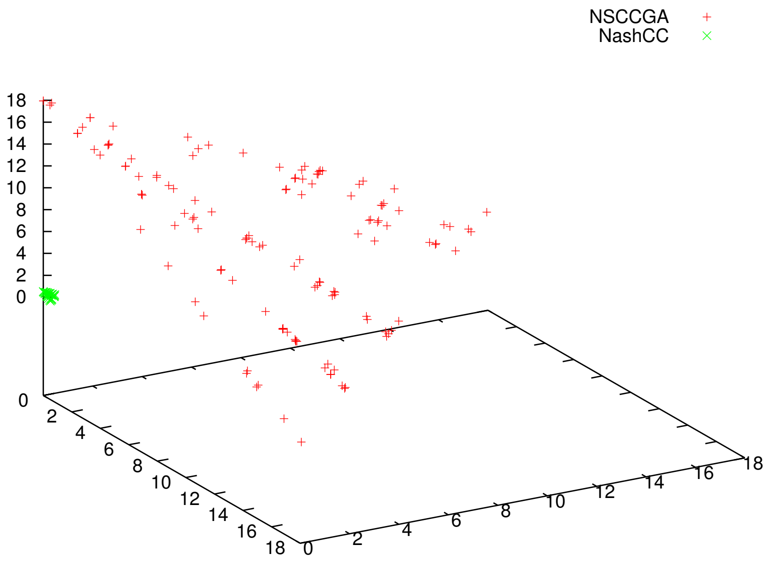

Figure 6.8: Pareto front approximations generated by NSCCGA and NashCC for DTLZ1.

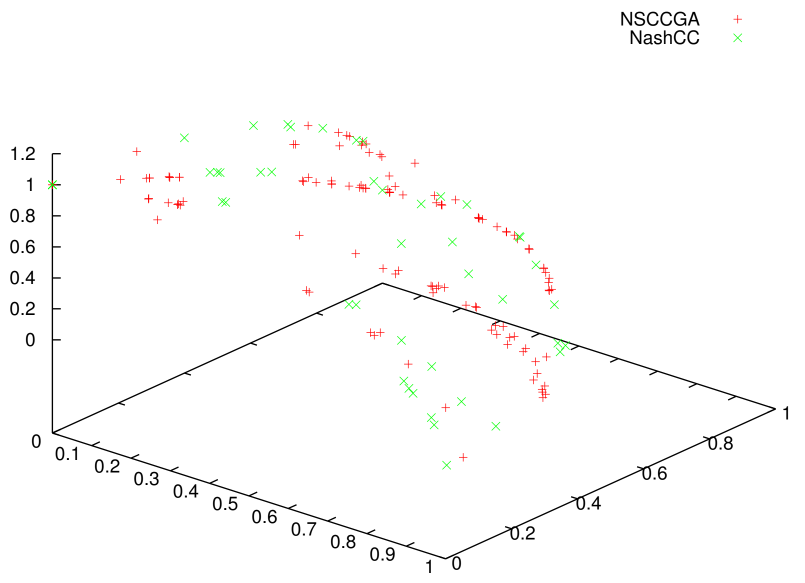

Figure 6.9: Pareto front approximations generated by NSCCGA and NashCC for DTLZ2.

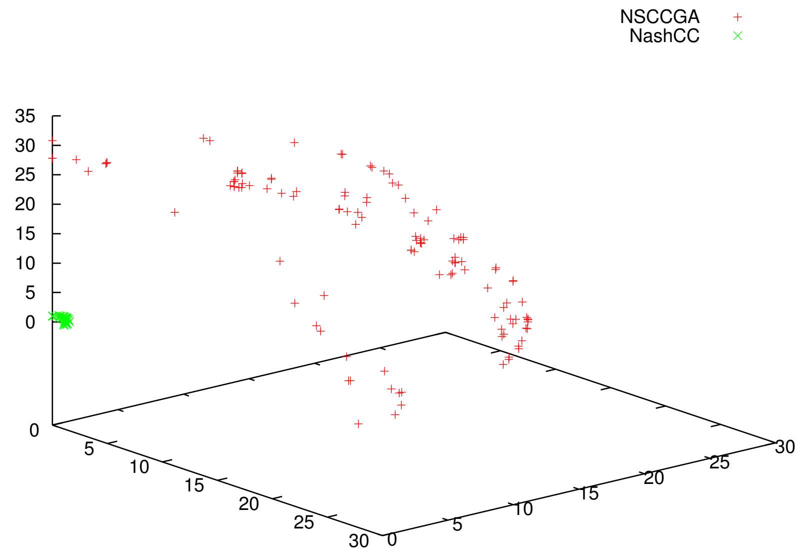

Figure 6.10: Pareto front approximations generated by NSCCGA and NashCC for DTLZ3.

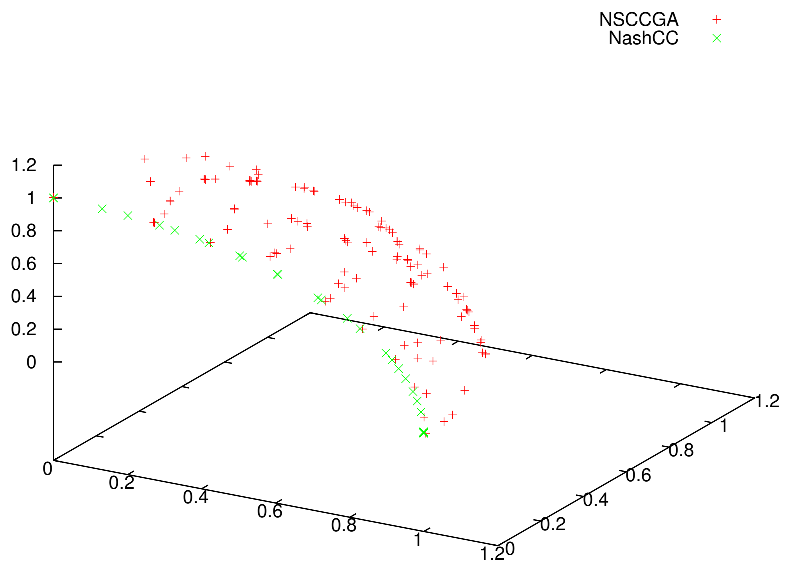

Figure 6.11: Pareto front approximations generated by NSCCGA and NashCC for DTLZ4.

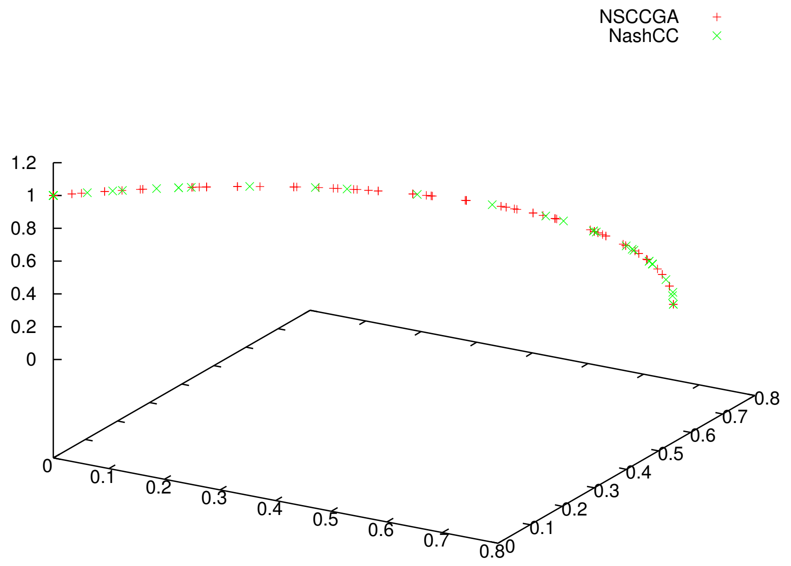

Figure 6.12: Pareto front approximations generated by NSCCGA and NashCC for DTLZ5.

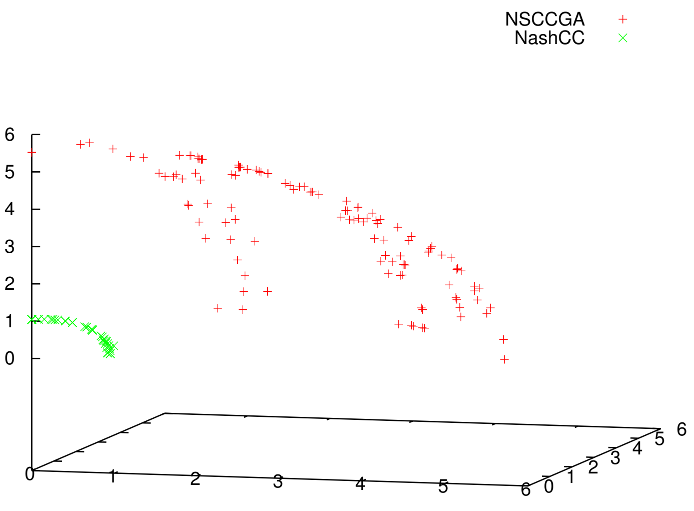

Figure 6.13: Pareto front approximations generated by NSCCGA and NashCC for DTLZ6.

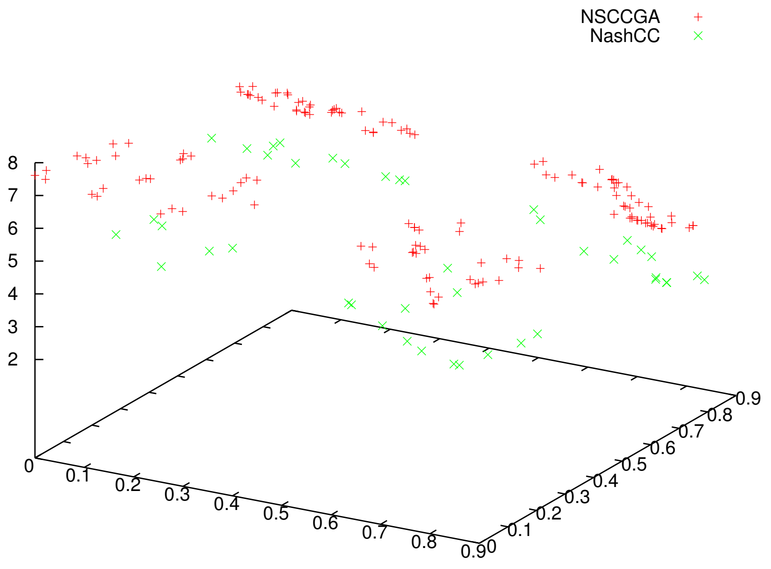

Figure 6.14: Pareto front approximations generated by NSCCGA and NashCC for DTLZ7.

6.2 Indicator Based Cooperative Coevolution for Multi-objective Optimization

Following the previous path. We developed another kind of cooperation scheme which, not only improves the performance of coevolution, but also allows to form more species than objectives functions in the MOPs being solved. As we have mention before, one of the main challenges when adapting cooperative coevolutionary framework to solve MOPs consists on how to select individuals from the other populations to assemble a complete solution. Previous work with the CCA on single objective problems has primarily selected the current *best* components from each species to merge into a collaboration, or performed a tournament with candidate solutions formed with the current *best* and randomly selected components. In a multi-objective scenario, there may be a number of individuals in each sub-population which are parts of overall solutions that are non-dominated in relation to each other, so the question is: how can we select one over the other?. The usual mechanism used by CCMOEAs is to select collaborators randomly from the set of non-domination solutions found so far in each especies' sub-population. However, this way of selecting individuals for collaboration make coevolving populations to be attracted to areas of the search space in which there are many strategies that perform well when combined with individuals from the other populations and this tends to generate individuals that have poor performance in the general context. So far, there is no mechanism which selects an individual to form a collaboration based on the contribution of the solution components and this would be a more suitable way to create collaborations in a multi-objective scenario. In this chapter we propose to make use of an indicator-based selection scheme to look for better collaborations into the CC framework. With this, we aim to get an improvement on the performance of the CC framework for multi-objective optimization.

6.3 Description of The Proposed Approach

Our approach to cooperative coevolution uses the first cooperative coevolution scheme proposed by Potter [73], where each population contains individuals that represent a particular component (in decision variables space) of the problem, so that one member from each population is needed in order to assemble a complete solution. Evaluation of an individual from a particular population is performed by joining the individual with collaborating partners from other populations. As mention before, the common way to perform the evaluation of each individual is by taking one representative from the other populations that belongs to the best set of non-dominated solutions found so far. However, there is a potential problem with this approach; there is no mechanism which awards a ranking based on the contribution of the solution components. Without such a mechanism, potentially good solutions are lost because species may participate with individuals which create poor new components in a candidate solution. Besides, this kind of interaction makes coevolution to only explore narrow regions of the collaboration space, which suggests that evolution is strongly attracted to certain regions of the search space. However, these regions do not necessarily correspond to (fitness-based) optimal solutions, and coevolution often converges to sub-optimal equilibria. Several recent studies have shown that Pareto-based multi-objective evolutionary algorithms (MOEAs) do not perform properly when dealing with problems having more than three objectives [68]. This has motivated the development of new selection schemes from which the use of quality assessment indicators has been the most promising choice [131]. The idea when using indicator-based selection is to maximize a quality assessment indicator that provides a good ordering among sets that represent Pareto approximations. We believe that taking this indicator-based approach into the selection mechanism for collaboration that CCMOEAs use, one can get an improvement on the performance of this sort of algorithms. So we use this idea to develop a new CCMOEA which uses a novel selection mechanism for collaborations, which, instead of looking for best non-dominated solutions, will look for collaborations which take into account the contribution of the solution components over a quality assessment indicator.

Such approach works as follows: at the beginning, it divides the vector of decision variables $\vec{x}$ of dimension $D \in \mathbb{N}$ into $S \in \mathbb{N}$ subcomponents, where S is equal to the number of objective functions in the problem. Each subcomponent is created from a random grouping of decision variables in order to increase the probability of grouping interacting variables in non-separable problems. At the same time, S subpopulations (species) are created, each one with NP individuals, and these S subpopulations are assigned their corresponding decision variables in a random way. This means that to each subpopulation, it corresponds a subcomponent from S which had been already created. Thus, every subpopulation will have a total of m decision variables. Once the subpopulations are created, the algorithm does a random initialization of all the individuals across all subpopulations. Aside from evaluation, the populations are evolved independently using independent MOEAs for each species. Then, the algorithm performs the *cycles* in which the evolution of each of the subpopulations is done for a given number of *generations*. This process continues until the stopping condition is reached, and at the end, the solutions that are globally non-dominated (i.e., with respect to all the subpopulations), constitute the outcome of the algorithm.

Collaboration among the subpopulations takes place in the next way: in the first generation, random collaborations are formed and evaluated, obtaining a random individual from each subpopulation and forming a complete set of solutions to be evaluated in their objective functions. Then, the results from the evaluation are assigned back to the individual under evaluation. After the first generation, the resulting child subpopulations Q_1 to Q_S will be evaluated by forming collaborations with individuals from the other species which have the best contribution to a quality assessment indicator.

Within the cooperative coevolutionary framework, we can use this idea to take, as a member for collaboration from each species, the individual which contributes the most on the hypervolume indicator. So, we propose to make collaborations by finding the best individuals according to the hypervolume indicator contribution values of the solutions found so far in each species. A summary of the way in which our approach works is presented in Algorithm 9.

As mention before, aside from evaluation, the populations are evolved independently using independent MOEAs for each species. For this sake, we adopt the use of differential evolution algorithm [132] operators, along some NSGA-II[75] techniques, as the multi-objective optimizer used by each species. The MOEA operates as follow: It starts with a population of random solutions, which becomes the current population. At each generation, an offspring population is created using the differential evolution operators; then, the current population for the next generation is updated using the solutions of both, the offspring and the parent populations. Before proceeding to the next generation, the size of the population is reduced using non-dominated sorting and a pruning technique aimed at diversity preservation, in a similar way as NSGA-II.

Algorithm 9 CCMOEA

```
Input: $NP$, $Cycles$, $Gmax$, $NumEsp$
Output: $SolutionSet$
  $Pobs \leftarrow Populations(NP, NumEsp)$
  $InitializeSpecies(Pobs)$
  for $j \leftarrow 1$ to $Cycles$ do
    for $i \leftarrow 1$ to $NumEsp$ do
      for $k \leftarrow 1$ to $Gmax$ do
        $MOEA(Pobs[i])$
      end for
    end for
  end for
  $SolutionSet \leftarrow ObtainNonDominatedSet(Pobs)$
  return $SolutionSet$
```

6.3.1 Experimental Studies

For the purposes of this study, we adopted the Deb-Thiele-Laumanns-Zitzler (DTLZ) test suite [3], with MOPs of three objective functions and 12 decision variables. The main characteristics of these problems are described next: DTLZ1 is a separable and multi-modal problem with a linear Pareto front. DTLZ2 is a separable and uni-modal problem with a concave Pareto front. DTLZ3 is a separable and multi-modal problem with a concave Pareto front. DTLZ4 is a separable and uni-modal problem with a concave Pareto front. DTLZ5 is a uni-modal problem with a degenerated arc Pareto front. DTLZ6 is an uni-modal problem with a degenerated arc front. Finally, DTLZ7 has a disconnected and mixed Pareto front.

6.3.2 Methodology

Since the main objective of this work is to study the impact selection mechanism for collaboration has in a CCMOEA and evaluate the performance of our approach, we will analyze our results with respect to those of the same new approach we just described, but using the usual selection mechanism for collaboration which takes one representative from the other populations that belongs to the set of non-dominated solutions found so far. We decided to called both versions as, Indicator-based CCMOEA (IBCCMOEA) and Pareto-based CCMOEA (PBCCMOEA). We established a predefined number of function evaluations that the algorithms can use, to analyze how they behave with the same resources. For measuring the results we adopted the hypervolume, but this time as a performance indicator [76]. The hypervolume is obtained by computing the volume (in objective function space) of the non-dominated set of solutions, given as the final result, of each CCMOEA. The aim of this study is to identify which of the CCMOEAs being compared is able to get closer to the true Pareto front using the same number of objective function evaluations.

6.3.3 Parameterization

The parameters of each CCMOEA used in our study were chosen in such a way that we could do a fair comparison among them. Since both CCMOEAs are of the same nature, we will use the same parameters for both approaches. We used 2 species for each problem with a populations size of 50 individuals for each species population. Each CCMOEA used 50 cycles and just one generation for each species per cycle for both approaches (which means each CCMOEA used 5000 function evaluations in the experiments). Finally, the F and CR values for differential evolution, used by the multi-objective species optimizer, were set to 0.5 and 0.5 respectively. All of this in order to use the same number of function evaluations in both CCMOEAs and to allow for a fair comparison of results.

6.3.4 Analysis of results

In our experiments, we obtained the hypervolume value over the 25 independents runs performed. Table 6.3 shows the average hypervolume of each of the CCMOEAs being compared for each test problem adopted, as well as the results of the statistical analysis that we made to validate our experiments, for which we used Wilcoxon's rank sum. To ease the analysis of the results in these tables, the cells containing the best hypervolume mean value for each problem have a grey colored background.

From Figures 6.23 to 6.29, we plot the results of the median of the 25 runs. We can observe that, using the same number of function evaluations, our proposed IBCCMOEA is able to get closer than PBCCMOEA to the Pareto front in all the problems. It is clear our approach has a better performance than the Pareto based approach, in terms of number of function evaluations and according to the Wilcoxon's rank sum test results, the null hypothesis ("medians are equal") can be rejected. Therefore, results confirm that the way selection is done for collaborations in CCMOEAs has an important effect. The new indicator-based approach presented here showed to very effective in improving CCMOEA performance.

Table 6.3: Average of the hypervolume indicator values of the results obtained for the DTLZ test problems. We show average (mean) values over 25 independent runs, as well as the standard deviation (SD) for each problem's results. The cells containing the best hypervolume value for each problem have a grey colored background. The P(H) columns show the results of the statistical analysis applied to our experiments using Wilcoxon's rank sum. P is the probability of observing the given result (the null hypothesis is true). Small values of P cast doubt on the validity of the null hypothesis. $H = 0$ indicates that the null hypothesis ("medians are equal") cannot be rejected at the 5% level. $H = 1$ indicates that the null hypothesis can be rejected at the 5% level.

Algorithm	IBCCMOEA		PBCCMOEA		IBCC-PBCC
	Mean	SD	Mean	SD	P(H)
Problem	DTLZ1				
	3.3136	0.0295	1.6261	0.5835	0.000000 (1)
	DTLZ2				
	2.5789	0.1207	2.4674	0.0367	0.000275 (1)
	DTLZ3				
	2.6911	0.0105	2.4665	0.0220	0.000000 (1)
	DTLZ4				
	2.6061	0.0984	2.4689	0.1282	0.000117 (1)
	DTLZ5				
	2.0235	0.0076	1.9940	0.0055	0.000000 (1)
	DTLZ6				
	2.0260	0.0109	1.9835	0.0069	0.000000 (1)
	DTLZ7				
	2.2282	0.0442	2.2078	0.0168	0.012087 (1)

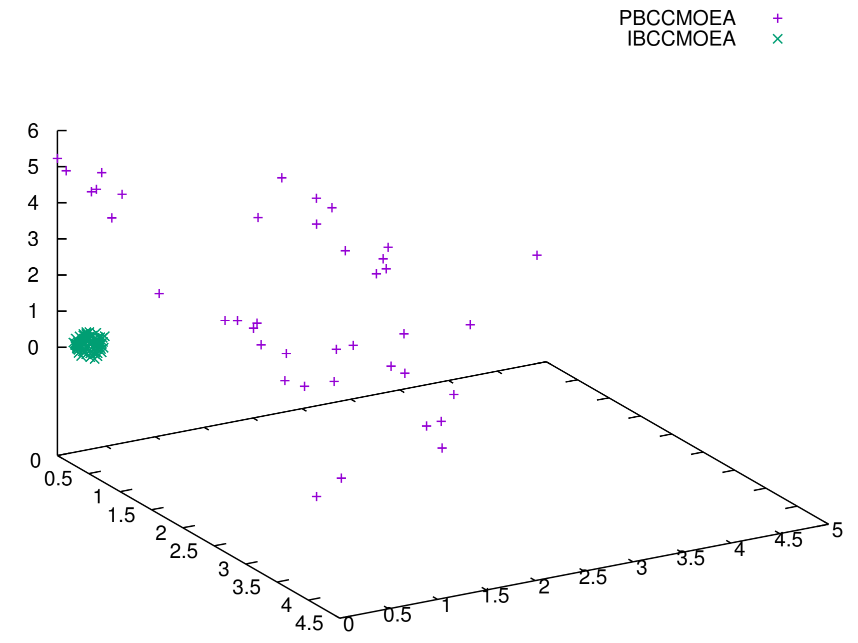

Figure 6.15: Plot of CCMOEAs for DTLZ1.

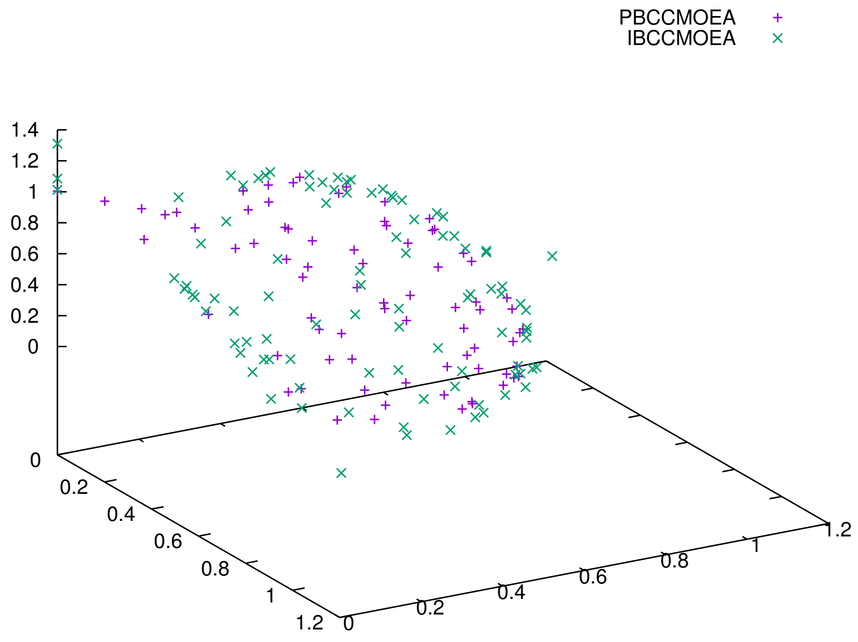

Figure 6.16: Plot of CCMOEAs for DTLZ2.

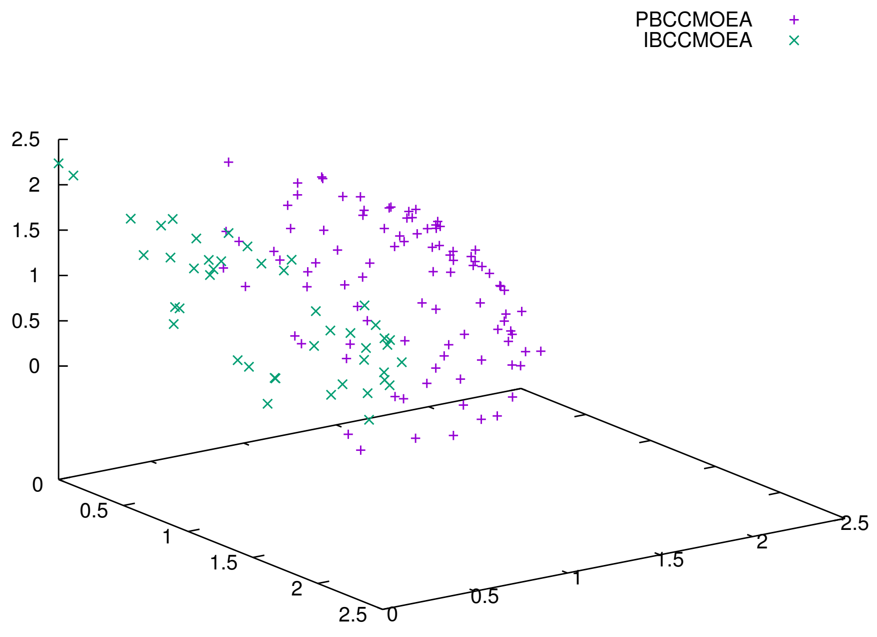

Figure 6.17: Plot of CCMOEAs for DTLZ3.

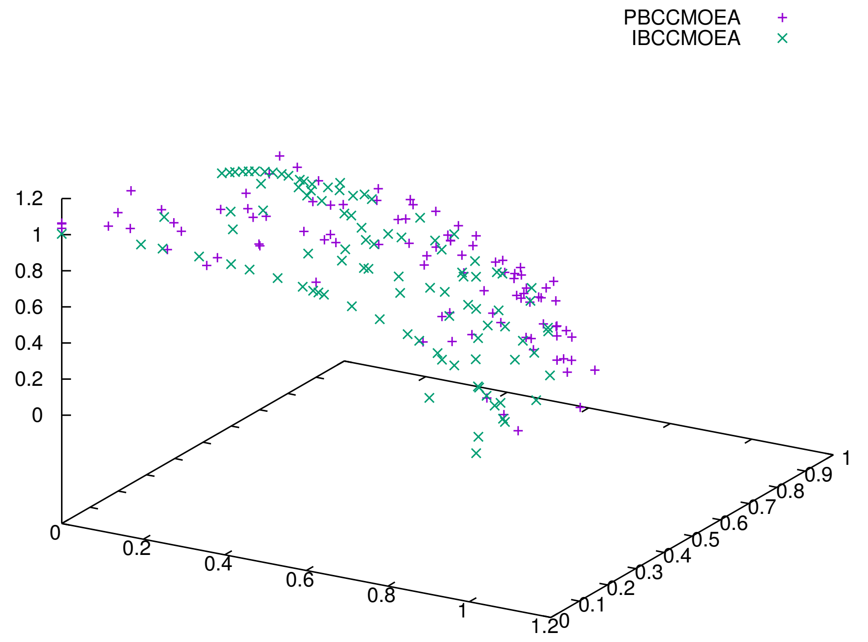

Figure 6.18: Plot of CCMOEAs for DTLZ4.

