Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

UNIDAD ZACATENCO

DEPARTAMENTO DE COMPUTACIÓN

Diseño de Algoritmos Meméticos Multi-Objetivo paralelos en GPUs

Tesis que presenta

Edgar Manoatl Lopez

Para obtener el grado de

Doctor en ciencias en computación

Director de tesis

Dr. Carlos Artemio Coello Coello

Ciudad de México **Febrero, 2019**

Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

ZACATENCO CAMPUS

COMPUTER SCIENCE DEPARTMENT

Design of Multi-Objective Memetic Algorithms using Graphical Processing Units

PhD thesis presented by

Edgar Manoatl Lopez

Advisor:

Dr. Carlos Artemio Coello Coello

Mexico City **February, 2019**

Resumen

En el mundo real existen muchos problemas de optimización en diversas disciplinas dentro de las ciencias, la ingeniería y la industria. Muchos de estos problemas requieren de la optimización simultánea de varias funciones objetivo las cuales suelen estar en conflicto entre sí. A éstos se les conoce como Problemas de Optimización Multi-Objetivo y no tienen una solución única, sino un conjunto de ellas, las cuales representan los mejores compromisos posibles entre todos los objetivos (es decir, soluciones en las cuales no es posible mejorar un objetivo sin empeorar otro). A dicho conjunto se le denomina óptimos de Pareto y su imagen en el espacio de las funciones objetivo se conoce como frente de Pareto.

Aunque existen diversas técnicas de programación matemática para resolver problemas multi-objetivo, éstas presentan varias limitantes, de entre las que destacan su sensibilidad a la forma y continuidad del frente de Pareto. Adicionalmente, la mayoría de estas técnicas suelen generar un elemento del conjunto de óptimos de Pareto por ejecución. Estas limitantes han motivado el uso de algoritmos evolutivos para resolver problemas multi-objetivo, pues presentan mayor generalidad y requieren poca información del problema.

En años recientes, la incorporación de buscadores locales a algoritmos evolutivos multi-objetivo se ha vuelto un área atractiva para varios investigadores, dando pie a los denominados algoritmos meméticos multi-objetivo. Sin embargo, la mayor parte de este trabajo se ha enfocado a problemas discretos y a la fecha existe muy poca investigación en torno a la solución de problemas continuos. La motivación principal para desarrollar algoritmos meméticos multi-objetivo es acelerar la convergencia hacia el frente de Pareto verdadero. No obstante, el diseño de estos algoritmos dista de lo trivial, pues es importante diseñar buenos buscadores locales y saber equilibrar su uso con respecto al buscador global. Adicionalmente, muchos problemas del mundo real tienen funciones objetivo costosas, las cuales pueden resolverse más eficazmente utilizado Unidades de Procesamiento Gráfico (GPUs, por sus siglas en inglés), cuyo costo económico es mucho más bajo que el de otras tecnologías disponibles para implementar algoritmos paralelos.

En esta tesis se utiliza el indicador de desempeño denominado *Inverted Generational Distance plus* (IGD$^+$) para guiar el motor de búsqueda local de un algoritmo memético multi-objetivo, incorporando el uso de GPUs. Se proponen dos nuevos algoritmos basados en IGD$^+$, los cuales se muestra que son competitivos con respecto a los algoritmos evolutivos multi-objetivo del estado del arte. El primero de ellos combina un optimizador global basado en el uso del hipervolumen y un motor de búsqueda local basado en IGD$^+$. Esta primera propuesta se compara con respecto al *S-Metric Selection Evolutionary Multiobjective Optimization Algorithm* (SMS-EMOA), siendo capaz de superarlo. Debido a que el uso del hipervolumen resulta computacionalmente costoso, se propone realizar una implementación paralela de este algoritmo mediante el uso de GPUs.

El segundo algoritmo propuesto en esta tesis consiste de una mejora del primero, de tal forma que puedan resolverse adecuadamente problemas multi-objetivo con frentes de Pareto complicados (p.ej., frentes de Pareto con geometrías irregulares). Esta segunda propuesta incorpora un mecanismo para guiar el proceso de optimización, el cual se adapta a cualquier forma geométrica del frente de Pareto. Este segundo algoritmo fue validado usando varios conjuntos de prueba, comparándose su desempeño con respecto al de nuestra primera propuesta. Adicionalmente, se le validó en un problema de diseño de vehículos en el que se busca maximizar la seguridad contra choques. El algoritmo propuesto no sólo es capaz de resolver éxitosamente una amplia variedad de problemas multi-objetivo, sino que también logra una aceleración significativa (de hasta 22 veces) mediante el uso de GPUs.

Abstract

Many real-world optimization problems exist in a wide variety of disciplines within science, engineering and industry. Many of these problems require the simultaneous optimization of several objective functions which are normally in conflict with each other. These are the so-called Multi-Objective Optimization Problems and they do not have a single solution, but a set of them, representing the best possible trade-offs among all the objectives (i.e., solutions in which it is not possible to improve one objective without worsening another one). This is the so-called Pareto Optimal set, and its image in objective function space is known as the Pareto front.

In spite of the existence of several mathematical programming techniques for solving multi-objective problems, they have several limitations, from which the main one is their sensitivity to the shape and continuity of the Pareto front. Additionally, most of these techniques generate a single element of the Pareto optimal set per run. These limitations have motivated the use of evolutionary algorithms for solving multi-objective problems, since they provide a greater generality and require little domain-specific information.

In recent years, the incorporation of local search engines into multi-objective evolutionary algorithms has become an attractive area for several researchers, giving rise to the so-called multi-objective memetic algorithms. However, most of this work has focused on discrete problems and to date, there is very little research related to the solution of continuous problems. The main motivation to develop multi-objective memetic algorithms is to speed up convergence towards the true Pareto front. Nevertheless, the design of these algorithms is far from trivial, because it is important to design good local search engines and to know how to balance their use with respect to that of the global search engine. Additionally, many real-world problems can be solved in a more efficient manner using Graphics Processing Units (GPUs) which have a lower cost than that of other technologies available for implementing parallel algorithms.

In this thesis, the performance indicator called *Inverted Generational Distance plus* (IGD$^+$) is adopted to guide the local search engine of a multi-objective memetic algorithm, incorporating the use of GPUs. Two new algorithms based on IGD$^+$ are proposed, both of which are found to be competitive with respect to state-of-the-art multi-objective evolutionary algorithms. The first algorithm combines a global search engine based on the use of the hypervolume and a local search engine based on IGD$^+$. This first proposal is compared with respect to the *S-Metric Selection Evolutionary*

Multi-objective Optimization Algorithm (SMS-EMOA), being able to outperform it. Since the use of the hypervolume involves a high computational cost, we propose a parallel implementation of this algorithm using GPUs.

The second algorithm proposed in this thesis consists of an improved version of the first one, such that it can properly solve multi-objective problems with complicated Pareto fronts (e.g., Pareto fronts with irregular geometries). This second proposal incorporates a mechanism to guide the optimization process, which is able to adapt to any geometrical shape of the Pareto front. This second algorithm was validated using several test problems, comparing its performance with respect to that of our first proposed algorithm. Additionally, it was validated using a vehicle design problem in which the aim is to maximize safety against collisions. Our proposed algorithm is not only able to successfully solve a wide variety of multi-objective optimization problems, but it can also achieve a significant speed up (of up to 22 times) through the use of GPUs.

Acknowledgements

Otra etapa más a terminado, con "altas y bajas", con "regaños y consejos", con "triunfos y fracasos". Tengo que decir que el ingreso al doctorado me enseño muchas alegrías y desdichas, sin en cambio, me encuentro de pie y listo a comenzar una nueva etapa de mi vida. Como bien dice mi canta-autor, escritor y sociólogo favorito NACH, tengo una vida por delante que descubrir y que aun tengo que vivir. El comienzo de esa nueva etapa surge con la culminación de mi tesis doctoral. Antes de comenzar esa nueva etapa tengo que empezar agradecer a todas las personas que contribuyeron en mi "camino del guerrero" (es decir, el camino que he tomado hasta aquí y poder decir que termine mi posgrado). En primera fila se encuentra mi madre, a ella le agradezco lo mucho que influyo en mi y poder decir que por ella, "yo soy lo que soy". Agradezco a toda mi familia, que por supuesto ellos me han motivado de forma implícita a esté logro. También quiero agradecer al Dr. Carlos A. Coello Coello, por todas sus enseñanzas y consejos que me brindo durante esté etapa, por supuesto que agradezco esos regaños que llego a decirme durante este proceso de mi formación como científico. Me agrado mucho tener que debatir con usted, y espero que no se quede con una mala impresión de mí. También quiero agradecer a una "personita" muy valiosa para mí, la cual me ha enseñado mucho más de esta vida que cualquier otro mentor, si eres tu "Nita" -"Gracias por todo"-. Gracias por tus buenos consejos, y por supuesto por tu atención que has tenido conmigo. Me has ayudado cuando más lo he necesitado, te agradezco de todo corazón.

Esta tesis fue derivada de los proyectos titulados "Esquemas de Selección Alternativos para Algoritmos Evolutivos Multi-Objetivo" (Ref. 1920, Convocatoria Fronteras de la Ciencia 2016 de CONACyT) y "Nuevos Esquemas de Selección para Algoritmos Evolutivos Multi-Objetivo basados en Indicadores de Desempeño" (No. de solicitud 4, Convocatoria 2018 del Fondo SEP-Cinvestav). El responsable técnico de ambos proyectos es el Dr. Carlos A. Coello Coello. Agradezco a los sinodales que me han apoyado a evaluar mi trabajo. También, agradezco a todas aquellas personas que se me olvidaron mencionar pero que saben que estoy muy agradecido de ellos.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Most practical real-world problems have several objectives (these objectives are often in conflict) which need to be optimized at the same time. These types of problems appear in many areas of our life such as engineering, chemistry, medicine and ecology, just to name a few. These sort of problems are known as Multi-objective Optimization Problems (MOPs) and their solution involves finding the best possible trade-offs among all the objectives. This set of trade-offs, when defined in decision variable space, is known as the *Pareto optimal set* (PS). The image of the Pareto optimal set is called the Pareto front (PF).

Mathematical programming techniques have been found to be very effective technique for solving MOPs, at a reasonably low computational cost. However, in real-world applications, there exist several MOPs for which mathematical programming techniques cannot guarantee that the solution obtained is optimum. These methods can be inefficient and inapplicable for solving these sort of problems. A possible way to deal with more complex optimization problems is to use meta-heuristics. One of them is the well-known Multi-Objective Evolutionary Algorithms (MOEAs). In the last 30 years, MOEAs have become very popular because of their conceptual simplicity and efficiency for solving MOPs.

However, there exists evidence that several types of MOPs are still very challenging to solve even for meta-heuristics. For example, in some MOPs, the Pareto optimal solutions are clustered in a small region of the search space. There are also MOPs with very complex Pareto front geometries (i.e., MOPs with irregular Pareto front shapes). The success of local search techniques in the solution of complex combinatorial optimization problems has motivated their incorporation into MOEAs, giving rise to the so-called Multi-Objective Memetic Algorithms (MOMAs) (see for example [1, 2]). The main advantage of adopting this sort of hybridization is to speed up convergence towards the Pareto front. However, the use of MOMAs introduces new issues, such as how to select the solutions to which the local search engine will be applied. In combinatorial optimization, there exists a lot of evidence indicating that the hybridization of the global optimizer with a local search engine is relatively easy to achieve (it is possible to establish a discrete neighborhood for each solution)[1]. However, it is not trivial to implement a local search engine for solving MOPs in continuous spaces, since in this case, defining a neighborhood structure is more difficult.

1.1 Problem Statement

Some researchers, such as [3], have raised some specific questions related to the effectiveness and efficiency of local search engines:

- How often should the local search (LS) be applied based upon a probability, PLS?
- On which k solutions should LS be used given a neighborhood $N(x)$ where x is a current solution?
- How long should LS be applied defined by a time period T?
- How efficient does LS need to be versus its effectiveness?

These questions involve some difficulties in designing new MOMAs. It is worth mentioning that combining a global optimizer with a local search technique for specific MOPs is critical to achieving good results, if the fitness function computation in real-world MOPs takes a considerable amount of running time. Nowadays, these limitations can be addressed using massive parallel processors such as a Graphics Processing Units (GPUs), since there is plenty of evidence that indicates that GPU-based approaches can reduce the running time without losing the advantages of CPU-based approaches (for more details see [4, 5, 6])).

1.2 Our proposal

Although MOMAs are very useful in solving complex MOPs, these sort of algorithms require more function evaluations than other multi-objective evolutionary algorithms,

[1]The neighborhood is used for establishing the region on which the local search will explore the solutions.

which makes them computationally expensive. For this reason, the main motivation of this work arises from the need to solve real-world MOPs more efficiently without losing the features of CPU-based memetic algorithms. In this thesis, we investigate different strategies to hybridize a global optimizer with a local search engine, in order to create a new MOMA based on the use of a GPU. The resulting GPU-based MOMA should be able to solve complicated MOPs more efficiently than its sequential version.

1.3 General and Specific goals

1.3.1 Main Goals

Since the main goal of this research is to advance the state-of-the-art, we create different strategies for the design of a new multi-objective memetic algorithm. To decrease its running time, we adopt the use of Graphics Processing Units. The aim was to create a new GPU-based MOMA, which is able to solve complex MOPs (i.e., MOPs with complicated Pareto front shapes).

1.3.2 Specific Goals

- Study the state-of-the-art multi-objective memetic algorithms, analyzing their main advantages and disadvantages with the aim of creating a new multi-objective memetic algorithm for continuous search spaces.
- Propose at least one new multi-objective memetic algorithm. This algorithm should be competitive with respect to state-of-the-art multi-objective evolutionary algorithms.
- Propose at least one GPU-based multi-objective memetic algorithm, which should be able to decrease the running time of its CPU-based version.
- Evaluate the proposed algorithms using standard test problems and performance indicators reported in the specialized literature.
- Perform a detailed statistical study of the proposed algorithms to determine the parameters to which they are most sensitive.
- Apply the proposed multi-objective memetic algorithm to solve a real-world multi-objective optimization problem.

1.4 Contributions

In the following, we present the publications obtained during the development of this thesis.

- Edgar Manoatl Lopez and Carlos A. Coello Coello. IGD+-EMOA: A Multi-Objective Evolutionary Algorithm based on IGD+, in *2016 IEEE Congress on Evolutionary Computation (CEC-2016)*, pp. 999-1006, IEEE Press, Vancouver, Canada, 24-29 July, 2016, ISBN 978-1-5090-0623-9.

- Edgar Manoatl Lopez and Carlos A. Coello Coello. A Parallel Multi-objective Memetic Algorithm Based on the IGD+ Indicator, in Julia Handl, Emma Hart, Peter R. Lewis, Manuel Lopez-Ibaez, Gabriela Ochoa and Ben Paechter (Editors), *Parallel Problem Solving from Nature PPSN XIV, 14th International Conference*, pp. 473-482, Springer. Lecture Notes in Computer Science Vol. 9921, Edinburgh, UK, September 17-21, 2016, ISBN 978-3-319-45822-9.

- Edgar Manoatl Lopez and Carlos A. Coello Coello. Improving the integration of the IGD+ indicator into the selection mechanism of a Multi-objective Evolutionary Algorithm, in *2017 IEEE Congress on Evolutionary Computation (CEC 2017)*, IEEE Press, San Sebastian, Spain 5-8 June 2017, 2017, ISBN 978-1-5090-4601-0.

- Edgar Manoatl Lopez and Carlos A. Coello Coello. An Improved Version of a Reference-Based Multi-Objective Evolutionary Algorithm based on IGD+, in *2018 Genetic and Evolutionary Computation Conference (GECCO'2018)*, IEEE Press, Kyoto, Japan July 15–19 2018, 2018, ISBN 978-1-4503-5618-3

- Edgar Manoatl Lopez and Carlos A. Coello Coello. Use of Reference Point Sets in a Decomposition-based Multi-Objective Evolutionary Algorithm, in *Parallel Problem Solving from Nature PPSN XV, 15th International Conference*, Coimbra, Portugal September 8–12 2018, 2018, ISBN 978-3-319-99258-7

1.5 Structure of the Thesis

This document is organized in ten chapters. In Chapter 2 we describe a brief introduction of optimization, including basic concepts and some methods to solve optimization problems. Chapter 3 presents a introduction of multi-objective optimization, including the definition of some performance indicators. In Chapter

4 we define the structure of multi-objective memetic algorithms and we review the most representative multi-objective memetic algorithms reported in the specialized literature. Our proposed multi-objective evolutionary algorithm based on the IGD$^+$ indicator and some strategies to create multi-objective evolutionary algorithms are described in detail in the Chapter 5 and Chapter 6, respectively. Our first proposal of a new multi-objective memetic algorithm is presented in Chapter 7. Our improved version of our proposed IGD$^+$-based multi-objective evolutionary algorithm is presented in Chapter 8. Chapter 9 presents an improved version of our IGD$^+$-based multi-objective memetic algorithm. Finally, we provide our final remarks and some possible paths for future work in Chapter 10.

Chapter 2

Background

This chapter presents some concepts related to optimization. The most important objective of this chapter is that the reader familiarizes with the basic concepts required to understand the contents of this thesis. The remainder of this chapter is organized as follows. Section 2.1 provides the conceptual and theoretical basis for global optimization. Likewise, a classification of different mathematical programming methods for nonlinear optimization problems is presented in Section 2.2. Finally, Section 2.3 describes a generic Evolutionary Algorithm (EA), and we also show the main advantages and disadvantages of this technique.

2.1 Optimization Problem

In mathematics, optimization refers to the process of finding the minimum or maximum point of a function, probably subject to some constraints on its decision variables. Throughout this document, we use the following notation:

- $\vec{x}$ is the vector of decision variables, also called parameters.
- f is the objective function, that we want to maximize or minimize.
- g and h represent constraints imposed on the objective function.

An optimization problem can be formally stated as follows:

Definition 2.1. Find the vector $\vec{x}$ which minimizes the function $f(\vec{x})$ subject to $\vec{x} \in \Omega \subseteq \mathbb{N}^n$, where Ω is the feasible region which satisfies the p inequality constraints:

$$g_i(\vec{x}) \leq 0;\ i = 1, ..., p$$

and the q equality constraints:

$$h_j(\vec{x}) = 0;\ j = 1, ..., q$$

The feasible solution $\vec{x}^* \in \Omega$ which corresponds to the minimum value of the objective function in all the search space is called “global optimum”.

Optimization problems can be classified according to the nature of the objective function and constraints (e.g., linear, nonlinear, convex), the number of decision variables (large or small) and to the features of the objective function (i.e., differentiable or non-differentiable). Although most problems in real-world applications have constraints, in this thesis we only deal with unconstrained optimization problems.

2.1.1 Notions of Optimality

It is necessary to introduce some relationships among the objective function, the constraints and the range of the decision variables. To make of this a self-contained chapter, we present next some basic concepts frequently used in optimization.

Definition 2.2. Global minimum: Given a function $f : \Omega \subseteq \mathbb{R}^n \to \mathbb{R}$, for $\vec{x}^* \in \Omega$ the value $f^* = f(\vec{x}^*) > -\infty$ is called global minimum, if and only if:

$$\forall \vec{x} \in \Omega : f(\vec{x}^*) \leq f(\vec{x}) \tag{2.1}$$

where $\vec{x}^*$ is a global minimum vector.

Definition 2.3. Local minimum: Given a function $f : \Omega \subseteq \mathbb{R}^n \to \mathbb{R}$, a solution $\vec{x}^o \in \Omega$ is called local minimum point, if and only if:

$$\forall \vec{x} \in \Omega : f(\vec{x}^o) \leq f(\vec{x}), \tag{2.2}$$

such that $||\vec{x} - \vec{x}^o|| < \epsilon$, where $\epsilon \in \mathbb{R}$ and the value $f(\vec{x}^o)$ is called local minimum.