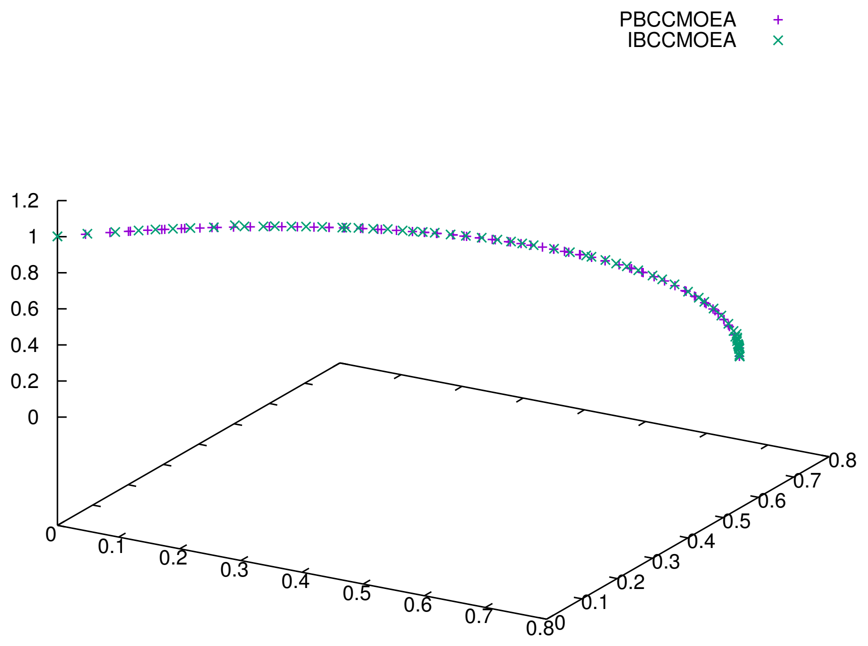

Figure 6.19: Plot of CCMOEAs for DTLZ5.

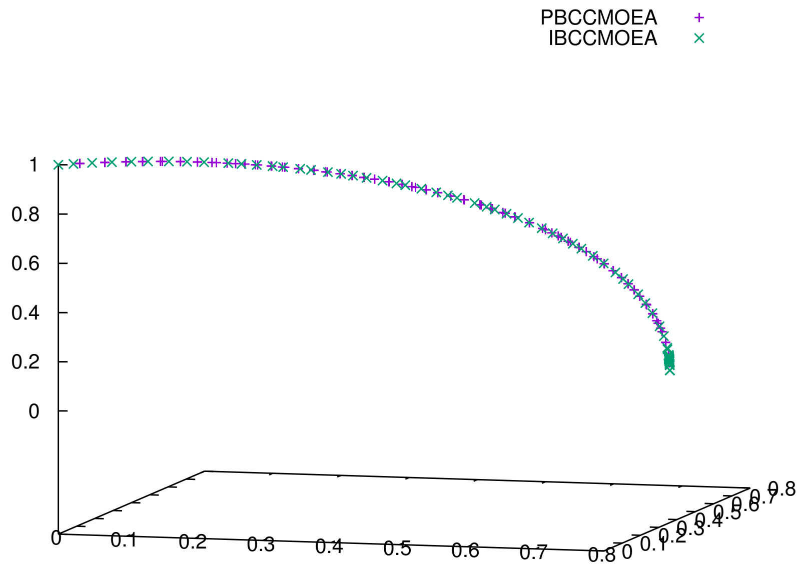

Figure 6.20: Plot of CCMOEAs for DTLZ6.

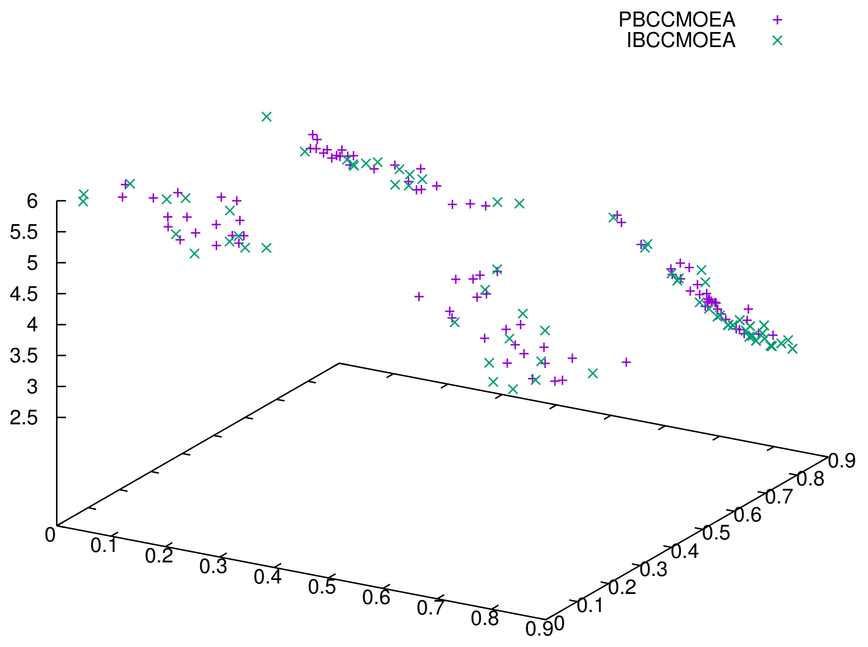

Figure 6.21: Plot of CCMOEAs for DTLZ7.

6.4 MOEA based on double decomposition

If we analyze the two previous frameworks, we can conclude that even though both approaches improve the performance of the cooperative coevolutionary framework based on Pareto optimal concept, these approaches make use of an unaffordable computational time when dealing with a large number of decision variables. This since, dealing with more decision variables implies to make use of more species, which at the same time means to make use of more collaborations between species and this in conclusion takes more time and computational resources. This drawback asks for another alternative to make use of the cooperative coevolutionary framework, which make us to look towards another kind of decomposition or transformation of the MOP. Decomposition, in terms of objective functions, is a well-established mathematical programming technique for dealing with multi-objective optimization problems (MOPs), which has been found to be extremely efficient and effective when coupled to evolutionary algorithms, as evidenced by MOEA/D. MOEA/D decomposes a MOP into several single-objective subproblems by means of well-defined scalarizing functions. It has been shown that MOEA/D is able to generate a set of evenly distributed solutions for different multi-objective benchmark functions with a relatively low number of decision variables. Next we study the effect of parameter scalability in MOEA/D and show how its efficacy decreases as the number of decision variables of the MOP increases. Thereafter, we investigate the improvements that MOEA/D can achieve when combined with cooperative coevolutionary techniques, giving rise to a novel MOEA which decomposes the MOP both in objective and in decision variables space.

The main idea of our proposed approach is to make use of the divide-and-conquer technique, used by the cooperative coevolutionary framework for large scale single objective optimization, and incorporate such concept into MOEA/D. Our motivation is that it is very natural to use scalar optimization methods in MOEA/D, since each solution is associated with a scalar optimization problem, in contrast with non-decomposition MOEAs where in most cases there is no easy way for them to take advantage of scalar optimization methods. Next, we give a brief description of both MOEA/D and cooperative coevolution.

6.4.1 MOEA/D

The *multi-objective evolutionary algorithm based on decomposition* (MOEA/D) [30] has gained growing interest from the community, due to its simplicity and to its effectiveness when applied to a broad range of MOPs. MOEA/D decomposes the MOP into a set of single-objective subproblems and solves these subproblems simultaneously using an evolutionary algorithm. Such decomposition is based on a set of weights each of which corresponds to a single subproblem. Each weight vector is used as a search direction to define a scalar function. For this sake, the so called Tchebycheff decomposition is the most widely used. Given a weight vector $\lambda = [\lambda_1, \ldots, \lambda_n]^T$ the corresponding subproblem is defined as:

$$\text{minimize} \quad g^{te}(x|\lambda, z^*) = \max_{1\leq i\leq n} \lambda_i |f_i(x) - z_i^*| \tag{6.3}$$

where z^* is the reference point chosen as the minimum of the objective function values found during the evolutionary process. The main advantage of the Tchebycheff approach is that it works regardless of the shape of the Pareto front, while other decomposition approaches (like the weighted sum approach) only work for convex Pareto fronts. The weights are also used to define neighborhoods of the subproblems. The neighborhood relations among these subproblems are defined based on the distances between their aggregation coefficient vectors. At each generation, a new individual is generated and evaluated using its own neighborhood of weights, with the idea that any information about these closest weight vectors should be helpful for optimizing the current individual's subproblem. Once this new individual is created, it is compared to its parent and in case the offspring is better, it replaces the parent. Moreover, it is also compared to other individuals in its neighborhood and is allowed to replace some of them. Therefore, at each generation, the population is composed of the best solution found so far (i.e., since the start of the run of the algorithm) for each subproblem.

6.4.2 Description of our proposed approach

If we are to extend the basic computational model of cooperative coevolution into an approach that already uses a decomposition strategy as the one adopted by MOEA/D, we must address the issues of a second problem decomposition, as well as other issues such as the interdependencies among subcomponents, credit assignment, and the maintenance of diversity. In order to do so and to provide reasonable opportunities for the success of coadapted subcomponents and an increase in efficiency when dealing with large scale MOPs, we can not use the whole model of cooperative coevolution, since it would be much more costly (due to the use of multiple subpopulations) than the use of MOEA/D as a standalone algorithm. Instead, we only incorporate into MOEA/D a coevolutionary step where we make use of the divide-and-conquer technique that splits the MOP to be solved, but in *decision variables* space.

Problem decomposition in the decision variables context consists in determining an appropriate number of subcomponents and the role that each of them will play in the overall search process. For some problems, an appropriate form of decomposition may be known a priori but for others, this may not be possible. Let's consider the problem of optimizing a function of m independent variables. It may be reasonable to decompose the problem into m subtasks, with each of them being assigned to the optimization of a single variable. However, there are many problems for which we have little or no information related to the number or roles of subcomponents that should be in the decomposition. This occurs in non-separable functions. Here, non-separable means that the vector of decision variables is composed by elements that interact with each other and are not independent. As a response to this problem, it has been found that dividing the problem into random groups provides better results than applying a deterministic division scheme, when dealing with non-separable functions [85, 86].

Motivated by this previous work, our proposed approach divides the vector of decision variables into S subcomponents (species), each one representing a subset of all the decision variables at a time rather than taking only one variable per subcomponent. We assign each decision variable to its correspondent subcomponent in a random way, trying to increase the chance of optimizing some interacting variables together. However, it is important to note that the cooperative coevolutionary adaptation presented here does not work as in the original framework, since we do not intend to use several subpopulations for each subcomponent of the problem and we will not need individuals from the other species to assemble a complete solution in order to perform a fitness evaluation. Here, we only use decision variable decomposition to make operations (crossover and mutation) more effective and with this, we can manage in a better way the *curse of dimensionality* (the performance of an evolutionary algorithm deteriorates rapidly as the dimensionality of the search

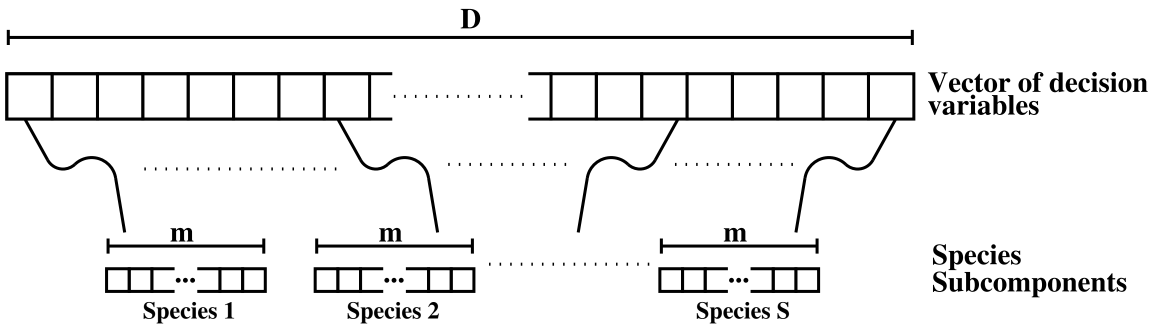

Figure 6.22: Graphical representation of the subcomponents (species) creation. Here, we assume a vector of decision variables of dimension D which is divided into S subcomponents of dimension m, created in a random way from the original vector of decision variables and assigned to the S existing species, where $D = m * S$.

space increases [133]) present in MOEAs. So, individuals will still be representing a whole solution, but operators will be applied based on the corresponding species, and not based on the individuals. The algorithm of our proposed MOEA based on double decomposition (MOEA/D^2) works as follows:

Input:

- The MOP
- N: The number of subproblems considered in MOEA/D
- S: The number of species for decision variables decomposition
- A set of N uniform spread weight vectors:
  $\lambda^1, \dots, \lambda^N$
- T: The neighborhood size

Output:

- PS: the final solutions found during the search

Step 1) Initialization:

Step 1.1) Set the external population of final solutions $PS = \emptyset$.

Step 1.2) Find the T closest weight vectors to each weight vector. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$, where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i .

Step 1.3) Generate an initial population $x^1, \ldots x^N$ randomly or by a problem-specific method. Set $FV^i = f(x^i)$.

Step 1.4) Divide the problem into S subcomponents $c^1, \ldots, c^S$ each one of dimension m, created in a random way from the original vector of decision variables x of dimension D (as shown in Figure 6.22), where $D = m * S$, such that, for each $j = 1, \ldots, N$, $x^j = [c^1_j, \ldots, c^S_j]$.

Step 1.5) Initialize $z = [z_1, \ldots, z_k]^T$, where z_i is the best value found so far for objective f_i.

Step 2) Update:

For $i = 1, \ldots, N$ do

Step 2.1) Crossover and Mutation:

For $j = 1, \ldots, S$ do

Step 2.1.1) Randomly select two indexes p, q from $B(i)$, and then generate a new solution y^j_c from c^j_p and c^j_q using crossover.

Step 2.1.2) Apply a problem-specific repair improvement heuristic on y^j_c to produce y'^j_c.

Step 2.2) Assemble y' from $[y'^1_c, \ldots, y'^S_c]$, sorting the subcomponents to form the original vector of decision variables.

Step 2.3) For each $j = 1, \ldots, k$, if $z_j > f_j(y')$, then set $z_j = f_j(y')$.

Step 2.4) Update of Neighboring Solutions: For each index $j \in B(i)$, if $g^{te}(y'|\lambda^j, z^*) < g^{te}(x^j|\lambda^j, z^*)$, then $FV^j = f(y')$.

Step 2.5) Remove from the external population PS all the vectors dominated by $f(y')$. Add $f(y')$ to PS if no vectors in PS dominate it.

Step 3) Stopping Criterion: Stop if the termination criterion is satisfied. Otherwise, go to Step 2.

Since c_p^j and c_q^j in Step 2.1.1 are the current best subcomponent (in decision variables space) solutions to neighbors of the i^{th} subproblem (in objective function space) and their dimensions are less than the original vector of decision variables x, their offspring y'^j_c (already improved by mutation) should be a good contribution to the complete assemble of the new final solution y'. Therefore, the resultant solution is very likely to have a lower (improved) function value for the neighbors of the i^{th} subproblem. Also, by using only the decomposition nature of the cooperative coevolutionary framework, there is no need for extra function evaluations. Therefore, the efficiency of MOEA/D is not lost.

6.4.3 Experimental Results

We validated MOEA/D^2 comparing its performance with respect to that of the original MOEA/D and with respect to GDE3 [74]. Although GDE3 is not a decomposition based algorithm, in the studies of parameter scalability presented in [6] this differential evolution based MOEA obtained the best overall results, which is the reason why we decided to include it in our comparative study.

GDE3

GDE3 is the third version of the so-called Generalized Differential Evolution (GDE) algorithm [134], which is able to deal with multiple objectives. It starts with a population of random solutions, which becomes the current population. At each generation, an offspring population is created using the differential evolution operators; then, the current population for the next generation is updated using the solutions of both, the offspring and the parent populations. Before proceeding to the next generation, the size of the population is reduced using non-dominated sorting and a pruning technique aimed at diversity preservation, in a similar way as NSGA-II, although the pruning used in GDE3 modifies the crowding distance of NSGA-II in order to solve some of its drawbacks when dealing with problems having more than two objectives.

Methodology

For the purposes of this study, we adopted the Deb-Thiele-Laumanns-Zitzler (DTLZ) test suite [135] with instances of three objectives with a number of decision variables that ranges from 200 to 1200. In order to assess the performance of each approach, we selected the hypervolume indicator [70], since this measure can differentiate between degrees of complete outperformance of two sets.

The aim of this study is to identify which of the algorithms being compared is able to get closer to the true Pareto front using the same number of objective function evaluations and how they behave as the dimensionality of the MOP increases.

6.4.4 Parameterization

The parameters of each algorithm used in our study were chosen in such a way that we could do a fair comparison among them. For MOEA/D^2 and MOEA/D, we adopted SBX and polynomial-based mutation [75] as the crossover and mutation operators, respectively. The mutation probability was set to $p_m = 1/l$, where l is the number of decision variables; the distribution indexes for SBX and the polynomial-based mutation were set as: $\eta_c = 20$ and $\eta_m = 20$. For the case of MOEA/D^2, different numbers of species were used for each problem instance, in order to have 2 decision variables per species. So, for problems with 200 decision variables, 100 species were used, for problems with 400 decision variables 200 species were used, and so on. The maximum number of iterations adopted for all problems and MOEAs was set to 1000, regardless of their dimensionality. The F and CR values for GDE3 were set to 0.5. Finally, the population size for all algorithms in all problems instances was set to 100.

6.4.5 Discussion of Results

In our experiments, we obtained the hypervolume value over the 25 independents runs performed. Table 6.3 shows the average hypervolume of each of the MOEAs being compared for each test problem adopted, as well as the results of the statistical analysis that we conducted to validate our experiments, for which we used Wilcoxon's rank sum. Also, we show the improvement on the hypervolume value that our approach was able to obtain against the other algorithms. From Figures 6.23 to 6.29, we plotted the results of the median of the 25 runs for MOEA/D^2 and MOEA/D for the DTLZ test problems with 1200 decision variables. GDE3 presented the poorest performance in all problem instances. MOEA/D produced competitive results for DTLZ2 and DTLZ4, although it could not outperform our approach in any problem instance. According to Wilcoxon's test, we cannot reject the null hypothesis in only two cases when comparing our approach to MOEA/D, DTLZ2 and DTLZ4 with 200 decision variables, which means that in these cases both algorithms have a similar behavior. Respect to all other cases, our approach outperfomed MOEA/D and GDE3, and as the results show, as the dimensionality of the problems grows, the improvement obtained by our approach on the hypervolume value increases. Based on the results of Wilcoxon's test, we can confirm that the null hypothesis can be rejected, so MOEA/D^2 yields the best overall results. Also, from Figures 6.23 to 6.29 we can observe that, using the same number of function evaluations, MOEA/D^2 is able to get closer than MOEA/D to the true Pareto front in all problems.

6.5 MOEA/D^2 against cooperative coevolution and MOEADVA

Next, we validate MOEA/D^2 against two MOEAs created to deal with large scale MOPs, an improved version of CCDE3 described in Section 4.3 of this thesis (which changes differential evolution operators by SBX and PBM operators) and against the

		MOEA/D^2	MOEA/D	MOEA/D^2-MOEA/D	MOEA/D^2-MOEA/D	GDE3	MOEAD^2-GDE3	MOEAD^2-GDE3
Function	No. Vars	HV	HV	Improvement	P(H)	HV	Improvement	P(H)
DTLZ1	200	124999991998953.0000	124999970289543.0000	21709410.2344	0.000000 (1)	124923656113375.0000	76335885578.2031	0.000000 (1)
	400	124999755014274.0000	124999298073533.0000	456940740.9063	0.000000 (1)	124181465622337.0000	818289391936.5000	0.000000 (1)
	600	124998478272908.0000	124996030413092.0000	2447859816.0625	0.000000 (1)	122039040800943.0000	2959437471964.5000	0.000000 (1)
	800	124995060011719.0000	124985536236730.0000	9523774988.8594	0.000000 (1)	117702084444497.0000	7292975567222.5300	0.000000 (1)
	1000	124986970597954.0000	124955356063479.0000	31614534475.2500	0.000000 (1)	110256296271387.0000	14730674326567.2000	0.000000 (1)
	1200	124970550659900.0000	124894718579624.0000	75832080276.4531	0.000000 (1)	99191612716078.9000	25778937943821.2000	0.000000 (1)
DTLZ2	200	728999.3904	728999.3862	0.0043	0.712386 (0)	728989.2937	10.0967	0.000000 (1)
	400	728999.3808	728999.3680	0.0128	0.043602 (1)	728321.7458	677.6350	0.000000 (1)
	600	728999.3605	728999.3085	0.0520	0.000000 (1)	721831.2296	7168.1309	0.000000 (1)
	800	728999.2870	728999.1289	0.1581	0.000000 (1)	698874.5807	30124.7063	0.000000 (1)
	1000	728999.0954	728998.4267	0.6687	0.000000 (1)	653788.8012	75210.2941	0.000000 (1)
	1200	728998.4393	728994.9746	3.4647	0.000000 (1)	571671.4472	157326.9921	0.000000 (1)
DTLZ3	200	1727999970755560.0000	1727999849624200.0000	121131355.7500	0.000000 (1)	1727222040269000.0000	777930486563.2500	0.000000 (1)
	400	1727996400439630.0000	1727991944817710.0000	4455621923.0000	0.000000 (1)	1716392835963730.0000	11603564475898.3000	0.000000 (1)
	600	1727970985655110.0000	1727945403715830.0000	25581939288.2500	0.000000 (1)	1679379508439150.0000	48591477215964.8000	0.000000 (1)
	800	1727890440027340.0000	1727805158813020.0000	85281214323.2500	0.000000 (1)	1597662758376130.0000	130227681651214.0000	0.000000 (1)
	1000	1727715593620590.0000	1727460212199730.0000	255381420857.2500	0.000000 (1)	1463152563598520.0000	264563030022069.0000	0.000000 (1)
	1200	1727363193497010.0000	1726773363259150.0000	589830237867.0000	0.000000 (1)	1259639566256750.0000	467723627240265.0000	0.000000 (1)
DTLZ4	200	728999.4140	728999.4078	0.0062	0.277231 (0)	728991.3901	8.0240	0.000000 (1)
	400	728999.4065	728999.3896	0.1154	0.000000 (1)	728201.9349	434.6057	0.000000 (1)
	600	728999.3788	728999.3464	0.0324	0.000000 (1)	720704.2965	8295.0824	0.000000 (1)
	800	728999.3150	728999.1945	0.1206	0.000000 (1)	696046.7161	32952.5989	0.000000 (1)
	1000	728999.1477	728998.6192	0.5285	0.000000 (1)	644641.3868	84357.7609	0.000000 (1)
	1200	728998.5780	728994.9171	3.6609	0.000000 (1)	559363.2154	169635.3626	0.000000 (1)

Table 6.4: Average of the hypervolume indicator values of the results obtained for the DTLZ1, DTLZ2, DTLZ3 and DTLZ4 test problems. We show average values over 25 independent runs. The cells containing the best hypervolume value for each problem have a grey colored background. The improvement columns show the improvement on the hypervolume value that our approach was able to get against the other algorithms. The P(H) columns shows the results of the statistical analysis applied to our experiments using Wilcoxon's rank sum. P is the probability of observing the given result (the null hypothesis is true). Small values of P cast doubt on the validity of the null hypothesis. $H = 0$ indicates that the null hypothesis ("medians are equal") cannot be rejected at the 5% level. $H = 1$ indicates that the null hypothesis can be rejected at the 5% level.

		MOEA/D^2	MOEA/D	MOEA/D^2-MOEA/D	MOEA/D^2-MOEA/D	GDE3	MOEAD^2-GDE3	MOEAD^2-GDE3
Function	No. Vars	HV	HV	Improvement	P(H)	HV	Improvement	P(H)
DTLZ5	200	1727866.0538	1727865.9384	0.1154	0.013007 (1)	1727431.4481	434.6057	0.000000 (1)
	400	1727865.5061	1727864.6278	0.8784	0.000000 (1)	1721336.0150	6529.4911	0.000000 (1)
	600	1727863.4153	1727859.1967	4.2187	0.000000 (1)	1697620.6971	30242.7183	0.000000 (1)
	800	1727857.2346	1727840.7092	16.5255	0.000000 (1)	1635056.7976	92800.4370	0.000000 (1)
	1000	1727837.7148	1727760.3047	77.4101	0.000000 (1)	1510727.5139	217110.2010	0.000000 (1)
	1200	1727773.5677	1727524.1382	249.4296	0.000000 (1)	1313095.5122	414678.0555	0.000000 (1)
DTLZ6	200	999967922.4861	999899450.4162	68472.0699	0.000000 (1)	999330750.1795	637172.3066	0.000000 (1)
	400	998891441.7157	996820925.0570	2070516.6587	0.000000 (1)	987305237.5147	11586204.2010	0.000000 (1)
	600	990768252.6193	981381295.3432	9386957.2761	0.000000 (1)	948509739.2585	42258513.3608	0.000000 (1)
	800	964064335.0353	937687686.8457	26376648.1896	0.000000 (1)	874883514.5826	89180820.4527	0.000000 (1)
	1000	906131328.8124	855654049.3429	50477279.4695	0.000000 (1)	744576613.9787	161554714.8337	0.000000 (1)
	1200	803763312.2166	712087777.8538	91675534.3628	0.000000 (1)	551828946.6234	251934365.5932	0.000000 (1)
DTLZ7	200	2203.4849	2203.4656	0.0193	0.000000 (1)	2055.0598	148.4252	0.000000 (1)
	400	2193.4627	2192.7324	0.7303	0.000000 (1)	1699.4438	494.0190	0.000000 (1)
	600	2090.3379	2067.9036	22.4343	0.000413 (1)	1338.1314	752.2065	0.000000 (1)
	800	1842.2642	1815.7768	26.4875	0.013007 (1)	1059.6383	782.6260	0.000000 (1)
	1000	1605.8489	1526.3840	79.4649	0.000001 (1)	855.6279	750.2210	0.000000 (1)
	1200	1398.8865	1352.2878	46.5987	0.006223 (1)	718.3771	680.5094	0.000000 (1)

Table 6.5: Average of the hypervolume indicator values of the results obtained for the DTLZ5, DTLZ6 and DTLZ7 test problems. We show average values over 25 independent runs. The cells containing the best hypervolume value for each problem have a grey colored background. The improvement columns show the improvement on the hypervolume value that our approach was able to get against the other algorithms. The P(H) columns shows the results of the statistical analysis applied to our experiments using Wilcoxon's rank sum. P is the probability of observing the given result (the null hypothesis is true). Small values of P cast doubt on the validity of the null hypothesis. $H = 0$ indicates that the null hypothesis ("medians are equal") cannot be rejected at the 5% level. $H = 1$ indicates that the null hypothesis can be rejected at the 5% level.

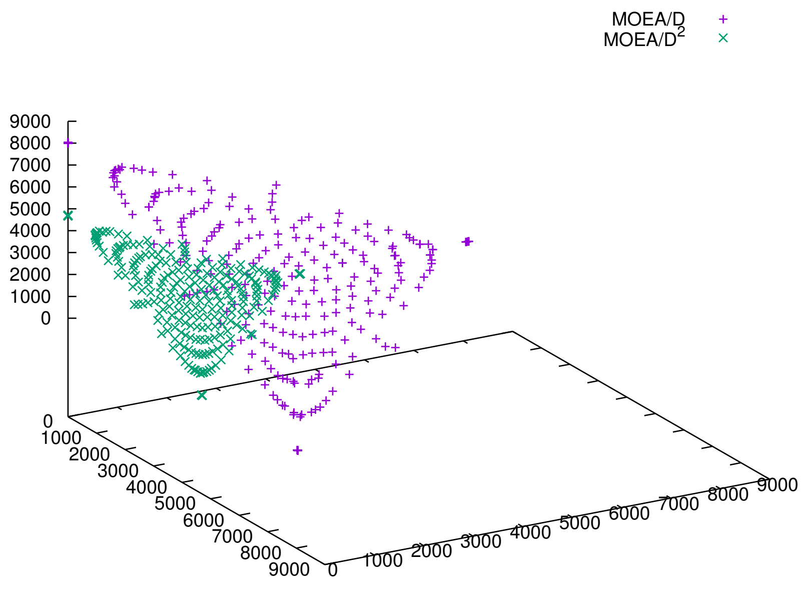

Figure 6.23: Pareto front approximations obtained by MOEA/D and MOEA/D^2 for DTLZ1 with 1200 decision variables.

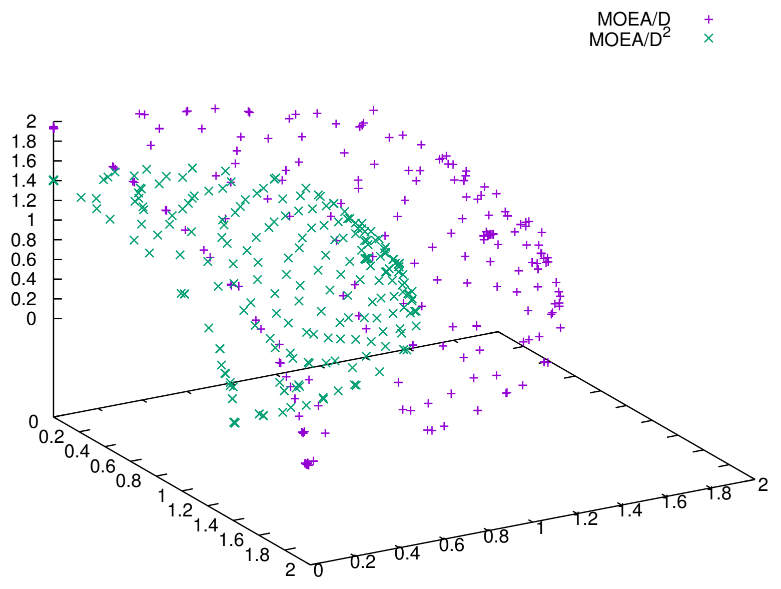

Figure 6.24: Pareto front approximations obtained by MOEA/D and MOEA/D^2 for DTLZ2 with 1200 decision variables.

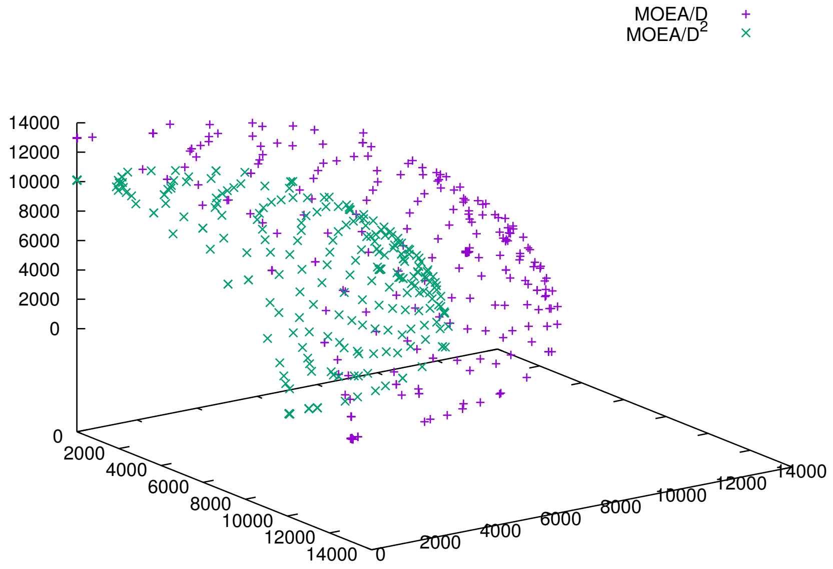

Figure 6.25: Pareto front approximations obtained by MOEA/D and MOEA/D^2 for DTLZ3 with 1200 decision variables.

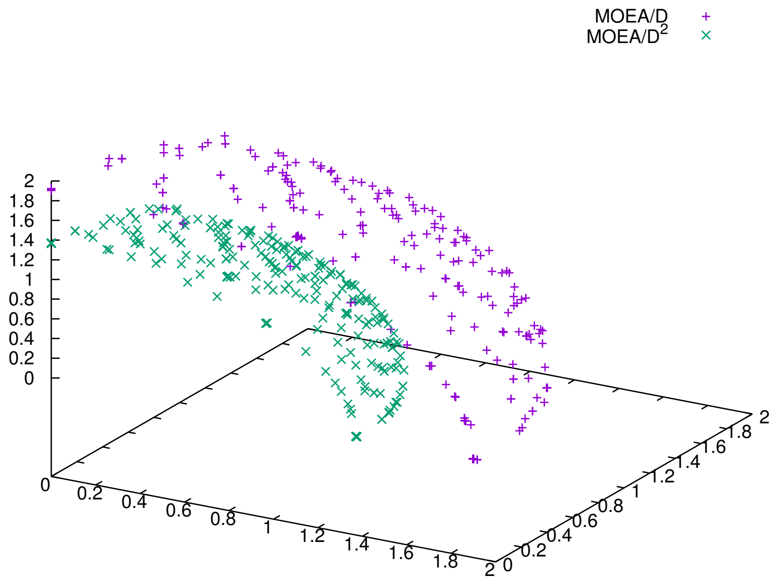

Figure 6.26: Pareto front approximations obtained by MOEA/D and MOEA/D^2 for DTLZ4 with 1200 decision variables.

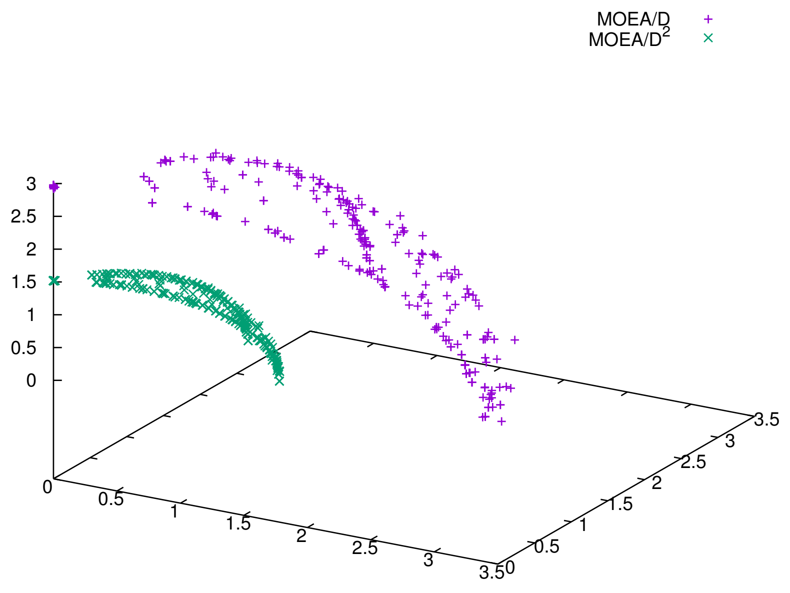

Figure 6.27: Pareto front approximations obtained by MOEA/D and MOEA/D^2 for DTLZ5 with 1200 decision variables.

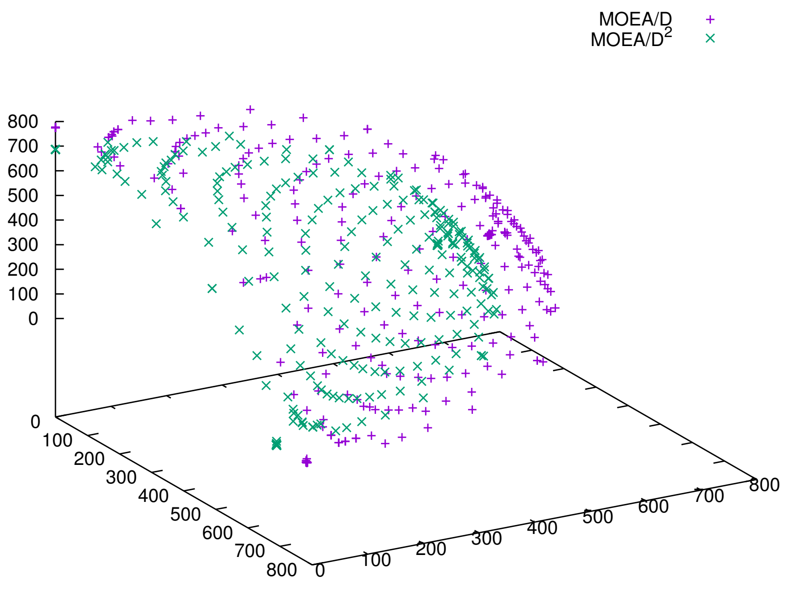

Figure 6.28: Pareto front approximations obtained by MOEA/D and MOEA/D^2 for DTLZ6 with 1200 decision variables.

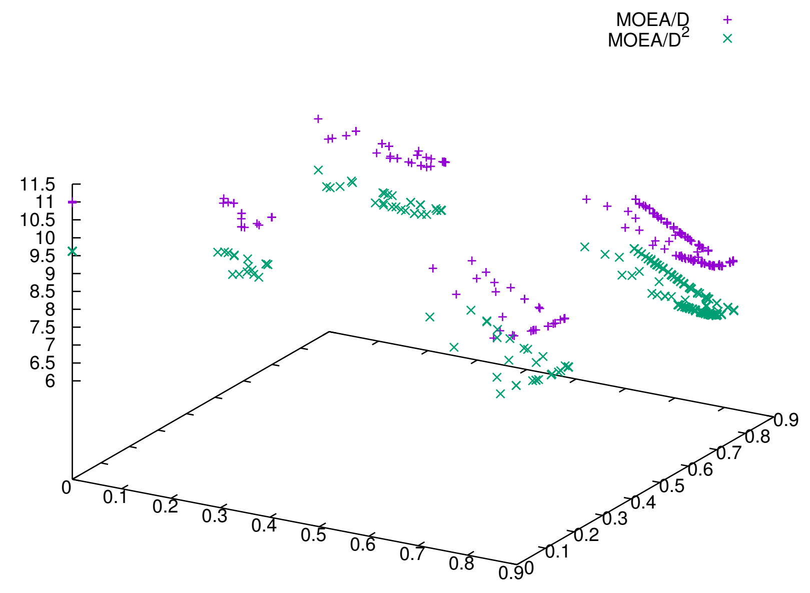

Figure 6.29: Pareto front approximations obtained by MOEA/D and MOEA/D^2 for DTLZ7 with 1200 decision variables.

MOEADVA, described in Section 4.4.

Methodology

For the purposes of this study, we adopted the Zitzler-Deb-Thiele (ZDT) [2], the Deb-Thiele-Laumanns-Zitzler (DTLZ) [3] and the Walking-Fish-Group (WFG) [4] test suites (see Appendix A for further details about the adopted problems), with instances of two objectives for the ZDT problems and three objectives for the DTLZ and WFG test problems with a number of decision variables that ranges from 200 to 1000. In order to assess the performance of each approach, we again make use of the hypervolume indicator [70], since this measure can differentiate between degrees of complete outperformance of two sets.

The aim of this study is to identify which of the algorithms being compared is able to get closer to the true Pareto front using the same number of objective function evaluations and how they behave as the dimensionality of the MOP increases.

6.5.1 Parameterization

The parameters of each algorithm used in our study were chosen in such a way that we could do a fair comparison among them. For MOEA/D^2 and the Cooperative Coevolutionary algorithm (CCMOEA), we adopted SBX and polynomial-based mutation [75] as the crossover and mutation operators, respectively. The mutation probability was set to $p_m = 1/l$, where l is the number of decision variables; the distribution indexes for SBX and the polynomial-based mutation were set as: $\eta_c = 20$ and $\eta_m = 20$. For the case of our approach[1], different numbers of species were used for each problem instance, in order to have 2 decision variables per species. So, for problems with 200 decision variables, 100 species were used, for problems with 400 decision variables 200 species were used, and so on. For the case of CCMOEA, we adopt a number of species which let us create species of 10 decision variables, so for problems with 200 decision variables, 20 species were used, for problems with 400 decision variables 40 species were used, and so on. In all cases a population size of 52 individual per species was adopted and one generation per cycle was use in order to allow the maximal number of collaborations between species. For the case of MOEADVA and MOEAD, we use a population size of 100 individuals for problems with 2 objectives functions and 120 individuals for problems with 3 objectives. The maximum number of function evaluations for each dimensionality in shown in Table 6.6.

6.5.2 Discussion of Results

In our experiments, we obtained the hypervolume value over the 25 independents runs performed. Tables 6.7 to **??** shows the average hypervolume of each of the MOEAs being compared for each test problem adopted, as well as the results of the statistical analysis that we conducted to validate our experiments, for which we used Wilcoxon's rank sum. Also, we show the improvement on the hypervolume

[1]To have a better understanding of the parameters of MOEA/D^2, a sensitivity analysis of parameters is presented in Appendix B.

No. Vars	FEs
200	400000
400	1500000
600	3300000
800	5800000
1000	9100000

Table 6.6: Number of function evaluations adopted for each problem dimension.

value that our approach was able to obtain against the other algorithms. When a cero values is presented, the algorithm could not obtained any solution which dominates the adopted reference points[2] for the hypervolume computation. From Figures 6.30 to 6.50, we plotted the results of the median of the 25 runs for MOEA/D^2 and MOEA/D for the DTLZ test problems with 200 decision variables. CCMOEA presented the poorest performance in all problem instances, although it was very competitive against MOEADVA. According to Wilcoxon's test, we cannot reject the null hypothesis in most ZDT problems, which means that in these cases the three approaches have a similar behavior. Respect to all other cases, our approach outperfomed MOEADVA and CCMOEA, and as the results show, as the dimensionality of the problems grows, the improvement obtained by our approach on the hypervolume value increases. Based on the results of Wilcoxon's test, we can confirm that the null hypothesis can be rejected, so MOEA/D^2 yields the best overall results. Also, from Figures 6.30 to 6.50, we can observe that, using the same number of function evaluations, MOEA/D^2 is able to get closer than MOEADVA to the true Pareto front in all problems. We did not include CCMOEA in these Figures since in some cases it it so far from the others that makes it difficult to appreciate the results of the other two approaches. Tables 6.12 to 6.14 shows the computational time of each algorithm in minutes, as can be observed the one that takes more time is MOEA/D^2, this since the nature of the improvement of the crossover technique it

[2]The adopted reference points were $[1.2, 16.1]$ and [200.1200.1200.1] for two and three objective problems respectively

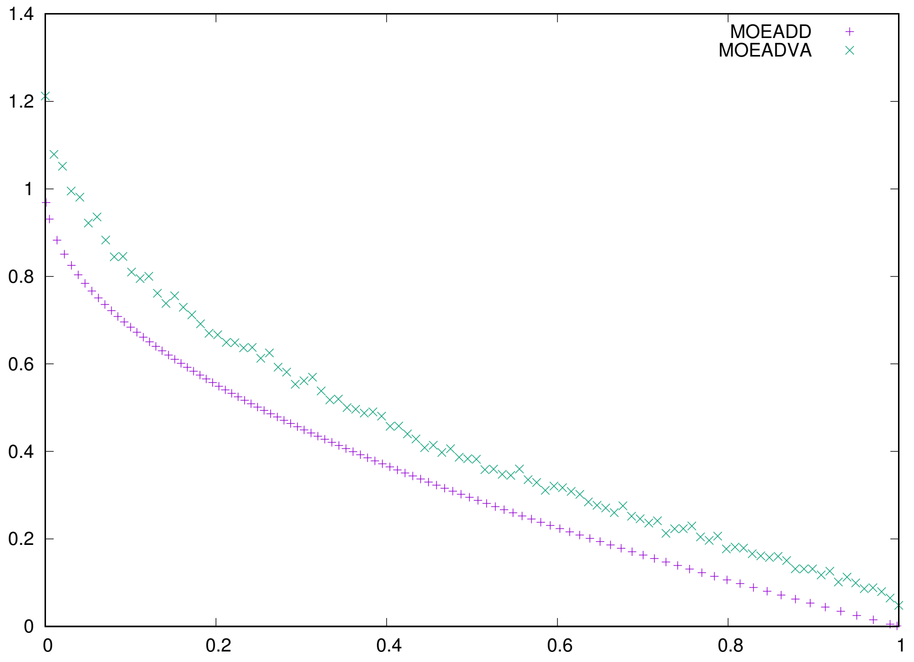

Figure 6.30: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for ZDT1 with 200 decision variables.

adopts uses more computational resources than the other techniques, however this resources are not function evaluations, therefore is an allowable spend.

		MOEA/D^2	MOEADVA	MOEA/D^2-MOEADVA	MOEA/D^2-MOEADVA	CCMOEA	MOEA/D^2-CCMOEA	MOEA/D^2-CCMOEA
Problem	No. Vars	HV	HV	Improvement	P(H)	HV	Improvement	P(H)
ZDT1	200	18.967	18.87	0.097	0.750832 (0)	18.86	0.107	0.402504 (0)
ZDT1	400	18.967	18.922	0.045	0.623605 (0)	18.875	0.092	0.544370 (0)
ZDT1	600	18.967	18.894	0.073	0.794969 (0)	18.876	0.091	0.976965 (0)
ZDT1	800	18.967	18.874	0.093	0.795012 (0)	18.881	0.086	0.795012 (0)
ZDT1	1000	18.967	18.875	0.092	0.623378 (0)	18.886	0.081	0.370532 (0)
ZDT2	200	18.633	18.428	0.205	1.000000 (0)	18.547	0.086	0.435731 (0)
ZDT2	400	18.633	18.524	0.109	0.930957 (0)	18.57	0.063	0.125857 (0)
ZDT2	600	18.633	18.467	0.166	0.156852 (0)	18.581	0.052	0.839722 (0)
ZDT2	800	18.633	18.437	0.196	0.750570 (0)	18.583	0.050	0.435231 (0)
ZDT2	1000	18.633	18.439	0.194	0.976945 (0)	18.588	0.045	0.750517 (0)
ZDT3	200	19.488	19.384	0.104	0.402504 (0)	19.386	0.102	0.193931 (0)
ZDT3	400	19.488	19.442	0.046	0.839826 (0)	19.384	0.104	0.435631 (0)
ZDT3	600	19.488	19.414	0.074	0.125940 (0)	19.414	0.074	0.236482 (0)
ZDT3	800	19.488	19.389	0.099	0.583360 (0)	19.428	0.060	0.214494 (0)
ZDT3	1000	19.488	19.391	0.097	0.930972 (0)	19.447	0.041	0.435631 (0)
ZDT4	200	11.536	0	11.536	0.021978 (1)	0	11.536	0.021978 (1)
ZDT4	400	14.802	0	14.802	0.021978 (1)	0	14.802	0.021978 (1)
ZDT4	600	16.064	0	16.064	0.021978 (1)	0	16.064	0.021978 (1)
ZDT4	800	16.463	0	16.463	0.021978 (1)	0	16.463	0.021978 (1)
ZDT4	1000	16.463	0	16.463	0.021978 (1)	0	16.463	0.021978 (1)
ZDT6	200	14.37	11.106	3.264	0.795012 (0)	12.744	1.626	0.707454 (0)
ZDT6	400	14.383	11.52	2.863	0.839826 (0)	13.11	1.273	0.583279 (0)
ZDT6	600	14.387	11.27	3.117	0.435631 (0)	13.252	1.135	0.174760 (0)
ZDT6	800	14.388	11.421	2.967	0.461538 (0)	13.38	1.008	0.193931 (0)
ZDT6	1000	14.388	11.537	2.851	0.461538 (0)	13.462	0.926	0.174760 (0)

Table 6.7: Average of the hypervolume indicator values of the results obtained for the ZDTs test problems. We show average values over 25 independent runs. The cells containing the best hypervolume value for each problem have a grey colored background. The improvement columns show the improvement on the hypervolume value that our approach was able to get against the other algorithms. The P(H) columns shows the results of the statistical analysis applied to our experiments using Wilcoxon's rank sum. P is the probability of observing the given result (the null hypothesis is true). Small values of P cast doubt on the validity of the null hypothesis. $H = 0$ indicates that the null hypothesis ("medians are equal") cannot be rejected at the 5% level. $H = 1$ indicates that the null hypothesis can be rejected at the 5% level.

Problem	No. Vars	MOEA/D^2 HV	MOEADVA HV	MOEA/D^2-MOEADVA Improvement	MOEA/D^2-MOEADVA P(H)	CCMOEA HV	MOEA/D^2-CCMOEA Improvement	MOEA/D^2-CCMOEA P(H)
DTLZ1	200	7272532.054	1538881.298	5733650.756	0.000000 (1)	0	7272532.054	0.000000 (1)
DTLZ1	400	7759193.731	637053.009	7122140.722	0.000000 (1)	0	7759193.731	0.000000 (1)
DTLZ1	600	7980879.18	0	7980879.180	0.000000 (1)	0	7980879.180	0.000000 (1)
DTLZ1	800	8009893.971	0	8009893.971	0.000000 (1)	0	8009893.971	0.000000 (1)
DTLZ1	1000	8011818.42	0	8011818.420	0.000000 (1)	0	8011818.420	0.000000 (1)
DTLZ2	200	8011978.698	8011949.081	29.617	0.000000 (1)	8011940.199	38.499	0.000000 (1)
DTLZ2	400	8011979.811	8011937.398	42.413	0.000000 (1)	8011889.452	90.359	0.000000 (1)
DTLZ2	600	8011976.993	8011936.571	40.422	0.000000 (1)	8011127.899	849.094	0.000000 (1)
DTLZ2	800	8011977.857	8011928.385	49.472	0.000000 (1)	8008194.338	3783.519	0.000000 (1)
DTLZ2	1000	8011978.256	8011927.397	50.859	0.000000 (1)	7998530.71	13447.546	0.000000 (1)
DTLZ3	200	5918889.483	0	5918889.483	0.000000 (1)	0	5918889.483	0.000000 (1)
DTLZ3	400	7752924.148	0	7752924.148	0.000000 (1)	0	7752924.148	0.000000 (1)
DTLZ3	600	8007318.671	0	8007318.671	0.000000 (1)	0	8007318.671	0.000000 (1)
DTLZ3	800	8011733.405	0	8011733.405	0.000000 (1)	0	8011733.405	0.000000 (1)
DTLZ3	1000	8011912.048	0	8011912.048	0.000000 (1)	0	8011912.048	0.000000 (1)
DTLZ4	200	8011966.489	8011841.928	124.561	0.000000 (1)	8007979.705	3986.784	0.000000 (1)
DTLZ4	400	8011948.377	8011845.097	103.280	0.000000 (1)	8007866.512	4081.865	0.000000 (1)
DTLZ4	600	8011936.587	8011831.641	104.946	0.000000 (1)	8011507.33	429.257	0.000000 (1)
DTLZ4	800	8011932.566	8011832.866	99.700	0.000000 (1)	8008491.248	3441.318	0.000000 (1)
DTLZ4	1000	8011929.679	8011836.416	93.263	0.000000 (1)	7999003.194	12926.485	0.000000 (1)

Table 6.8: Average of the hypervolume indicator values of the results obtained for the DTLZ1-4 test problems. We show average values over 25 independent runs. The cells containing the best hypervolume value for each problem have a grey colored background. The improvement columns show the improvement on the hypervolume value that our approach was able to get against the other algorithms. The P(H) columns shows the results of the statistical analysis applied to our experiments using Wilcoxon's rank sum. P is the probability of observing the given result (the null hypothesis is true). Small values of P cast doubt on the validity of the null hypothesis. $H = 0$ indicates that the null hypothesis ("medians are equal") cannot be rejected at the 5% level. $H = 1$ indicates that the null hypothesis can be rejected at the 5% level.

Problem	No. Vars	MOEA/D^2 HV	MOEADVA HV	MOEA/D^2-MOEADVA Improvement	MOEA/D^2-MOEADVA P(H)	CCMOEA HV	MOEA/D^2-CCMOEA Improvement	MOEA/D^2-CCMOEA P(H)
DTLZ5	200	8011770.318	8011729.558	40.760	0.000000 (1)	8011602.836	167.482	0.000000 (1)
DTLZ5	400	8011774.982	8011683.191	91.791	0.000000 (1)	8010964.755	810.227	0.000000 (1)
DTLZ5	600	8011758.987	8011678.262	80.725	0.000000 (1)	8008683.843	3075.144	0.000000 (1)
DTLZ5	800	8011763.281	8011676.16	87.121	0.000000 (1)	8004408.088	7355.193	0.000000 (1)
DTLZ5	1000	8011765.653	8011675.234	90.419	0.000000 (1)	7991717.465	20048.188	0.000000 (1)
DTLZ6	200	8010016.614	6645759.081	1364257.533	0.000000 (1)	5127288.267	2882728.347	0.000000 (1)
DTLZ6	400	8009820.7	1201970.295	6807850.405	0.000000 (1)	0	8009820.700	0.000000 (1)
DTLZ6	600	8008743.839	0	8008743.839	0.000000 (1)	0	8008743.839	0.000000 (1)
DTLZ6	800	8007469.038	0	8007469.038	0.000000 (1)	0	8007469.038	0.000000 (1)
DTLZ6	1000	8006158.393	0	8006158.393	0.000000 (1)	0	8006158.393	0.000000 (1)
DTLZ7	200	7901714.043	7886137.17	15576.873	0.000000 (1)	7781840.636	119873.407	0.000000 (1)
DTLZ7	400	7901723.838	7893094.014	8629.824	0.000000 (1)	7745357.704	156366.134	0.000000 (1)
DTLZ7	600	7906767.656	7887652.596	19115.060	0.000000 (1)	7710787.753	195979.903	0.000000 (1)
DTLZ7	800	7906767.197	7891466.363	15300.834	0.000000 (1)	7698037.71	208729.487	0.000000 (1)
DTLZ7	1000	7906767.371	7892924.008	13843.363	0.000000 (1)	7659711.369	247056.002	0.000000 (1)

Table 6.9: Average of the hypervolume indicator values of the results obtained for the DTLZ5-7 test problems. We show average values over 25 independent runs. The cells containing the best hypervolume value for each problem have a grey colored background. The improvement columns show the improvement on the hypervolume value that our approach was able to get against the other algorithms. The P(H) columns shows the results of the statistical analysis applied to our experiments using Wilcoxon's rank sum. P is the probability of observing the given result (the null hypothesis is true). Small values of P cast doubt on the validity of the null hypothesis. $H = 0$ indicates that the null hypothesis ("medians are equal") cannot be rejected at the 5% level. $H = 1$ indicates that the null hypothesis can be rejected at the 5% level.