2.1.2 Convexity

The concept of convexity is fundamental in optimization. It implies that it is possible to find in an exact manner the globally optimal solution. The formal definition of convex function is the following:

Definition 2.4. Convex function: A function $f : \mathbb{R}^n \to \mathbb{R}$ is called convex, where $\lambda \in \mathbb{R}$, if for any two vectors $\vec{x}, \vec{y} \in \mathbb{R}^n$ satisfies:

$$f(\lambda\vec{x} + (1-\lambda)\vec{y}) \leq \lambda f(\vec{x}) + (1-\lambda)f(\vec{y}), \tag{2.3}$$

The term convex programming is used to describe a special case of the constrained optimization problem in which:

- The objective function is convex
- The equality constraint functions are linear
- The inequality constraint functions are concave

On the other hand, if the function is non-convex, the optimization problem is much more difficult to solve, because the search space will be more difficult to explore.

2.1.3 Optimality Criterion

When the function f is smooth, there are several efficient and practical ways to identify the local minima. In particular, if f is twice continuously differentiable [7], the necessary conditions for optimality are derived by assuming that $\vec{x}^*$ is a local minimizer.

Definition 2.5. First-Order Necessary Conditions: If $\vec{x}$ is a local minimizer and f is continuously differentiable in an open neighborhood of $\vec{x}^*$, then $\nabla f(\vec{x}^*) = 0$.

Definition 2.6. Second-Order Necessary Conditions: Suppose that $\nabla^2 f$ is continuous in an open neighborhood of $\vec{x}^*$ and that $\nabla f(\vec{x}^*) = 0$ and $\nabla^2 f(\vec{x}^*)$ is positive definite. Then $\vec{x}^*$ is a strict local minimizer of f.

Although in some cases the necessary conditions are also sufficient for optimality, in general, additional information is necessary such as the derivative. Therefore, certain additional convexity assumptions are needed to guarantee that the solution $\vec{x}^*$ is optimal.

2.2 Classical Optimization Algorithms

Throughout the years, many mathematical programming techniques for solving optimization problems have been proposed. The classical or mathematical programming methods are deterministic algorithms, which exploit certain features of the optimization problems (e.g., convexity) and use additional information (e.g., the gradient). It is worth mentioning that these methods have been successfully used to solve engineering problems.

There are many techniques based on gradient information to solve nonlinear optimization problems such as Cauchy's method [8], Newton's method [9], and Fletcher and Reeves's method [10]. Although all of them solve optimization problems in a few steps, these techniques need that the search starts from an initial feasible point. On the other hand, there exist other techniques to solve non-differential problems which are called non-gradient techniques. Non-gradient methods or direct search methods, are techniques that do not require any information of the derivatives of the objective function. These sort of techniques constitute a good alternative when the problem is not differentiable and are also designed to solve one-dimensional and multi-dimensional optimization problems (see [11, 12]). Unfortunately, none of these mathematical methods guarantees convergence to the global optimum when dealing with the general nonlinear optimization problem. In most cases, these methods rely on an initial search point and, when dealing with multi-modal problems (i.e., problems that have several local optima) most of them get easily trapped in local optima and are unable to reach the global optimum. Despite the considerable variety of techniques developed in mathematical programming research and other disciplines to tackle those issues, the general nonlinear optimization problems calls for the use of remains alternative approaches to try to solve it.

2.3 Evolutionary Algorithms

Evolutionary Algorithms (EAs) are methods inspired on natural selection,(particularly, the "survival of the fittest" principle[1]), which are used to solve optimization problems. EAs operate on a population of solutions, i.e., they operate on a set of solutions instead of operating on one solution at a time, as traditional optimization methods. At each iteration of the EA, a competitive selection mechanism that tends to preserve the fittest individuals, is applied during the evolutionary process. The individuals with the highest fitness values have a higher probability to apply evolutionary operators to form new individuals, which are expected to be better than their predecessors. This process is repeated until reaching a maximum

[1]The survival of the fittest principle was proposed by Charles Darwin in his evolutionary theory [13].

(pre-defined) number of generations. The evolutionary operators associated with EAs are mutation and recombination. According to [3], an EA is described as follows:

Definition 2.7. Evolutionary Algorithm : Let I be an non-empty set (the individual space), $\{\mu^{(i)}\}$ where $i \in \mathbb{N}$ a sequence in $\mathbb{Z}^+$ (the parent population sizes), $\{\mu'^{(i)}\}$ a sequence in $\mathbb{Z}^+$ (the offspring population sizes), $\Phi :\longrightarrow \mathbb{R}$ a fitness function, $\bigcup_{i=1}^{\infty}(I^{\mu})^{(i)} \longrightarrow \{\text{true}, \text{false}\}$ (the termination criterion), $\mathcal{X} \in \{\text{true}, \text{false}\}$, r a sequence $\{r^{(i)}\}$ of recombination operators $r^{(i)} : \mathbb{X}_r^{(i)} \longrightarrow \mathcal{T}\Big(\Omega_r^{(i)}, \mathcal{T}\Big(I^{\mu^{(i)}}, I^{\mu'^{(i)}}\Big)\Big)$, m a sequence $\{m^{(i)}\}$ of mutation operators $m^{(i)} : \mathbb{X}_m^{(i)} \longrightarrow \mathcal{T}\Big(\Omega_m^{(i)}, \mathcal{T}\Big(I^{\mu^{(i)}}, I^{\mu'^{(i)}}\Big)\Big)$, s a sequence $\{s^{(i)}\}$ of selection operators $s^{(i)} : \mathbb{X}_s^{(i)} \times \mathcal{T}(I, \mathbb{R}) \rightarrow \mathcal{T}\Big(\Omega_s^{(i)}, \mathcal{T}\Big(\Big(I^{\mu'^{(i)} + \mathcal{X}\mu^{(i)}}\Big), I\mu^{(i+1)}\Big)\Big)$, $\Theta \in \mathbb{X}_r^{(i)}$ (the recombination parameters), $\Theta_m^{(i)} \in \mathcal{X}_m^{(i)}$ (the mutation parameters), and $\theta_s^{(i)} \in \mathbb{X}_s^{(i)}$ (the selection parameters). Then Algorithm 1 shows the pseudo-code of an Evolutionary Algorithm.

Algorithm 1 Evolutionary Algorithm

```
1:  $t := 0$;
2:  initialize $P(0) := \{\vec{a_1}, ..., \vec{a_\mu}\} \in I^{\mu^{(0)}}$ ;
3:  while $l(\{P(0), ..., P(t)\}) \neq True$ do
4:      recombine $P'(t) := r^{(t)}_{\Theta_r^{(t)}}(P(t))$;
5:      mutate $P''(t) := m^{(t)}_{\Theta_m^{(t)}}(P'(t))$;
6:      select:
7:      if $\mathcal{X}$ then
8:          $P(t+1) := s^{(t)}_{(\theta_s^{(t)}, \Phi)}(P''(t))$;
9:      else
10:         $P(t+1) := s^{(t)}_{(\theta_s^{(t)}, \Phi)}(P''(t) \cup P(t))$;
11:     end if
12:     $t := t + 1$;
13: end while
```

EAs are stochastic search methods. This implies that they cannot guarantee convergence to the global optimum of the problem. However, the use of evolutionary algorithms to solve optimization problems has been motivated mainly because of their population-based nature which allows them to generate several solutions in a single run. Additionally, evolutionary algorithms do not require specific domain information, which makes them a more general optimizer.

2.4 Evolutionary Computation Paradigms

The term Evolutionary Computation refers to a set bio-inspired techniques (which are based Darwin's evolutionary theory). In general terms, in order to simulate the evolutionary process it is worth to consider the following elements:

- Encoding
- Evolutionary Operators
- Fitness Function
- Selection Mechansim

Nowadays, there exist three main paradigms: a) Evolution strategies (ESs), b) Evolutionary Programming (EP) and c) Genetic Algorithms (GAs). These paradigms adopt all the conditions mentioned before. In the following we will briefly describe about the most important aspects of each of them.

2.4.1 Evolution Strategies

Evolution Strategies (ES) were developed by Rechenberg and Schwefel in Germany [14]. This algorithm was originally designed to solve hydrodynamic optimization problems. The original algorithm called (1+1)-ES adopted a single a father, which was mutated to create a single offspring. Although, the original proposal used a single solution, population-based ESs were introduced a few years later [14]. Rechenberg proposed the 1/5 rule in order to adjust the standard deviation value used to apply a Gaussian mutation during the evolutionary process for guaranteeing convergence to the global optimum (under certain assumptions).

2.4.2 Evolutionary Programming

Evolutionary Programming was proposed by Fogel in the USA [15]. The core idea of this algorithm is to simulate natural evolution as a learning process. EP uses a deterministic parent selection mechanism and each new member is created via mutation. In this paradigm, crossover is not used because evolution is simulated at a species level, and different species can't be recombined.

2.4.3 Genetic Algorithms

Genetic Algorithm were proposed in the early 1960s by Holland in the USA [16]. This algorithm was designed to solve machine learning problems. This sort of evolutionary algorithm adopts binary encoding of the individuals in the population. It uses a probabilistic selection mechanism and the crossover operator is considered its main evolutionary operator. However, a mutation operator is also adopted to improve the exploratory capabilities of the algorithm. Nowadays, the GA is the most popular evolutionary algorithm having an important number of applications [17, 18].

2.5 Advantages and Disadvantages of Evolutionary Algorithms

Evolutionary Algorithms have the advantage that their implementation is not complicated, because they are conceptually simple. EAs can be applied to any sort optimization problem without requiring problem specific information. EAs have been used to solve many real-world optimization problems in different areas such as Finance, Chemistry and Physics just to name a few. The complexities of many real-world problems (e.g., non-differentiable objectives, high dimensionality , etc.) makes them difficult to tackle using mathematical programming techniques. In contrast, EAs have been found to be a good choice to solve these problems. Their simplicity makes EAs easy to hybridize with other optimization approaches (e.g., classical optimization techniques). Additionally, EAs have little data dependency, which makes them easy to parallelize so that they can deal with problems having computationally expensive objectives. Classical optimization methods are not robust to dynamic changes in the environment and often require to restart the optimization process in order to provide a solution when such changes occur. In contrast, EAs can be used to adapt solutions to changing enviroments, which makes them advantageous to deal with dynamic environments. However, EAs require a fine-tuning of their parameters since it is not trivial to establish appropriate values for a specific problem. This is considered one of the main drawbacks of EAs. Furthermore, EAs do not guarantee convergence towards an optimal solution in a finite amount of time (i.e., generations). Finally, because of their stochastic nature, EAs need to be run several times in order to produce a statistically significant solution [3].

2.6 Other Bio-inspired Optimization Algorithms

Some researchers have proposed other bio-inspired approaches which have been widely used to solve optimization problems such as : Artificial Immune Systems (AIS) [19], Ant Colony Optimization (ACO) [20], Artificial Bee Colony (ABC) [21] and Particle swarm Optimization (PSO) [22].

Artificial Immune Systems are inspired by biological immune systems. They simulate the efficient immune system response when an animal is exposed to antigens. The response consists in generating antibodies with a high affinity to the antigens. The antibodies having the highest affinity are mutated (using a process known as hyper-mutation) and cloned. This model is known as clonal selection [19] and is the most popular in the specialized literature. Artificial immune systems have been adopted to solve classification and optimization problemas.

Ant Colony Optimization (ACO) was proposed in order to solve hard combinatorial optimization problems [20]. The inspiring source of ACO is the pheromone trails laid down by real ants which use pheromone as their communication channel. ACO implements a randomized construction bio-inspired heuristic which makes probabilistic decisions as a function of artificial pheromone. The artificial pheromone in ACO serves as numerical information which the ants use to construct solutions to an optimization problem.

The Artificial Bee Colony algorithm creates a colony which consists of three groups of bees: (1) employed bees, (2) onlookers and (3) scouts. The first half of the colony consists of the employed artificial bees and the second half includes the onlookers. For every food source there is only one employed bee. The artificial Bee Colony algorithm is considered as a swarm-based optimization algorithm, because all the bees work together in order to solve an optimization problem.

Particle Swarm Optimization simulates a swarm social model (to be more specific a bird flock or a fish school). Birds and fish adjust their physical movement to avoid predators, seek food and mate, which makes them to optimize environmental parameters such as temperature. In the conventional PSO, each particle in a swarm population updates its position in the search space according to the best particle, that has been found so far. The main core of PSO is to use particles with the best known positions to lead the swarm population in order to converge towards a single optimum in the search space. Particle swarm optimization is an algorithm that seems to be efficient and effective for optimizing a wide range real-world problems. Compared to evolutionary algorithms, PSO does not use evolutionary operators (i.e., crossover and mutation).

As we can see, there are several techniques for solving optimization problems. In this thesis we only will make reference to evolutionary algorithms, which are by far, the most popular bio-inspired meta-heuristic currently used for optimization.

Chapter 3

Multi-Objective Optimization

A large number of real-world problems that arise in academic and industrial areas, have several (often conflicting) objectives which need to be optimized at the same time. They are generically called Multi-objective Optimization Problems (MOPs) and their solution involves finding the best possible trade-offs among all their objectives. Formally a MOP is described as follows[1]:

$$\text{minimize} \quad \vec{f}(\vec{x}) := [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_m(\vec{x})]^T \tag{3.1}$$

subject to:

$$g_i(\vec{x}) \leq 0, \quad i = 1, 2, \ldots, p \tag{3.2}$$

$$h_j(\vec{x}) = 0, \quad j = 1, 2, \ldots, q \tag{3.3}$$

where $\vec{x} = [x_1, x_2, \ldots, x_n]$ is the vector of decision variables, $f_k : \mathbb{R}^n \rightarrow \mathbb{R}$, $k = 1, \ldots, m$ are the objective functions and $g_i, h_j : \mathbb{R}^n \rightarrow \mathbb{R}$, $i = 1, \ldots, p$, $j = 1, \ldots, q$ are the constraint functions of the problem. Equations (3.2) and (3.3) determine the feasible region, which is described by $\Omega \in \mathbb{R}^n$ and any decision vector $\vec{x} \in \Omega$ defines a feasible solution of the MOP.

However, the decision variables $x_i \forall i = 1, ..., n$ can be continuous or discrete. It is worth mentioning that in this work we are only interested in continuous domains which are contained on $\mathbb{R}^n$. When the functions g_i and h_i are not present in the MOP, the problem is called unconstrained MOP. On the other hand, if all functions and the constraint functions are linear, the problem is called a linear MOP. Thus, if at least one of the functions is nonlinear, the problem is named a nonlinear MOP. As

[1]We assume, without loss of generality, that all objectives are to be minimized.

mentioned before, in single-objective optimization, it is possible to determine between any given pair of solutions if one is better than any other (comparing their objective function values). As a result, we usually obtain a single optimal solution. On the other hand, solving a MOP involves finding the best possible trade-offs among all objectives of the MOP. These trade-offs, when defined in decision variable space, constitute the so-called Pareto optimal set. The image of the Pareto optimal set is called the Pareto front ($\mathcal{PF}$). According to the type of the functions f, g and h, we can classify MOPs as follows:

- **Multi-objective linear programming**. The objective functions and constraint functions, which are used on the MOP, are linear.
- **Nonlinear Multi-objective Optimization**. If at least one of the objective functions or constraint functions are nonlinear, the MOP is called nonlinear.
- **Convex Multi-Objective Optimization**. A multi-objective optimization problem is considered convex if all the objective functions and the feasible region are convex.

In this work we are interested in solving nonlinear unconstrained multi-objective optimization problems.

Definition 3.1. Decision Variables. The 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 presented by $\vec{x} = [x_1, x_2, \ldots, x_n]$.

Definition 3.2. Decision variable space. The decision variable space is the n-dimensional space of the decision variables, in which each coordinate axis corresponds with one component of the vector $\vec{x}$.

Definition 3.3. Objective functions. The objective functions evaluate how good a given solution is. Normally, they are denoted as $f_i : \mathbb{R}^n \to \mathbb{R}$. In MOPs, we have an objective function vector $\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_m(\vec{x})]$.

Definition 3.4. Objective function Space. The objective function space is the m-dimensional space of the objective functions, in which each coordinate axis corresponds with one element of the objective function vector $\vec{f}(\vec{x})$.

Definition 3.5. Feasible Region. We say that the feasible region is defined by all feasible vectors $\vec{x}$, where each vector $\vec{x}$ is feasible, if it satisfies all the constraints functions of the problem.

3.1 Notions of Optimality in Multi-Objective Optimization

As mentioned before, it is not possible to find a single solution that would be optimal for all the objective functions simultaneously when such objectives are in conflict with each other. Thus, in order to describe the concept of optimality in which we are interested on, the following concepts are introduced as follows:

Definition 3.6. Pareto Dominance. Let $\vec{x}, \vec{y} \in \mathbb{R}^m$, we say that $\vec{x}$ "dominates" $\vec{y}$ (denoted by $\vec{x} \prec \vec{y}$), if and only if: i) $x_i \leq y_i$ for all $i \in \{1, \ldots, m\}$ and ii) $x_j < y_j$ for at least one $j \in \{1, \ldots, m\}$.

Definition 3.7. Weak Dominance. We say that a vector $\vec{x}$ weakly dominates vector $\vec{y}$, denoted by $\vec{x} \preceq \vec{y}$, if $\vec{x}$ is not worse than $\vec{y}$ in all objectives.

Definition 3.8. Strict Dominance. We say that a vector $\vec{x}$ strictly dominates vector $\vec{y}$, denoted by $\vec{x} \prec\prec \vec{y}$, if and only if $f_i(\vec{x}) < f_i(\vec{y})$ for all $i \in \{1, \ldots, m\}$.

Definition 3.9. We say that a vector of decision variables $\vec{x} \in \mathcal{X} \subset \mathbb{R}^n$ is "non-dominated" with respect to $\mathcal{X}$, if there does not exist another $\vec{x}' \in \mathcal{X}$ such that $\vec{f}(\vec{x}') \prec \vec{f}(\vec{x})$.

Definition 3.10. We say that a vector of decision variables $\vec{x} \in \mathcal{F} \subset \mathbb{R}^n$ (where $\mathcal{F}$ is the feasible region) is "Pareto optimal" if it is non-dominated with respect to $\mathcal{F}$.

Definition 3.11. The Pareto Optimal Set $\mathcal{P}^*$ is defined by:

$$\mathcal{P}^* = \{\vec{x} \in \mathcal{F} | \vec{x} \text{ is Pareto optimal}\}$$

Definition 3.12. The "Pareto Front" $\mathcal{PF}^*$ is defined as follows:

$$\mathcal{PF}^* = \{\vec{f}(\vec{x}) \in \mathbb{R}^m | \vec{x} \in \mathcal{P}^*\}$$

In general, it is not possible to find an analytical expression that defines the $\mathcal{PF}$ of a MOP. Thus, the most common way to obtain the $\mathcal{PF}$ is to compute a sufficient number of points in the feasible region. Usually, we are interested in Pareto optimal solutions and can disregard the other feasible solutions. Exceptions to this practice are problems where one of the objective functions is an approximation of an unknown function. Then, the real Pareto optimal set is unknown.

Definition 3.13. Ideal Objective Vector. The ideal objective vector is denoted by $\vec{z}^* = [z_1^*, z_2^*, \ldots, z_k^*]^T$ and is obtained by minimizing each of the objective functions individually, subject to the constraints (i.e., $\vec{z}^* = \min f_i(\vec{x})$ subject to $\vec{x} \in \Omega$).

Definition 3.14. Utopian Objective Vector. The utopian objective vector is denoted by $\vec{z}^{**} = [z_1^{**}, z_2^{**}, \ldots, z_k^{**}]^T$. It is an infeasible objective vector whose components are formed by $z_i^{**} = z_i^* - \epsilon_i$, where z_i^* is the i^{th} component of the ideal objective vector and ϵ_i is a relatively small but computational significant scalar.

Definition 3.15. Nadir Objective Vector. The nadir objective vector is denoted by $\vec{z}^{nad} = [z_1^{nad}, z_2^{nad}, \ldots, z_k^{nad}]^T$ and its components are the upper bounds of the Pareto optimal set.

3.2 Optimization Methods for solving MOPs

The goal of most optimization methods is to find a reasonably good approximation of the Pareto optimal front. Currently, some researchers such as Zitzler et al. [23, 24] have suggested three desirable aspects of non-dominated sets, which are described as follows:

- **Convergence**: The distance between Pareto front approximation and Pareto optimal front should be minimized.
- **Distribution**: The distance between each point of the approximation should be as uniform as possible.
- **Spread**: The extent of the Pareto front approximation should be maximized (i.e., for each objective, a wide range of values should be covered by the approximation).