		MOEA/D^2	MOEADVA	MOEA/D^2-MOEADVA	MOEA/D^2-MOEADVA	CCMOEA	MOEA/D^2-CCMOEA	MOEA/D^2-CCMOEA
Problem	No. Vars	HV	HV	Improvement	P(H)	HV	Improvement	P(H)
WFG1	200	7922873.746	7835726.48	87147.266	0.000000 (1)	7871404.772	51468.974	0.000000 (1)
WFG1	400	7927101.242	7836910.834	90190.408	0.000000 (1)	7875468.421	51632.821	0.000000 (1)
WFG1	600	7927967.952	7835874.895	92093.057	0.000000 (1)	7879724.58	48243.372	0.000000 (1)
WFG1	800	7928807.4	7836348.646	92458.754	0.000000 (1)	7882709.959	46097.441	0.000000 (1)
WFG1	1000	7928194.311	7836868.956	91325.355	0.000000 (1)	7884204.399	43989.912	0.000000 (1)
WFG2	200	7990940.294	7956269.877	34670.417	0.000000 (1)	7950331.56	40608.734	0.000000 (1)
WFG2	400	7987179.889	7954683.512	32496.377	0.000000 (1)	7946993.508	40186.381	0.000000 (1)
WFG2	600	7986616.815	7951500.139	35116.676	0.000000 (1)	7943972.594	42644.221	0.000000 (1)
WFG2	800	7986203.275	7944821.24	41382.035	0.000000 (1)	7945270.074	40933.201	0.000000 (1)
WFG2	1000	7985901.771	7952169.208	33732.563	0.000000 (1)	7952829.666	33072.105	0.000000 (1)
WFG3	200	7985262.217	7979580.302	5681.915	0.000000 (1)	7972722.929	12539.288	0.000000 (1)
WFG3	400	7982286.141	7980830.47	1455.671	0.000000 (1)	7967912.231	14373.910	0.000000 (1)
WFG3	600	7981557.8	7975331.333	6226.467	0.000000 (1)	7962232.451	19325.349	0.000000 (1)
WFG3	800	7981287.38	7959773.467	21513.913	0.000000 (1)	7962962.917	18324.463	0.000000 (1)
WFG3	1000	7988856.468	7981036.671	7819.797	0.000000 (1)	7960257.929	28598.539	0.000000 (1)
WFG4	200	8011540.032	7923442.247	88097.785	0.000000 (1)	7982122.934	29417.098	0.000000 (1)
WFG4	400	8011569.601	7924408.716	87160.885	0.000000 (1)	7978893.296	32676.305	0.000000 (1)
WFG4	600	8011520.509	7924079.898	87440.611	0.000000 (1)	7977200.288	34320.221	0.000000 (1)
WFG4	800	8011533.212	7924377.544	87155.668	0.000000 (1)	7977115.255	34417.957	0.000000 (1)
WFG4	1000	8011523.357	7924722.19	86801.167	0.000000 (1)	7976237.29	35286.067	0.000000 (1)
WFG5	200	8005490.116	7974477.965	31012.151	0.000000 (1)	7966328.97	39161.146	0.000000 (1)
WFG5	400	8005531.567	7975345.255	30186.312	0.000000 (1)	7960925.545	44606.022	0.000000 (1)
WFG5	600	8005518.595	7974910.438	30608.157	0.000000 (1)	7958685.006	46833.589	0.000000 (1)
WFG5	800	8005524.006	7975258.84	30265.166	0.000000 (1)	7957232.91	48291.096	0.000000 (1)
WFG5	1000	8005537.837	7975527.59	30010.247	0.000000 (1)	7958307.011	47230.826	0.000000 (1)

Table 6.10: Average of the hypervolume indicator values of the results obtained for the WFG1-5 test problems. We show average values over 25 independent runs. The cells containing the best hypervolume value for each problem have a grey colored background. The improvement columns show the improvement on the hypervolume value that our approach was able to get against the other algorithms. The P(H) columns shows the results of the statistical analysis applied to our experiments using Wilcoxon's rank sum. P is the probability of observing the given result (the null hypothesis is true). Small values of P cast doubt on the validity of the null hypothesis. $H = 0$ indicates that the null hypothesis ("medians are equal") cannot be rejected at the 5% level. $H = 1$ indicates that the null hypothesis can be rejected at the 5% level.

		MOEA/D^2	MOEADVA	MOEA/D^2-MOEADVA	MOEA/D^2-MOEADVA	CCMOEA	MOEA/D^2-CCMOEA	MOEA/D^2-CCMOEA
Problem	No. Vars	HV	HV	Improvement	P(H)	HV	Improvement	P(H)
WFG6	200	8010650.276	7980329.793	30320.483	0.000000 (1)	7960448.383	50201.893	0.000000 (1)
WFG6	400	8011180.054	7985703.349	25476.705	0.000000 (1)	7953347.477	57832.577	0.000000 (1)
WFG6	600	8011312.014	7988445.759	22866.255	0.000000 (1)	7952137.625	59174.389	0.000000 (1)
WFG6	800	8011414.494	7989607.939	21806.555	0.000000 (1)	7951605.716	59808.778	0.000000 (1)
WFG6	1000	8011254.828	7990518.412	20736.416	0.000000 (1)	7946826.852	64427.976	0.000000 (1)
WFG7	200	8011362.288	8008668.662	2693.626	0.000000 (1)	7968167.391	43194.897	0.000000 (1)
WFG7	400	8011430.934	8007615.846	3815.088	0.000000 (1)	7963801.944	47628.990	0.000000 (1)
WFG7	600	8011462.834	8005213.926	6248.908	0.000000 (1)	7962932.903	48529.931	0.000000 (1)
WFG7	800	8011452.828	7993359.968	18092.860	0.000000 (1)	7961273.928	50178.900	0.000000 (1)
WFG7	1000	8011481.764	8005913.608	5568.156	0.000000 (1)	7961182.174	50299.590	0.000000 (1)
WFG8	200	7996726.765	7962918.085	33808.680	0.000000 (1)	7968797.162	27929.603	0.000000 (1)
WFG8	400	7996271.274	7968218.982	28052.292	0.000000 (1)	7964974.147	31297.127	0.000000 (1)
WFG8	600	7996282.528	7970145.483	26137.045	0.000000 (1)	7962911.357	33371.171	0.000000 (1)
WFG8	800	7996317.726	7972277.2	24040.526	0.000000 (1)	7962816.827	33500.899	0.000000 (1)
WFG8	1000	7996627.817	7972176.163	24451.654	0.000000 (1)	7962723.827	33903.990	0.000000 (1)
WFG9	200	8009971.451	7997740.588	12230.863	0.000000 (1)	7959506.023	50465.428	0.000000 (1)
WFG9	400	8009584.103	7995020.482	14563.621	0.000000 (1)	7950343.561	59240.542	0.000000 (1)
WFG9	600	8009417.728	7975631.928	33785.800	0.000000 (1)	7950172.817	59244.911	0.000000 (1)
WFG9	800	8009428.918	7987722.583	21706.335	0.000000 (1)	7949128.124	60300.794	0.000000 (1)
WFG9	1000	8009713.817	7987623.583	22090.234	0.000000 (1)	7951823.91	57889.907	0.000000 (1)

Table 6.11: Average of the hypervolume indicator values of the results obtained for the WFG6-9 test problems. We show average values over 25 independent runs. The cells containing the best hypervolume value for each problem have a grey colored background. The improvement columns show the improvement on the hypervolume value that our approach was able to get against the other algorithms. The P(H) columns shows the results of the statistical analysis applied to our experiments using Wilcoxon's rank sum. P is the probability of observing the given result (the null hypothesis is true). Small values of P cast doubt on the validity of the null hypothesis. $H = 0$ indicates that the null hypothesis ("medians are equal") cannot be rejected at the 5% level. $H = 1$ indicates that the null hypothesis can be rejected at the 5% level.

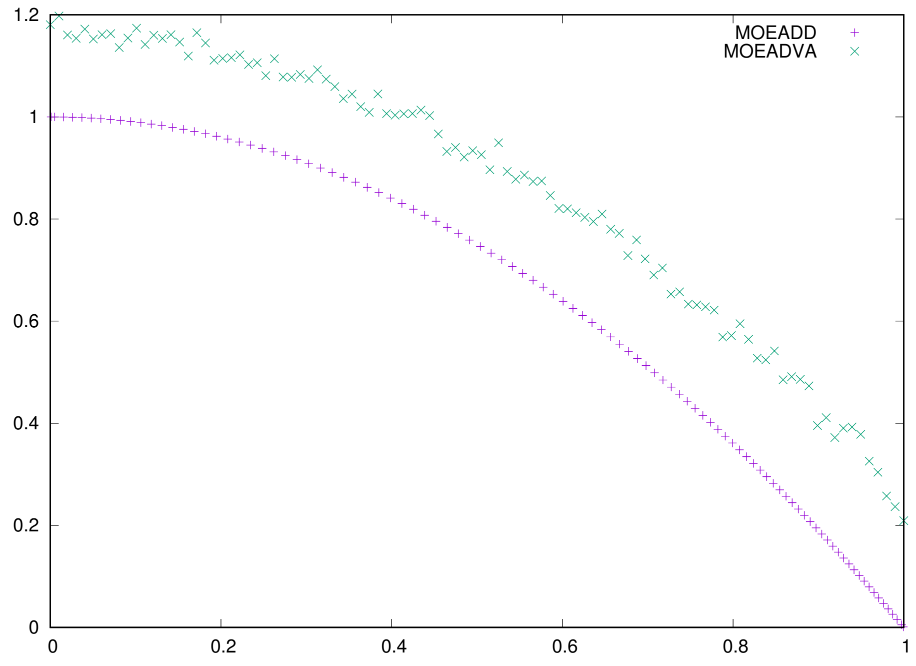

Figure 6.31: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for ZDT2 with 200 decision variables.

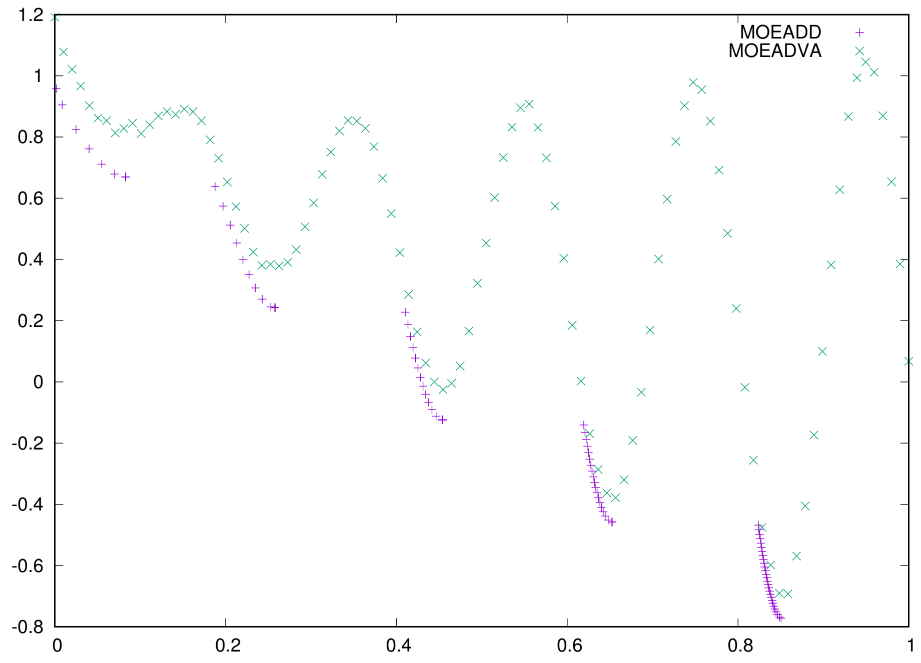

Figure 6.32: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for ZDT3 with 200 decision variables.

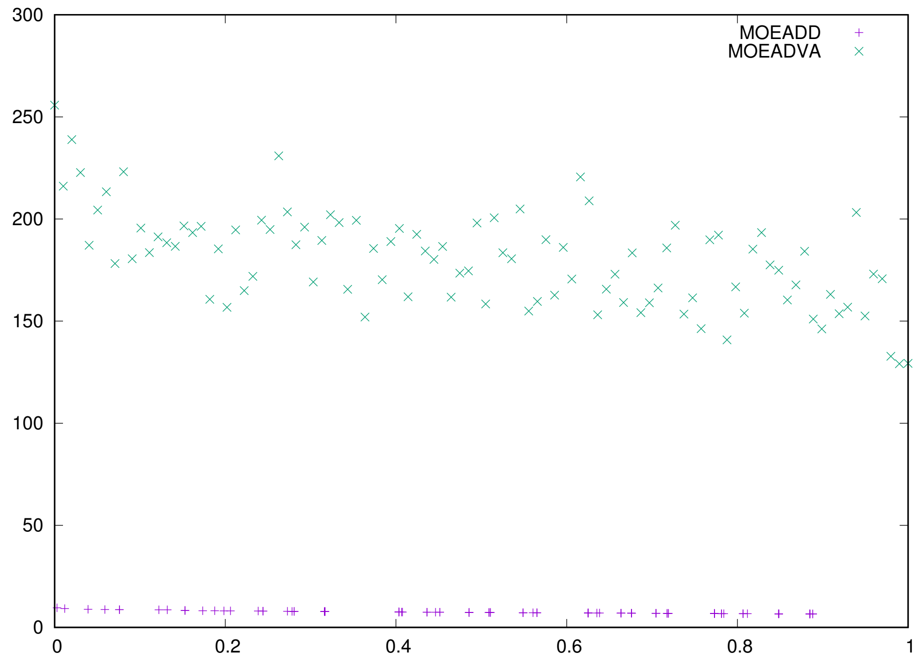

Figure 6.33: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for ZDT4 with 200 decision variables.

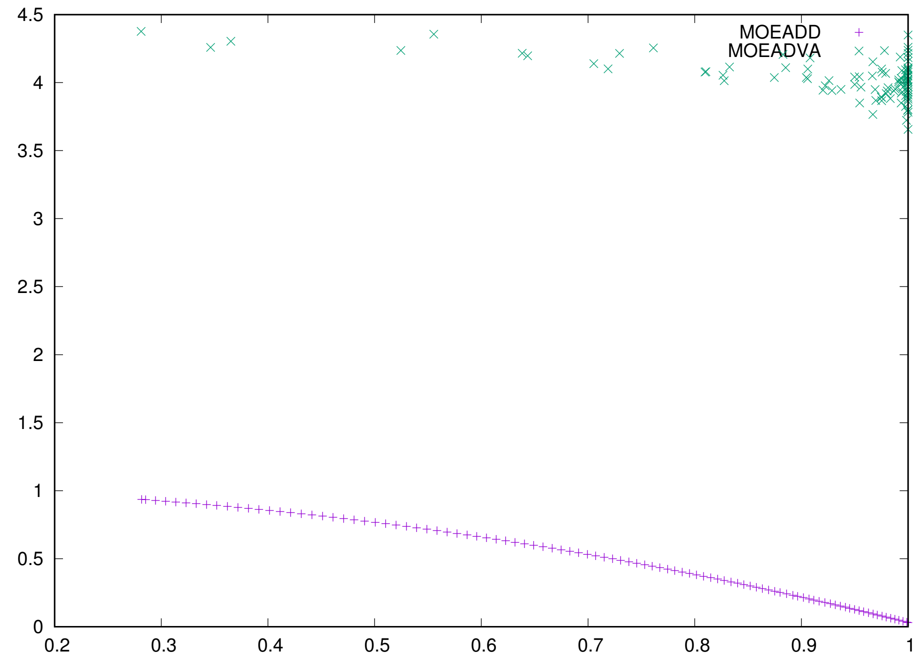

Figure 6.34: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for ZDT6 with 200 decision variables.

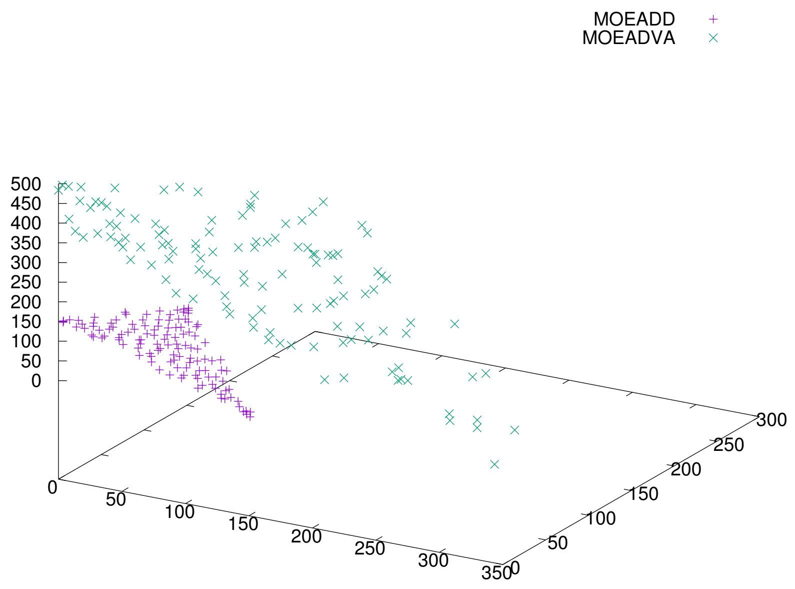

Figure 6.35: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for DTLZ1 with 200 decision variables.

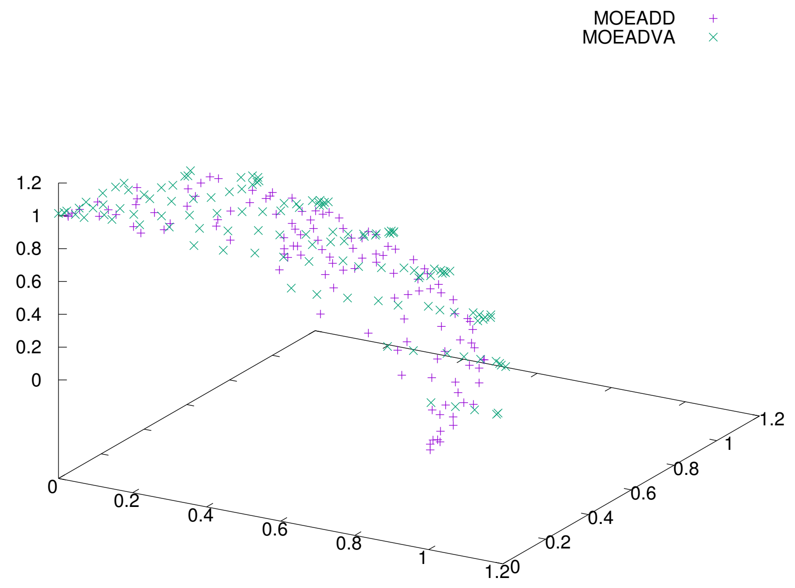

Figure 6.36: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for DTLZ2 with 200 decision variables.

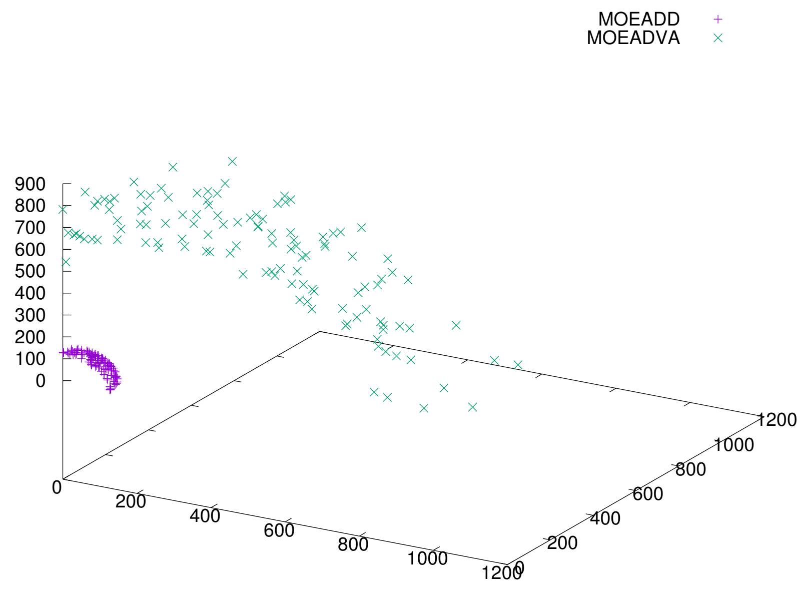

Figure 6.37: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for DTLZ3 with 200 decision variables.

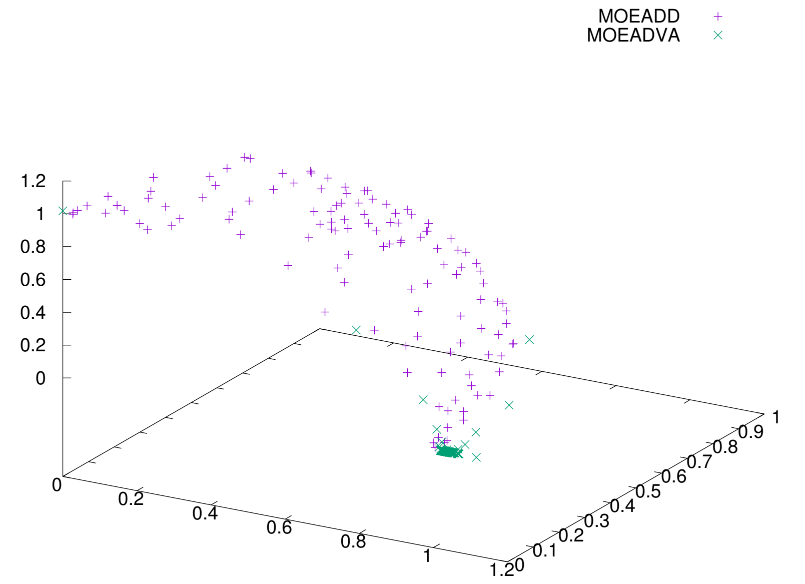

Figure 6.38: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for DTLZ4 with 200 decision variables.

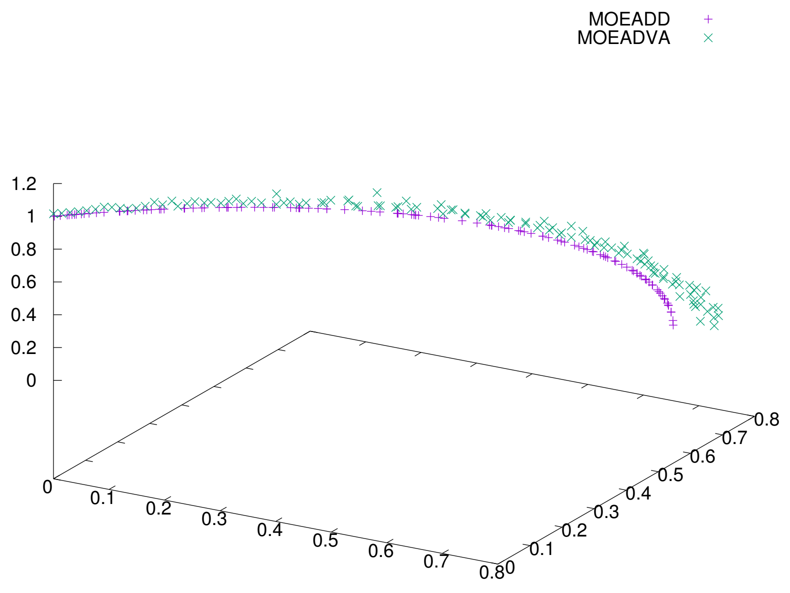

Figure 6.39: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for DTLZ5 with 200 decision variables.

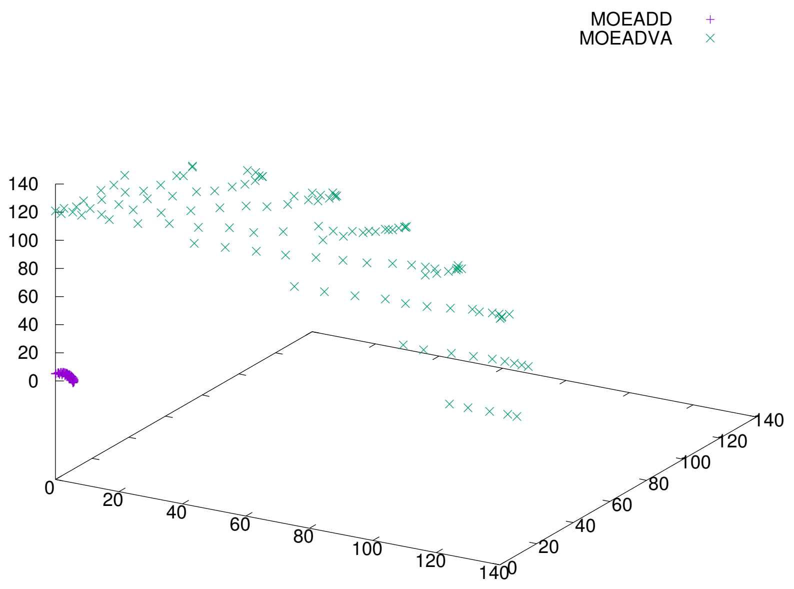

Figure 6.40: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for DTLZ6 with 200 decision variables.

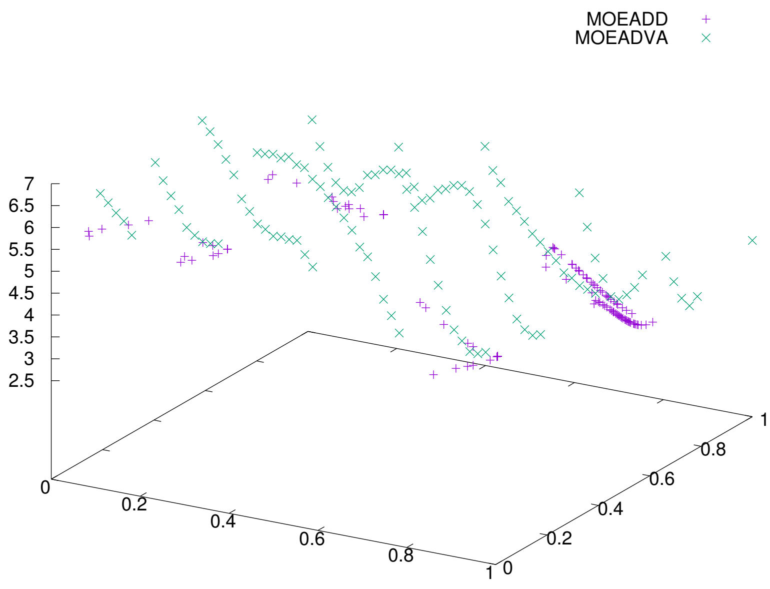

Figure 6.41: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for DTLZ7 with 200 decision variables.

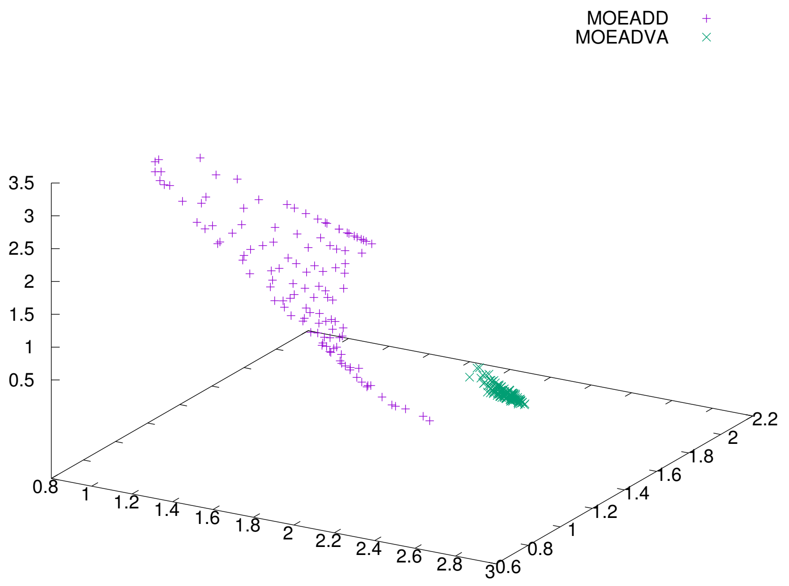

Figure 6.42: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for WFG1 with 200 decision variables.

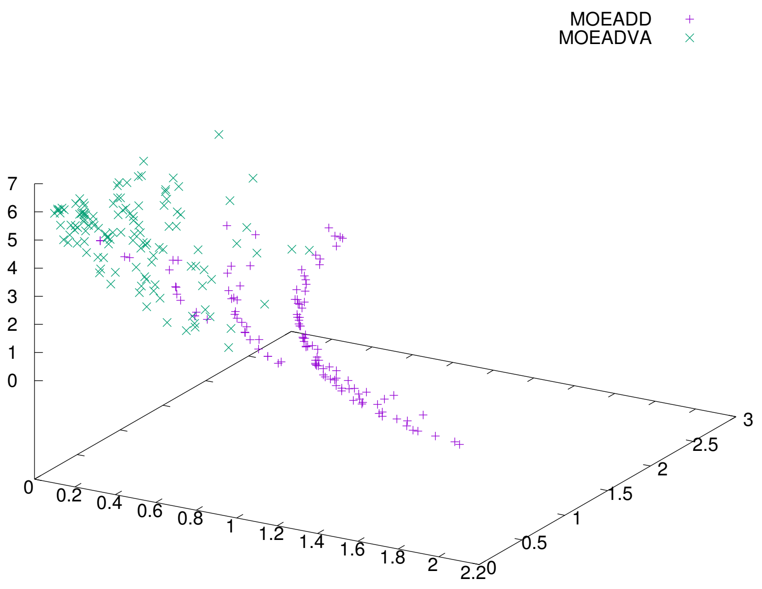

Figure 6.43: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for WFG2 with 200 decision variables.

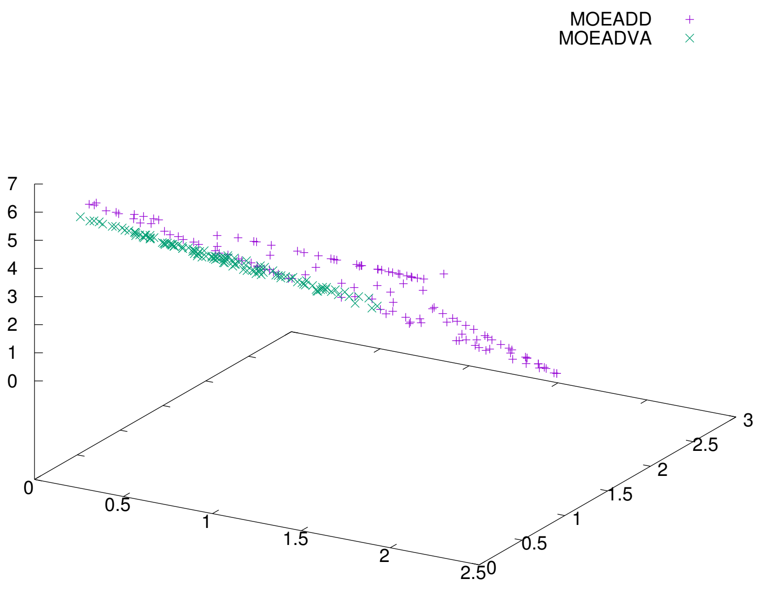

Figure 6.44: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for WFG3 with 200 decision variables.

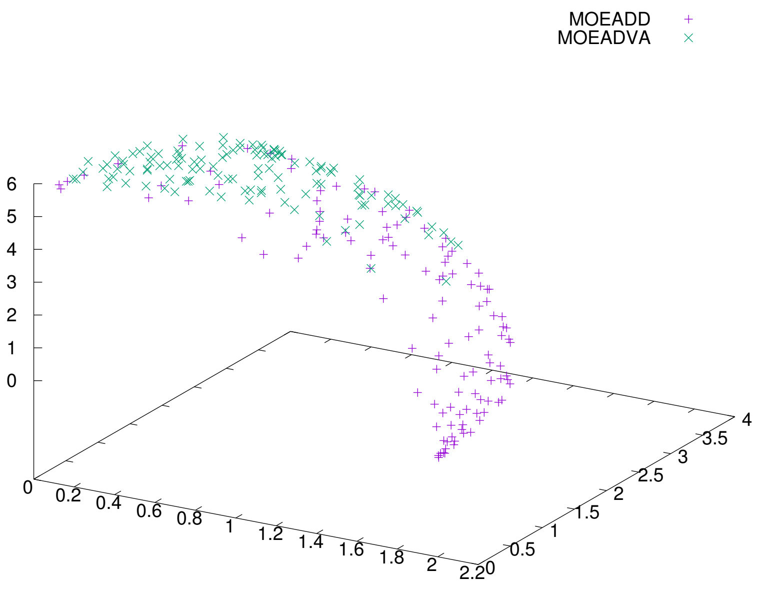

Figure 6.45: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for WFG4 with 200 decision variables.

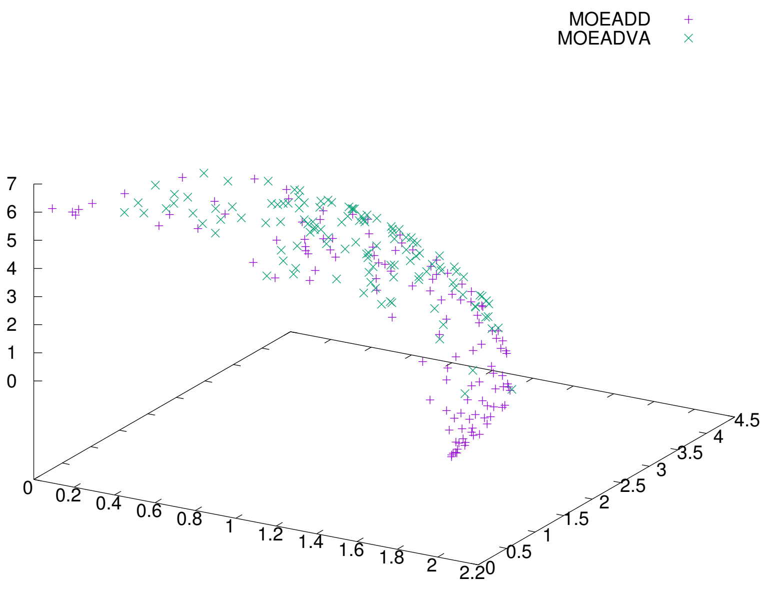

Figure 6.46: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for WFG5 with 200 decision variables.

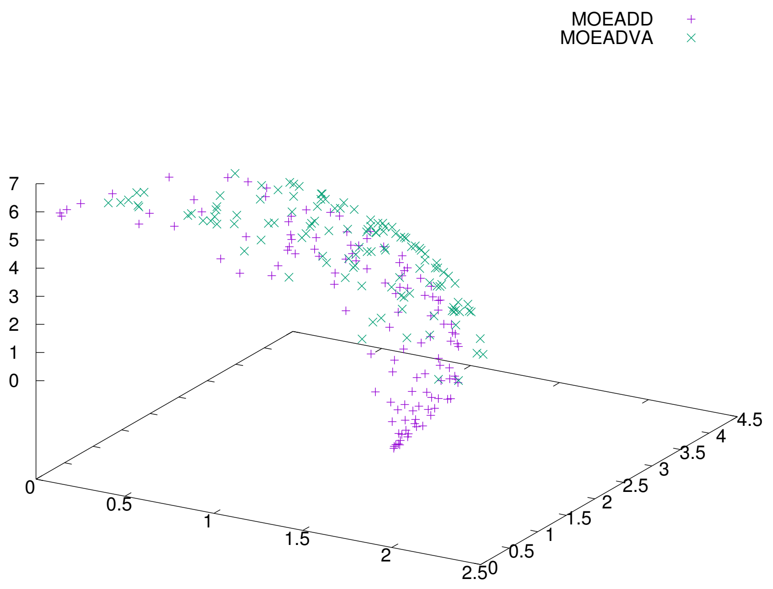

Figure 6.47: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for WFG6 with 200 decision variables.

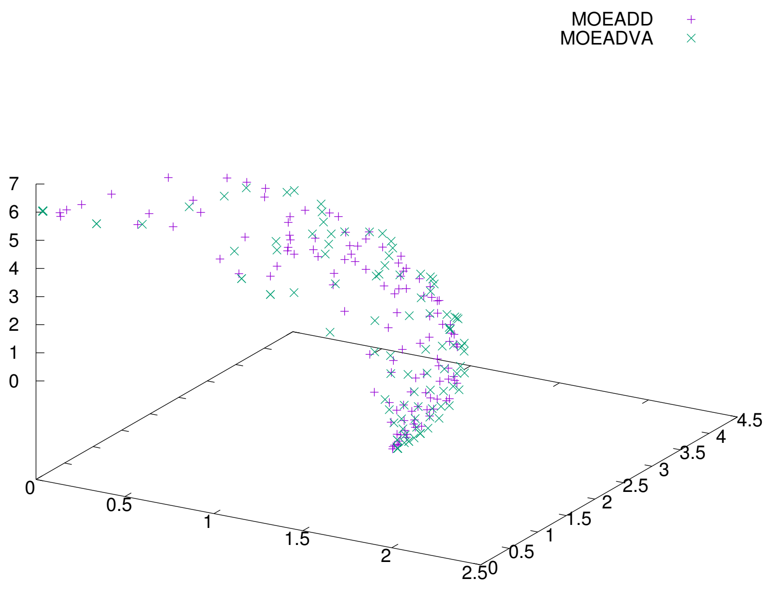

Figure 6.48: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for WFG7 with 200 decision variables.

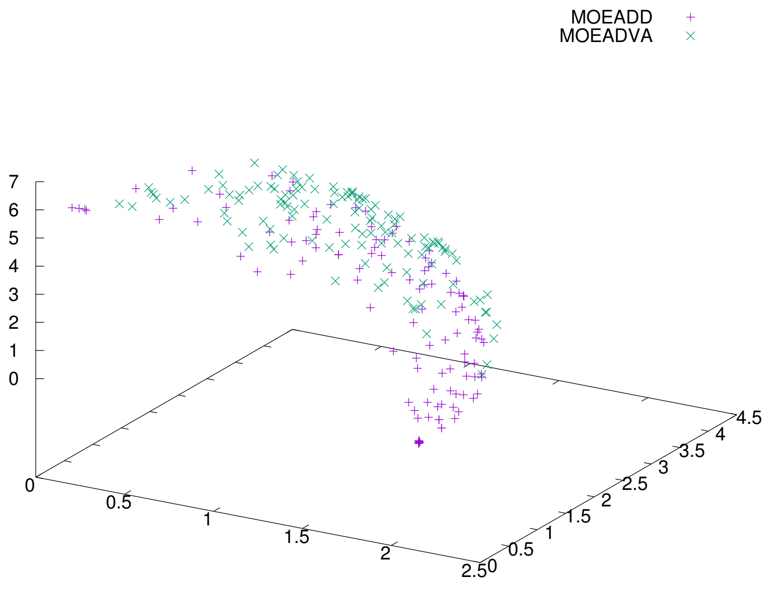

Figure 6.49: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for WFG8 with 200 decision variables.

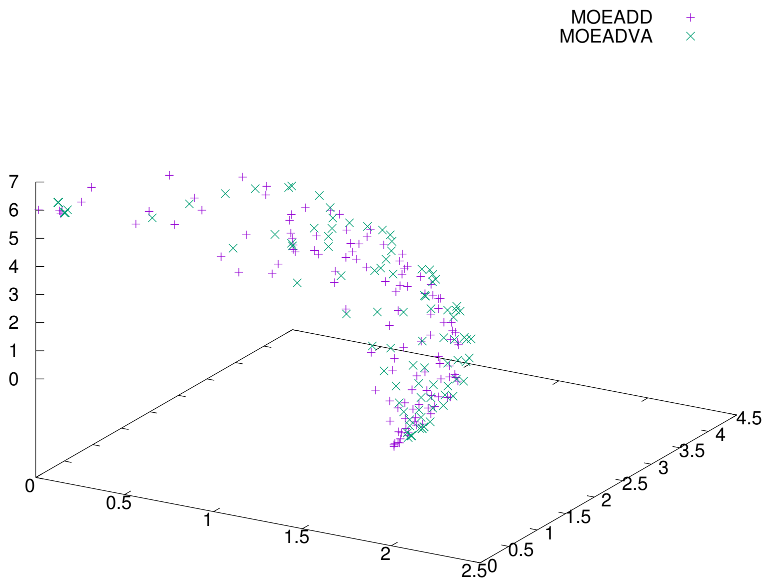

Figure 6.50: Pareto front approximations obtained by MOEADVA and MOEA/D^2 for WFG9 with 200 decision variables.

Problem	**No. Vars**	**MOEA/D^2**	**MOEADVA**	**CCMOEA**
ZDT1	200	0.401	0.018	0.037
ZDT1	400	3.174	0.181	0.260
ZDT1	600	10.857	0.503	1.002
ZDT1	800	21.685	1.117	2.419
ZDT1	1000	41.888	2.093	4.509
ZDT2	200	0.273	0.018	0.039
ZDT2	400	2.225	0.180	0.256
ZDT2	600	8.025	0.500	1.003
ZDT2	800	17.315	1.109	2.547
ZDT2	1000	33.406	2.098	4.604
ZDT3	200	0.467	0.019	0.036
ZDT3	400	2.869	0.182	0.261
ZDT3	600	11.451	0.505	0.963
ZDT3	800	21.613	1.118	2.365
ZDT3	1000	37.757	2.127	4.577
ZDT4	200	0.352	0.068	0.103
ZDT4	400	2.453	0.661	0.915
ZDT4	600	8.115	2.056	3.863
ZDT4	800	19.037	4.816	8.639
ZDT4	1000	35.664	9.320	22.336
ZDT6	200	0.306	0.021	0.046
ZDT6	400	2.185	0.204	0.318
ZDT6	600	7.140	0.566	1.294
ZDT6	800	17.007	1.265	2.217
ZDT6	1000	32.238	2.385	5.475

Table 6.12: Average of the computational time (in minutes) of each algorithm when solving ZDT test suite.

Problem	**No. Vars**	**MOEA/D^2**	**MOEADVA**	**CCMOEA**
DTLZ1	200	0.640	0.079	0.108
DTLZ1	400	6.611	0.854	0.945
DTLZ1	600	12.307	2.653	3.911
DTLZ1	800	25.876	6.204	7.187
DTLZ1	1000	46.969	11.948	22.429
DTLZ2	200	0.581	0.025	0.051
DTLZ2	400	3.981	0.241	0.397
DTLZ2	600	10.392	0.680	1.431
DTLZ2	800	21.553	1.534	3.367
DTLZ2	1000	38.880	2.906	7.088
DTLZ3	200	0.560	0.080	0.101
DTLZ3	400	4.370	0.858	0.901
DTLZ3	600	11.431	2.679	3.504
DTLZ3	800	24.528	6.248	9.740
DTLZ3	1000	44.646	11.984	19.799
DTLZ4	200	0.562	0.028	0.064
DTLZ4	400	3.822	0.251	0.452
DTLZ4	600	11.896	0.702	1.456
DTLZ4	800	22.145	1.558	3.905
DTLZ4	1000	39.293	2.928	9.884
DTLZ5	200	0.587	0.029	0.052
DTLZ5	400	3.464	0.298	0.372
DTLZ5	600	9.664	0.862	1.430
DTLZ5	800	20.187	1.957	3.851
DTLZ5	1000	36.926	3.701	7.943
DTLZ6	200	0.471	0.108	0.164
DTLZ6	400	3.799	1.079	1.167
DTLZ6	600	10.694	3.396	5.111
DTLZ6	800	23.173	7.946	12.286
DTLZ6	1000	43.950	15.337	25.866
DTLZ7	200	0.534	0.021	0.061
DTLZ7	400	5.112	0.189	0.328
DTLZ7	600	10.257	0.518	2.095
DTLZ7	800	20.888	1.152	2.706
DTLZ7	1000	38.434	2.123	5.562

Table 6.13: Average of the computational time (in minutes) of each algorithm when solving DTLZ test suite.

Problem	**No. Vars**	**MOEA/D^2**	**MOEADVA**	**CCMOEA**
WFG1	200	0.852	0.319	0.795
WFG1	400	5.830	2.875	5.661
WFG1	600	17.537	8.784	21.852
WFG1	800	38.798	20.042	61.693
WFG1	1000	74.124	38.343	139.664
WFG2	200	1.520	0.860	1.009
WFG2	400	10.598	6.912	10.316
WFG2	600	32.562	22.146	30.711
WFG2	800	72.749	51.482	85.584
WFG2	1000	149.247	99.002	169.681
WFG3	200	1.532	0.916	1.037
WFG3	400	10.915	6.899	8.929
WFG3	600	35.006	22.157	35.369
WFG3	800	78.331	50.137	95.147
WFG3	1000	151.851	99.405	177.624
WFG4	200	0.911	0.262	0.645
WFG4	400	6.010	1.915	4.748
WFG4	600	18.261	5.796	17.594
WFG4	800	40.089	13.252	45.641
WFG4	1000	153.642	25.114	99.873
WFG5	200	0.717	0.193	0.524
WFG5	400	5.075	1.311	3.939
WFG5	600	15.627	3.811	15.083
WFG5	800	34.430	8.360	37.665
WFG5	1000	132.923	15.436	78.130

Table 6.14: Average of the computational time (in minutes) of each algorithm when solving WFG test suite.

Chapter 7

Conclusions and Future Work

In this thesis, we have studied the way some approaches and frameworks can be adopted to develop MOEAs for large scale MOPs.

First, we have developed a new cooperative coevolutionary framework for solving MOPs using a novel cooperation strategy based on the Nash equilibrium. With this, we presented a novel cooperative coevolutionary MOEA, called NashCC, which was shown to be able to successfully deal with the ZDT and DTLZ test suites. We have studied the convergence rate of our proposed NashCC with respect to that of NSCCGA and GCEA. The results confirmed that our proposed approach outperforms the other two CCAs and that the collaboration framework has a great impact in CCAs. We confirm that the tendency of CCAs to fall into ESS is also present when dealing with MOPs and that looking for ideal collaborations is a good way to alleviate this problem.

Another outstanding example is the development of a novel decomposition-based MOEA called MOEA/D^2, which adopts decomposition based techniques used by co-operative coevolutionary algorithms. MOEA/D^2 uses a double decomposition of the MOP, one in objective functions space, as MOEA/D, and another in decision variables space. Our experimental results indicate that MOEA/D^2 clearly outperforms MOEA/D, MOEADVA, CCMOEA and GDE3 in MOPs having from 200 up to 1200 decision variables. This approach was able to deal with all the difficulties presented in the ZDT, DTLZ and WFG test suite, even in high dimensionality. The results confirmed that our proposed approach is very effective and efficient in tackling large scale MOPs.

As part of our future work, we intend to study other decomposition techniques for decision variable space. We are also interested in studying the possible use of other (computationally inexpensive) methods to generate a set of weight vectors more uniformly distributed for MOEA/D^2. Also, we aim to develop a parallel version of some of the proposed approaches in order to increase their efficiency and save computational time.

Appendix A

Test Problems

A.1 Zitzler-Deb-Thiele Test Suite

The Zitzler-Deb-Thiele Test (ZDT) test suite [2] includes 5 representative bi-objective test problems for comparing optimizers, which are scalable to any number of decision variables. The majority of these problems are separable, including degenerated and multimodal Pareto optimal fronts, of which the exact shape and location are known.

A.1.1 ZDT1

The test function ZDT1 has a convex Pareto-optimal front (see Figure A.1):

$$\begin{array}{ll}
\text{Given} & \vec{x} = \{x_1, x_2, \ldots, x_m\} \\
\text{Minimize} & f_1(x_1) = x_1 \\
\text{where} & g(x_2, \ldots, x_m) = 1 + 9 \cdot \sum_{i=2}^{m} x_i/(m-1) \\
 & h(f_1, g) = 1 - \sqrt{f_1/g} \\
\text{subject to} & 0 \leq x_i \leq 1, \text{ for } i = 1, 2, \ldots, m.
\end{array} \tag{A.1}$$

where m is the number of decision variables. The Pareto-optimal front is formed with $g(\vec{x}) = 1$.

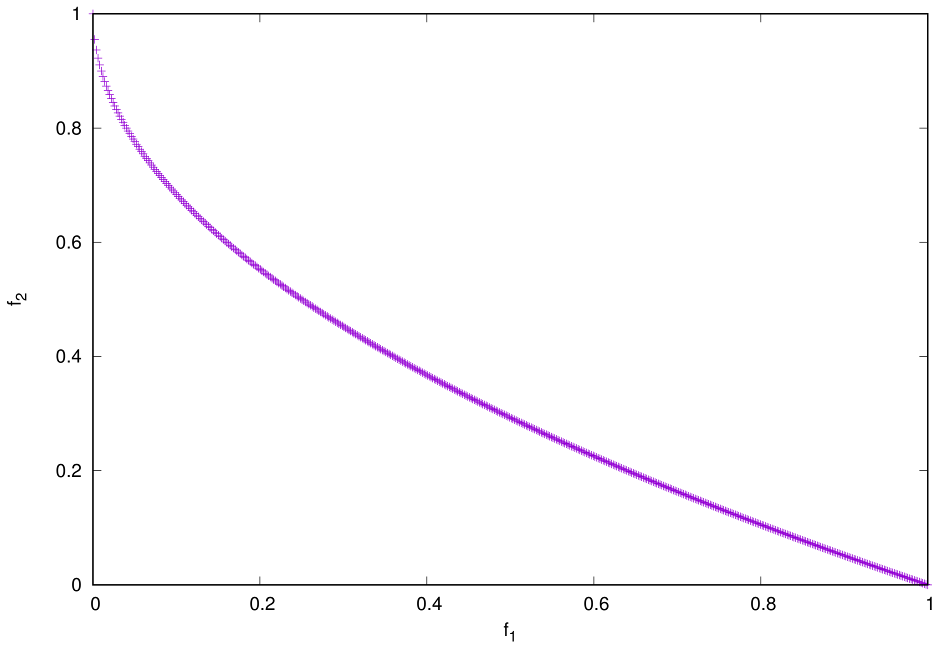

Figure A.1: ZDT1 Pareto optimal front.

A.1.2 ZDT2

The test function ZDT2 is the nonconvex counterpart to ZDT1 (see Figure A.2) :

$$
\begin{aligned}
&\text{Given} && \vec{x} = \{x_1, x_2, \ldots, x_m\} \\
&\text{Minimize} && f_1(x_1) = x_1 \\
&\text{where} && g(x_2, \ldots, x_m) = 1 + 9 \cdot \sum_{i=2}^{m} x_i/(m-1) \\
& && h(f_1, g) = 1 - (f_1/g)^2 \\
&\text{subject to} && 0 \leq x_i \leq 1, \text{ for } i = 1, 2, \ldots, m.
\end{aligned}
\tag{A.2}
$$

where m is the number of decision variables. The Pareto-optimal front is formed with $g(\vec{x}) = 1$.

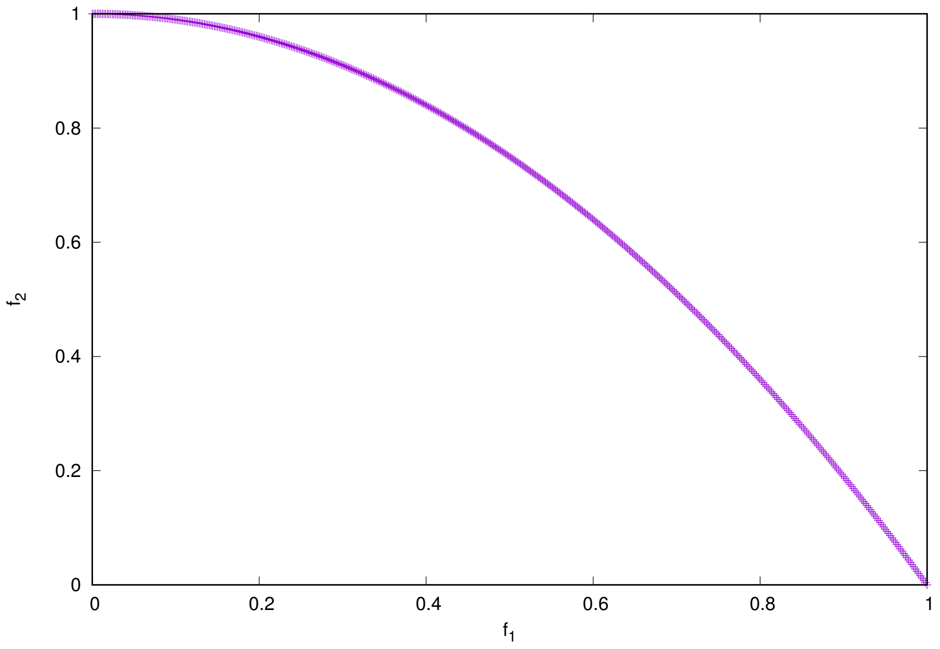

Figure A.2: ZDT2 Pareto optimal front.

A.1.3 ZDT3

The test function ZDT3 represents the discreteness feature; its Pareto-optimal front consists of several noncontiguous convex parts (see Figure A.3):

$$\begin{aligned}
&\text{Given} && \vec{x} = \{x_1, x_2, \dots, x_m\} \\
&\text{Minimize} && f_1(x_1) = x_1 \\
&\text{where} && g(x_2, \dots, x_m) = 1 + 9 \cdot \sum_{i=2}^{m} x_i/(m-1) \\
& && h(f_1, g) = 1 - \sqrt{f_1/g} - (f_1/g)\sin(10\pi f_1) \\
&\text{subject to} && 0 \leq x_i \leq 1, \text{ for } i = 1, 2, \dots, m.
\end{aligned} \tag{A.3}$$

where m is the number of decision variables. The Pareto-optimal front is formed with $g(\vec{x}) = 1$. The introduction of the sine function in h causes discontinuity in the Pareto-optimal front. However, there is no discontinuity in the parameter space.

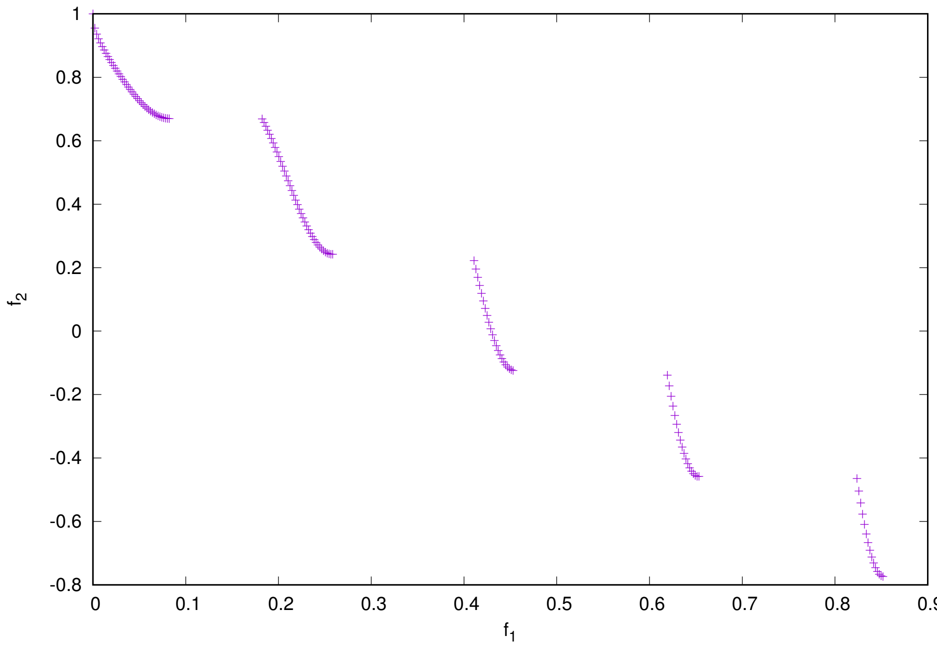

Figure A.3: ZDT3 Pareto optimal front.

A.1.4 ZDT4

The test function ZDT4 contains 21^9 local Pareto-optimal fronts (see Figure A.4) and, therefore, tests for the MOEAs ability to deal with multimodality:

$$
\begin{aligned}
&\text{Given} && \vec{x} = \{x_1, x_2, \ldots, x_m\} \\
&\text{Minimize} && f_1(x_1) = x_1 \\
&\text{where} && g(x_2, \ldots, x_m) = 1 + 10(m-1) + \sum_{i=2}^{m}(x_i^2 - 10\cos(4\pi x_i)) \\
& && h(f_1, g) = 1 - \sqrt{f_1/g} \\
&\text{subject to} && 0 \leq x_1 \leq 1 \\
& && -5 \leq x_i \leq 5, \text{ for } i = 2, \ldots, m.
\end{aligned}
\tag{A.4}
$$

where m is the number of decision variables. The Pareto-optimal front is formed with $g(\vec{x}) = 1.25$. Note that not all local Pareto-optimal sets are distinguishable in the objective space.