The optimization methods to solve MOPs can be classified in many ways such as enumerative, deterministic and stochastic methods [3]. Enumerative methods are considered as the simplest search strategies, since they evaluate each possible solution within some finite search space. It is worth mentioning that these methods become impractical when the search space is too large. Deterministic methods incorporate problem-domain knowledge. Some examples of this sort of algorithms are greedy methods, hill-climbing methods or gradient methods. These algorithms have been successfully used in solving a wide variety of MOPs, but when the MOP is high-dimensional, discontinuous or multimodal, they are often ineffective. On the

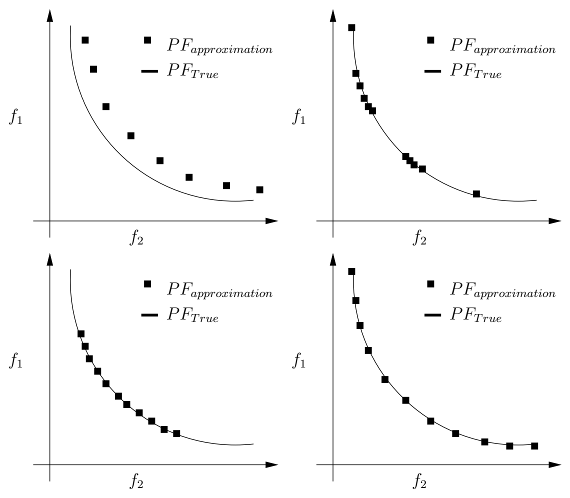


Figure 3.1: Examples of deficiency in the PFs

other hand, stochastic methods such as sampling methods, simulated annealing, tabu search, evolutionary algorithms and memetic algorithms have been developed to solve irregular MOPs. These sort of methods require a function to assign fitness values to the possible solutions of the MOP. They adopt some mechanisms in order to map between the problem and the domain of the algorithm. Stochastic methods cannot guarantee finding the optimal solution, but they can eventually find a local optimal. Nowadays, stochastic search methods provide good solutions to a wide range MOPs which traditional traditional deterministic search methods cannot properly solve.

3.3 Multi-Objective Evolutionary Algorithms

The first suggestion for using EAs to solve Multi-Objective Optimization Problems was hinted at the end of the 1960s by Rosenberg [25]. However, it was until 1984, when David Schaffer [26] proposed the first actual implementation of a Multi-Objective Evolutionary Algorithm (MOEA).

Evolutionary algorithms seem particularly suitable to solve MOPs, because they can deal simultaneously with a set of possible solutions (the so-called population). This allows finding several members of the Pareto optimal set in a single run of the algorithm, instead of having to perform a series of separate runs as in the case of traditional mathematical programming techniques [3]. Additionally, MOEAs are less susceptible to the shape or continuity of the Pareto front which means that they can properly deal with discontinuous or concave Pareto fronts. On the other hand, mathematical programming techniques normally have difficulties to deal with such problems.

EAs and MOEAs share a similar structure. Their main difference is the fitness assignment mechanism because a MOEA deals with more than one objective function at the same time. Therefore, the fitness assignment scheme must consider different desirable aspects of non-dominated solutions. MOEAs have been classified into three major categories. These categories and the specific techniques within each of them are commonly classified based on how and when they incorporate preferences from the Decision Maker (DM) into the search process. The classification is the following:

- **A Priori Approaches**. The DM defines the importance of the objectives before starting the search. Some examples are the Lexicographic method, linear fitness aggregation methods, and nonlinear fitness aggregation methods.

- **Progressive Techniques**. A search techniques produces solutions and the DM progressively provides preference information to narrow down the search so that the most preferred solutions can be found.

- **A Posteriori Techniques**. The optimizer produces non-dominated solutions and then the DM chooses the most preferred solutions according to his preferences. Some examples are Pareto-based MOEAs, niche-based MOEAs, aggregation-based MOEAs and Indicator-based MOEAs, just to named a few.

As mentioned before, the goal of MOEAs is to minimize the distance of the solutions obtained to the true Pareto front as well as to make the distribution of the solutions as uniform as possible Pareto front. Therefore, the fitness assignment of MOEAs must consider a way to evaluate those goals. In a MOEA, we need an additional process to transform a fitness vector into a scalar value. Thus, the selection mechanism of MOEAs is affected based on the fitness assignment scheme that is adopted. This implies that MOEAs can be classified in different groups. In the following, we present the most popular MOEAs developed over the years, some of which are referred to in this thesis:

1. **Pareto-based MOEAs**. These approaches adopt Pareto optimality in their selection mechanism in order to evaluate each individual from the population.

2. **Decomposition-based MOEAs**. These techniques transform a MOP into several scalar optimization subproblems, all of which are simultaneously solved.

3. **Indicator-based MOEAs**. These approaches use performance indicators (also called quality indicators) to assign the fitness of each individual based on how much a solution contributes to a given indicator.

3.4 Performance Assessment

The comparison of the performance of different MOEAs is an important issue in multi-objective optimization. Since the comparison needs to satisfy the three criteria mentioned before (see subsection 3.2), there exist different performance measures which allow a quantitative comparison of results among the different algorithms. Thus, the performance measure or Indicator $I(.)$ is a mapping from a set of objective vectors to a real number. Mathematically, we can say that $I := \mathbb{R}^{m\times k} \longrightarrow \mathbb{R}$, where m is the number of objectives and k is number of members in the non-dominated set. In order to describe the indicators which area used in this document, we need to provide first some definitions.

Definition 3.16. Pareto Compliant Indicator, if $A \rhd B$ holds between two non-dominated sets A and B, $I(A) < I(B)$ always holds: $A \rhd B \Longrightarrow I(A) < I(B)$.

Definition 3.17. Weak Pareto Compliant Indicator, if $A \preceq B$ holds between two non-dominated sets A and B, $I(A) \leq I(B)$ always holds: $A \preceq B \Longrightarrow I(A) \leq I(B)$.

We expect that performance indicators must have compatibility with the Pareto dominance relation. However, many of them are Pareto non-compliant. Next, we introduce some performance indicators which are used in this document.

3.4.1 Hypervolume Indicator

The hypervolume has become very popular in current multi-objective optimization research (see [27]). The hypervolume is able to assess convergence and maximum spread along the Pareto front and it is Pareto complaint. Because of its highly desirable features, it has been used not only as a quality measure for comparing final results of multi-objective evolutionary algorithms (MOEAs), but also as a selection operator (it is, for example, very suitable for *many-objective optimization problems*). However, it has one serious drawback: computing the exact hypervolume is highly costly. The best known algorithms to compute the hypervolume are polynomial in the number of points, but their cost grows exponentially with the number of objectives.

Mathematically if Λ denotes the Lebesgue measure, the hypervolume can be described as:

$$I_H(\mathcal{A}, \vec{y_{ref}}) = \Lambda \left(\bigcup_{\vec{y} \in \mathcal{A}} \{\vec{x} | \quad \vec{y} \prec \vec{x} \prec \vec{y_{ref}}\} \right) \tag{3.4}$$

where $\mathcal{A}$ is the approximation of the Pareto front optimal set and $\vec{y_{ref}} \in \mathbb{R}^k$ denotes the reference point. The hypervolume is obtained by computing the union of whole hypercubes in the objective space, where each hypercube is generated by every solution in $\mathcal{A}$ and the reference point $\vec{y}_{ref}$.

3.4.2 Generational Distance Indicator

The Generational Distance indicator (GD [28]) is defined as the averaged semi-distance from the image of a candidate set $\mathcal{A}$ to our discretization of the true Pareto front (i.e., the reference point set $\mathcal{Z}$). Next, we present its formal definition.

Given a candidate set $\mathcal{A} \subset \mathbb{R}^m$ and a reference set $\mathcal{Z} \subset \mathbb{R}^m$, then:

$$GD(\mathcal{A}, \mathcal{Z}) = \frac{1}{|\mathcal{A}|} \left(\sum_{j=1}^{|\mathcal{A}|} d_j(\vec{z}, \vec{a})^p \right)^{1/p} \tag{3.5}$$

where d_j denotes the nearest Euclidean distance from $a_i \in \mathcal{A}$ to $\mathcal{Z}$.

3.4.3 Inverted Generational Distance Indicator

The Inverted Generational Distance indicator, which is also called IGD indicator, is similar to the GD indicator, since both need a reference point set $\mathcal{Z}$. The formal definition of the IGD indicator is the following [29]:

$$IGD(\mathcal{A}, \mathcal{Z}) = \frac{1}{|\mathcal{Z}|} \left(\sum_{j=1}^{|\mathcal{Z}|} d'_j(\vec{z}, \vec{a})^p \right)^{1/p} \tag{3.6}$$

where d'_j denotes the minimal Euclidean Distance from $z_j \in \mathcal{Z}$ to $\mathcal{A}$.

Δ_p Indicator

Another reference set-based indicator is Δ_p [30], which is considered as an "Averaged Hausdorff Distance" between the approximate set and the reference set. This indicator is based on both GD and IGD. It is defined as:

$$\Delta_p = \max(GD(\mathcal{A}, \mathcal{Z}), IGD(\mathcal{A}, \mathcal{Z})) \tag{3.7}$$

In spite of the fact that Δ_p is a pseudo-metric which simultaneously evaluates proximity to the Pareto front and spread of solutions along it. It is not Pareto compliant.

3.4.4 $R2$ Indicator

This indicator belongs to the family of R indicators proposed by Hansen and Jaszkiewicz [31, 32]. This family of indicators use utility functions for evaluating approximate Pareto fronts. The $R2$-indicator is weakly monotonic and simultaneously evaluates several desired aspects of an approximate Pareto front. The formal definition of the $R2$-indicator is:

$$I_{R2}(\mathcal{A},\mathcal{U}) = -\frac{1}{|\mathcal{U}|}\left(\sum_{u\in\mathcal{U}} \max_{\vec{a}\in\mathcal{A}}\{u(\vec{a})\}\right) \tag{3.8}$$

where $\mathcal{U}$ denotes a utility function. The most commonly adopted utility function is the weighted Tchebycheff metric. The $R2$-indicator is defined in terms of the Tchebycheff metric as follows:

$$R2(\mathcal{A}, \mathcal{W}, \vec{z}^{*}) = \frac{1}{|\mathcal{W}|}\sum_{\vec{w}\in\mathcal{W}} \min_{\vec{a}\in\mathcal{A}}\left(\max_{i=1,\ldots,k}\{w_i|a_i - z_i^{*}|\}\right) \tag{3.9}$$

3.4.5 The modified Inverted Generational Distance Indicator

Recently, a variation of the well-known Inverted Generational Distance (IGD) was introduced in [33, 34]. This indicator, which is called IGD+ was shown to be weakly Pareto compliant, and represents some evident advantages with respect to the original IGD indicator. The IGD$^+$ indicator is defined as follows:

$$\text{IGD}^{+}(\mathcal{A}, \mathcal{Z}) = \frac{1}{|\mathcal{Z}|}\left(\sum_{j=1}^{|\mathcal{Z}|} d' +_j (\vec{z}, \vec{a})^{p}\right)^{1/p} \tag{3.10}$$

where $d'+$ is the nearest modified Euclidean distance from $z_j \in \mathcal{Z}$ to $\mathcal{A}$ and the modified Euclidian distance $d+(\vec{z},\vec{a})$ is defined as:

$$d+(\vec{z},\vec{a}) = \sqrt{(\max\{a_1 - z_1, 0\})^2 + \cdots + (\max\{a_m - z_m, 0\})^2}. \tag{3.11}$$

Therefore, a low IGD$^+$ value means that the set $\mathcal{A}$ constitutes a better approximation to the real $\mathcal{PF}$ if we consider the reference set as $\mathcal{PF}_{True}$. The Pareto compliance property between two objective vectors (i.e., $\vec{a} \prec \vec{b} \implies I(\vec{a},\vec{z}) < I(\vec{b},\vec{z})$) does not always hold when the Euclidean distance is used. The authors of IGD$^+$ proved that it is weakly Pareto compliant. Additionally, they also showed that IGD and Δ_p have inconsistencies with respect to the Pareto dominance relation since if a reference point $\vec{z}$ and an objective vector $\vec{a}$ are non-dominated with each other, it is possible to obtain $I(\vec{a},\vec{z}) > I(\vec{b},\vec{z})$ for $\vec{a} \prec \vec{b}$. The IGD$^+$ indicator overcomes the main drawbacks of both IGD and Δ_p.

3.5 Multi-Objective Evolutionary Algorithms based on a Reference Set

For several years, MOEAs adopted selection mechanisms based on Pareto optimality. However, in recent studies, it has been found that Pareto-based MOEAs can not properly solve many-objective problems (problems with more than three objectives) [35]. This has motivated the development of indicator-based selection mechanisms, since this sort of mechanism seems to work properly in many-objective problems.

Previously, we said that the performance indicator which has been most commonly used for the selection mechanism of a MOEA, is the hypervolume [36, 37].

One of the best hypervolume-based selection mechanisms currently available is the one incorporated in the *S Metric Selection-Evolutionary Multi-Objective Optimization Algorithm* (SMS-EMOA) [38]. However, the high cost involved in computing exact hypervolume contributions limits the practical use of SMS-EMOA in many-objective problems. In order to address these limitations, some researchers have developed different alternatives, such as the use of reference sets. Next, we will briefly discuss some reference-based MOEAs. Our discussion will focus specifically on approaches that adopt reference weight vectors for leading the optimization process. The approaches discussed next are divided in two main groups: (a) decomposition-based MOEAs and (b) indicator-based MOEAs which rely on the use of reference sets.

The *Multi-objective Evolutionary Algorithm Based on Decomposition* (MOEA/D)

[39] is the best well-known decomposition-based MOEA. MOEA/D divides the whole $\mathcal{PF}$ into a group of sub-spaces, each of which can be regarded as a sub-MOP. Another example of a MOEA that belongs in this category is NSGA-III [40], which uses a distributed set of reference points to manage the diversity of the candidate solutions, whose aim is to improve convergence. Although NSGA-III has been found to be a very competitive many-objective optimizer, its implementation is not trivial, being particularly sensitive to the construction of the hyper-planes. There is also an extension of MOEA/D which includes the Pareto dominance relation to select candidate solutions. This MOEA is called MOEA/DD [41], and is able to outperform the original MOEA/D, particularly in many-objective problems (the authors of MOEA/DD validated it with unconstrained and constrained benchmark problems having up to 15 objectives). The *Reference Vector Guided Evolutionary Algorithm* (RVEA) [42] is a very promising MOEA that provides very competitive results in MOPs with complicated Pareto fronts (i.e., MOPs with irregular Pareto front shapes). RVEA incorporates a novel method to preserve good candidate solutions, which consists of an adaptive technique for adjusting the reference vectors in order to balance the convergence and diversity of the solutions in high-dimensional objective spaces.

Regarding indicator-based MOEAs which rely on the use of reference sets, our discussion will focus specifically on approaches that adopt either IGD$^+$ or Δ_p (both of which are based on the use of reference sets). The first MOEA based on Δ_p was DDE [43], which uses differential evolution (DE) as its search engine. Its authors showed that DDE was able to converge rapidly towards the true Pareto front and that it could properly solve many-objective problems. However, this approach had some limitations related to the use of Δ_p (e.g., it does not properly work with multi-frontal and degenerate MOPs). On the other hand, Δ_p-EMOA [44] is another approach based on the use of Δ_p, which is inspired on SMS-EMOA [38] and incorporates a novel mechanism for building the reference set. This algorithm builds a reference point set by linearizing the non-dominated front of the current population. Δ_p-EMOA was designed to solve only bi-objective problems. Nevertheless, there is an extension of this approach, which is able to solve three-objective problems [45]. Another approach based on the Δ_p indicator is the *Reference Indicator-Based Evolutionary Multi-Objective Algorithm* (RIB-EMOA) [46]. This MOEA integrates a mechanism for building a reference set by using a family of curves. RIB-EMOA uses a weight vector set for approximating the reference point set. Although promising for many-objective optimization, this approach is not able to solve MOPs with complicated Pareto fronts. Another MOEA based on Δ_p was proposed in [47]. This algorithm uses ϵ-dominance in order to build a reference point set. This novel algorithm was validated using standard test functions, having from three to six objective functions. Its authors showed that this algorithm is able to solve convex, concave and disconnected MOPs. Additionally, they provided a promising solution to avoid the use of reference weight vectors for guiding the optimization process.

In recent years, some researchers have proposed other types of indicator-based reference-based MOEAs, such as R2-MOEA, which is based on the $R2$ indicator [31]. Same as when using Δ_p, the use of $R2$ requires a reference set of weights in order to compute its value. Another $R2$-based MOEA is the *Many Objective Meta-heuristic Based on the $R2$ indicator* (MOMBI) [48]. This algorithm adopts the use of weight vectors and the $R2$ indicator [31], and both mechanisms lead the optimization process. Although MOMBI is very competitive, it loses diversity in high dimensionality. This motivated the development of an improved version of this approach, called MOMBI-II [49], which uses an aggregation function and statistical information for selecting the candidate solutions.

In this thesis, our proposed MOEAs are categorized as indicator-based MOEAs which rely on the use of reference sets. To be more specific, our MOEAs are based on IGD$^+$. In Chapter 5 and Chapter 8, we provide more details about these MOEAs.

Chapter 4

Multi-Objective Memetic Algorithms

In order to design Multi-objective Evolutionary Algorithms (MOEAs) that are more efficient for solving real-world problems, some researchers have proposed to hybridize local search engines with MOEAs in order to drive the search toward the true Pareto front more effectively and efficiently, giving rise to the so-called Multi-Objective Memetic Algorithms (MOMAs). The main advantage of adopting this sort of hybridization is to speed up convergence to the Pareto front. However, the use of MOMAs introduces new issues, such as how to select the solutions to which the local search will be applied and for how long to run the local search engine (the use of such a local search engine has an extra computational cost). This chapter deal with the design of new MOMAs. We start by providing a brief definition of a Multi-Objective Memetic Algorithm and by explaining how it works. Additionally, we describe the advantages and disadvantages of MOMAs and we review some examples of them that have been proposed in the specialized literature.

4.1 About Multi-Objective Memetic Algorithms

The term "memetic" has its roots in the word "meme", which was originally described by Richard Dawkins in his classical book "The Selfish Gene" [3]. Pablo Moscato introduced the concept of "Memetic Algorithm" to denote the combination of local search heuristics with a population-based strategy. According to Moscato [50], the hybridization of a global optimizer (e.g., an evolutionary algorithm) with a Local Search engine (LS) is called a Memetic Algorithm (MA). These methods are inspired by models of natural systems which combine the evolutionary adaptation of a population with individual learning. In the case of MAs, “meme” refers to the strategies (e.g., local refinement or perturbation, just to name a few.) that are employed to improve individuals (or solutions). The success of local search techniques in the solution of combinatorial optimization problems [51, 52, 53, 54] has motivated their incorporation into MOEAs, giving arise the well-known Multi-Objective Memetic Algorithms (MOMAs). MAs and MOMAs share the same structure (i.e., both include a global optimizer and local search engine). However, MAs are able to solve single-objective optimization problems and the local search engine is applied to a certain percentage of solutions. The neighborhood, where the local search is applied, is created in an implicit way. On the other hand, MOMAs need to create, in an explicit way, the neighborhood, since solving MOPs involves finding a set of optimal solutions. The obvious choice for creating the neighborhood, is to cluster solutions into different regions of the objective space. Although MAs and MOMAs share the same structure, both change the rule of creating the neighborhood for applying the local search engine. It is worth mentioning that combining a global optimizer with a local search technique for specific MOPs is critical to achieve good results, if the fitness function computation in real-world MOPs takes a considerable amount of running time.

Thus, some researchers, such as [3], have raised some specific questions related to the effectiveness and efficiency of Local Search (LS) engines:

- How often should the LS be applied based upon a probability, PLS?
- On which k solutions should LS be used given a neighborhood $N(x)$ where x is a current solution?
- How long should LS be run, defined by a time period T?
- How efficient does LS need to be versus its effectiveness?