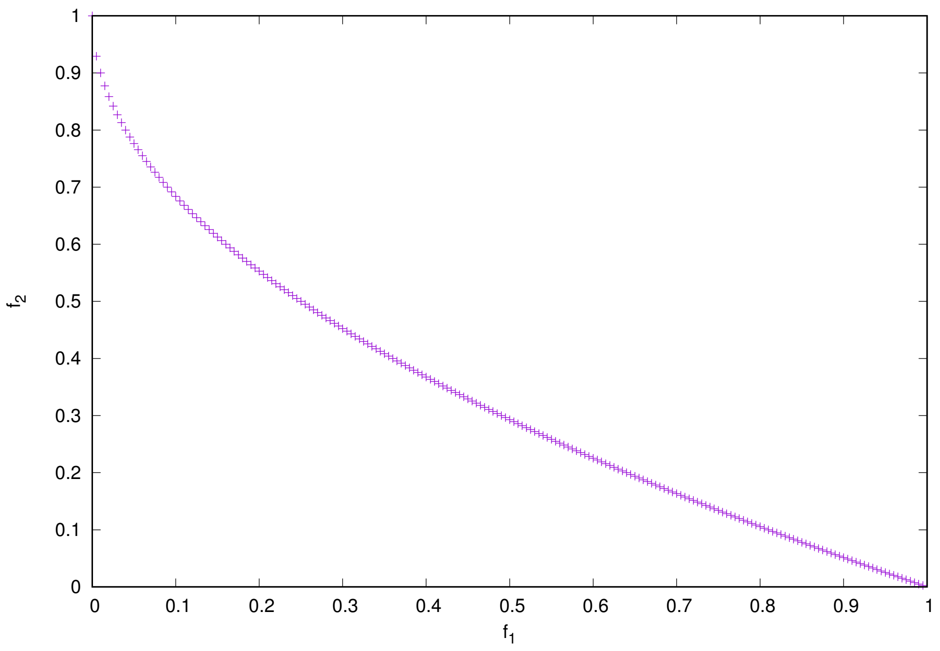

Figure A.4: ZDT4 Pareto optimal front.

A.1.5 ZDT6

The test function ZDT6 includes two difficulties caused by the nonuniformity of the search space: first, the Pareto-optimal solutions are nonuniformly distributed along the global Pareto front (the front is biased for solutions for which $f_1(\vec{x})$ is near one); second, the density of the solutions is lowest near the Pareto-optimal front and highest away from the front (see Figure A.5):

$$\begin{array}{lr}
\text{Given} & \vec{x} = \{x_1, x_2, \ldots, x_m\} \\
\text{Minimize} & f_1(x_1) = 1 - exp(-4x_1)\sin^6(6\pi x_1) \\
\text{where} & g(x_2, \ldots, x_m) = 1 + 9 \cdot ((\sum_{i=2}^{m} x_i)/(m-1))^{0.25} \\
 & h(f_1, g) = 1 - (f_1/g)^2 \\
\text{subject to} & 0 \leq x_i \leq 1, \text{ for } i = 1, 2, \ldots, m.
\end{array} \tag{A.5}$$

where m is the number of decision variables. The Pareto-optimal front is formed with $g(\vec{x}) = 1$ and is nonconvex.

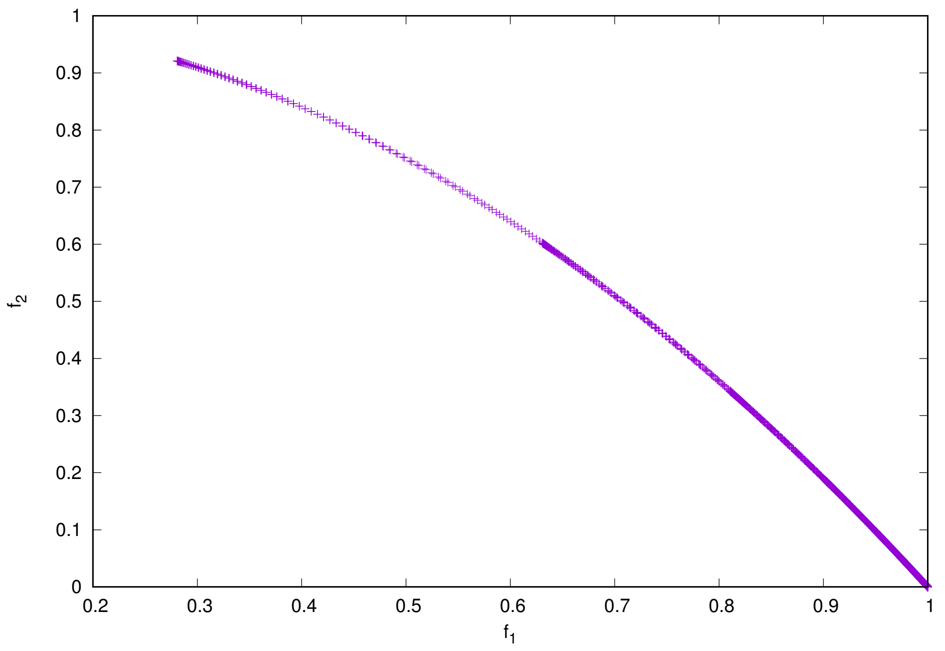

Figure A.5: ZDT6 Pareto optimal front.

A.2 Deb-Thiele-Laumanns-Zitzler Test Suite

The Deb-Thiele-Laumanns-Zitzler (DTLZ) test suite [3] includes nine representative test problems for comparing optimizers, which are scalable to any number of decision variables and objectives. The majority of these problems are separable, including degenerated and multimodal Pareto optimal fronts, of which the exact shape and location are known. Next, we present the seven unconstrained problems of the DTLZ test suite. Here, the total number of decision variables is given by $n = m+k-1$, where m represents the number of objectives and k is the number of distance parameters.

A.2.1 DTLZ1

This test problem is separable and multimodal. Its Pareto optimal front is linear and is given by the following expression:

$$\begin{aligned}
\text{Given}\vec{x} &= \{x_1, \dots, x_{m-1}, x_m, \dots, x_n\} \\
\text{Minimize } f_1(\vec{x}) &= 0.5(1+g(\vec{y}))\prod_{i=1}^{m-1} x_i \\
f_{j=2:m-1}(\vec{x}) &= 0.5(1+g(\vec{y}))(1-x_{m-j+1})\prod_{i=1}^{m-j} x_i \\
f_m(\vec{x}) &= 0.5(1+g(\vec{y}))(1-x_1) \\
\text{where} y_{i=1:k} &= \{x_m, x_{m+1}, \dots, x_n\} \\
g(\vec{y}) &= 100\left[k + \sum_{i=1}^{k}(y_i - 0.5)^2 - \cos(20\pi(y_i - 05))\right] \\
\text{subject to} \quad 0 \leq & x_i \leq 1, \text{ for } i = 1, 2, \dots, n.
\end{aligned} \tag{A.6}$$

The number of decision variables is given by $n = m + k - 1$. All objective function values lie on the linear hyper-plane $\sum_{i=1}^{m} f_i = 0.5$. The Pareto optimal solution corresponds to $\vec{y} = (0, 0, \dots)^T$. The difficulty in this problem is to converge to the hyper-plane. The search space contains $(11^k - 1)$ local Pareto optimal fronts, each of which can attract an optimizer. The Pareto optimal front is shown in Figure A.6.

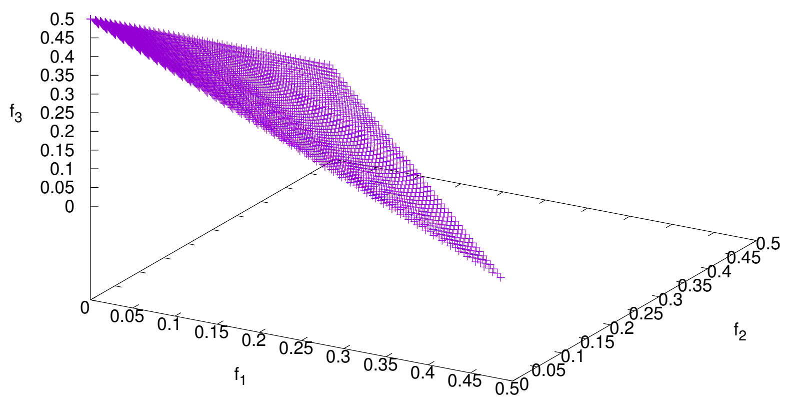

Figure A.6: Pareto optimal front of DTLZ1 with 3 objective functions.

A.2.2 DTLZ2

This problem is separable and unimodal. The geometry of its Pareto optimal front is concave (see Figure A.7), and is defined as follows:

$$\begin{aligned}
\text{Given } \vec{x} &= \{x_1, \ldots, x_{m-1}, x_m, \ldots, x_n\} \\
\text{Minimize } f_1(\vec{x}) &= (1 + g(\vec{y})) \prod_{i=1}^{m-1} \cos\left(\frac{x_i \pi}{2}\right) \\
f_{j=2:m-1}(\vec{x}) &= (1 + g(\vec{y})) \left(\prod_{i=1}^{m-j} \cos\left(\frac{x_i \pi}{2}\right)\right) \sin\left(\frac{x_{m-j+1}\pi}{2}\right) \\
f_m(\vec{x}) &= (1 + g(\vec{y})) \sin\left(\frac{x_1 \pi}{2}\right) \\
\text{where } y_{i=1:k} &= \{x_m, x_{m+1}, \ldots, x_n\} \\
g(\vec{y}) &= \sum_{i=1}^{k} (y_i - 0.5)^2 \\
\text{subject to } \quad & 0 \le x_i \le 1, \text{ for } i = 1, 2, \ldots, n.
\end{aligned} \tag{A.7}$$

The number of decision variables is given by $n = m + k - 1$. The Pareto optimal solutions correspond to $\vec{y} = (0.5, 0.5, \ldots)^T$ and all objective functions values must

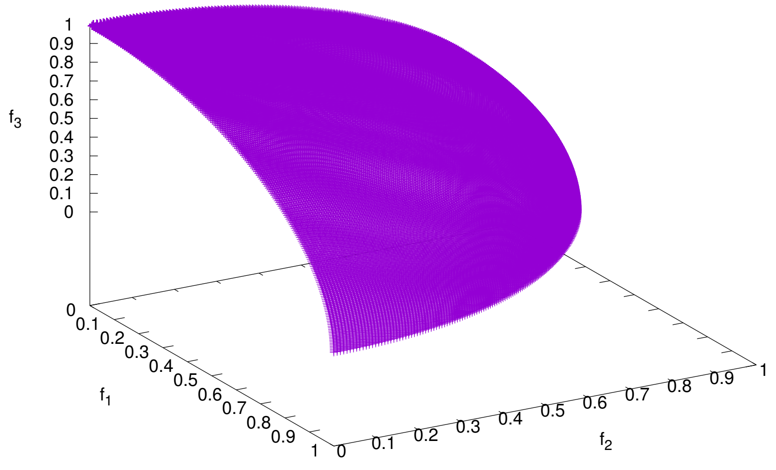

Figure A.7: Pareto optimal front of DTLZ2 with 3 objective functions.

satisfy that $\sum_{i=1}^{m}(f_i)^2 = 1$.

A.2.3 DTLZ3

This problem is the same as DTLZ2 except for a new g function, that makes it multimodal. The definition is given as follows:

$$\begin{aligned}
\text{Given } \vec{x} &= \{x_1, \ldots, x_{m-1}, x_m, \ldots, x_n\} \\
\text{Minimize } f_1(\vec{x}) &= (1+g(\vec{y}))\prod_{i=1}^{m-1}\cos\left(\frac{x_i\pi}{2}\right) \\
f_{j=2:m-1}(\vec{x}) &= (1+g(\vec{y}))\left(\prod_{i=1}^{m-j}\cos\left(\frac{x_i\pi}{2}\right)\right)\sin\left(\frac{x_{m-j+1}\pi}{2}\right) \\
f_m(\vec{x}) &= (1+g(\vec{y}))\sin\left(\frac{x_1\pi}{2}\right) \qquad \text{(A.8)}\\
\text{where } y_{i=1:k} &= \{x_m, x_{m+1}, \ldots, x_n\} \\
g(\vec{y}) &= 100\left[k + \sum_{i=1}^{k}(y_i - 0.5)^2 - \cos(20\pi(y_i - 05))\right] \\
\text{subject to} \quad & 0 \le x_i \le 1, \text{ for } i = 1, 2, \ldots, n.
\end{aligned}$$

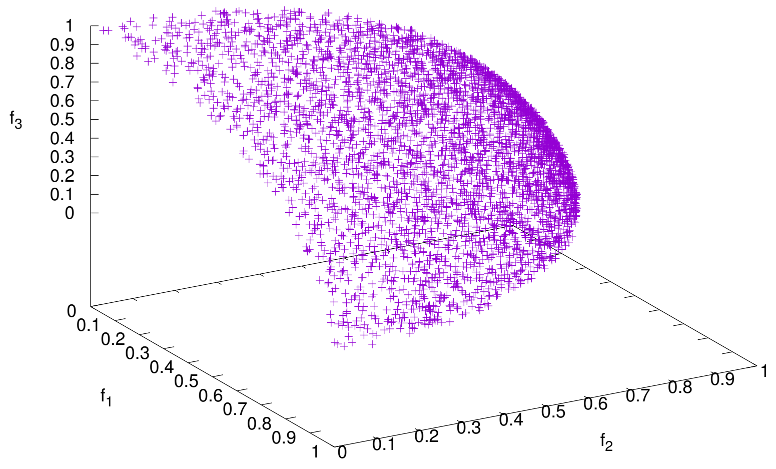

Figure A.8: Pareto optimal front of DTLZ3 with 3 objective functions.

The above function g introduces $(3^k - 1)$ local Pareto optimal fronts, and one global Pareto optimal front (see Figure A.8). All local Pareto optimal fronts are parallel to the global Pareto optimal front and an optimizer can get stuck at any of these local Pareto optimal fronts, before converging to the global Pareto optimal front at g = 0. The global Pareto optimal front corresponds to $\vec{y} = (0.5, 0.5, \dots)^T$. The next local Pareto optimal front is at g = 1.

A.2.4 DTLZ4

This problem is concave, separable and unimodal (see Figure A.9). It tests an optimizer's ability to maintain a good distribution of solutions, and is defined as follows:

Given

$$\vec{x} = \{x_1, \ldots, x_{m-1}, x_m, \ldots, x_n\}$$

Minimize

$$\begin{aligned}
f_1(\vec{x}) &= (1 + g(\vec{y})) \prod_{i=1}^{m-1} \cos\left(\frac{x_i^{\alpha}\pi}{2}\right) \\
f_{j=2:m-1}(\vec{x}) &= (1 + g(\vec{y})) \left(\prod_{i=1}^{m-j} \cos\left(\frac{x_i^{\alpha}\pi}{2}\right)\right) \sin\left(\frac{x_{m-j+1}^{\alpha}\pi}{2}\right) \\
f_m(\vec{x}) &= (1 + g(\vec{y})) \sin\left(\frac{x_1^{\alpha}\pi}{2}\right)
\end{aligned}$$

where

$$\begin{aligned}
y_{i=1:k} &= \{x_m, x_{m+1}, \ldots, x_n\} \\
g(\vec{y}) &= \sum_{i=1}^{k} (y_i - 0.5)^2
\end{aligned}$$

$$\text{subject to} \quad 0 \leq x_i \leq 1, \text{ for } i = 1, 2, \ldots, n. \tag{A.9}$$

This problem allows a dense set of solutions to exist near the $f_m - f_1$ plane. It is interesting to note, that although the search space has a variable density of solutions, the classical weighted- sum approaches or other directional methods may not have any added difficulty in solving this problem compared to DTLZ2.

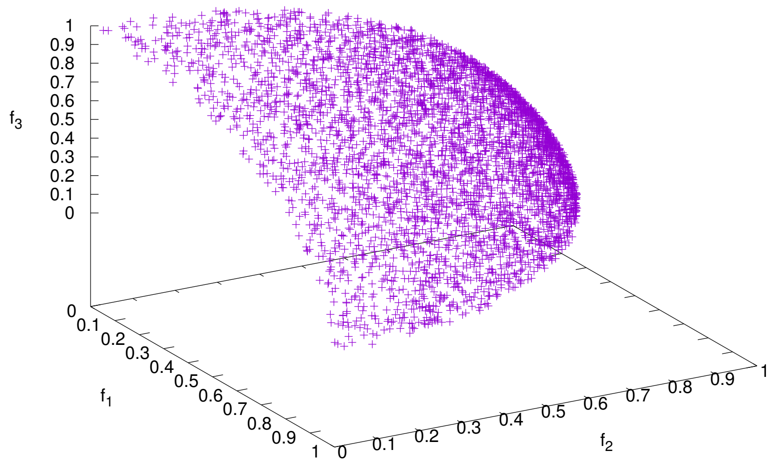

Figure A.9: Pareto optimal front of DTLZ4 with 3 objective functions.

A.2.5 DTLZ5

This problem is unimodal and degenerated (see Figure A.10). It is defined as:

$$
\begin{aligned}
&\text{Given } \vec{x} = \{x_1, \ldots, x_{m-1}, x_m, \ldots, x_n\} \\
&\text{Minimize} \\
f_1(\vec{x}) &= (1 + g(\vec{y})) \prod_{i=1}^{m-1} \cos\left(\frac{x_i \pi}{2}\right) \\
f_{j=2:m-1}(\vec{x}) &= (1 + g(\vec{y})) \left(\prod_{i=1}^{m-j} \cos\left(\frac{\theta_i \pi}{2}\right)\right) \sin\left(\frac{\theta_{m-j+1}\pi}{2}\right) \\
f_m(\vec{x}) &= (1 + g(\vec{y})) \sin\left(\frac{x_1 \pi}{2}\right) \\
\text{where } y_{i=1:k} &= \{x_m, x_{m+1}, \ldots, x_n\} \\
\theta_i &= \begin{cases} x_i, & i = 1 \\ \frac{1+2g(\vec{y})x_i}{2(1+g(\vec{y}))}, \text{for} & i = 2, \ldots, m-1 \end{cases} \\
g(\vec{y}) &= \sum_{i=1}^{k} (y_i - 0.5)^2 \\
&\text{subject to} \quad 0 \le x_i \le 1, \text{ for } i = 1, 2, \ldots, n.
\end{aligned}
\tag{A.10}
$$

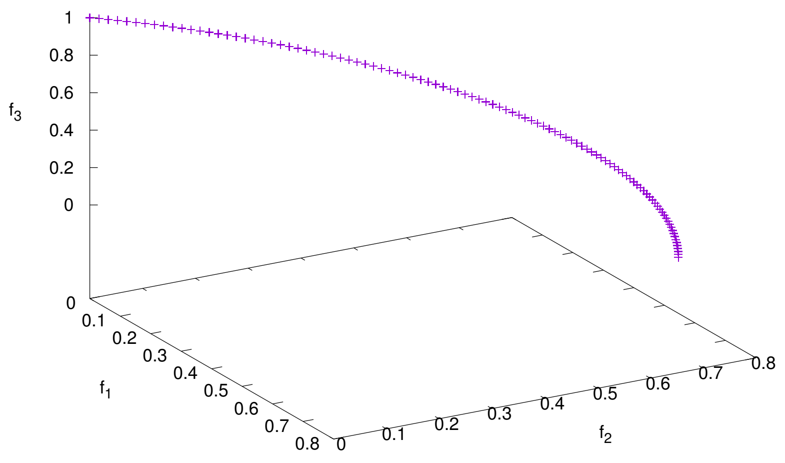

Figure A.10: Pareto optimal front of DTLZ5 with 3 objective functions.

This problem tests the optimizer's ability to converge to a curve. The Pareto optimal front corresponds to $\vec{y} = (0.5, 0.5, \dots)^T$, and all objective function values must satisfy $\sum_{i=1}^{m}(f_i)^2 = 1$.

A.2.6 DTLZ6

Modifying DTLZ5, a harder problem evolves by changing the function g. The resulting problem is unimodal and degenerated:

$$
\begin{aligned}
&\text{Given } \vec{x} = \{x_1, \ldots, x_{m-1}, x_m, \ldots, x_n\} \\
&\text{Minimize} \\
&f_1(\vec{x}) = (1 + g(\vec{y})) \prod_{i=1}^{m-1} \cos\left(\frac{x_i \pi}{2}\right) \\
&f_{j=2:m-1}(\vec{x}) = (1 + g(\vec{y})) \left(\prod_{i=1}^{m-j} \cos\left(\frac{\theta_i \pi}{2}\right)\right) \sin\left(\frac{\theta_{m-j+1}\pi}{2}\right) \\
&f_m(\vec{x}) = (1 + g(\vec{y})) \sin\left(\frac{x_1 \pi}{2}\right) \\
&\text{where } y_{i=1:k} = \{x_m, x_{m+1}, \ldots, x_n\} \\
&\theta_i = \begin{cases} x_i, & i = 1 \\ \frac{1+2g(\vec{y})x_i}{2(1+g(\vec{y}))}, \text{for } & i = 2, \ldots, m-1 \end{cases} \\
&g(\vec{y}) = \sum_{i=1}^{k} y_i^{0.1} \\
&\text{subject to} \quad 0 \le x_i \le 1, \ \text{for } i = 1, 2, \ldots, n.
\end{aligned}
\tag{A.11}
$$

The Pareto optimal front corresponds to $\vec{y} = (0, 0, \ldots)^T$ and is shown in Figure A.11. The lack of convergence to the Pareto optimal front in this problem makes optimizers to find a dominated surface as the obtained front, whereas the true Pareto optimal front is a curve. In real-world problems, this aspect may provide misleading information about the properties of the Pareto optimal front.

A.2.7 DTLZ7

This problem has a disconnected set of $2^{m?1}$ Pareto optimal regions in the search space and will test an algorithm?s ability to maintain subpopulations in different Pareto

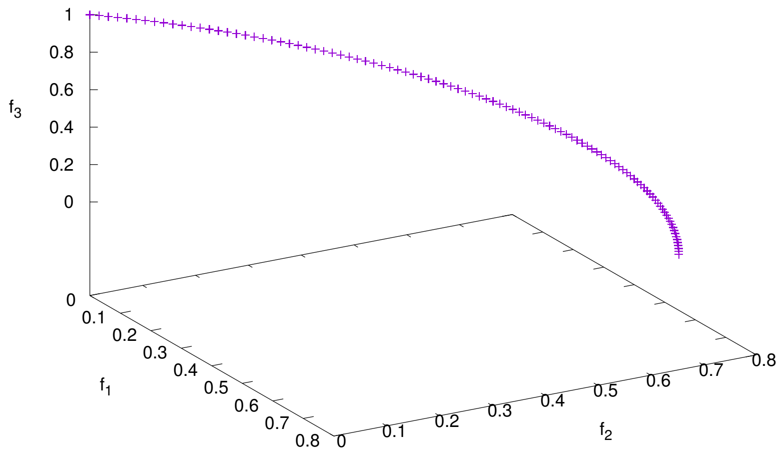

Figure A.11: Pareto optimal front of DTLZ6 with 3 objective functions.

optimal regions.

$$\text{Given } \vec{x} = \{x_1, \ldots, x_{m-1}, x_m, \ldots, x_n\}$$

$$\text{Minimize}$$

$$f_{j=1:m-1}(\vec{x}) = x_m$$

$$f_m(\vec{x}) = (1 + g(\vec{y}))(m - \sum_{i=1}^{m-1} \left[\frac{f_i}{1 + g(\vec{y})} (1 + \sin(3\pi f_i)) \right]) \tag{A.12}$$

$$\text{where } y_{i=1:k} = \{x_m, x_{m+1}, \ldots, x_n\}$$

$$g(\vec{y}) = 1 + \frac{9}{k} \sum_{i=1}^{k} y_i$$

$$\text{subject to} \quad 0 \leq x_i \leq 1, \text{ for } i = 1, 2, \ldots, n.$$

The Pareto optimal front corresponds to $\vec{y} = (0, 0, \ldots)^T$ and is shown in Figure A.12.

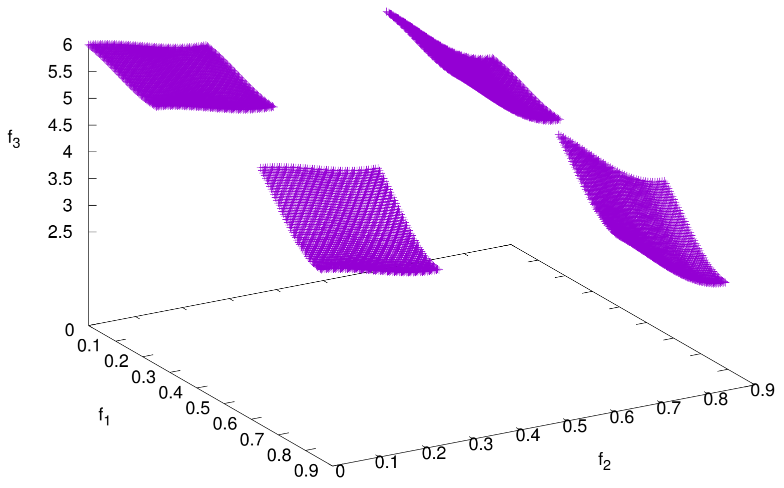

Figure A.12: Pareto optimal front of DTLZ7 with 3 objective functions.

A.3 Walking Fish Group Test Suite

The Walking-Fish-Group test suite [4], suggests nine multi-objective test problems (WFG1-WFG9), that are scalable with respect to both objectives and decision variables, and have known Pareto optimal sets. These problems include a wide variety of Pareto optimal geometries. Moreover, characteristics such as bias, multi-modality, and non-separability are defined by a set of transformations. Next, we present these benchmark problems. Here, m represents the number of objectives, and each problem is defined in terms of an underlying vector of parameters $\vec{x} \in \mathbb{R}^m$ that defines the fitness space. x_m is known as the underlying distance parameter, and $x_1 : m-1$ are the underlying position parameters. The vector $\vec{x}$ is derived, via a series of transition vectors, from a vector of working parameters $\vec{z} \in \mathbb{R}^n$ (also known as vector of variables). It is worth noting that $n \geq m$ and the number of decision variables is given by $n = k + l$. The first $k \in \{m-1, 2(m-1), 3(m-1), \dots\}$ working parameters are the position related parameters and the last $l \in \{1, 2, \dots\}$ working parameters are the distance related parameters. Each transition vector adds complexity to the underlying problem. The optimizer directly manipulates $\vec{z}$, through which $\vec{x}$ is indirectly manipulated.

A.3.1 WFG1

This problem is separable and unimodal, but it has a polynomial and flat region. It is strongly biased toward small values of the variables, which makes it very difficult for some optimizers. It is defined as follows:

$$\begin{aligned}
&\text{Given } \vec{z} = \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
\text{Minimize } f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}\left(1-\cos\left(\frac{x_i\pi}{2}\right)\right) \\
f_{j=2:m-1} &= x_m + 2j\left(\prod_{i=1}^{m-j}\left(1-\cos\left(\frac{x_i\pi}{2}\right)\right)\right)\left(1-\sin\left(\frac{x_{m-j+1}\pi}{2}\right)\right) \\
f_m(\vec{x}) &= x_m + 2m\left(1 - x_1 - \frac{\cos\left(\frac{10\pi x_1}{2}\right)}{10\pi}\right)
\end{aligned}$$

where

$$\begin{aligned}
x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \left\{\frac{2(i-1)k}{(m-1)}+1, \ldots, \frac{2ik}{(m-1)}\right\}\right) \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_n\}, \{2(k+1), \ldots, 2n\}\right) \\
y_{i=1:n} &= \text{b\_poly}(y'_i, 0.02) \\
y'_{i=1:k} &= y''_i \\
y'_{i=k+1:n} &= \text{b\_flat}(y''_i, 0.8, 0.75, 0.85) \\
y''_{i=1:k} &= \frac{z_i}{2i} \\
y''_{i=k+1:n} &= \text{s\_linear}\left(\frac{z_i}{2i}, 0.35\right) \\
\text{subject to} \quad & 0 \le x_i \le 1, \text{ for } i = 1, 2, \ldots, m. \\
& 0 \le z_i \le 2i, \text{ for } i = 1, 2, \ldots, n.
\end{aligned} \tag{A.13}$$

The Pareto optimal front is shown in Figure A.13.