These questions raise several issues when designing new MOMAs and it is worth mentioning that we need to consider them for developing a MOMA. MOMAs are employed both to explore and to exploit the objective space. A generic MOMA is described in Algorithm 2 [3]. The algorithm starts with a population $\mathbb{P}_g$ which contains $\mathcal{N}$ randomly generated individuals. The algorithm is divided into two main processes: (1) the global optimizer and (2) the local search engine (LS). The aim of the global optimizer is to explore the objective space. On the other hand, the local search engine refines the optimization search along the Pareto front. LS techniques employ

Algorithm 2 Multi-Objective Memetic Algorithm

```
1: Randomly initialize population $\mathbb{P}_g$ with $\mathcal{N}$ individuals;
2: Evaluate fitness $\vec{f}_m(\vec{x})$ of each individual $\vec{x} \in \mathbb{P}_g$ ;
3: while Termination condition false do
4:     $g = g + 1$;
5:     Select $\mathbb{P}_g^{'}$ from $\mathbb{P}_{(g-1)}$ based on fitness $\vec{f}_k(\vec{x})$ ($k$ = number of objectives);
6:     Apply evolutionary operators to $\mathbb{P}_g^{'} \longrightarrow \mathbb{P}_g^{''}$;
7:     Local Search in $\mathbb{P}_g^{''}$ neighborhood; $\mathbb{P}_g^{''} \longrightarrow \mathbb{P}_g^{'''}$;
8:     Evaluate fitness $\vec{f}_m(\vec{x})$ of each individual in $(\mathbb{P}_g^{''}, \mathbb{P}_g^{'''})$;
9:     Select $\mathbb{P}_g$ from $(\mathbb{P}_{(g-1)}, \mathbb{P}_g^{'}, \mathbb{P}_g^{''}, \mathbb{P}_g^{'''})$;
10: end while
```

decision space neighborhoods whose selected points generate vectors in objective space. As mentioned before, the local search process is controlled by a certain number of iterations T and it is applied based upon a probability, PLS. The structure of a generic MOMA has been used for solving combinatorial MOPs. However, to design new MOMAs for solving MOPs in continuous spaces is not trivial, since to establish the neighborhood we need to define regions of the objective space in an explicit way. A possible way to deal with this problems is to cluster solutions (i.e., applying a niching technique) or to establish reference points (see Figure 4.1). There is plenty of evidence regarding the difficulties to improve a candidate solution when the LS engine is launched within a MOMA [55, 56, 57, 58], since finding the direction on which the LS engine has to make its next move depends on the following criteria: 1) the definition of the neighborhood (i.e., the size of the neighborhood and the selection the candidate solutions), 2) the number decision variables, and 3) the number of objectives.

For the purposes of this thesis, we classify MOMAs depending on the type of adopted LS engine. The two main categories are: mathematical programming techniques and stochastic techniques. Figure 4.2 shows a taxonomy of MOMAs, where each category is divided by the type of LS engine adopted. The use of mathematical programming methods into MOMAs ensures finding local optimal solutions (occasionally in a few steps), since they have specific rules to move from one solution to another. However, these types of MOMAs have some limitations (e.g., they use gradient information to guide the search). On the other hand, MOMAs based on stochastic techniques are able to solve non-differentiable MOPs. These are considered as derivative free-based MOMAs. Both types of MOMAs constitute a good alternative for solving MOPs in continuous search spaces, but it is worth mentioning

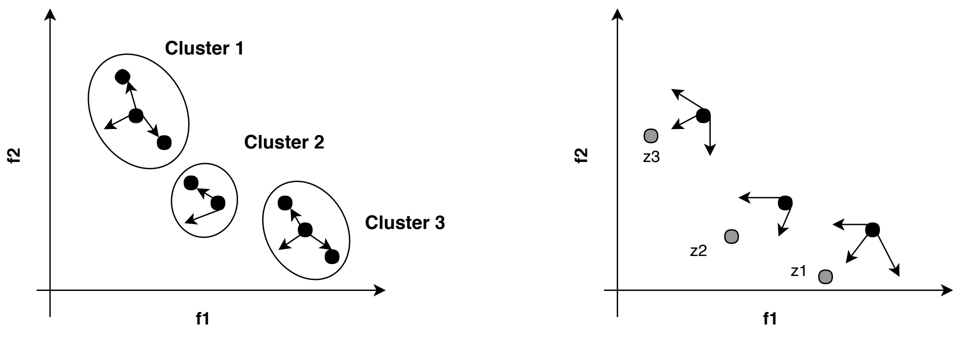


Figure 4.1: Graphical representation of two scenarios for applying the LS engine. The left hand figure shows a case in which the LS engine is launched using several clusters and the right hand figure shows the cases in which the LS is led by the reference point. Both mechanisms are applied in objective space.

that the use of a particular type of MOMA depends on the type of MOP that we aim to solve.

In the following, we review MOMAs, which are divided by two groups. The first group includes the MOMAs based on the use of mathematical programming techniques, which are shown in Section 4.2. On the other hand, Section 4.3 shows the second group, which is composed by MOMAs based on the use of stochastic search techniques.

4.2 Multi-Objective Memetic Algorithms based on Mathematical Programming techniques

Gradient-based optimizers are very efficient at finding local minima in a few steps, when the optimization problem being solved fulfills certain mathematical properties (e.g., convexity). In single-objective numerical optimization, the gradient of the function f to be optimized transmits useful information. For any dimensional point $\vec{y}$, the gradient at the point $\vec{x}$ ($\nabla f(\vec{y})$) is the direction of the greatest decrease of f starting from point $\vec{y}$. Hence, this direction can be used in an algorithm to find the local minimum of f. An obvious question, therefore, is how to extend gradient-based techniques to a multi-objective setting in an attempt to improve upon existing MOEAs. In multi-objective optimization, the situation is evidently more complicated. For instance, there are several good directions towards which we can possibly move (at least one by each objective) based on gradient information. Nowadays, several

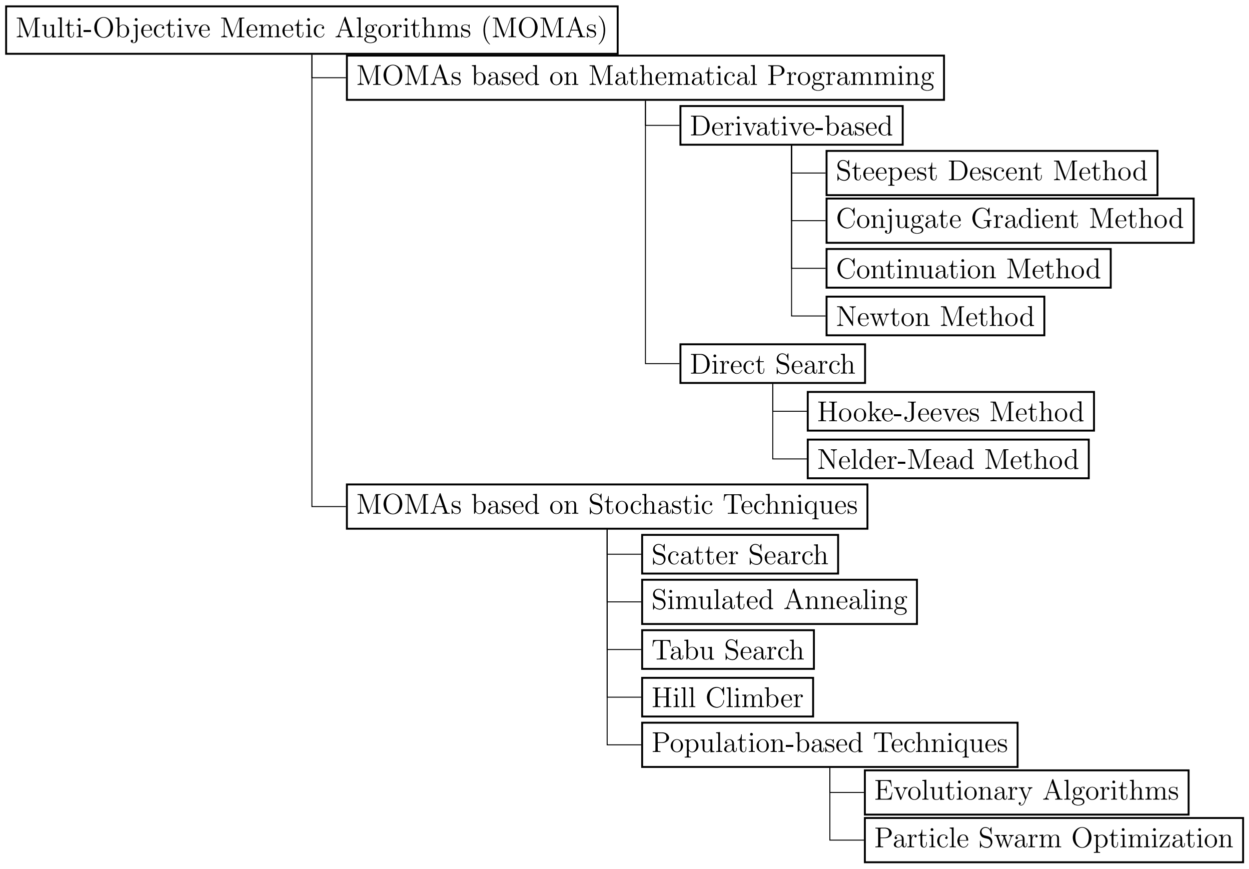


Figure 4.2: Taxonomy of Multi-Objective Memetic Algorithms.

researchers have proposed to hybridize a global optimizer with a gradient-based technique or a direct-based method. They have shown that the use of these techniques can address the optimization process with or without gradient information. In the following, we review state-of-the-art MOMAs based on mathematical programming techniques. We place special emphasis on the hybridization of MOEAs with gradient-based LS engines and non-gradient-based LS engines (see Table 4.1).

4.2.1 Exploiting Gradient Information in Numerical Multi-Objective Evolutionary Optimization

This MOMA was proposed by Bosman et al. [59]. This memetic algorithm hybridizes the extension of the Estimation of Distribution Algorithm (EDA[2]), called MIDEA [66], with a gradient-based LS engine. The LE engine adopts the conjugate gradient method for leading the optimization process. The LS engine is applied to a randomly chosen objective using the gradient of the current objective function. The authors named the resulting strategy as Random-Objective Conjugate Gradients. The hybridization scheme that they employed is a generational one. At the end of a generation, i.e., after one step of selection, variation and re-evaluation are finished, a set of candidate solutions is determined. The proposed MOMA launches the gradient-based LS engine for each candidate solution, where each of them is considered as a starting point. To launch the LS engine, the authors proposed to adopt a clustering technique for defining the ratio ρ_p of the neighborhood. The gradient-based local search operator is applied only as long as the current ratio is below a certain percentage. Their experiments showed that the best way to exploit gradient information in multi-objective optimization is to use a set of non-dominated solutions. This MOMA is capable of improving multiple solutions at the same time, which leads to a lower number of evaluations. Although they tested their approach on differentiable MOPs, this MOMA, it was not adopted to solve real-world MOPs.

4.2.2 Combining Gradient Techniques for Numerical Multi-Objective Evolutionary Optimization

This MOMA, which was proposed by Bosman et al. [60], is an extension of [59], where the authors proposed to use a line search method as an LS engine. The LS

[2]Estimation of distribution algorithms (EDAs) are a class of evolutionary algorithms that build at each generation a probabilistic model from the selected solutions (for more details about EDAs see [66]).

MOMA	**Global Optimizer**	**LS engine**	**Gradient Information**	**MOP**	**Performance Evaluation**	**Year**
MOMA based on MIDEA [59]	EDA	Conjugate gradient method	Yes	No	Generational Distance (GD)	2005
MOMA based on MIDEA [60]	EDA	Conjugate gradient Method and Combined-Objectives repeated Line-Search (CORL)	Yes	No	Generational Distance (GD)	2006
MOMA based on NSGA-II [61]	NSGA-II	Schaffler's Stochastic Method (SSM) and Timmel's Population-Based Method (TPM)	Yes	ZDT test suite	Generational Distance (GD), spread, and binary ϵ indicator	2009
MOMA based on SMS-EMOA [58]	SMS-EMOA	Steepest Descent method and Hooke-Jeeves method	Yes	ZDT test suite	Hypervolume	2009
MOMA based on HCS [62]	NSGA-II and SPEA	Hill Climber with Sidestep (HCS)	Yes	DTLZ and ZDT test suites	Generational Distance (GD) and Inverted Generartional Distance (IGD)	2010
MOMA based on hypervolume Indicator [63]	SMS-EMOA	Newton's method and Ordinary Differential Equation (ODE) maximizing the Hpervolume indicator	Yes	Convex and differentiable MOPs	Hypervolume	2013
MOMA based on MOEA/D [64]	MOEA/D	Directed Search method and Continuation method	Yes	ZDT test suite	Δ_p	2015
RDS-NSGA-II [65]	R-NSGA-II	Directed search method (DS) and Niching technique.	Yes[1]	DTLZ and ZDT test suites	Inverted Generational Distance (IGD)	2017

Table 4.1: Gradient-based MOMAs.

engine tries to find non-dominated solutions using the gradient descent direction. This MOMA shares the same structure of its predecessor. The LS engine optimizes a single objective and finds the best non-dominated solution. This strategy is called Combined-Objectives Repeated Line-Search (CORL). The global optimizer adopted by this MOMA is MIDEA. The naive MIDEA uses a simple univariate factorized probability distributions in each cluster, where each cluster is used for creating the neighborhood of the LS engine. The authors only tested their MOMA using differentiable MOPs, and they adopted the Generational Distance (GD) indicator for assessing its performance. Their experiments showed that the proposed MOMA was able to adapt the line search method to different types of MOPs. Additionally, this proposed approach is able to adapt the line search algorithm for decreasing the number of function evaluations performed.

4.2.3 Gradient Based Stochastic Mutation Operators in Evolutionary Multi-objective Optimization

This MOMA, proposed by Shukla et al. [61], is constituted by the hybridization of the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [67] and two classical LS engines. The first LS engine incorporates the use of Scheffler's Stochastic Method (SSM). SSM is based on the solution of a set of stochastic differential equations. This LS engine requires the objective functions to be twice continuously-differentiable, where the gradients are obtained using a stochastic perturbation method (i.e., by the mutation operator). On the other hand, the second LS engine adopts the use of Timmel's Population-Based Method (TPM). TPM creates feasible solutions using the descent direction and a step size. This method guarantees finding non-dominated solutions along the Pareto front. The authors proposed the use of the Finite Differences (FD) method for estimating the gradient information. The authors only tested their MOMA using the ZTD test suite, and adopted the Generational Distance (GD), and the binary ϵ indicator indicator for assessing its performance. Their numerical results show that multi-frontal MOPs affect the performance of the LS engines since both of them converge towards local regions when dealing with such problems.

4.2.4 On the Hybridization of SMS-EMOA and Local Search for Continuous Multi-Objective Optimization

This MOMA was proposed by Beume et al. [58], which hybridizes the S-Metric Selection Evolutionary Multi-Objective Algorithm (SMS-EMOA [38]) with a gradient-based LS engine and with a direct search-based LS engine. The first LS engine is based on the use of the steepest descent method and the second LS engine uses the Hooke-Jeeves method to lead the optimization process. The authors adopted both methods to compare the performance of each LS engine in different scenarios (i.e., uni-modal MOPs and multi-frontal MOPs). To apply the steepest descent method, the authors proposed to compute the Jacobian of the MOP (i.e., $J_f = \nabla f_1 \cdots \nabla f_m$). The LS search engine computes a descent direction υ using the gradient information from the Jacobian at a point x. The MOMA is based on a parameterized probability function to control the LS engine. The authors only tested their MOMA using the ZTD test suite, and they adopted the hypervolume indicator for assessing its performance. Their experimental results on academic test functions showed an increased convergence speed as well as an improved accuracy of the solutions set with respect to the stand-alone SMS-EMOA. Their authors showed that the use of a direct search method provides better performance than the use of the steepest descent method when it solves multi-frontal MOPs.

4.2.5 HCS: A New Local Search Strategy for Memetic Multi-objective Evolutionary Algorithms

This approach was proposed by Lara et al. [62], and hybridizes the Hill Climber with Sidestep (HCS), used as an LS engine, with the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) and the Improved the Strength Pareto Evolutionary Algorithm 2 (SPEA2 [68]). The authors proposed two ways to apply the hill climber method: 1) with gradient information and 2) without gradient information. Gradient-based HCS is capable of finding the descent direction using a QR-factorization of the transposed of the Jacobian matrix of the MOP at the point $\vec{x}$. On the other hand, non-gradient-based HCS generates randomly descent direction υ, which are computed by the geometry of the directional cones. The gradient-based LS engine uses a continuation method for linearizing the Pareto front to refine the current solution. The authors proposed to use the finite-difference method for approximating the gradient of the MOP. The LS engine is launched when certain probability is reached and is applied for each solution from the main population. The authors showed that the use of HCS as an LS engine combined with an MOEA improves performance. However, the numerical results show that this MOMA has some difficulties when it tries to solve multi-frontal MOPs. It is worth mentioning that this approach only works with bi-objective MOPs.

4.2.6 The hypervolume based directed search method for multi-objective optimization problems

As mentioned before, performance indicators have been used in the selection process of a MOEA, giving rise the well-known indicator-based MOEAs. These types of MOEAs have been shown to have good performance when solving complex MOPs. In this work, the authors made the first attempt to incorporate a performance indicator into an LS engine. This approach was proposed by Schütze et al. [63], and was called SMS-EMOA-HVDS. The authors proposed to adopt the SMS-EMOA as a global optimizer and Newton's Method for the local search engine. Newton's method uses the gradient and the Hessian of the hypervolumen indicator for maximizing its value. Thus, this local search engine is applied to whole solutions of the population at the same time. It is worth mentioning that it is not trivial to compute the first and the second derivative of the hypervolume indicator. For this reason, this approach only works with bi-objective problems. The authors showed that the local search significantly improves the performance of SMS-EMOA. Likewise, they showed that it is possible to drive the search of LS using multi-objective performance indicators such as the hypervolume. This approach is considered as an indicator-based LS engine. This MOMA was tested using convex and differentiable MOPs, and it would be interesting to extend it for solving many-objective problems.

4.2.7 The directed search method for multi-objective memetic algorithms

This approach was proposed by Schütze et al. [69], and consisted of two mechanisms to adopt a Direct search method: 1) using a gradient-based line search method to lead the LS engine towards the Pareto front and 2) using the gradient-based continuation method for sampling the shape of the Pareto front. For gradient-based line search, the authors proposed to solve a root finding problem, which was defined as $J(\vec{x})\upsilon = 0$ (where $J(\vec{x})$ defines the Jacobian of the MOP at the point $\vec{x}$). The authors established that the greedy direction υ is maximized when $< J(\vec{x}, \upsilon) >= 0$, i.e., the direction

v is orthogonal to the Jacobian at the point $\vec{x}$. The authors used v as the descent direction for the LS engine. On the other hand, the continuation method adopted two main procedures. The first procedure finds the predicted direction on which the continuation method has to move. In order to predict the direction, the authors proposed to compute the QR-factorization of the Jacobian vector at the point $\vec{x}$. After that, the continuation method corrects the predicted point along the Pareto front. These LS engines were hybridized with MOEA/D [39], where the LS engine is applied when certain percent is reached. Their experimental results on the ZDT test suite showed an increased convergence speed as well as improved accuracy of the solution set with respect to the use of the stand-alone MOEA/D.

4.2.8 A Multi-objective GA-Simplex Hybrid Algorithm

This MOMA, proposed by Zapotecas et al. [70], belongs to the class of MOMAs based on the use of a direct search method. This approach was proposed by Koduru et al. [71], and adopts a fuzzy dominance and the nonlinear simplex search algorithm (normally called Nelder-Mead algorithm) [12]. The proposed algorithm combines the exploratory nature of genetic algorithms with the exploitative behavior of simplex search. The proposed memetic approach is employed to estimate the parameters of a gene regulatory network for flowering time control in rice. The authors showed that their hybridization worked as they expected. They showed that their approach outperforms SPEA [68].

4.2.9 Diversifying Multi-Objective Gradient Techniques and their Role in Hybrid Multi-Objective Evolutionary Algorithms

This MOMA was proposed by Pirpinia et al. [72], the authors proposed a diversification technique to exploit the objective space using gradient information. For this sake, they used a Monte-Carlo method to obtain a discrete, spatially-uniformly distributed approximation of the Pareto-optimal set. This memetic algorithm adopts an Estimation of Distribution Algorithm (EDA) as its global optimizer. As mentioned before, EDAs are among the most robust and powerful optimization algorithms, providing the best balance between proximity and diversity. Their experimental results show that this MOMA is able to successfully solve deformable medical image registration problems.

4.3 Multi-Objective Memetic Algorithms based on Stochastic Algorithms

These MOMAs adopt a stochastic algorithm (e.g., an evolutionary algorithm, simulated annealing or tabu search) as an LS engine which is combined with a Multi-Objective Evolutionary Algorithm which acts as the global optimizer. In the last few years, the development of MOEAs hybridized with stochastic search methods has attracted the attention of several researchers mainly because in this case no derivatives are required to guide the search. In the following, we present some approaches that have reported improvements with respect to the use of a stand-alone MOEA .

4.3.1 A Scatter Tabu Search Procedure for Non-Linear Multi-objective Optimization

The authors proposed a Scatter Tabu Search Procedure for Non-Linear Multi-objective Optimization (SSPMO) [55], which adopts scatter search as an LS engine. This LS engine adopts the $||\vec{x}||_\infty$ for guiding the search along the Pareto front. It mimics the updating process used in a traditional scatter search implementation for single objective optimization (this makes the LS engine to optimize a single objective at a time, i.e., the approach LS engine optimizes the Lebesgue metric). The adopted global optimizer is a multi-objective tabu search algorithm [73], which utilizes a tabu list for avoiding repeated candidate solutions. It is worth mentioning that this is a continuous version of tabu search (i.e., it can solve continuous MOPs) of its original implementation. The authors mention that the scatter search phase of SSPMO is designed to address the main problem found in the approximation methods for multi-objective optimization (i.e., premature convergence towards local regions). Their experimental results on the ZDT and the DTLZ test suites show an increased convergence speed as well as an improved accuracy of the solution set with respect to the use of a stand-alone optimization algorithm.

4.3.2 An Improved Particle Swarm Pareto Optimizer with Local Search and Clustering

This is another stochastic-based MOMA, which was created by Ching et al. [74]. The authors proposed to use of a Multi-Objective Particle Swarm Optimizer (MOPSO) [57] as a global optimizer and simulated annealing as an LS engine. The main idea of the LS engine is to find new candidate solutions without converging towards local regions. This MOMA adopts the Hierarchical clustering algorithm [56], which uses a linkage criterion to determine the distance between two sets. The neighborhood of the LS engine is created explicitly by the clustering algorithm. The authors show that their proposed approach can avoid premature convergence, and the clustering technique maintains diversity in the population. Their experimental results show that this MOMA outperforms the stand-alone MOPSO regarding spread, spacing, and convergence.

4.3.3 Multi-objective memetic algorithm based on decomposition

This approach was proposed by Wali et al. [75], and hybridizes a decomposition-based MOEA with a Particle Swarm Optimization (PSO) algorithm [22]. PSO acts as an LS engine and it is able to refine the candidate solutions obtained by the global optimizer. The LS engine starts when certain percentage is reached, and its neighborhood is defined in an implicit way by the niche of the MOEA/D [76] (they adopt a version on differential evolution). The authors conclude that the power of the Differential Evolution (DE) operator is a good option for creating a global optimizer and that PSO refines the optimization search along the Pareto front. Although this MOMA outperforms MOEA/D when it solves some ZDT problems, this sort of hybridization cannot solve some instances from the UF test suite [77].

4.3.4 MOEA/D with opposition-based learning for multi-objective optimization problem

In this work, the authors proposed an Opposition-Based Learning technique (OBL) as an LS engine, which considers an estimate and its corresponding opposite value synchronously to obtain a better approximation for the optimal solution. The authors show that opposite points are more likely to approach the optimal solution than randomly generated points. The opposition-based learning strategy combines the evolutionary operators (to be more specific the differential evolution operator) to speed up the convergence of MOEA/D (i.e., the global optimizer). The opposition-based operator is used to initialize the population, and it creates multiples initial solutions to launch multiple LS engines. The opposition-based LS engine creates a neighborhood based on the use of a hyper-rectangle, which is formed by an interval.

The authors proposed a mechanism for changing the probability of applying the LS engine. This mechanism can select on which moment the global optimizer or the LS engine will be launched. This MOMA was tested on the ZDT, DTLZ and WFG test suites. Their numerical results show that this MOMA outperforms MOEA/D in terms of the IGD and hypervolume indicators.

4.3.5 A Decomposition based Memetic Multi-objective Algorithm for Continuous Multi-objective Optimization Problem

This MOMA was proposed by Wang et al. [78], and was one the first attempts to hybridize a decomposition-based MOEA with a greedy algorithm based on DE operators. The LS engine works as a hill climber method, but it uses the DE operators to create new candidate solutions. The neighborhood of the LS engine shares the same neighborhood structure of the global optimizer. The LS engine is applied when certain percentage of the total number of function evaluations is reached for each generation of the MOEA/D. This MOMA was tested on the ZDT test suites and Fonseca's test problem [79]. The experimental results show that this MOMA produces better solutions than MOEA/D regarding IGD and hypervolume. This hybridization turns of to be a good stochastic-based MOMA for solving muti-frontal MOPs.

4.3.6 Multi-objective Hybrid PSO Using ϵ-Fuzzy Dominance

This MOMA, proposed by Koduru et al. [80], hybridizes the Nelder-Mead algorithm with a Particle Swarm Optimizer (PSO, for more details, see [81]) algorithm, where PSO algorithm works as a global optimizer. A Nelder-Mead-based LS engine creates each neighborhood using a clustering technique (to be more specific, the k-means algorithm). The authors split the main population into smaller subpopulations (or clusters), where each sub-population is improved separately. The MOMA is shown to perform better than the original PSO on several test problems (from the DTLZ and ZDT test suites) as well as for the optimization of a genetic model for flowering time control.

In this thesis, we are interested in designing stochastic-based MOMAs that rely on the use of performance indicators, since as mentioned before there is some evidence that leads to believe that indicator-based MOMAs have shown to have a better performance with respect to other MOMAs (e.g., gradient-based MOMAs). Chapters 7 and 9 provide more details about our proposed MOMAs.

Chapter 5

Multi-Objective Algorithm based on the IGD$^+$ Indicator

As mentioned before, Multi-Objective Evolutionary Algorithms (MOEAs) have been developed during the last 30 years, from which the last 15 years, have had a very intense activity [82, 83]. For several years, MOEAs adopted selection mechanisms based on Pareto optimality [17, 23, 35]. However, recent studies have shown that Pareto-based multi-objective evolutionary algorithms do not perform properly when dealing with problems having more than three objectives (the so-called *many-objective optimization* problems) [35]. For this reason, some researchers have investigated the development of new selection schemes. One of the current research trends in this area is to use a performance indicator in the current selection mechanism of a MOEA because they provide a good ordering among sets that represent Pareto approximations. A number of performance indicators have been proposed, from which the hypervolume is, with no doubt, the most popular so far, mainly because of its nice mathematical properties (it's the only unary indicator which is known to be Pareto compliant [36, 84, 31]). However, the main drawback of hypervolume-based MOEAs is the high computational cost associated with the computation of the exact hypervolume contributions, which becomes unaffordable in many-objective optimization problems.

A possible way to deal with this limitation is to adopt a different indicator to select solutions in a MOEA such as reference-based performance indicators. Here, we propose a selection scheme based on the modified inverted generational distance (IGD) indicator, which was recently proposed by Ishibuchi [33, 34] (for more details see Chapter 2). This new version, called IGD$^+$ has a very low computational cost, even in high dimensional problems. Although, IGD$^+$ is weakly Pareto compliant, its main drawback is that it requires a reference set to compute the indicator value. In this chapter, we propose a MOEA based on the use of IGD$^+$. We also propose a technique to construct such a reference set and we show that the resulting MOEA has a competitive performance with respect to two state-of-the-art MOEAs (SMS-EMOA [38] and MOEA/D [39]), even when dealing with problems having a high number of objectives, while keeping a very affordable computational cost. The main goal of this chapter is to analyze and exploit the properties of the IGD$^+$ indicator. We show that the IGD$^+$ indicator is able to properly drive the search towards the Pareto Front of a MOP. In the following section, we describe in detail our proposed approach.

5.1 General Framework

Our approach starts with a population $\mathcal{P}_t$ which contains N randomly generated individuals. A new offspring is created by choosing two different parents from $\mathcal{P}$. The parents are recombined using Simulated Binary Crossover (SBX) and the resulting offspring are mutated (in this case, using Polynomial-Based Mutation [85]). This process is repeated until having λ offspring. The second step is to combine the parents and the offspring population to form the so-called Q set. The new population at generation $t+1$ is generated using an IGD$^+$-based selection mechanism. The pseudo-code of the multi-objective approach is illustrated by Algorithm 3. Next, we will provide more details of our proposed approach.

5.2 Selection Mechanism

Since we intend to use IGD$^+$ in the selection mechanism of a MOEA, we propose to transform the selection mechanism into a linear assignment problem (LAP). The LAP is the problem of choosing an optimal assignment of n items (e.g., jobs) to m machines (or workers). Formally, the LAP can be expressed as:
Given two sets, $A = \{a_1, \ldots, a_n\}$ and $T = t_1, \ldots, t_n$ with the same cardinality, and a cost function $C : A \times T \to \mathbb{R}$ and having $\Phi : A \to T$ as the set of all bijections between A and T, the LAP can be formulated as follows:

$$\min_{\phi \in \Phi} \sum_{a \in A} C(a, \Phi(a)) \tag{5.1}$$

Normally, the cost is also presented as a squared matrix C, where each element $Ci, j = C(a_i, t_j)$ represents the relationship between a_i and t_j.

Algorithm 3 General Framework

```
Input: A MOP ($F$), a stopping criterio, an uniform spread of $N$ weight reference
    vectors: $\lambda^1 \dots \lambda^N$.
Output: Approximation of the MOP
 1: $t \leftarrow 0$;
 2: Generate an initial population randomly $(x_i, \dots, x_N) \in X$ to create $\mathcal{P}_0$ ;
 3: while $t < gen_{max}$ do
 4:     $\mathcal{Q} \leftarrow \mathcal{P}_t$;
 5:     for each new offspring do
 6:         Apply evolutionary operators: Randomly select two parents from $\mathcal{P}_t$ to
            produce $u_i$;
 7:         $\mathcal{Q} \leftarrow \mathcal{Q} \bigcup \{F(u_i)\}$;
 8:     end for
 9:     Create the reference set ($\mathcal{Z}$) using the supersphere curve method with the
        current population $\mathcal{Q}$;
10:     $\mathcal{P}_{t+1} \leftarrow$ IGD$^+$-based selection mechanism ($\mathcal{Z}$, $\mathcal{Q}$); /*for more details see section
        5.2*/
11:     $t \leftarrow t + 1$;
12: end while
13: $\mathcal{Q} \leftarrow$ non-Dominated $\mathcal{P}_t$;
14: return $\mathcal{Q}$;
```

A linear assignment problem can be created in terms of a MOP, by using the m-dimensional objective vectors which represent individuals from the population and the reference set. So, a cost matrix is created using the modified distance d^+ between each element in the reference set and all objective vectors in the population. This transformation aims to find the best relationship between them. As evidenced in [86], the solution of this LAP allows convergence to the true Pareto front and, at the same time, produces a good distribution of solutions along the Pareto front. It is worth noticing, however, that the reference set needs to be well-distributed in objective function space and needs to dominate our current approximation of the Pareto front. In order to solve the LAP, we can make use of the Kuhn-Munkres Algorithm, also known as the Hungarian algorithm, which is able to solve LAP instances in polynomial time (it is $\mathcal{O}(n^3)$ [87] for squared matrices). An extension of this algorithm for rectangular matrices was introduced by Bourgeois [88]. The extension to rectangular matrices allows the algorithm to operate in situations where the numbers of reference points and individuals from the population are not equal.

We need to normalize the objective vectors of the current population, in order to handle objectives having different units. This normalization can be expressed as:

$$f_i' = \frac{f_i}{u_i} \tag{5.2}$$

where $\vec{u} \in \mathcal{R}^m$ and its i^{th}-element is defined as $u_i = \max\limits_{j=1,\ldots,\mu+\lambda} f_i(\vec{x}_j)$, $i = 1, \ldots, m$. The second step is to compute the C cost matrix and we can then express each element of the C cost matrix as follows:

$$C_{i,j} = d^+(a_i, z_j), \quad i = 1, \ldots, n, \quad j = 1, \ldots, n. \tag{5.3}$$

The solution to our assignment problem is found by identifying the combination of values in C resulting in the smallest sum. This solution corresponds to the best relationship of the current points of the population with respect to a reference set.

5.3 Approximating the Reference Set

In most multi-objective optimization problems, the geometrical shape of the true PF is unknown. However, we can approximate certain types of PFs (i.e., at least those having a smooth convex or concave surface) using superspheres. A γ-supersphere is a type of curve and it is defined as follows:

$$\{(y_1, \ldots, y_m) \in \mathbb{R}^m_+ | \quad y_1^\gamma + \cdots + y_m^\gamma = 1\} \tag{5.4}$$

where $\gamma \in \mathbb{R}_+$ is an arbitrary and fixed value. We only consider the "positive" parts of the γ-superspheres. According to [89], we can view the positive parts of the γ-superspheres as concave if $\gamma > 1$ or as convex if $0 < \gamma < 1$.

Clearly, we can see that a set of weight vectors satisfies equation (5.4) when $\gamma = 1$, since a weight vector is defined as:

Definition 5.1. Let $\vec{w} = [w_1, \ldots, w_m] \in \mathcal{R}^m$. We say that $\vec{w}$ is a weight vector if $\sum_{j=1}^m w_j = 1$ and $w_j \geq 0$.

In order to build the reference set, we assume that we have a set of weight vectors which is used to construct the reference set. We need to find the γ-value which will be used to transform the weights set into the reference set.

Clearly, in order to find the γ-value, equation (5.4) would be come a root-finding problem and we can say that the γ-value needs to satisfy:

$$y_1^{\gamma} + \cdots + y_m^{\gamma} - 1 = 0 \tag{5.5}$$

For solving equation (5.5), we propose to use Newton's method for approximating the γ-value. Now, we can see that the next approximation to the root is defined as:

$$\gamma_{k+1} = \gamma_k - \frac{(\sum_{j=1}^{m} y_j^{\gamma_k}) - 1}{\sum_{j=1}^{m} y_j^{\gamma_k} \log(y_j)} \tag{5.6}$$

Then, the computation of the reference set is described according to the following description.

Let $\mathcal{Q}$ be the current set which was created combining the parent and offspring population. Thus, the reference set is created by Algorithm 4.

Algorithm 4 Computation of the reference set which is based on supersphere curves

```
Input: A current set $\mathcal{Q} \subset \mathbb{R}^m$ and a set of weighted vec-
    tors $W \subset \mathbb{R}^m$, where $m$ is the number of objectives
Output: The reference set $\mathcal{Z}$ which is the best approximation of the set $Q$
 1: Finding the nondominated points from $\mathcal{Q}$ and
    save to $\mathcal{Q}'$
 2: for each $\vec{p} \in \mathcal{Q}'$ do
 3:     for each $\vec{w} \in \mathcal{W}$ do
 4:         Compute $d^{\perp}(\vec{p}, \vec{w}) = \| \vec{p} - \vec{w}^T \vec{s} \vec{w} / \| \vec{w} \|^2 \|$
 5:     end for
 6:     Assign $r(w) = \underset{\vec{p} \in \mathcal{Q}'}{\operatorname{argmin}} \; d^{\perp}(\vec{p}, \vec{w})$
 7: end for
 8: $j \leftarrow 0$
 9: for each $\vec{w} \in \mathcal{W}$ do
10:     $stepsize \leftarrow \vec{p}_{r(\vec{w})} \cdot \vec{w} / \| \vec{w} \|$
11:     $\vec{y} \leftarrow stepsize * \vec{w}$
12:     Approximate the $\gamma$ value using equation (5.5)
13:     Compute a supersphere point as $z_{j,k} \leftarrow w_{j,k}^{\gamma}$ for all $j = 1, \ldots, m$
14:     $j \leftarrow j + 1$
15: end for
```

In the first part of the algorithm, we find the non-dominated points which will be used as a reference for building the curve (these points establish the non-dominated region). After that, we search the best relationship between each weighted vector $\vec{w}$ and the non-dominated points. For this reason, we calculate the perpendicular distance between both sets. In order to construct the reference surface, we project the nearest non-dominated point to a specific weight vector $\vec{w}$. Once this is done, we can search the γ-value using Newton's method, which is described by equation (5.6). Finally, the reference point is computed using the γ-value. We can see that this process is repeated for all weight vectors of set $\mathcal{Z}$.

It is worth noting that this approach uses a predefined set of weights in order to ensure diversity. We adopted Das and Dennis' approach which places points on an $(m-1)$-dimensional hyperplane [90]. The total number of vectors is represented by the combinatorial number C_{m-1}^{H+m-1}, where H is the number of divisions of the objective space.

5.4 Experimental results

We compare the performance of our proposed algorithm with respect to that of two state-of-the-art MOEAs. The first is the *S Metric Selection-Evolutionary Multi-objective Optimization Algorithm* (SMS-EMOA) [38]. SMS-EMOA is a steady state evolutionary algorithm in which each newly created solution is ranked and a solution is removed from the worst ranked front in order to keep the same population size. The solution that contributes the least to the hypervolume of the worst ranked front is then discarded (see [38] for details). We use here a version that incorporates the algorithm proposed in [91] for estimating the hypervolume using Monte Carlo sampling, instead of the exact hypervolume calculations adopted in the original implementation of SMS-EMOA. The second approach adopted for our comparative study is the multi-objective evolutionary algorithm based on decomposition (MOEA/D) [39], which transforms a multi-objective problem into several single-objective optimization problems which are simultaneously optimized.

5.4.1 Test problems

For our comparative study, we adopted two benchmarks: the Deb-Thiele-Laumanns-Zitzler (DTLZ) test suite [92] and the Walking-Fish-Group (WFG) test suite [93]. These problems include aspects such as separability and multifrontality which make them more difficult to solve.

Problem	**Reference points**
DTLZ1	$(1, 1, 1, \ldots, 1)$
DTLZ2-6	$(2, 2, 2, \ldots, 2)$

Table 5.1: Reference points used for the hypervolume indicator.

5.4.2 Methodology

For our comparative study, we decided to adopt the hypervolume indicator, which assesses both convergence and maximum spread along the Pareto front. To compute the hypervolume indicator, we used the reference points shown in Table 5.1.

Additionally, we also compared the running time of each MOEA, which was measured in seconds.

5.4.3 Parameterization

In the DTLZ test suite, the total number of decision variables is given by $n = m+k-1$, where m is the number of objectives and k was set to 5 for DTLZ1 and to 10 for DTLZ2-6. The number of decision variables in WFG was set to 24, and the position-related parameter was set to $m - 1$.

The parameters of each MOEA used in our study were chosen in such a way that we could do a fair comparison among them. The distribution indexes for the SBX and polynomial-based mutation operators [85], used by our approach and SMS-EMOA, 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 $p_m = 1/L$, where L is the number of decision variables. Otherwise, the number of samples was set to 100,000. The total number of function evaluations was set in such a way that it did not exceed 60,000.

In MOEA/D and our proposed approach, the number of weight vectors was set to the same value as the population size. The population size N is dependent on H which specifies the number of divisions in objective space. H was set in such a way that N took a value not greater than 130. MOEA/D used the Tchebycheff approach with a neighborhood size of 20. The main characteristics of the hardware used for the experiments are the following: An Intel Core i7-3930k CPU running at 2.30 GHz, with 8GB of RAM.