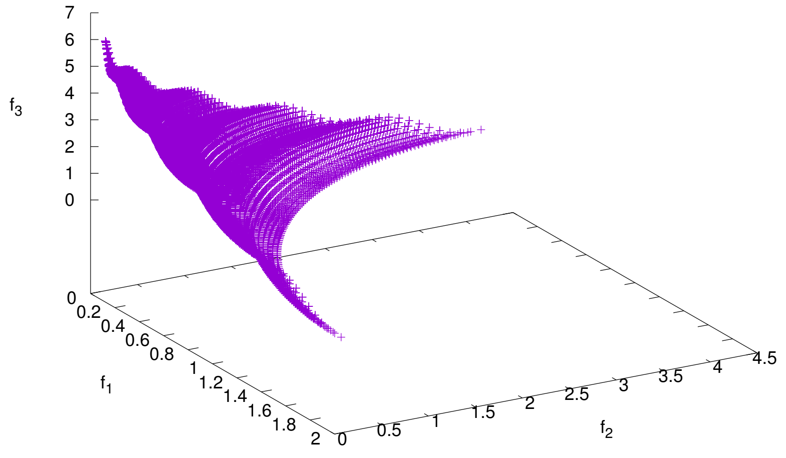

Figure A.13: Pareto optimal front of WFG1 with 3 objective functions.

A.3.2 WFG2

This problem is separable and multimodal. The Pareto optimal front is disconnected (see Figure A.14)

$$\begin{aligned}
\text{Given } \vec{z} &= \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
\text{Minimize } f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}\left(1-\cos\left(\frac{x_i\pi}{2}\right)\right) \\
f_{j=2:m-1} &= x_m + 2j\left(\prod_{i=1}^{m-j}\left(1-\cos\left(\frac{x_i\pi}{2}\right)\right)\right)\left(1-\sin\left(\frac{x_{m-j+1}\pi}{2}\right)\right) \\
f_m(\vec{x}) &= x_m + 2m\left(1 - x_1\cos^2(5x_1\pi)\right)
\end{aligned}$$

where

$$\begin{aligned}
x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \{1, \ldots, 1\}\right) \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_{k+l/2}\}, \{1, \ldots, 1\}\right) \\
y'_{i=1:k} &= y'_i \\
y'_{i=k+1:k+l/2} &= \text{r\_nonsep}(\{y'_{k+2(i-k)-1}, y'_{k+2(i-k)}\}, 2) \\
y'_{i=1:k} &= \frac{z_i}{2i} \\
y'_{i=k+1:n} &= \text{s\_linear}\left(\frac{z_i}{2i}, 0.35\right) \\
\text{subject to} \quad & 0 \le x_i \le 1, \text{ for } i = 1, 2, \ldots, m. \\
& 0 \le z_i \le 2i, \text{ for } i = 1, 2, \ldots, n.
\end{aligned} \tag{A.14}$$

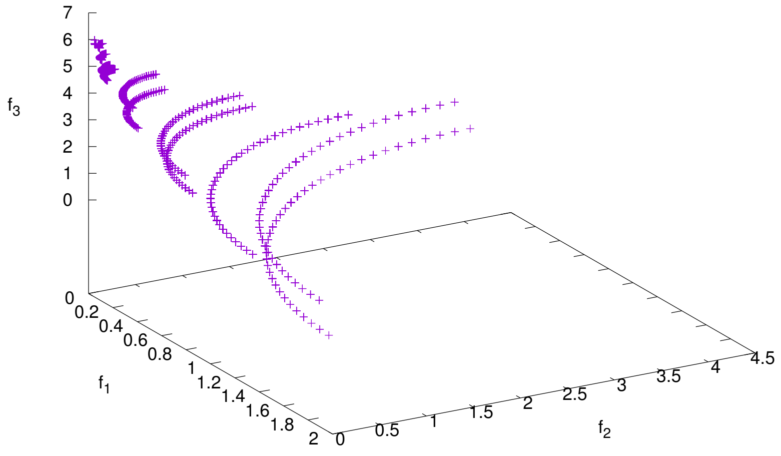

Figure A.14: Pareto optimal front of WFG2 with 3 objective functions.

A.3.3 WFG3

This problem is nonseparable but unimodal. It has a linear and degenerated Pareto optimal front (see Figure A.15), which is given by the following expression:

$$
\begin{aligned}
\text{Given } \vec{z} &= \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
\text{Minimize } f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}(x_i) \\
f_{j=2:m-1} &= x_m + 2j\left(\prod_{i=1}^{m-j} x_i\right)(1 - x_{m-j+1}) \\
f_m(\vec{x}) &= x_m + 2m(1 - x_1) \\
\text{where } x_{i=1} &= u_i \\
x_{i=2:m-1} &= x_m(u_i - 0.5) + 0.5 \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_{k+l/2}\}, \{1, \ldots, 1\}\right) \\
u_i &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \{1, \ldots, 1\}\right) \\
y'_{i=1:k} &= y'_i \\
y'_{i=k+1:k+l/2} &= \text{r\_nonsep}(\{y'_{k+2(i-k)-1}, y'_{k+2(i-k)}\}, 2) \\
y'_{i=1:k} &= \frac{z_i}{2i} \\
y'_{i=k+1:n} &= \text{s\_linear}\left(\frac{z_i}{2i}, 0.35\right) \\
\text{subject to } \quad & 0 \le x_i \le 1, \text{ for } i = 1, 2, \ldots, m. \\
& 0 \le z_i \le 2i, \text{ for } i = 1, 2, \ldots, n.
\end{aligned}
\tag{A.15}
$$

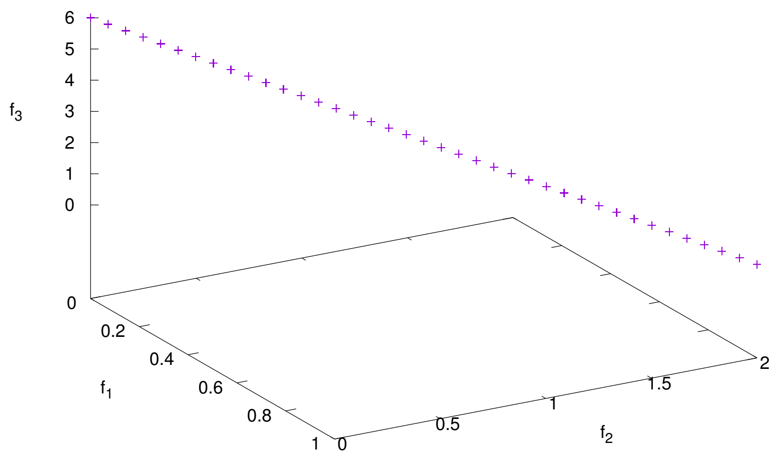

Figure A.15: Pareto optimal front of WFG3 with 3 objective functions.

A.3.4 WFG4

In this case, the problem is separable, but highly multimodal. The Pareto optimal front is concave (see Figure A.16) and is defined as follows:

$$
\begin{aligned}
\text{Given } \vec{z} &= \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
\text{Minimize } f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}(\sin(x_i\pi/2)) \\
f_{j=2:m-1} &= x_m + 2j\left(\prod_{i=1}^{m-j}\sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
f_m(\vec{x}) &= x_m + 2m\cos(x_1\pi/2) \\
\text{where } x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \{1, \ldots, 1\}\right) \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_n\}, \{1, \ldots, 1\}\right) \\
y_{i=1:n} &= \text{s\_multi}\left(z_i/2i, 30, 10, 0.35\right) \\
\text{subject to} \quad & 0 \leq x_i \leq 1, \text{ for } i = 1, 2, \ldots, m. \\
& 0 \leq z_i \leq 2i, \text{ for } i = 1, 2, \ldots, n.
\end{aligned}
\tag{A.16}
$$

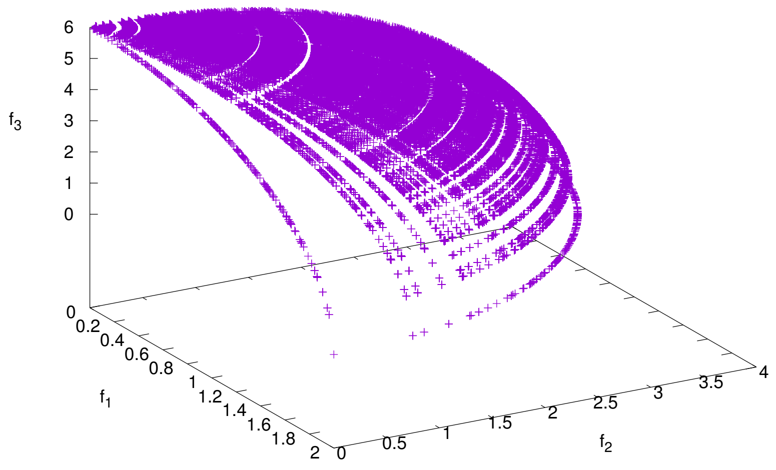

Figure A.16: Pareto optimal front of WFG4 with 3 objective functions.

A.3.5 WFG5

A deceptive and separable problem. The Pareto optimal front is concave (see Figure A.17) and is defined by the following expression:

$$
\begin{aligned}
\text{Given } \vec{z} &= \{z_1, \dots, z_k, z_{k+1}, \dots, z_n\} \\
\text{Minimize } f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}(\sin(x_i\pi/2)) \\
f_{j=2:m-1} &= x_m + 2j\left(\prod_{i=1}^{m-j}\sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
f_m(\vec{x}) &= x_m + 2m\cos(x_1\pi/2) \\
\text{where } x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \dots, y_{ik/(m-1)}\}, \{1, \dots, 1\}\right) \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \dots, y_n\}, \{1, \dots, 1\}\right) \\
y_{i=1:n} &= \text{s\_decept}\left(z_i/2i, 0.35, 0.001, 0.05\right) \\
\text{subject to} \quad & 0 \le x_i \le 1, \text{ for } i = 1, 2, \dots, m. \\
& 0 \le z_i \le 2i, \text{ for } i = 1, 2, \dots, n.
\end{aligned}
\tag{A.17}
$$

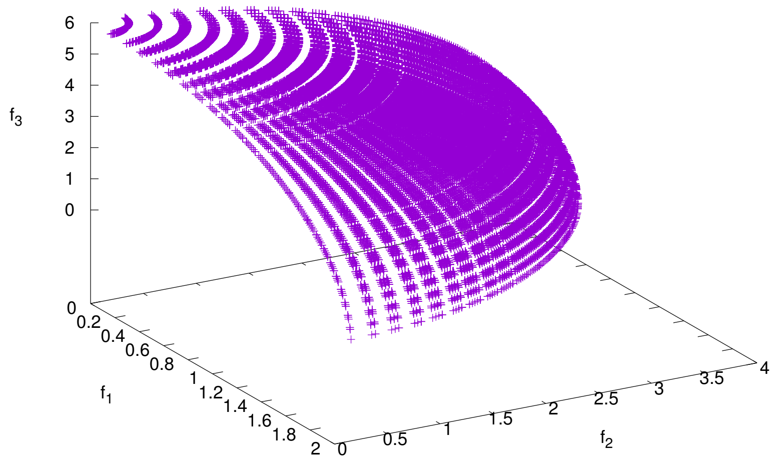

Figure A.17: Pareto optimal front of WFG5 with 3 objective functions.

A.3.6 WFG6

This problem is nonseparable and unimodal. Its Pareto optimal front is concave (see Figure A.18), and is defined as follows:

$$\begin{aligned}
\text{Given } \vec{z} &= \{z_1, \dots, z_k, z_{k+1}, \dots, z_n\} \\
\text{Minimize } f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}(\sin(x_i\pi/2)) \\
f_{j=2:m-1} &= x_m + 2j\left(\prod_{i=1}^{m-j}\sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
f_m(\vec{x}) &= x_m + 2m\cos(x_1\pi/2) \\
\text{where } x_{i=1:m-1} &= \text{r\_nonsep}\left(\{y_{(i-1)k/(m-1)+1}, \dots, y_{ik/(m-1)}\}, \{1, \dots, 1\}\right) \qquad \text{(A.18)} \\
x_m &= \text{r\_nonsep}\left(\{y_{k+1}, \dots, y_n\}, \{1, \dots, 1\}\right) \\
y_{i=1:k} &= z_i/2i \\
y_{i=k+1:n} &= \text{s\_linear}\left(z_i/2i, 0.35\right) \\
\text{subject to} \quad & 0 \le x_i \le 1, \ \text{for } i = 1, 2, \dots, m. \\
& 0 \le z_i \le 2i, \ \text{for } i = 1, 2, \dots, n.
\end{aligned}$$

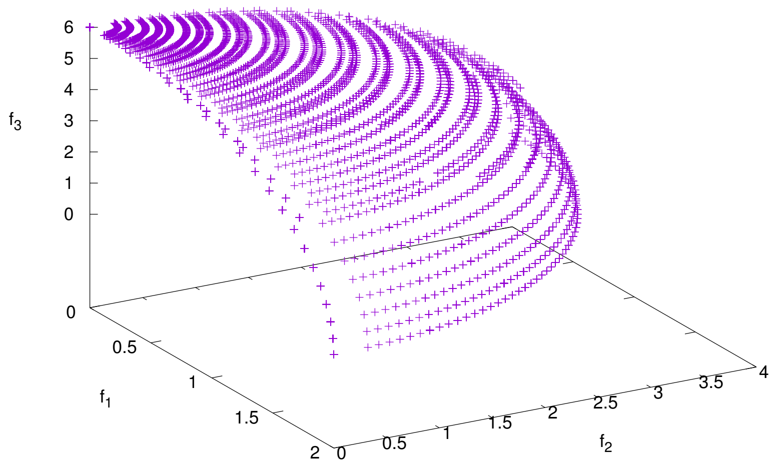

Figure A.18: Pareto optimal front of WFG6 with 3 objective functions.

A.3.7 WFG7

Having a parameter dependent bias, this problem is also separable and unimodal. The concave Pareto optimal front is depicted in Figure A.19, and is defined as:

$$
\begin{aligned}
\text{Given } \vec{z} &= \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
\text{Minimize } f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}(\sin(x_i\pi/2)) \\
f_{j=2:m-1} &= x_m + 2j\left(\prod_{i=1}^{m-j}\sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
f_m(\vec{x}) &= x_m + 2m\cos(x_1\pi/2) \\
\text{where } x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \{1, \ldots, 1\}\right) \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_n\}, \{1, \ldots, 1\}\right) \\
y_{i=1:k} &= y'_i \\
y_{i=k+1:n} &= \text{s\_linear}\left(y'_i, 0.35\right) \\
y'_{i=1:k} &= \text{b\_param}(\{z_i/2i, \text{r\_sum}\left(\{z_{i+1}/(2(i+1)), \ldots, z_n/2n\}, \{1, \ldots, 1\}\right), \\
&\quad 0.98/49.98, 0.02, 50) \\
y'_{i=k+1:n} &= z_i/2i \\
\text{subject to } \quad & 0 \leq x_i \leq 1, \text{ for } i = 1, 2, \ldots, m. \\
& 0 \leq z_i \leq 2i, \text{ for } i = 1, 2, \ldots, n.
\end{aligned}
\tag{A.19}
$$

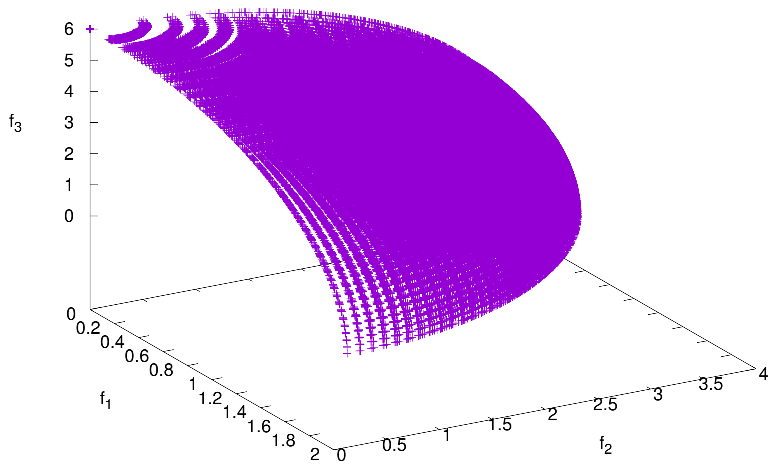

Figure A.19: Pareto optimal front of WFG7 with 3 objective functions.

A.3.8 WFG8

This problem has a parameter dependent bias, but is nonseparable and unimodal. The concave Pareto optimal front (see Figure A.20) is given by the expression:

$$
\begin{aligned}
\text{Given } \vec{z} &= \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
\text{Minimize } f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}(\sin(x_i\pi/2)) \\
f_{j=2:m-1} &= x_m + 2j\left(\prod_{i=1}^{m-j}\sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
f_m(\vec{x}) &= x_m + 2m\cos(x_1\pi/2) \\
\text{where } x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \{1, \ldots, 1\}\right) \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_n\}, \{1, \ldots, 1\}\right) \\
y_{i=1:k} &= y'_i \\
y_{i=k+1:n} &= \text{s\_linear}\left(y'_i, 0.35\right) \\
y'_{i=k} &= z_i/2i \\
y'_{i=k+1:n} &= \text{b\_param}(\{z_i/2i, \text{r\_sum}\left(\{z_1/2, \ldots, z_{i-1}/2(i-1)\}, \{1, \ldots, 1\}\right), \\
&\quad 0.98/49.98, 0.02, 50) \\
\text{subject to} \quad & 0 \le x_i \le 1, \text{ for } i = 1, 2, \ldots, m. \\
& 0 \le z_i \le 2i, \text{ for } i = 1, 2, \ldots, n.
\end{aligned}
\tag{A.20}
$$

A.3.9 WFG9

The last problem of the suite is nonseparable, multimodal, deceptive, and has a parameter dependent bias. All these features make it a very difficult problem. This

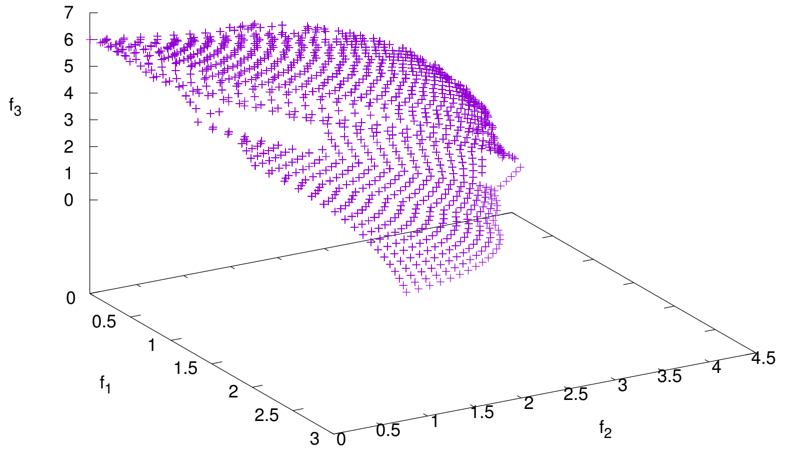

Figure A.20: Pareto optimal front of WFG8 with 3 objective functions.

problem has a concave Pareto optimal front (see Figure A.21) and is defined as:

$$
\begin{aligned}
\text{Given } \vec{z} &= \{z_1,\dots,z_k,z_{k+1},\dots,z_n\}\\
\text{Minimize } f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}(\sin(x_i\pi/2))\\
f_{j=2:m-1} &= x_m + 2j\left(\prod_{i=1}^{m-j}\sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2)\\
f_m(\vec{x}) &= x_m + 2m\cos(x_1\pi/2)\\
\text{where } x_{i=1:m-1} &= \text{r\_nonsep}\left(\{y_{(i-1)k/(m-1)+1},\dots,y_{ik/(m-1)}\},\{1,\dots,1\}\right)\\
x_m &= \text{r\_nonsep}\left(\{y_{k+1},\dots,y_n\},\{1,\dots,1\}\right)\\
y_{i=1:k} &= \text{s\_decept}(y'_i,0.35,0.001,0.05)\\
y_{i=k+1:n} &= \text{s\_multi}(y'_i,30,95,0.35)\\
y'_{i=1:n-1} &= \text{b\_param}(\{z_i/2i,\text{r\_sum}\left(\{z_{i+1}/(2(i+1)),\dots,z_n/(2n)\},\{1,\dots,1\}\right),\\
&\quad 0.98/49.98,0.02,50)\\
y'_{i=n} &= z_n/2n\\
\text{subject to } &\quad 0\le x_i\le 1,\ \text{for } i=1,2,\dots,m.\\
&\quad 0\le z_i\le 2i,\ \text{for } i=1,2,\dots,n.
\end{aligned}
\tag{A.21}
$$

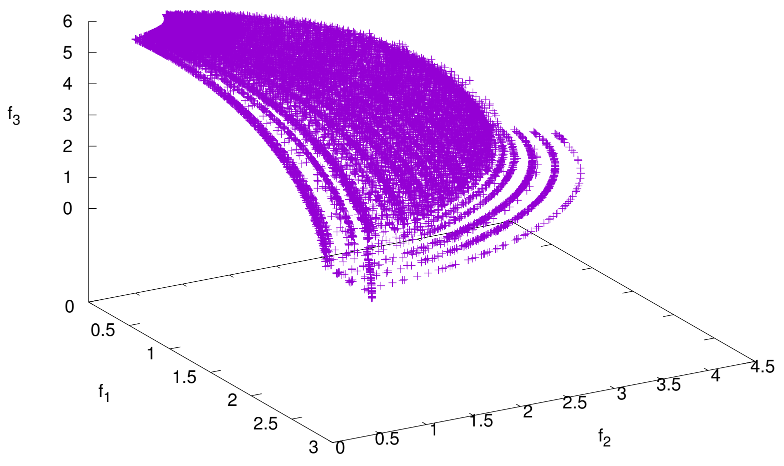

Figure A.21: Pareto optimal front of WFG9 with 3 objective functions.

Appendix B

Sensitivity Analysis of Parameters

Next we study the sensitivity that MOEA/D^2 has over the following parameters: the number of species, the neighborhood size and the populations size.

To study how MOEA/D^2 is sensitive to the above parameters, multiple data for each parameter have been tried. Four values of the neighborhood: 5, 10, 15 and 20. Four values of the number of species: 5, 10, 25 and 50. Four values of the population size: 50, 100, 150 and 200. We analyze this values when solving ZDT1 test problem with 100 decision variables. 30 independent runs have been conducted for each configuration on the MOP, this restricted by the computational time of each of the experiments. Therefore, as the data of 30 independent executions is clearly not Normal, we adopted Bootstrapping [136] with 1000 resamplings from the 30 executions of each combination from which we obtained: the hypervolume mean, the standard error, the bias, and the confidence intervals of 95% for each sample. Table B.1 show the obtained results.

Niche	Species	PopSize	MaxGen	Mean	Bias	SE	CI
5	5	50	400	3.120546	0.000480	0.027549	(3.069,3.176)
5	5	100	200	2.937600	-0.000235	0.025049	(2.886,2.986)
5	5	150	133	2.772640	0.000516	0.018781	(2.737,2.811)
5	5	200	100	2.667199	-0.000443	0.017679	(2.633,2.702)
5	10	50	400	3.193081	0.000494	0.018466	(3.158,3.231)
5	10	100	200	3.097153	-0.000044	0.017172	(3.060,3.128)

5	10	150	133	2.927517	-0.000305	0.017996	(2.892,2.962)
5	10	200	100	2.820841	-0.000416	0.017682	(2.790,2.859)
5	25	50	400	3.285972	-0.000575	0.021485	(3.236,3.323)
5	25	100	200	3.167240	-0.000277	0.015155	(3.141,3.198)
5	25	150	133	3.025194	0.000261	0.015874	(2.994,3.057)
5	25	200	100	2.916356	0.000009	0.017005	(2.883,2.949)
5	50	50	400	3.294808	0.000497	0.019255	(3.256,3.329)
5	50	100	200	3.191176	-0.000360	0.017123	(3.152,3.219)
5	50	150	133	3.033892	0.000273	0.012931	(3.008,3.058)
5	50	200	100	2.918768	0.000355	0.016551	(2.886,2.951)
10	5	50	400	3.210825	-0.000121	0.025451	(3.158,3.259)
10	5	100	200	3.105953	0.000982	0.031779	(3.043,3.165)
10	5	150	133	2.969539	0.000127	0.028132	(2.905,3.015)
10	5	200	100	2.869539	0.000764	0.022936	(2.829,2.917)
10	10	50	400	3.228278	0.001742	0.024554	(3.177,3.271)
10	10	100	200	3.204958	0.000116	0.017138	(3.172,3.241)
10	10	150	133	3.132529	0.000551	0.023225	(3.084,3.175)
10	10	200	100	3.011906	0.000887	0.019007	(2.975,3.048)
10	25	50	400	3.281102	0.000290	0.026095	(3.218,3.325)
10	25	100	200	3.232674	0.000651	0.017708	(3.195,3.266)
10	25	150	133	3.187071	-0.000841	0.025903	(3.124,3.227)
10	25	200	100	3.130193	-0.000311	0.018740	(3.093,3.167)
10	50	50	400	3.263799	0.000295	0.020146	(3.223,3.303)
10	50	100	200	3.265180	0.000184	0.021240	(3.222,3.304)
10	50	150	133	3.230954	0.000202	0.016822	(3.194,3.259)
10	50	200	100	3.174513	0.001003	0.017887	(3.137,3.208)
15	5	50	400	3.238675	0.000437	0.020173	(3.192,3.272)
15	5	100	200	3.122657	0.001129	0.026264	(3.057,3.167)
15	5	150	133	3.061123	0.000474	0.022979	(3.010,3.103)
15	5	200	100	3.029161	-0.000502	0.020045	(2.990,3.072)
15	10	50	400	3.284053	0.000988	0.020370	(3.248,3.326)
15	10	100	200	3.213139	0.000906	0.026120	(3.156,3.258)
15	10	150	133	3.181154	-0.000627	0.019448	(3.142,3.217)

15	10	200	100	3.099957	-0.001031	0.028583	(3.041,3.155)
15	25	50	400	3.312518	0.000151	0.018904	(3.273,3.350)
15	25	100	200	3.290444	0.000909	0.017617	(3.254,3.324)
15	25	150	133	3.235889	0.000251	0.018752	(3.196,3.271)
15	25	200	100	3.217832	0.000094	0.020774	(3.176,3.259)
15	50	50	400	3.277960	-0.000393	0.023350	(3.219,3.317)
15	50	100	200	3.285106	-0.000743	0.015696	(3.254,3.314)
15	50	150	133	3.284448	-0.000295	0.022088	(3.239,3.325)
15	50	200	100	3.250647	-0.000215	0.020404	(3.202,3.285)
20	5	50	400	3.284726	-0.000201	0.018326	(3.246,3.318)
20	5	100	200	3.215011	-0.000136	0.025099	(3.165,3.263)
20	5	150	133	3.154259	0.000164	0.017647	(3.121,3.190)
20	5	200	100	3.097857	-0.000619	0.016838	(3.065,3.130)
20	10	50	400	3.305992	-0.000119	0.022581	(3.252,3.343)
20	10	100	200	3.233349	-0.000060	0.018517	(3.194,3.267)
20	10	150	133	3.203719	-0.000435	0.020129	(3.166,3.244)
20	10	200	100	3.121219	0.000934	0.028804	(3.066,3.179)
20	25	50	400	3.324003	-0.000028	0.016223	(3.289,3.353)
20	25	100	200	3.276982	-0.001890	0.017912	(3.242,3.313)
20	25	150	133	3.292737	0.000306	0.017269	(3.257,3.325)
20	25	200	100	3.213864	0.000507	0.020900	(3.169,3.252)
20	50	50	400	3.331659	0.000021	0.019033	(3.290,3.367)
20	50	100	200	3.306242	0.000021	0.025083	(3.248,3.349)
20	50	150	133	3.294139	-0.000252	0.017128	(3.264,3.331)
20	50	200	100	3.267183	0.000335	0.017770	(3.230,3.300)

Table B.1: Results for the sensitivity analysis of MOEA/^{2} parameters.