Objectives	**H**	**Population size**
2	119	120
3	14	120
4	7	120
5	5	126
6	4	126
7	3	84
8	3	120

Table 5.2: Parameterization values

5.5 Discussion of Results

Table 5.3 provides the average hypervolume over the 30 independent executions of each compared MOEA for each instance of the DTLZ test suite. The best results are presented in **boldface**. We used Wilcoxon's rank sum for the statistical assessment of our results. It is clear that the winner in this experimental study is our proposed approach, since it was able to outperform SMS-EMOA-HYPE in all problems. We can observe that the null hypothesis can be rejected in all cases (the medians of two results are distinct), which means that the differences obtained are statistically significant.

Table 5.4 shows the comparison of results with respect to MOEA/D. As can be observed, our approach was able to outperform MOEA/D in twenty-four cases and in a few more, both approaches obtained similar results. The null hypothesis cannot be rejected in only three cases (DTLZ2 and DTLZ6 with 7, 2 and 6 dimensions, respectively). In the other cases, the differences obtained are statistically significant. For DTLZ5, MOEA/D performs better than our proposed approach in all the instances of this problem. The reason is probably that the true Pareto front of this problem is linear, which makes the approximations produced by our approach to converge to a single region. Table 5.5 provides the average running time over the 30 independent executions of each compared MOEA. Table 5.5 indicates that MOEA/D has the lowest average running times. However, our proposed approach was able to solve the problems in a reasonably low running time (if we consider the running time of SMS-EMOA, which would be significantly higher if exact hypervolume contributions had been computed). Overall, our proposed approach produced results that are very competitive, while requiring a reasonably low computational cost.

Figures 5.1, 5.2, 5.3, 5.4 present a graphical representation of the approximations of the Pareto front obtained by our proposed approach in some of the WFG test problems adopted with 24 variables and 3 objectives. These plots correspond to the mean hypervolume value from 30 independent executions. It is interesting to note that our proposed approach is able to properly converge to the true Pareto front of WFG2 which is disconnected. This confirms that the tendency of selection mechanism based on indicators such as IGD$^+$ is an effective way to solve many-objective problems.

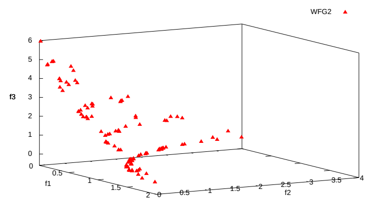


Figure 5.1: Solutions obtained by IGD$^+$-EMOA for WFG2.

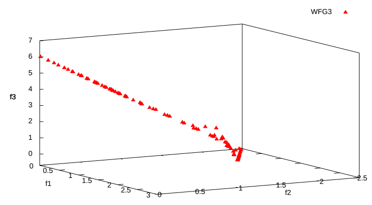


Figure 5.2: Solutions obtained by IGD$^+$-EMOA for WFG3.

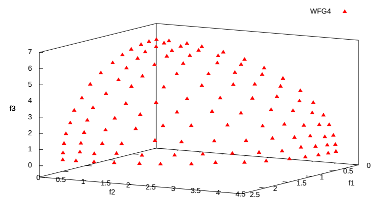


Figure 5.3: Solutions obtained by IGD$^+$-EMOA for WFG4.

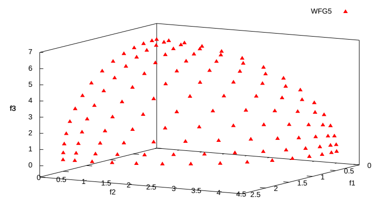


Figure 5.4: Solutions obtained by IGD$^+$-EMOA for WFG5.

5.6 Final Remarks

We have proposed a new indicator-based approach, which relies on the use of reference sets, for solving many-objective problems. This method constitutes a proof-of-principle regarding the suitability of the IGD$^+$ indicator to approximate the true Pareto Front of a MOP. The core idea of our proposed algorithm is to adopt the IGD$^+$ performance indicator in the selection mechanism of a MOEA. Our proposal includes a new method for constructing the reference set which is based on Newton's Method using super-spheres. Our results indicate that our proposed approach is very competitive with respect to two state-of-the-art MOEAs (SMS-EMOA and

MOEA/D), while requiring a relatively low computational cost (lower than that required by SMS-EMOA). It is worth noticing that the technique for building the reference set produces a good approximations to problems with concave, convex and disconnected Pareto fronts.

The main motivation for the algorithm presented in this chapter, has been to show that it is possible to design a competitive MOEA based on the IGD$^+$ indicador. This fact implies that it is possible to create a new Multi-Objective Memetic Algorithm based on the IGD$^+$ indicator, where the local search engine is led by a reference point set. It is worth mentioning that the reference point set could be established by the Decision Maker, which would allow to explore only a single region of the search space rather than all of it.

Motivated by the last advantage of this proposed approach, in Chapter 6, we describe a proposal that incorporates the IGD$^+$ indicator into a MOMA.

Objectives	**IGD$^+$-EMOA (I_H)**	**SMS-EMOA (I_H)**	$P(H)$
m	DTLZ1		
2	**0.873783426**	0.603475637	0.0000(1)
3	**0.97402027**	0.630213232	0.0000(1)
4	**0.994388397**	0.661586784	0.0000(1)
5	**0.9919105**	0.768368948	0.0000(1)
6	**0.892359232**	0.776399218	0.0000(1)
7	**0.854507595**	0.738417486	0.0000(1)
8	**0.861695367**	0.814606014	0.0000(1)
DTLZ2			
2	**3.210821317**	1.863197179	0.0000(1)
3	**7.421812488**	3.375262182	0.0000(1)
4	**15.56741135**	6.747488035	0.0000(1)
5	**31.66763799**	14.90159836	0.0000(1)
6	**58.07564274**	33.83189984	0.0000(1)
7	**116.7077942**	65.20591399	0.0000(1)
8	**201.1809779**	150.6662452	0.0000(1)
DTLZ3			
2	**3.204155341**	1.88237071	0.0000(1)
3	**7.355071834**	6.411731511	0.0000(1)
4	**15.53694896**	14.12060072	0.0000(1)
5	**31.63537503**	30.64233556	0.0000(1)
6	56.53710167	**59.64630155**	0.0000(1)
7	**96.20407137**	60.7350761	0.0000(1)
8	**198.890958**	129.4343786	0.0000(1)
DTLZ4			
2	**3.210795477**	2.083725111	0.0000(1)
3	**7.082647895**	3.866538974	0.0000(1)
4	**15.20291716**	7.715636261	0.0000(1)
5	**29.27224775**	17.69125542	0.0000(1)
6	**59.24047887**	39.52867707	0.0000(1)
7	**113.5313178**	74.07283972	0.0000(1)
8	**245.9811967**	168.8209989	0.0000(1)
DTLZ5			
2	**3.210822115**	1.859887057	0.0000(1)
3	**4.042831297**	3.575537122	0.0000(1)
4	**8.003414255**	6.12118542	0.0000(1)
5	**16.00248339**	11.30170488	0.0000(1)
6	**31.99996981**	21.59984771	0.0000(1)
7	**63.99998659**	39.53980236	0.0000(1)
8	**127.999845**	80.08300234	0.0000(1)
DTLZ6			
2	**3.106126387**	1.714714707	0.0000(1)
3	**5.66835877**	3.019997564	0.0000(1)
4	**7.437133923**	5.724252402	0.0000(1)
5	**14.82343849**	10.70214298	0.0000(1)
6	**29.76776111**	20.11681978	0.0000(1)
7	**58.53573762**	36.44267101	0.0000(1)
8	**117.596708**	75.84189126	0.0000(1)

Table 5.3: Results obtained in the DTLZ test problems by SMS-EMOA(Hype) and our proposed IGD$^+$-EMOA, using the hypervolume indicator (I_H). The third column shows the results of the statistical analysis applied to our experiments using Wilcoxon's rank sum, where P is the probability of observing the given result (the null hypoteshis is true). If the P-value is small, this indicates that the null hypothesis can be rejected at the 5% level and we can conclude that the two results are distinct ($H = 1$) and their difference is statistically significant.

Objectives	**IGDplus-EMOA (I_H)**	**MOEA/D (I_H)**	$P(H)$
m	DTLZ1		
2	0.873783426	**0.873862461**	0.0000(1)
3	**0.97402027**	0.968914097	0.0000(1)
4	**0.994388397**	0.970400938	0.0000(1)
5	**0.9919105**	0.724927498	0.0000(1)
6	**0.892359232**	0.762830249	0.0001(1)
7	**0.854507595**	0.656643222	0.0000(1)
8	**0.861695367**	0.402984134	0.0000(1)
DTLZ2			
2	3.210821317	**3.210869248**	0.0000(1)
3	**7.421812488**	7.382922569	0.0000(1)
4	**15.56741135**	13.31884553	0.0000(1)
5	**31.66763799**	27.15582568	0.0000(1)
6	**58.07564274**	53.83549288	0.0000(1)
7	**116.7077942**	115.8005853	0.2890(0)
8	201.1809779	**216.0379868**	0.0000(1)
DTLZ3			
2	3.204155341	**3.206666982**	0.0009(1)
3	7.355071834	**7.374166479**	0.0215(1)
4	**15.53694896**	12.91659236	0.0000(1)
5	**31.63537503**	24.2992914	0.0000(1)
6	**56.53710167**	48.35142688	0.0000(1)
7	96.20407137	**100.9289111**	0.0000(1)
8	198.890958	**224.3638975**	0.0000(1)
DTLZ4			
2	**3.210795477**	2.395550932	0.0001(1)
3	**7.082647895**	6.233954506	0.0003(1)
4	**15.20291716**	11.7039561	0.0000(1)
5	**29.27224775**	22.23443593	0.0000(1)
6	**59.24047887**	47.35243703	0.0000(1)
7	**113.5313178**	93.66821328	0.0000(1)
8	**245.9811967**	184.5468339	0.0000(1)
DTLZ5			
2	3.210822115	**3.210869361**	0.0000(1)
3	4.042831297	**6.091452678**	0.0000(1)
4	8.003414255	**10.80302487**	0.0000(1)
5	16.00248339	**16.15695638**	0.0037(1)
6	31.99996981	**37.8234936**	0.0000(1)
7	63.99998659	**78.10923993**	0.0000(1)
8	127.999845	**150.5444428**	0.0000(1)
DTLZ6			
2	**3.106126387**	3.056665166	0.0000(1)
3	5.66835877	**5.801128621**	0.5792(0)
4	7.437133923	**8.877262691**	0.0000(1)
5	**14.82343849**	11.95437872	0.0000(1)
6	**29.76776111**	28.53823221	0.5793(0)
7	58.53573762	**71.77293638**	0.0000(1)
8	117.596708	**131.4029039**	0.0000(1)

Table 5.4: Results obtained in the DTLZ test problems by MOEA/D and our proposed IGD$^+$-EMOA, using the hypervolume indicator (I_H). The third column shows the results of the statistical analysis applied to our experiments using Wilcoxon's rank sum, where P is the probability of observing the given result (the null hypothesis is true). If the P-value is large, the data do not give any reason to reject the null hypothesis and we can conclude that the two results are the same ($H = 0$). Otherwise, if the P-value is small, the null hypothesis can be rejected at the 5% level and the two results are distinct ($H = 1$) and their difference is statistically significant.

Objectives	IGD$^+$-EMOA	MOEA/D	SMS-EMOA
m	DTLZ1		
2	5.638792	**0.451672**	929.482702
3	10.521783	**0.529422**	1230.958571
4	13.902771	**0.245426**	2111.873245
5	28.46273	**0.435324**	2014.102227
6	32.162026	**0.566348**	2082.745213
7	19.710173	**0.27731**	3510.322044
8	46.412152	**0.067746**	5038.534303
	DTLZ2		
2	29.264559	**0.618055**	1259.578953
3	42.169365	**0.694392**	2200.86988
4	54.027164	**0.32018**	3689.959608
5	90.588769	**0.575118**	4539.503577
6	107.13428	**0.86347**	5563.185747
7	12.719386	**0.337427**	5290.054592
8	27.656435	**0.085446**	6934.750096
	DTLZ3		
2	8.252379	**0.573409**	800.213934
3	49.478735	**0.662682**	999.843201
4	23.923694	**0.295604**	1185.619497
5	48.132021	**0.533946**	1052.808177
6	74.052142	**0.733004**	1505.031887
7	15.886005	**0.337056**	3180.209343
8	27.307364	**0.076317**	4069.487323
	DTLZ4		
2	7.620927	**0.49825**	1353.118432
3	10.11606	**0.698973**	2337.602668
4	13.871761	**0.265944**	3948.222166
5	22.35422	**0.55728**	4213.046424
6	25.576331	**0.78043**	5512.432487
7	15.039888	**0.315422**	5241.15327
8	31.035888	**0.070369**	6897.864537
	DTLZ5		
2	7.983771	**0.600385**	1238.279526
3	26.447281	**0.714718**	1977.824246
4	19.755846	**0.242651**	4047.525445
5	22.538165	**0.315002**	3862.21531
6	21.331467	**0.334229**	6723.838925
7	72.28353	**0.347957**	6149.572591
8	23.338065	**0.397939**	8314.476698
	DTLZ6		
2	7.171413	**0.578733**	2007.65159
3	33.712275	**0.623824**	2526.082216
4	44.556186	**0.200934**	3039.004372
5	52.202535	**0.472714**	3307.124979
6	56.476568	**0.349478**	3985.982441
7	43.149346	**0.296292**	4645.460144
8	99.324617	**0.384934**	7241.438203

Table 5.5: Here, we show the computational time (measured in seconds) required by each execution of the MOEAs compared. All algorithms were compiled using the GNU C compiler and they were executed on the same computer.

Chapter 6

A new selection mechanism based on the IGD$^+$ Indicator

6.1 Introduction

In this chapter, we propose a selection mechanism (called IGD$^+$-H) which is based on the combination of the Inverted Generational Distance$^+$ (IGD$^+$) indicator [33, 34] and the Kuhn-Munkres' (Hungarian) algorithm to solve Linear Assignment Problems (LAPs). The proposed selection scheme is compared with respect to other selection mechanisms (i.e., steady-state selection mechanism) based on the IGD indicator and with respect to the use of the Δ_p indicator. Our proposed technique is incorporated into a MOEA and is validated using standard test problems. We show that our proposed IGD$^+$-H-based selection mechanism is able to achieve a significant speedup (of up to 200x) with respect to the exclusive use of any of the indicators adopted in our study. We analyze here the impact on this selection mechanism (using a transformation to a linear assignment problem) when it interacts with different indicators based on reference sets and we show that such a mechanism is able to drive the search towards the Pareto Front. Indeed, it is possible to incorporate our proposed IGD$^+$-H selection mechanism into a multi-objective memetic algorithm as we will see later on. This chapter is organized as follows. Section 6.2 describes our proposed approach. Our experimental study is presented in Section 6.3. Finally, Section 6.4 provides our final remarks.

6.2 Our Proposed Approach

6.2.1 General Framework

In Chapter 5, we described a new MOEA based on the use of the IGD$^+$ indicator. Here we adopt the same structure of the general framework previously proposed (i.e., the general framework of IGD$^+$-MOEA), which starts with a population $\mathcal{P}_t$ which contains N randomly generated individuals. A new offspring is created by choosing two different parents from $\mathcal{P}$. The parents are recombined using evolutionary operators (we adopted Simulated Binary Crossover (SBX) and Polynomial-based Mutation [85]). Thereafter, the resulting offspring are added to the new set. This process is repeated until having a total of λ offspring. After that, the algorithm combines the parents and the offspring populations to form the so-called Q set. The new population at generation $t+1$ is generated using different selection mechanisms. The pseudo-code of the multi-objective approach is presented in Algorithm 5. Next, we will provide more details of the technique that we propose for selecting the new population.

Algorithm 5 General Framework

```
Input: A MOP ($F$), a stopping criterion and a uniform spread of $N$ reference vectors
    $\mathcal{Z}$.
Output: Approximation of the MOP
 1: $t \leftarrow 0$;
 2: Generate an initial population randomly $(x_i, \ldots, x_N) \in X$ to create $\mathcal{P}_0$ ;
 3: while $t < gen_{max}$ do
 4:     $\mathcal{Q} \leftarrow \mathcal{P}_t$;
 5:     for each new offspring do
 6:         Apply evolutionary operators: Randomly select two parents from $\mathcal{P}_t$ to
            produce $u_i$;
 7:         $\mathcal{Q} \leftarrow \mathcal{Q} \bigcup \{F(u_i)\}$;
 8:     end for
 9:     $\mathcal{P}_{t+1} \leftarrow$ IGD$^+$-based selection mechanism $(\mathcal{Z}, \mathcal{Q})$;
10:     $t \leftarrow t + 1$;
11: end while
12: $\mathcal{Q} \leftarrow$ non-Dominated $\mathcal{P}_t$;
13: return $\mathcal{Q}$;
```

6.2.2 Selection Mechanism

We propose to use two different selection techniques. The first technique transforms the selection mechanism into a Linear Assignment Problem (LAP), which uses different cost functions for defining the LAP. This selection mechanism is the same that uses the IGD$^+$-EMOA (see Chapter 5). In order to explain our technique, we need to provide first more details about the LAP.

The LAP consists choosing an optimal assignment of n items (e.g., jobs) to m machines (or workers). Mathematically, the LAP can be expressed as: Given two sets, $A = \{a_1, \ldots, a_n\}$ and $T = \{t_1, \ldots, t_n\}$ with the same cardinality, and a cost function $C : A \times T \rightarrow \mathbb{R}$ and having $\Phi : A \rightarrow T$ as the set of all bijections between A and T, the LAP can be formulated as follows:

$$\min_{\phi \in \Phi} \sum_{a \in A} C(a, \Phi(a)) \tag{6.1}$$

Normally, the cost of the problem is also described as a squared matrix C, where each element $C_{i,j} = C(a_i, t_j)$ represents the relationship between a_i and t_j. The multi-objective selection mechanism seems to be very related to the LAP. Since the LAP is commonly solved using jobs and workers, here we want to find the best relationship between reference points and candidate solutions.

The cost matrix in terms of a MOP is created as:

$$C_{i,j} = d+(a_i, z_j), \quad i = 1, \ldots, |\mathcal{A}|, \quad j = 1, \ldots, |\mathcal{Z}|. \tag{6.2}$$

where $a_i \in \mathcal{Q}$ is the i^{th} point from the population $\mathcal{Q}$, $z_j \in \mathcal{Z}$ is the reference point and $d+$ is the modified Euclidean distance. Analogously, that process can be combined with the normal Euclidean distance.

In order to solve the LAP, we can make use of the Kuhn-Munkres Algorithm, also known as the Hungarian algorithm, which is able to solve LAP instances in polynomial time $\mathcal{O}(n^3)$ [87] for squared matrices. An extension of this algorithm for rectangular matrices was introduced by Bourgeois [88]. The extension to rectangular matrices allows the algorithm to operate in situations where the numbers of reference points and individuals from the population are not equal. The optimal solution to our assignment problem is found by identifying the combination of values in the cost matrix C resulting in the smallest sum. This solution corresponds to the best relationship between the current points of the population and the elements of a reference set. Example 6.1 (see Figure 6.1) shows a cost matrix with 9 rows and

$$\begin{pmatrix}
0.350 & 0.727 & 0.007 & 0.165 & 0.221 \\
0.007 & 0.943 & 0.223 & 0.381 & 0.064 \\
0.485 & 0.567 & 0.138 & 0.006 & 0.356 \\
0.663 & 0.011 & 0.317 & 0.183 & 0.534 \\
0.130 & 0.884 & 0.163 & 0.322 & 0.005 \\
0.671 & 0.061 & 0.325 & 0.191 & 0.542 \\
0.677 & 0.025 & 0.331 & 0.197 & 0.548 \\
0.349 & 0.726 & 0.006 & 0.164 & 0.219 \\
0.360 & 0.722 & 0.014 & 0.160 & 0.231
\end{pmatrix}$$

Figure 6.1: Initial Cost matrix with 9 solutions and 5 reference points, which was generated using the modified Euclidean distance for DTLZ2.

$$\begin{pmatrix}
0.343 & 0.717 & 0.002 & 0.159 & 0.216 \\
0.000 & 0.933 & 0.217 & 0.375 & 0.059 \\
0.478 & 0.556 & 0.133 & 0.000 & 0.351 \\
0.656 & 0.000 & 0.311 & 0.177 & 0.529 \\
0.124 & 0.873 & 0.157 & 0.316 & 0.000 \\
0.665 & 0.050 & 0.320 & 0.185 & 0.538 \\
0.671 & 0.014 & 0.325 & 0.191 & 0.544 \\
0.342 & 0.715 & 0.000 & 0.158 & 0.215 \\
0.353 & 0.711 & 0.008 & 0.153 & 0.226
\end{pmatrix}$$

Figure 6.2: Final Cost matrix with 9 solutions and 5 reference points, which contains the optimal solution of the LAP for DTLZ2.