Bibliography

[1] W.J. Cook. *Mathematical Programming Computation*. Springer, mathematical optimization society edition, 2009.

[2] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. Technical Report 70, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, December 1999.

[3] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable test problems for evolutionary multi-objective optimization. Technical report, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), 2001.

[4] Simon Huband, Phil Hingston, Luigi Barone, and Lyndon While. A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, October 2006.

[5] Juan J. Durillo, Antonio J. Nebro, Carlos A. Coello Coello, Francisco Luna, and Enrique Alba. A Comparative Study of the Effect of Parameter Scalability in Multi-Objective Metaheuristics. In *2008 Congress on Evolutionary Computation (CEC'2008)*, pages 1893–1900, Hong Kong, June 2008. IEEE Service Center.

[6] J.J. Durillo, A.J. Nebro, C.A. Coello Coello, J. Garcia-Nieto, F. Luna, and E. Alba. A Study of Multiobjective Metaheuristics When Solving Parameter Scalable Problems. *IEEE Transactions on Evolutionary Computation*, 14(4):618–635, August 2010.

[7] Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Springer, New York, second edition, September 2007.

[8] Vilfredo Pareto. *Cours D'Economie Politique*, volume I and II. F. Rouge, Lausanne, 1896.

[9] Jared L Cohon and David H Marks. A review and evaluation of multiobjective programing techniques. *Water Resources Research*, 11(2):208–220, 1975.

[10] A. Charnes, W. W. Cooper, and R. O. Ferguson. Optimal estimation of executive compensation by linear programming. *Management Science*, 1(2):138–151, 1955.

[11] Abraham Charnes and William W. Cooper. *Management Models and Industrial Applications of Linear Programming*, volume 1. John Wiley & Sons Inc, New York, December 1961.

[12] Kaisa M. Miettinen. *Nonlinear Multiobjective Optimization.* Kluwer Academic Publishers, Boston, Massachusetts, 1999.

[13] N Arunkumar, L Karunamoorthy, S Anand, and T Ramesh Babu. Linear approach for solving a piecewise linear vendor selection problem of quantity discounts using lexicographic method. *The International Journal of Advanced Manufacturing Technology*, 28(11-12):1254–1260, 2006.

[14] Saul Gass and Thomas Saaty. The computational algorithm for the parametric objective function. *Naval Research Logistics Quarterly*, 2(1-2):39–45, 1955.

[15] L. Zadeh. Optimality and non-scalar-valued performance criteria. *IEEE Transactions on Automatic Control*, 8(1):59–60, Jan 1963.

[16] D.A. Wismer Y.Y. Haimes, L.S. Lasdon. On a bicriterion formulation of the problems of integrated system identification and system optimization. *IEEE Transactions on Systems, Man and Cybernetics*, 1:296–297, 1971.

[17] M. Zeleny. Compromise programming. In J. Cochrane and M. Zeleny, editors, *Multiple Criteria Decision Making*, pages 262–301. University of South Carolina Press, Columbia, 1973.

[18] Jr. Bowman, V.Joseph. On the relationship of the tchebycheff norm and the efficient frontier of multiple-criteria objectives. In Herv Thiriez and Stanley Zionts, editors, *Multiple Criteria Decision Making*, volume 130 of *Lecture Notes in Economics and Mathematical Systems*, pages 76–86. Springer Berlin Heidelberg, 1976.

[19] RalphE. Steuer. The tchebycheff procedure of interactive multiple objective programming. In Birsen Karpak and Stanley Zionts, editors, *Multiple Criteria Decision Making and Risk Analysis Using Microcomputers*, volume 56 of *NATO ASI Series*, pages 235–249. Springer Berlin Heidelberg, 1989.

[20] J.T. Buchanan. A naïve approach for solving mcdm problems: The guess method. *Journal of the Operational Research Society*, 48(2):202–206, 1997.

[21] Richard Rosenberg. *Simulation of genetic populations with biochemical properties*. PhD thesis, Department of Communication Sciences, University of Michigan, Ann Arbor, Michigan, USA, June 1967.

[22] John David Schaffer. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. PhD thesis, Vanderbilt University, Nashville, Tennessee, USA, 1984.

[23] Carlos A. Coello Coello and Gregorio Toscano Pulido. A Micro-Genetic Algorithm for Multiobjective Optimization. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 126–140. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[24] N. Srinivas and Kalyanmoy Deb. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221–248, Fall 1994.

[25] Eckart Zitzler and Lothar Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, November 1999.

[26] Jeffrey Horn and Nicholas Nafpliotis. Multiobjective Optimization using the Niched Pareto Genetic Algorithm. Technical Report IlliGAl Report 93005, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1993.

[27] Carlos M. Fonseca and Peter J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.

[28] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In K. Giannakoglou, D. Tsahalis, J. Periaux, P. Papailou, and T. Fogarty, editors, *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pages 95–100, Athens, Greece, 2002.

[29] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[30] Qingfu Zhang and Hui Li. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, December 2007.

[31] Michael Emmerich, Nicola Beume, and Boris Naujoks. An EMO Algorithm Using the Hypervolume Measure as Selection Criterion. In Carlos A. Coello Coello, Arturo Hernández Aguirre, and Eckart Zitzler, editors, *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*, pages 62–76, Guanajuato, México, March 2005. Springer. Lecture Notes in Computer Science Vol. 3410.

[32] Joshua Knowles and David Corne. Properties of an Adaptive Archiving Algorithm for Storing Nondominated Vectors. *IEEE Transactions on Evolutionary Computation*, 7(2):100–116, April 2003.

[33] Nicola Beume, Boris Naujoks, and Michael Emmerich. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, 16 September 2007.

[34] Christian Igel, Nikolaus Hansen, and Stefan Roth. Covariance Matrix Adaptation for Multi-objective Optimization. *Evolutionary Computation*, 15(1):1–28, Spring 2007.

[35] Sanaz Mostaghim, Jürgen Branke, and Hartmut Schmeck. Multi-Objective Particle Swarm Optimization on Computer Grids. In Dirk Thierens, editor, *2007 Genetic and Evolutionary Computation Conference (GECCO'2007)*, volume 1, pages 869–875, London, UK, July 2007. ACM Press.

[36] Heike Trautmann, Tobias Wagner, and Dimo Brockhoff. R2-emoa: Focused multiobjective search using r2-indicator-based selection. In Giuseppe Nicosia and Panos Pardalos, editors, *Learning and Intelligent Optimization*, Lecture Notes in Computer Science, pages 70–74. Springer Berlin Heidelberg, 2013.

[37] A Diaz-Manriquez, G. Toscano-Pulido, C.AC. Coello, and R. Landa-Becerra. A ranking method based on the r2 indicator for many-objective optimization. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 1523–1530, June 2013.

[38] D.H. Phan and J. Suzuki. R2-ibea: R2 indicator based evolutionary algorithm for multiobjective optimization. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 1836–1845, June 2013.

[39] K. Gerstl, G. Rudolph, O. Schutze, and H. Trautmann. Finding evenly spaced fronts for multiobjective control via averaging hausdorff-measure. In *Electrical Engineering Computing Science and Automatic Control (CCE), 2011 8th International Conference on*, pages 1–6, Oct 2011.

[40] Heike Trautmann, Günter Rudolph, Christian Dominguez-Medina, and Oliver Schütze. Finding Evenly Spaced Pareto Fronts for Three-Objective Optimization Problems. In Oliver Schütze, Carlos A. Coello Coello, Alexandru-Adrian Tantar, Emilia Tantar, Pascal Bouvry, Pierre Del Moral, and Pierrick Legrand, editors, *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation II*, pages 89–105. Springer, Advances in Intelligent Systems and Computing Vol. 175, Berlin, Germany, 2012. ISBN 978-3-642-31519-0.

[41] Cynthia A. Rodríguez Villalobos and Carlos A. Coello Coello. A New Multi-Objective Evolutionary Algorithm Based on a Performance Assessment Indicator. In *2012 Genetic and Evolutionary Computation Conference (GECCO'2012)*, pages 505–512, Philadelphia, USA, July 2012. ACM Press. ISBN: 978-1-4503-1177-9.

[42] Carlos A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Kluwer Academic Publishers, New York, May 2002. ISBN 0-3064-6762-3.

[43] David E. Goldberg and Jon Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*, pages 41–49, Hillsdale, NJ, USA, 1987. L. Erlbaum Associates Inc.

[44] Kalyanmoy Deb and David E. Goldberg. An investigation of niche and species formation in genetic function optimization. In *Proceedings of the 3rd International Conference on Genetic Algorithms*, pages 42–50, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[45] Marco Laumanns, Lothar Thiele, Kalyanmoy Deb, and Eckart Zitzler. Combining Convergence and Diversity in Evolutionary Multi-objective Optimization. *Evolutionary Computation*, 10(3):263–282, Fall 2002.

[46] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.

[47] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M. Fonseca, and Viviane Grunert da Fonseca. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, April 2003.

[48] Joshua D. Knowles, David W. Corne, and Mark Fleischer. Bounded Archiving using the Lebesgue Measure. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, volume 4, pages 2490–2497, Canberra, Australia, December 2003. IEEE Press.

[49] 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. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601, Aug 2014.

[50] David A. Van Veldhuizen and Gary B. Lamont. On Measuring Multiobjective Evolutionary Algorithm Performance. In *2000 Congress on Evolutionary Computation*, volume 1, pages 204–211, Piscataway, New Jersey, July 2000. IEEE Service Center.

[51] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. KanGAL report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[52] Eckart Zitzler and Lothar Thiele. Multiobjective Optimization Using Evolutionary Algorithms—A Comparative Study. In A. E. Eiben, editor, *Parallel Problem Solving from Nature V*, pages 292–301, Amsterdam, September 1998. Springer-Verlag.

[53] Viviane Grunert da Fonseca, Carlos M. Fonseca, and Andreia O. Hall. Inferential performance assessment of stochastic optimisers and the attainment function. In *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 213–225. Springer-Verlag, 2001.

[54] P. Czyzak and A. Jaszkiewicz. Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 7:34–47, 1998.

[55] Joshua Knowles, Lothar Thiele, and Eckart Zitzler. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland, feb 2006. revised version.

[56] David A. Van Veldhuizen and Gary B. Lamont. Multiobjective Evolutionary Algorithm Test Suites. In Janice Carroll, Hisham Haddad, Dave Oppenheim, Barrett Bryant, and Gary B. Lamont, editors, *Proceedings of the 1999 ACM*

Symposium on Applied Computing, pages 351–357, San Antonio, Texas, 1999. ACM.

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

[58] David A. Van Veldhuizen and Gary B. Lamont. Evolutionary Computation and Convergence to a Pareto Front. In John R. Koza, editor, *Late Breaking Papers at the Genetic Programming 1998 Conference*, pages 221–228, Stanford University, California, July 1998. Stanford University Bookstore.

[59] Carlos A. Coello Coello and Nareli Cruz Cortés. Solving Multiobjective Optimization Problems using an Artificial Immune System. *Genetic Programming and Evolvable Machines*, 6(2):163–190, June 2005.

[60] Jason R. Schott. Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1995.

[61] M. P. Hansen and A. Jaszkiewicz. Evaluating the quality of approximations to the non-dominated set. Technical Report Technical Report IMM-REP-1998-7, Technical University of Denmark, March 1998.

[62] Dimo Brockhoff, Tobias Wagner, and Heike Trautmann. On the Properties of the $R2$ Indicator. In *2012 Genetic and Evolutionary Computation Conference (GECCO'2012)*, pages 465–472, Philadelphia, USA, July 2012. ACM Press. ISBN: 978-1-4503-1177-9.

[63] Oliver Schütze, Xavier Esquivel, Adriana Lara, and Carlos A. Coello Coello. Using the Averaged Hausdorff Distance as a Performance Measure in

Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 16(4):504–522, August 2012.

[64] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, Summer 2000.

[65] Thomas Weise, Raymond Chiong, and Ke Tang. Evolutionary optimization: Pitfalls and booby traps. *Journal of Computer Science and Technology*, 27(5):907–936, 2012.

[66] Zhenyu Yang, Ke Tang, and Xin Yao. Large scale evolutionary optimization using cooperative coevolution. *Inf. Sci.*, 178(15):2985–2999, August 2008.

[67] Mohammad Nabi Omidvar, Xiaodong Li, and Ke Tang. Designing benchmark problems for large-scale continuous optimization. *Inf. Sci.*, 316(C):419–436, September 2015.

[68] Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. Evolutionary many-objective optimization: A short review. In *2008 IEEE Congress on Evolutionary Computation CEC'2008 (IEEE World Congress on Computational Intelligence)*, pages 2424–2431, Hong Kong, June 2008.

[69] José A. Molinet Berenguer and Carlos A. Coello Coello. Evolutionary Many-objective Optimization based on Kuhn-Munkres' Algorithm. In Antonio Gaspar Cunha et al., editor, *Evolutionary Multi-Criterion Optimization, 8th International Conference, EMO 2015*, Porto, Portugal, 29 March - 1 April 2015. Springer.

[70] Eckart Zitzler, Dimo Brockhoff, and Lothar Thiele. The Hypervolume Indicator Revisited: On the Design of Pareto-compliant Indicator Via Weighted Integration. In Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni,

Tomoyuki Hiroyasu, and Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*, pages 862–876, Matshushima, Japan, March 2007. Springer. Lecture Notes in Computer Science Vol. 4403.

[71] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. Technical Report 139, Computer Engineering and Networks Laboratory, ETH Zurich, June 2002.

[72] Luis Miguel Antonio and Carlos A. Coello Coello. Use of Cooperative Coevolution for Solving Large Scale Multiobjective Optimization Problems. In *2013 IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 2758–2765, Cancún, México, 20-23 June 2013. IEEE Press. ISBN 978-1-4799-0454-9.

[73] Mitchell A. Potter and Kenneth A. De Jong. A cooperative coevolutionary approach to function optimization. In *Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature: Parallel Problem Solving from Nature*, PPSN III, pages 249–257, London, UK, UK, 1994. Springer-Verlag.

[74] Saku Kukkonen and Jouni Lampinen. GDE3: The third Evolution Step of Generalized Differential Evolution. In *2005 IEEE Congress on Evolutionary Computation (CEC'2005)*, volume 1, pages 443–450, Edinburgh, Scotland, September 2005. IEEE Service Center.

[75] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. KanGAL report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[76] Eckart Zitzler and Lothar Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, November 1999.

[77] X. Ma, F. Liu, Y. Qi, X. Wang, L. Li, L. Jiao, M. Yin, and M. Gong. A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables. *IEEE Transactions on Evolutionary Computation*, 20(2):275–298, April 2016.

[78] Hai-Lin Liu, Fangqing Gu, and Qingfu Zhang. Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems. 18:450–455, 06 2014.

[79] Xingyi Zhang, Ye Tian, Ran Cheng, and Yaochu Jin. A decision variable clustering based evolutionary algorithm for large-scale many-objective optimization. PP, 08 2016.

[80] Zhenyu Yang., Ke Tang., and Xin Yao. Multilevel Cooperative Coevolution for Large Scale Optimization. In *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 1663–1670, 2008.

[81] S. Rahnamayan, H.R. Tizhoosh, and M.M.A. Salama. Opposition-based differential evolution. *Evolutionary Computation, IEEE Transactions on*, 12(1):64–79, 2008.

[82] Nasimul Noman and Hitoshi Iba. Enhancing differential evolution performance with local search for high dimensional function optimization. In *Proceedings of the 2005 conference on Genetic and evolutionary computation*, GECCO 2005, pages 967–974, New York, NY, USA, 2005. ACM.

[83] Weicai Zhong, Jing Liu, Mingzhi Xue, and Licheng Jiao. A multiagent genetic algorithm for global numerical optimization. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 34(2):1128–1141, 2004.

[84] Janez Brest, Ales Zamuda, Iztok Fister, and Mirjam Sepesy Maucec. Large scale global optimization using self-adaptive differential evolution algorithm. *Evolutionary Computation, IEEE Transactions on*, 12(2032-2039):1–8, July 2010.

[85] Xin Yao Zhenyu Yang, Ke Tang. Differential evolution for high-dimensional function optimization. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, volume 1, sept. 2007.

[86] Xin Yao Mohammad Nabi, Zhenyu Yang. Cooperative co-evolution for large scale optimization through more frequent random grouping. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, volume 1, pages 1– 8 Vol.1, sept. 2010.

[87] Jan Paredis. Coevolutionary computation. *Artif. Life*, 2(4):355–375, August 1995.

[88] Mitchell A. Potter and Kenneth A. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evol. Comput.*, 8(1):1–29, March 2000.

[89] C Rosin and R Belew. New methods for competitive coevolution. *Evolutionary Computation*, 5(1):1–29, March 1997.

[90] SevanGregory Ficici. Multiobjective optimization and coevolution. In *Multiobjective Problem Solving from Nature*, Natural Computing Series, pages 31–52. Springer Berlin Heidelberg, 2008.

[91] Luis Miguel Antonio and Carlos A. Coello Coello. A Non-cooperative Game for Faster Convergence in Cooperative Coevolution for Multi-objective

Optimization. In *2015 IEEE Congress on Evolutionary Computation (CEC'2015)*, pages 109–116, Sendai, Japan, 25-28 May 2015. IEEE Press. ISBN 978-1-4799-7492-4.

[92] Paul R. Ehrlich and Peter H. Raven. Butterflies and Plants: A Study in Coevolution. *Evolution*, 18(4):586–608, 1964.

[93] R.R.N. Mohammad and Z. Kobti. A new strategy to detect variable interactions in large scale global optimization. In *Swarm Intelligence (SIS), 2014 IEEE Symposium on*, pages 1–8, Dec 2014.

[94] Ales Zamuda, Janez Brest, Borko Boskovic, and Viljem Zumer. Large scale global optimization using differential evolution with self-adaptation and cooperative co-evolution. In *IEEE Congress on Evolutionary Computation*, pages 3718–3725. IEEE, 2008.

[95] Helio JC Barbosa and André MS Barreto. An interactive genetic algorithm with co-evolution of weights for multiobjective problems. In *Genetic and Evolutionary Computation Conference (GECCO-2001), San Francisco, California, July*, pages 7–11. Citeseer, 2001.

[96] Nattavut Keerativuttiumrong, Nachol Chaiyaratana, and Vara Varavithya. Multi-objective Co-operative Co-evolutionary Genetic Algorithm. In Juan Julián Merelo Guervós, Panagiotis Adamidis, Hans-Georg Beyer, José-Luis Fernández-Villaca nas, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature—PPSN VII*, pages 288–297, Granada, Spain, September 2002. Springer-Verlag. Lecture Notes in Computer Science No. 2439.

[97] Antony W. Iorio and Xiaodong Li. A Cooperative Coevolutionary Multiobjective Algorithm Using Non-dominated Sorting. In Kalyanmoy Deb et al., editor, *Genetic and Evolutionary Computation–GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part I*, pages 537–548,

Seattle, Washington, USA, June 2004. Springer-Verlag, Lecture Notes in Computer Science Vol. 3102.

[98] Kuntinee Maneeratana, Kittipong Boonlong, and Nachol Chaiyaratana. Multi-objective Optimisation by Co-operative Co-evolution. In *Parallel Problem Solving from Nature - PPSN VIII*, pages 772–781, Birmingham, UK, September 2004. Springer-Verlag. Lecture Notes in Computer Science Vol. 3242.

[99] Tse Guan Tan, Hui Keng Lau, and J. Teo. Cooperative coevolution for pareto multiobjective optimization: An empirical study using spea2. In *TENCON 2007 - 2007 IEEE Region 10 Conference*, pages 1–4, Oct 2007.

[100] Yu-Jun Zheng and Sheng-Yong Chen. Cooperative particle swarm optimization for multiobjective transportation planning. *Applied Intelligence*, 39(1):202–216, 2013.

[101] Bernabé Dorronsoro, Grégoire Danoy, Pascal Bouvry, and Antonio J. Nebro. Multi-objective cooperative coevolutionary evolutionary algorithms for continuous and combinatorial optimization. In Pascal Bouvry, Horacio González-Vélez, and Joanna Kołodziej, editors, *Intelligent Decision Systems in Large-Scale Distributed Environments*, volume 362 of *Studies in Computational Intelligence*, pages 49–74. Springer Berlin Heidelberg, 2011.

[102] TseGuan Tan and Jason Teo. Improving the performance of multiobjective evolutionary optimization algorithms using coevolutionary learning. In Raymond Chiong, editor, *Nature-Inspired Algorithms for Optimisation*, volume 193 of *Studies in Computational Intelligence*, pages 457–487. Springer Berlin Heidelberg, 2009.

[103] TseGuan Tan, HuiKeng Lau, and Jason Teo. Cooperative versus competitive coevolution for pareto multiobjective optimization. In Kang Li, Minrui Fei, GeorgeWilliam Irwin, and Shiwei Ma, editors, *Bio-Inspired Computational*

Intelligence and Applications, volume 4688 of *Lecture Notes in Computer Science*, pages 63–72. Springer Berlin Heidelberg, 2007.

[104] K.C. Tan, Y.J. Yang, and C.K. Goh. A distributed cooperative coevolutionary algorithm for multiobjective optimization. *Evolutionary Computation, IEEE Transactions on*, 10(5):527–549, October 2006.

[105] Tse Guan Tan, J. Teo, and Hui Keng Lau. Performance scalability of a cooperative coevolution multiobjective evolutionary algorithm. In *Computational Intelligence and Security, 2007 International Conference on*, pages 119–123, Dec 2007.

[106] K. C. Tan, Y. H. Chew, T. H. Lee, and Y. J. Yang. A cooperative coevolutionary algorithm for multiobjective optimization. In *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, volume 1, pages 390–395 vol.1, October 2003.

[107] Wenjing Zhao, Sameer Alam, and Hussein A. Abbass. Mocca-ii: A multi-objective co-operative co-evolutionary algorithm. *Appl. Soft Comput.*, 23:407–416, October 2014.

[108] Carlos A. Coello Coello and Margarita Reyes Sierra. A Coevolutionary Multi-Objective Evolutionary Algorithm. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, volume 1, pages 482–489, Canberra, Australia, December 2003. IEEE Press.

[109] Joshua D. Knowles and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

[110] Carlos A. Coello Coello and Gregorio Toscano Pulido. A Micro-Genetic Algorithm for Multiobjective Optimization. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, *First*

International Conference on Evolutionary Multi-Criterion Optimization, pages 126–140. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[111] Joshua D. Knowles and David W. Corne. The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Multiobjective Optimisation. In *1999 Congress on Evolutionary Computation*, pages 98–105, Washington, D.C., July 1999. IEEE Service Center.

[112] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. Technical Report 70, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, December 1999.

[113] Ian Parmee and Andrew H. Watson. Preliminary Airframe Design using Co-Evolutionary Multiobjective Genetic Algorithms. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark Jakiela, and Robert E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 1657–1665, Orlando, Florida, USA, 13-17 July 1999. Morgan Kaufmann.

[114] H. Wang, L. Jiao, and X. Yao. Two_arch2: An improved two-archive algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 19(4):524–541, Aug 2015.

[115] M. Li, S. Yang, and X. Liu. Pareto or non-pareto: Bi-criterion evolution in multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 20(5):645–665, Oct 2016.

[116] Zhi-Hui Zhan, Jingjing Li, Jiannong Cao, Jun Zhang, Henry Shu-Hung Chung, and Yu-Hui Shi. Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems. *Cybernetics, IEEE Transactions on*, 43(2):445–463, 2013.

[117] Carlos A. Coello Coello and Nareli Cruz Cortés. Solving Multiobjective Optimization Problems using an Artificial Immune System. *Genetic Programming and Evolvable Machines*, 6(2):163–190, June 2005.

[118] Qingfu Zhang, Aimin Zhou, Shizheng Zhao, Ponnuthurai Nagaratnam Suganthan, Wudong Liu, and Santosh Tiwari. Multiobjective optimization test instances for the cec 2009 special session and competition. *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, pages 1–30, 2008.

[119] Rui Wang, Robin C. Purshouse, and Peter J. Fleming. Preference-inspired co-evolutionary algorithms using weight vectors. *European Journal of Operational Research*, 243(2):423 – 441, 2015.

[120] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, May 2001.

[121] Richard A. Watson and Jordan B. Pollack. Coevolutionary dynamics in a minimal substrate. In *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, GECCO'01, pages 702–709, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[122] Sean Luke. When coevolutionary algorithms exhibit evolutionary dynamics. In *2002 Genetic and Evolutionary Computation Conference Workshop Program*, pages 236–241, 2002.

[123] Liviu Panait, R. Paul Wiegand, and Sean Luke. *A Sensitivity Analysis of a Cooperative Coevolutionary Algorithm Biased for Optimization*, pages 573–584. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[124] Edwin D. de Jong, Juan Julián Merelo-Guervós, John A. Bullinaria, Jonathan E. Rowe, Peter Tiňo, Ata Kabán, and Hans-Paul Schwefel. *Intransitivity in Coevolution*, pages 843–851. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[125] S. Nolfi and D. Floreano. Coevolving predator and prey robots: Do arms races arise in artificial evolution? *Artificial Life*, 4(4):311–335, Oct 1998.

[126] John Cartlidge and Seth Bullock. Combating coevolutionary disengagement by reducing parasite virulence. *Evol. Comput.*, 12(2):193–222, June 2004.

[127] Dave Cliff and Geoffrey F. Miller. *Tracking the red queen: Measurements of adaptive progress in co-evolutionary simulations*, pages 200–218. Springer Berlin Heidelberg, Berlin, Heidelberg, 1995.

[128] Thomas Lange Vincent and Joel Steven Brown. *Evolutionary game theory, natural selection, and Darwinian dynamics.* Cambridge University Press, Cambridge, New York, 2005. Autre tirage : 2007.

[129] J. Nash. Non-cooperative games. *The annals of Mathematics*, 54(2):286–295, 1951.

[130] Kwee-Bo Sim, Dong-Wook Lee, and Ji-Yoon Kim. Game theory based coevolutionary algorithm: A new computational coevolutionary approach. *INTERNATIONAL JOURNAL OF CONTROL AUTOMATION AND SYSTEMS*, 2:463–474, 2004.

[131] Eckart Zitzler and Simon Künzli. Indicator-based Selection in Multiobjective Search. In Xin Yao, EdmundK. Burke, JoSE. Lozano, Jim Smith, Juan Julian Merelo-Gueraes, JohnA. Bullinaria, JonathanE. Rowe, Peter Tiao, Ata Kaban, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature PPSN VIII*, volume 3242 of *Lecture Notes in Computer Science*, pages 832–842. Springer Berlin Heidelberg, Birmingham, UK, September 2004.

[132] Rainer Storn and Kenneth Price. Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, 11(4):341–359, December 1997.

[133] F. van den Bergh and A. P. Engelbrecht. A cooperative approach to particle swarm optimization. *Trans. Evol. Comp*, 8(3):225–239, June 2004.

[134] J. Lampinen. DE's selection rule for multiobjective optimization. Technical report, Lappeenranta University of Technology, Department of Information Technology, 2001.

[135] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable Test Problems for Evolutionary Multiobjective Optimization. In Ajith Abraham, Lakhmi Jain, and Robert Goldberg, editors, *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, pages 105–145. Springer Berlin Heidelberg, USA, 2005.

[136] Bradley Efron. *Bootstrap Methods: Another Look at the Jackknife*, pages 569–593. Springer New York, New York, NY, 1992.