5 columns, where the number of columns expresses the cardinality of the reference set. We can see in Figure 6.2 that the i^{th} row from the final cost matrix with zero values represents the selected candidate solution. The optimal solution of the LAP is obtained by selecting the solutions 1, 2, 3, 4 and 7 from the final cost matrix. The smallest sum of the cost matrix is $0.007+0.006+0.011+0.005+0.006 = 0.035$, which represents the best relationship between the reference set and the objective solutions.

Figure 6.3 shows an example of how the selection technique works. In this case, each reference point is represented by a white square and the optimal solution of the LAP is indicated using black circles. Our proposed selection mechanism works in the same manner as the one proposed in Chapter 5, but with one important difference: here, the construction of the reference set is performed at the beginning of the search and remains static (i.e., without changes), whereas we mentioned that it is possible to compute the reference set at each generation.

Finally, we adopted another way to reduce the size of the current population $\mathcal{Q}$ (for more details see [46]). This selection mechanism chooses the N best individuals

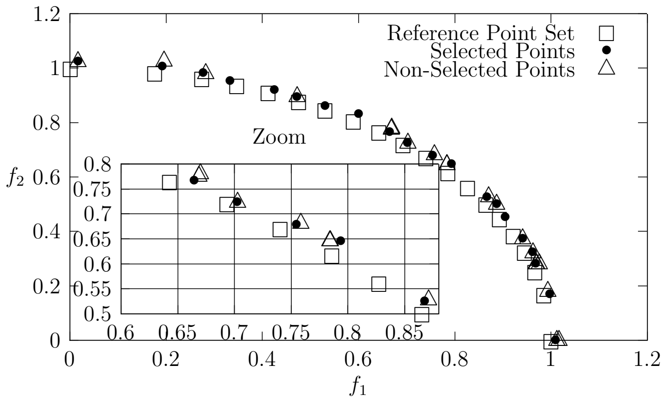


Figure 6.3: Example of our proposed selection mechanism based on IGD$^+$ and the Hungarian Algorithm for DTLZ1 with 20 reference points.

from $\mathcal{Q}$ using any performance indicator such as IGD, Δ_p or IGD$^+$. This selection mechanism works as the one adopted by SMS-EMOA, which discards one solution from the population. In SMS-EMOA, the individual that is removed is the one that minimizes the exclusive contribution of the hypervolume indicator. This algorithm adopts a steady-state selection mechanism. The exclusive contribution of a solution using any reference set indicator (I_x) is described as:

$$EI_x(a, \mathcal{A}, \mathcal{Z}) = I_x(\mathcal{A} \backslash \{a\}, \mathcal{Z}). \tag{6.3}$$

where $\mathcal{A}$ is the population and $\mathcal{Z}$ is the reference set. However, this process works only if the cardinality of the offspring population is one. As mentioned before, this selection mechanism woks with the IGD, Δ_p and IGD$^+$ indicators.

6.2.3 Approximating the Reference Set

There are several methods available for building the reference set. One of them was proposed by Menchaca et al. in [47], and uses ϵ-dominance to establish a lower region. They proposed to use an identification vector for splitting the space into hypercubes. Each component of the vector keeps the ϵ distance, which is established for each dimension of the space. This is a novel approach but, unfortunately, it does not provide solutions with a good (i.e., uniform) distribution. There is another approach, which was explained in Chapter 5, which tries to approximate the reference set using super-spheres.

In this study, we aim to analyze the impact of using different selection mechanisms based on reference sets. For this reason, in order to approximate the reference set,

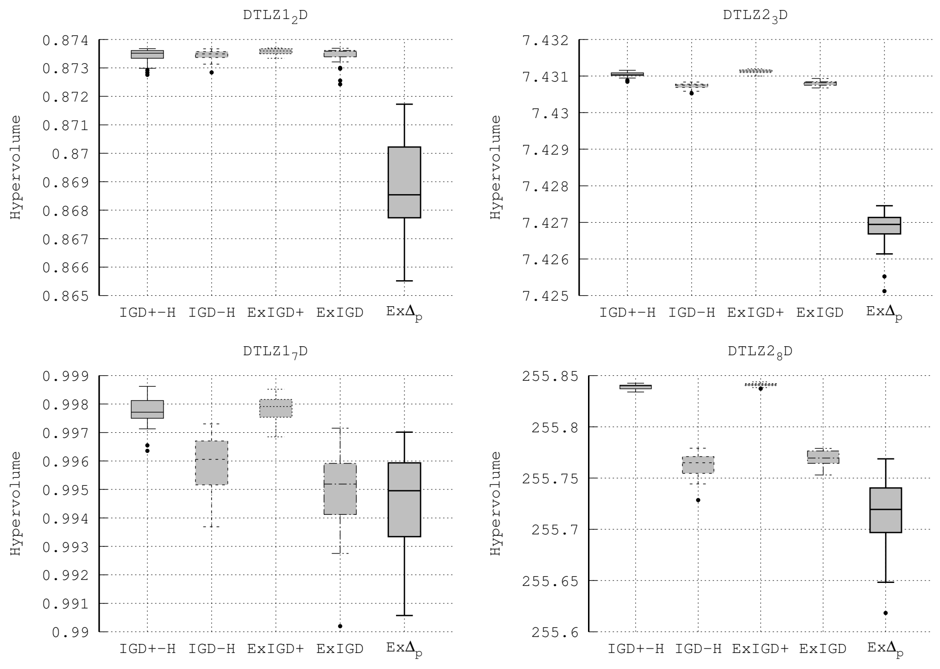


Figure 6.4: Performance comparison among MOEAs, where each plot was obtained from 30 independent executions solving DTLZ1 and DTLZ2 with 2 to 8 objectives.

we used a sample of the true Pareto front, which was randomly generated with 3000 points. The size of the reference set was also reduced to N points. Indeed, we selected the N points that maximize the hypervolume indicator. In order to do this, we adopted a greedy algorithm based on the hypervolume contributions to reduce the number of reference points to a certain specific size.

6.3 Experimental results

We compared the performance of each selection mechanism previously discussed. For this sake, each selection mechanism was included into a general MOEA (see subsection 6.2.1). The parameters of each MOEA used in our study were chosen in such a way that we could do a fair comparison among them and we could adopt the same evolutionary operators for each version of the MOEA.

6.3.1 Test problems

For our comparative study, we adopted the Deb-Thiele-Laumanns-Zitzler (DTLZ) test suite [92]. This set of problems includes aspects such as separability and multi-

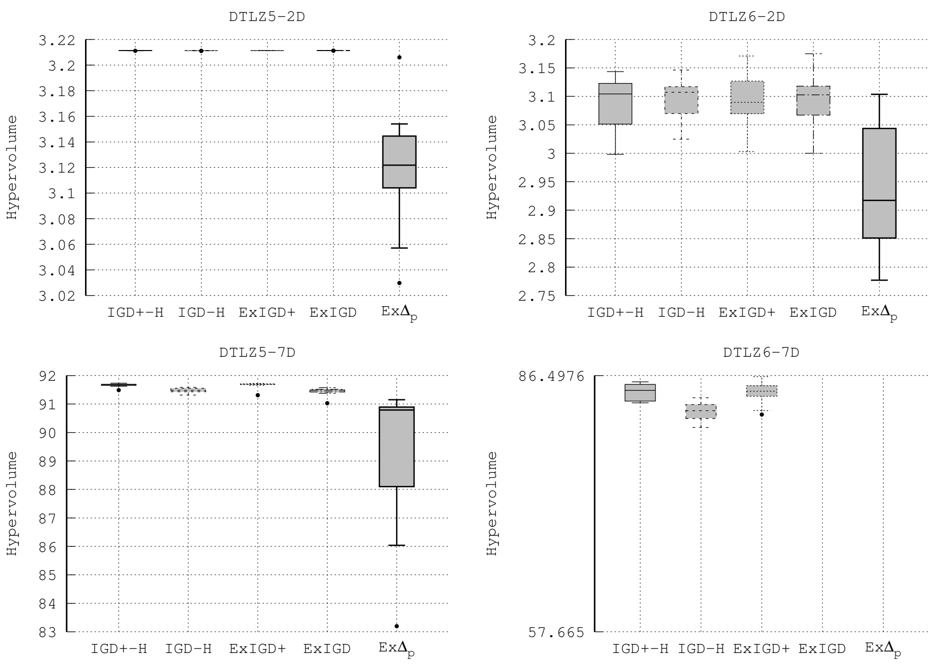


Figure 6.5: Performance comparison among MOEAs, where each plot was obtained by 30 independent executions solving DTLZ5 and DTLZ6 with 2 to 7 objectives.

Table 6.1: Reference points used to compute the Hypervolume indicator for each DTLZ test problem.

Problem	**Reference points**
DTLZ1	$(1, 1, 1, \ldots, 1)$
DTLZ2-6	$(2, 2, 2, \ldots, 2)$

frontality which make them more difficult to solve. We selected problems in such a way that we had different Pareto front shapes such as linear, concave and degenerate linear, since we aimed to observe the impact of each of the selection mechanisms previously discussed.

6.3.2 Methodology

For our comparative study, we decided to adopt the hypervolume indicator, which assesses both convergence to the true Pareto front and maximum spread along it. To compute I_H, we used the reference points shown in Table 6.1. Additionally, we also compared the running time of each version of MOEA, which was measured in minutes.

6.3.3 Parameterization

In the DTLZ test suite, the total number of decision variables is given by $n = m+k-1$, where m is the number of objectives and k was set to 5 for DTLZ1 and to 10 for DTLZ2 to DTLZ6. The number of objectives was set from 2 to 8. The parameters of each MOEA used in our study were chosen in such a way that the MOEAs were able to converge to the true Pareto Front of the test instances adopted. The distribution indexes for the SBX and polynomial-based mutation operators [85], adopted by each MOEA, 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 $p_m = 1/L$, where L is the number of decision variables. The total number of function evaluations was not allowed to exceed 50,000. We used a population size of 110 individuals and we iterated during 450 generations. For each MOEA, we used the same reference set, which was generated for each test problem (DTLZ1 to DTLZ6). Our experiments were run on a computer with an Intel Core i5-3930k processor running at 2.70 GHz, with 8GB of RAM.

6.3.4 Numerical Results

Table 6.2 provides the average hypervolume value over the 30 independent executions of each compared MOEA for each instance of the DTLZ test suite. The best results are presented in **boldface** and grey-colored cells show the second best results. The running time is shown in parentheses. It is clear that the winner in this experimental study is the IGD$^+$-based selection mechanism since the IGD$^+$-based MOEAs were able to outperform both the IGD-based MOEAs and the Δ_p-based MOEA in all the test problems in terms of the hypervolume indicator.

We can observe in Figures 6.4 and 6.5 that the medians and variances of all results are different, which means that the differences obtained are statistically significant. Likewise, we can see that the Δ_p-based MOEA has a lower hypervolume value than both the IGD$^+$-H-based MOEA and the IGD-H-based MOEA. It is worth noting that, since Δ_p adopts both IGD and GD, the GD indicator affects the performance of the Δ_p indicator. The reason is that GD calculates the average distance from each solution to its closest reference point, and this causes the selection mechanism based on the Δ_p indicator to select dominated solutions. For this reason, the Δ_p-based MOEA is not able to converge. As can be observed in Figures 6.4 and 6.5, the Δ_p-based selection mechanism incorporates a change between GD an IGD which makes the variance to increase, since these two selection schemes perform differently.

As shown in Table 6.2, IGD$^+$-H-MOEA and ExIGD$^+$-MOEA were able to outperform IGD-H-EMOA, ExIGD-EMOA and ExΔ_p-EMOA in all cases. These two approaches (IGD$^+$-H-MOEA and ExIGD$^+$-MOEA) obtained similar results. Although IGD-H-EMOA works similarly to IGD$^+$-H-EMOA (since both incorporate the same selection mechanism based on LAP, but have a different cost function), IGD-H-EMOA was not able to converge to the true Pareto front, whereas IGD$^+$-H-EMOA was able to do it. The main reason for this is that the use of the Euclidean distances affects the dominance relation because the calculation of the Euclidean distance is inconsistent with the Pareto dominance relation when the reference point does not dominate the solutions. This makes the IGD-based selection mechanism to choose solutions which are close to the reference set, and avoids selecting non-dominated solutions. ExIGD-EMOA and ExΔ_p-EMOA are unable to converge to the Pareto front in DTLZ5 and DTLZ6 with 5, 6, 7 and 8 objectives since the use of Euclidean distances affects their selection mechanisms.

The main reason for which the IGD$^+$-based MOEAs showed a better performance than MOEAs based on IGD and Δ_p, is the incorporation of the modified Euclidean distance since this distance adopts an inferiority vector, which can be viewed as the minimum amount of the increase from a z reference point so that the result vector is weakly dominated by the objective point. That modification makes possible to consider non-dominated points when the IGD$^+$ indicator is used as our selection mechanism. The modified Euclidean distance solves the drawbacks of IGD and Δ_p. We showed that the use of the modified Euclidean distance significantly improves the performance of the selection mechanism. Notwithstanding, the running time of ExIGD$^+$-EMOA is higher than that of IGD$^+$-H-EMOA.

Table 6.2 indicates that IGD$^+$-H-EMOA has the lowest running times. However, ExIGD$^+$-EMOA was able to solve the test problems adopted in a reasonably low running time (particularly for the instances having 2 and 3 objectives).

Figures 6.6 and 6.7 present a graphical representation of the running time of each MOEA for DTLZ1 and DTLZ5. These plots correspond to the average time value from 30 independent executions. We can observe that the LAP reduces the running time of a MOEA since the optimal solution of the LAP guarantees the best relationship between the reference point and the solutions. This allows this selection mechanism to converge faster than the use of the Exclusive-based selection mechanism.

Thus, IGD$^+$-H-EMOA is computationally cheaper than ExIGD-EMOA, ExIGD$^+$-EMOA and ExΔ_p-EMOA. It is worth indicating that IGD$^+$-H-EMOA is able to

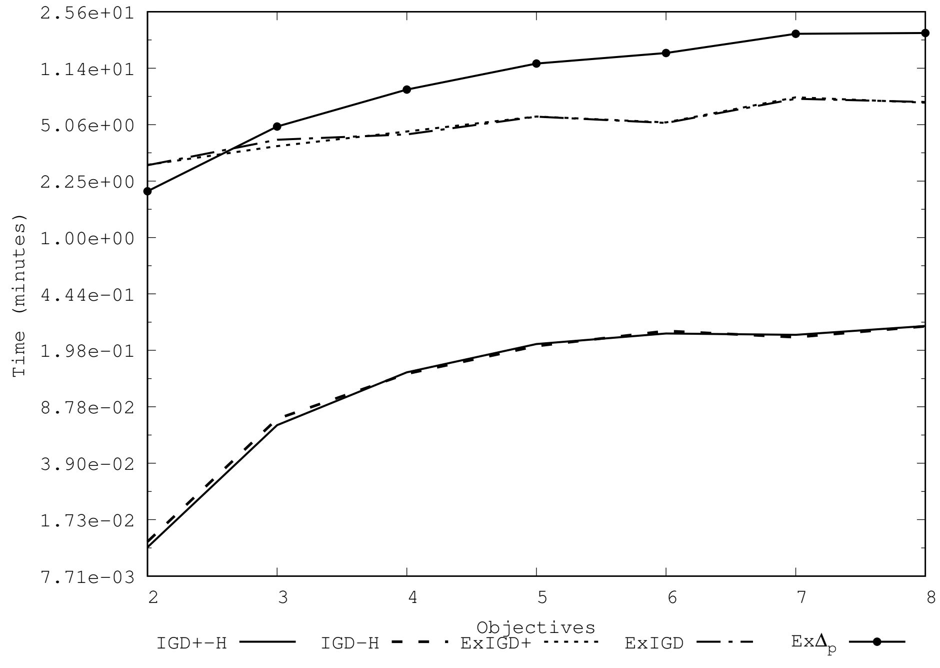


Figure 6.6: Graph showing the running time of each MOEA for DTLZ1.

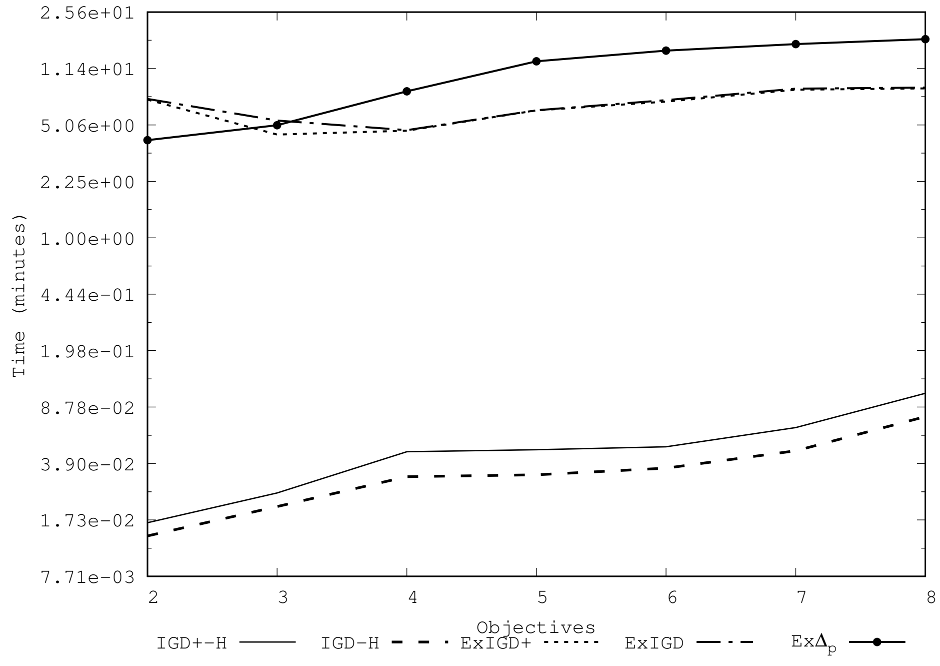


Figure 6.7: Graph showing the running time of each MOEA for DTLZ5.

achieve a significant speedup (of up to 200x) with respect to ExIGD$^+$-EMOA. This confirms that the use of a selection mechanism based on IGD$^+$ is an effective way to solve some many-objective problems.

	IGD+-H-EMOA	**IGD-H-EMOA**	**ExIGD+-EMOA**	**ExIGD-EMOA**	**ExΔ_p-EMOA**
m			DTLZ1		
2	0.87343 (0.0117 min)	0.87345 (0.0126 min)	**0.87358** (2.8401 min)	0.87345 (2.8238 min)	0.86872 (1.9446 min)
3	0.97371 (0.0674 min)	0.97338 (0.0738 min)	**0.97386** (3.7193 min)	0.97351 (4.075 min)	0.97266 (4.9237 min)
4	0.99379 (0.1442 min)	0.99353 (0.1411 min)	**0.99386** (4.5933 min)	0.99308 (4.3992 min)	0.99121 (8.3803 min)
5	0.99371 (0.2163 min)	0.99275 (0.2103 min)	**0.99404** (5.6831 min)	0.99223 (5.6811 min)	0.99045 (12.1779 min)
6	0.99054 (0.2517 min)	0.98847 (0.2612 min)	**0.99057** (5.2241 min)	0.98726 (5.1929 min)	0.98831 (14.1643 min)
7	0.99776 (0.2471 min)	0.99592 (0.2386 min)	**0.99783** (7.4982 min)	0.99479 (7.3405 min)	0.99462 (18.6716 min)
8	**0.99586** (0.2801 min)	0.99344 (0.2792 min)	0.9958 (6.9452 min)	0.9921 (7.0153 min)	0.99168 (18.8965 min)
			DTLZ2		
2	3.21147 (0.0125 min)	3.21151 (0.0113 min)	**3.21156** (4.7688 min)	3.21153 (4.7642 min)	3.12112 (3.8215 min)
3	7.43104 (0.0301 min)	7.43073 (0.0276 min)	**7.43113** (5.9651 min)	7.4308 (7.2221 min)	7.42677 (8.182 min)
4	15.58708 (0.0451 min)	15.5851 (0.0396 min)	**15.58732** (6.2784 min)	15.5852 (6.3927 min)	15.58049 (9.9203 min)
5	31.69208 (0.0559 min)	31.68537 (0.0477 min)	**31.69251** (8.1911 min)	31.68585 (8.4751 min)	31.68244 (13.9195 min)
6	63.76177 (0.0608 min)	63.74695 (0.0482 min)	**63.76286** (9.2453 min)	63.74789 (9.2507 min)	63.71765 (18.4489 min)
7	127.81248 (0.0725 min)	127.78877 (0.054 min)	**127.81384** (10.7463 min)	127.7894 (10.9737 min)	127.76065 (20.8394 min)
8	255.83892 (0.0756 min)	255.76216 (0.059 min)	**255.84114** (11.7623 min)	255.76964 (18.0689 min)	255.71304 (21.9143 min)
			DTLZ5		
2	3.21127 (0.0167 min)	3.21123 (0.0138 min)	3.21131 (7.2897 min)	**3.21131** (7.3568 min)	3.132 (4.0653 min)
3	6.1043 (0.0255 min)	6.10427 (0.021 min)	**6.10436** (4.4098 min)	6.10441 (5.388 min)	5.92074 (5.0434 min)
4	12.00938 (0.0462 min)	12.00646 (0.0323 min)	**12.00975** (4.6598 min)	12.00786 (4.6972 min)	11.92451 (8.201 min)
5	23.82496 (0.0476 min)	23.81727 (0.0332 min)	**23.82717** (6.24 min)	23.81817 (6.2444 min)	23.75192 (12.6418 min)
6	47.39216 (0.0496 min)	47.35934 (0.0365 min)	**47.39692** (7.0745 min)	47.36511 (7.2339 min)	46.70794 (14.7261 min)
7	**91.66916** (0.0653 min)	91.47108 (0.0468 min)	91.66848 (8.3923 min)	91.45174 (8.5274 min)	89.56395 (16.1746 min)
8	**145.92603** (0.107 min)	145.63295 (0.0765 min)	145.87716 (8.555 min)	135.55127 (8.6648 min)	126.21945 (17.3731 min)
			DTLZ6		
2	3.08569 (0.0303 min)	3.0933 (0.031 min)	**3.09634** (3.7122 min)	3.08765 (3.6532 min)	2.9441 (6.0076 min)
3	**5.9579** (0.2248 min)	5.91691 (0.1842 min)	5.94636 (2.4423 min)	5.85907 (2.4041 min)	5.72483 (5.5973 min)
4	11.50411 (0.3443 min)	11.53968 (0.2563 min)	**11.62126** (3.9025 min)	0 (3.9697 min)	11.92451 (7.6791 min)
5	21.77326 (0.3251 min)	21.6063 (0.2376 min)	**22.29951** (5.6863 min)	0 (5.6109 min)	0 (12.0119 min)
6	41.68516 (0.3037 min)	40.5511 (0.2284 min)	**41.98876** (6.5304 min)	0.56907 (6.7096 min)	2.36595 (13.8672 min)
7	**84.29418** (0.3171 min)	81.75814 (0.2262 min)	84.2936 (7.5512 min)	0 (7.441 min)	0 (15.1196 min)

Table 6.2: Results obtained in the DTLZ test problems by each MOEA. We compared the performance of each MOEA using the hypervolume indicator. The values in parentheses correspond to the computational time (measured in minutes) required by each execution of the MOEAs compared. This table was obtained from 30 independent runs. All the algorithms were compiled using the GNU C compiler and they were executed on the same computer.

6.4 Final Remarks

We have proposed several selection mechanisms for indicator-based MOEAs which use a reference set. The core idea of our proposed algorithm is to adopt the IGD$^+$ performance indicator in the selection mechanism of a MOEA. Here, we showed that the use of the modified Euclidean distance significantly improves the performance of the selection mechanism. Additionally, the transformation of the selection mechanism into a LAP reduces the running time, which makes possible a significant speedup (of up to 200x). Our experimental results showed that a selection mechanism based on Δ_p has some drawbacks when it tries to solve problems with degenerate Pareto fronts. This selection mechanism was not able to solve degenerate multi-objective problems with more than 5 objectives. As can be observed, the Pareto compliant property between two-objective vectors is of utmost importance and improves the performance of the selection mechanism of a MOEA. Our preliminary experimental results showed that our proposed IGD$^+$-based selection mechanism is an effective way to lead the search to a single region of the objective space. Thus, we will design a local search engine based on the IGD$^+$ indicator. In the next chapter, we elaborate on this assumption, and we show that it is possible to create a multi-objective memetic algorithm based on the IGD$^+$ indicator.

Chapter 7

Multi-Objective Memetic Algorithm based on the IGD$^+$ Indicator

In this chapter, we propose a new Multi-Objective Memetic Algorithm (MOMA) which uses a Local Search technique (LS) based on the modified inverted generational distance (IGD$^+$) combined with a hypervolume-based global optimizer [38]. We want to combine different properties of each indicator for improving the performance of the overall MOMA. This is possible, since these indicators have nice properties (i.e., hypervolume is Pareto compliant and IGD$^+$ is weakly Pareto compliant). However, the main drawback of the hypervolume is the high computational cost associated with its computation. This high computational cost limits the use of this indicator, particularly in problems having many objectives. On the other hand, IGD$^+$ has a very low computational cost, even in high dimensional problems. As mentioned before (see Chapter 6), the IGD$^+$ indicator is able to lead the optimization process toward the Pareto front, and it is possible to use this indicator for leading the optimization of the local search engine. In spite of the fact that this hybridization is possible (i.e., hybridize the hypervolume indicator and the IGD$^+$ indicator), there are still some drawbacks which limit the use of this type of combination, since computing the exact hypervolume contribution is highly costly.

Nowadays, this sort of limitations can be addressed by using massive parallel processors such as Graphics Processing Units (GPUs). There is plenty of evidence that indicates that GPU-based approaches can reduce the running time without losing the advantages of CPU-based approaches (for more details see [4, 5, 6]). For this reason, we develop here a parallel implementation of our MOMA and illustrate its performance when using both indicators. We hypothesized that an appropriate combination could improve the performance of a MOMA and the experiments reported in this chapter validate our hypothesis.

This chapter is organized as follows. Section 7.2 describes our proposed approach. The experimental study is presented in Section 7.3. Finally, Section 7.4 provides our final remarks.

7.1 Graphics Processing Units

Before explaining our proposed approach we need to provide more details about Graphics Processing Units (GPUs), which are many-core processors capable of achieving high-performance computing [94]. More specifically, GPUs are well-suited for addressing problems that can be expressed as data-parallel computations, since the same program is executed on many data elements in parallel. Data-parallel processing maps data elements to parallel processing threads. Many applications that process large data sets can use a data-parallel programming model to speed up the computations. Some real-world applications solved using GPUs, are 3D rendering, video encoding and decoding, image scaling, stereo vision, and pattern recognition. Although GPUs were created accelerate image and video processing, it is possible to use them to tackle other types of problems (i.e., numerical application problems, which typically appear of science and engineering). Nowadays, GPUs are general-purpose processors with specially designed APIs like CUDA [95] and OpenCL [96]. The main key abstractions in the GPU are: (1) a hierarchy of thread groups, (2) a hierarchy of memory and (3) barrier synchronization. These abstractions provide fine grained data parallelism and thread parallelism, those are nested within coarse grained data parallelism and task parallelism. GPUs are able to split the problem into coarse sub-problems that can be solved independently in parallel by blocks of threads, where each sub-problem can be solved cooperatively in parallel by all threads within the block. Here, we propose to incorporate the use of GPUs into a multi-objective memetic algorithm for decreasing its running time. Next, we will provide more details about the way in which our proposal works.

7.2 Multi-Objective Memetic Algorithm

Our MOMA consists of two different approaches. The first one is a global optimizer which is based on SMS-EMOA [38]. The second method is our local search (LS) technique which uses an IGD$^+$-based search engine. A local search method can improve a candidate solution very quickly, but its aims to find only local optima (i.e., solutions which are optimal in a certain region of the objective space close to the candidate solution from which the local search is launched). On the other hand, a global optimizer is meant to explore the whole search space for finding different regions, and eventually, the global optimum (or a good approximation of it).

7.2.1 Our Global Optimizer

Our global optimizer starts with an initial population of N individuals. Then, a new individual is created through the use of evolutionary operators (i.e., SBX and polynomial-based mutation). This new individual will become a member of the next population, if replacing an existing individual leads to a higher quality of the population with respect to the hypervolume contribution. Afterwards, one individual is discarded from the worst ranked front in order to maintain the same population size. If the cardinality of this front is larger than 1, the individual which minimizes the hypervolume contribution is eliminated. The LS technique is launched when a certain percentage of the total number of generations is reached. Next, we will provide more details of the way in which our LS works. The pseudo-code of our proposed memetic approach is presented in Algorithm 7.

Algorithm 6 General Framework

```
Input: A MOP, a stopping criterion and a uniform spread of $N$ reference vectors
Output: Approximation of the MOP
 1: $P_0 \leftarrow$ init();
 2: $t \leftarrow 0$;
 3: while $t < gen_{max}$ do
 4:     $Q_{t+1} \leftarrow$ generate($P_t$); /*Generate offspring using evolutionary operators*/
 5:     /*Hypervolume-based selection mechanism for finding $N$ best individuals*/
 6:     $P_{t+1} \leftarrow$ reduce($P_t \cup \{Q_{t+1}\}$);
 7:     if $PL > P_{percent}$ then
 8:         /*Applying Local Search Engine*/
 9:         $P_{t+1} \leftarrow$ LS Engine($P_{t+1}$);
10:     end if
11:     $t \leftarrow t + 1$;
12: end while
13: return $P_t$;
```

7.2.2 Our Local Search Engine

We can see that if we attempt to find the solution of a MOP, the resulting points in $\mathcal{PF}$ are clustered in various regions of objective space. Thus, it is possible to compute new points using local search engines which exploit certain regions of objective space. Local search techniques have been proposed to lead the search within a certain region towards the $\mathcal{PF}$ more effectively and efficiently.

We focused on how to select the k^{th} solution to which LS should be applied. A straightforward solution is to apply LS to all the individuals in the population. Although this involves a higher computational cost, in our case, this sort of scheme is possible using a GPU-based implementation. Thus, our proposal is to apply several local search engines on different regions of the search space, which are specified by a clustering technique, based on the IGD$^+$ indicator. It is worth noting that this indicator requires a reference set $\mathcal{Z}$. Our proposed approach creates different neighborhoods for each point in the reference set. The i^{th} neighborhood is created by N points from the population. Such points have the nearest distance with respect to the i^{th} reference point in terms of the $d+$ distance (see equation (3.11)). Our LS technique starts with a population $\mathcal{P}$ which contains N individuals obtained by our global search engine. The new i^{th} offspring is created by choosing three different parents from its neighborhood. The parents are recombined using the differential evolution operator, where the first parent is selected by the nearest distance in terms of the $d+$ distance and the rest of the parents are randomly chosen. The second step is to combine the parents and the offspring of each neighborhood to form the so-called Q set. The new population at generation $t + 1$ is generated by finding the nearest point from Q for each z reference point in $\mathcal{Z}$. This process is repeated until the stopping criterion is satisfied (we use a maximum number of iterations as our stopping criterion).

7.2.3 Reference Set

As mentioned in Chapter 5, we can approximate the geometrical shape of certain types of Pareto Fronts (PFs) using superspheres. In order to build the reference set, we adopt the same technique presented in Chapter 5, thus we assume that we have a set of weight vectors which is used to construct the reference set. We need to find the γ-value which will be used to transform the weights set into the reference set. Clearly, in order to find the γ-value, equation (5.4) (for more details see Chapter 5) would become a root-finding problem and we can say that the γ-value needs to satisfy:

$$y_1^\gamma + \cdots + y_m^\gamma - 1 = 0 \tag{7.1}$$

For solving equation (7.1), we use Newton's method for approximating the γ-value. Now, we can see that the next approximation to the root is defined as:

$$\gamma_{k+1} = \gamma_k - \frac{(\sum_{j=1}^m y_j^{\gamma_k}) - 1}{\sum_{j=1}^m y_j^{\gamma_k} \log(y_j)} \tag{7.2}$$

Let $\mathcal{Q}$ be the current set which was created combining the parent and offspring population. Thus, the reference set is created by Algorithm 7. We added some new operations for building the reference point set. Particularly, we added to provide expand and translate operations.

Algorithm 7 Computation of the reference set which is based on supersphere curves

Input: A current set $\mathcal{Q} \subset \mathbb{R}^m$, a set of weighted vectors $W \subset \mathbb{R}^m$, where m is the number of objectives, expand value $e \subset \mathbb{R}$ and translate value $t \subset \mathbb{R}$

Output: The reference set $\mathcal{Z}$ which is the best approximation of the set Q

```
 1: Find the nondominated points from $\mathcal{Q}$ and
    save to $\mathcal{Q}'$
 2: for each $\vec{p} \in \mathcal{Q}'$ do
 3:     for each $\vec{w} \in \mathcal{W}$ do
 4:         Compute $d^\perp(\vec{p}, \vec{w}) = \| \vec{p} - \vec{w}^T \vec{p} \vec{w} / \| \vec{w} \|^2 \|$
 5:     end for
 6:     Assign $r(w) = \operatorname{argmin}_{\vec{p} \in \mathcal{Q}'} \; d^\perp(\vec{p}, \vec{w})$
 7: end for
 8: $j \leftarrow 0$
 9: for each $\vec{w} \in \mathcal{W}$ do
10:     $stepsize \leftarrow \vec{p}_{r(\vec{w})} \cdot \vec{w} / \| \vec{w} \|^2$
11:     $\vec{y} \leftarrow stepsize * \vec{w}$
12:     Approximate the $\gamma$ value using equation (7.1)
13:     Compute the supersphere point as $z_{j,k} \leftarrow e(w_{j,k}^\gamma) - t$ for all $j = 1, \ldots, m$
14:     $j \leftarrow j + 1$
15: end for
```

In the first step of the algorithm, we find the non-dominated points from set $\mathcal{Q}$ which will establish the non-dominated region. After that, in the first loop, we search the nearest perpendicular distance between each weighted vector $\vec{w}$ and the non-dominated points (we find the best relationship between each weighted vector and each non-dominated point). In order to construct the reference surface, we project the nearest non-dominated point to a specific weighted vector $\vec{w}$. Once this is done, we can search the γ-value using Newton's method, which is described by equation (7.2). Finally, the reference point is computed using the γ-value. After that, we apply the expand and translate operations. These operations transform the surface for spreading the reference set along the objective space. We can see that this process is considered as a generation and is repeated for each weighted vector. For generating the weighted vectors, we adopted Das and Dennis' approach [90] and the number of weighted vectors was set to N. However, it is possible to adopt another algorithm for creating the weighted vectors.

7.2.4 Parallel Multi-Objective Memetic Algorithm

The main idea of our parallel implementation is to use all the available hardware resources for improving the performance of our proposed MOMA. For this reason, in order to simplify the parallelization we focused only on the most time-consuming parts of the algorithm. Our implementation is based on two different parallel implementations, one for handling the local search technique, and another one, which is responsible of computing the hypervolume contribution from the global search engine. As we indicated in Section 7.2.2, the LS procedure is composed by a clustering technique, a procedure for generating new offspring as well as one for the evaluation of the objective functions. This process is repeated for a certain number of iterations. The main idea is to apply the LS technique to all the individuals of the population. For this reason, the parallelization of this procedure is done in the following way: we adopt a SIMD[1] model to apply the clustering technique to create each sub-region on the objective space at the same time. Thus, this procedure creates different blocks of threads, where each thread computes the $d+$ value (see equation (3.11)) between each reference point in $\mathcal{Z}$ and each current point in the population $\mathcal{Q}$. After that, each block searches the nearest distance (this process is repeated until having b elements for building the clustering region). After this is done, we create m offspring, each of them residing in a specific sub-region (the i^{th} neighborhood) using a thread of the GPU for each of them. Thus, each thread in the block can assess the new offspring in the neighborhood. It is worth mentioning that this process needs to normalize all points for each generation of the local search technique, in order to handle objectives having different units.

[1]SIMD (Single Instruction Multiple Data) is a computer architecture which can handle only one instruction but applies it to many data streams simultaneously [97].

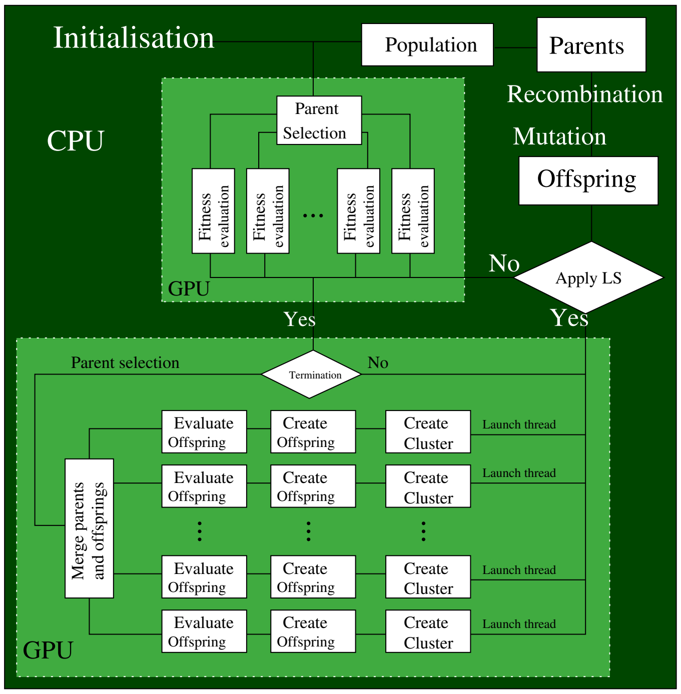


Figure 7.1: Block diagram of GPU-based MOMA.

In [4], the use of a GPU-based approach showed that it is possible to find a good approximation of MOPs using the hypervolume indicator as a selection mechanism without losing the advantages of a sequential approach. For this reason, we adopted this approach for implementing the second part of our MOMA.[2] Figure 7.1 describes the flow of our proposed parallel proposed approach, which gives more details on how the GPU-based implementation works.

7.3 Experimental Results

We compare the performance of our memetic algorithm with respect to SMS-EMOA which has two different variants. The first version uses exact calculation of the hypervolume contribution for each generation of the search process. The second version incorporates the algorithm proposed in [91] for estimating the hypervolume using Monte Carlo sampling, instead of the exact hypervolume calculations adopted in the original implementation of SMS-EMOA. Our MOMA was compared with respect to its GPU-based implementation. Our proposed approach was implemented in CUDA-C.[3]

[2]The GPU-based approach computes in a faster way the hypervolume contribution of a point.
[3]The GPU platform and API developed by Nvidia called CUDA [95] (Computer Unified Device Architecture), which is the one adopted in this work, is based on the CUDA-C language, which is

7.3.1 Test problems

For our comparative study, we adopted two benchmarks: (1) the Deb-Thiele-Laumanns-Zitzler (DTLZ) test suite [92] and (2) the Walking-Fish-Group (WFG) test suite [93]. These problems include different aspects which make them more difficult to solve (for more details see [92, 93]).

7.3.2 Methodology

For our comparative study, we decided to adopt the hypervolume indicator, which assesses both convergence and maximum spread along the Pareto front. In order to compute I_H, we used different reference points for each test suite, which were set to $(1, \ldots, 1)$ for DTLZ1, $(2, \ldots, 2)$ for DTLZ2 to DTLZ6, $(2, \ldots, 2, 7)$ for DTLZ7 and $(3, 5, \ldots, 2m + 1)$ for the WFG test problems. Additionally, we also compared the running time of each MOEA, which was measured in minutes. We also incorporated the speedup for comparing our GPU-based MOMA with each of its competitors.

7.3.3 Parameterization

For the DTLZ test suite, the total number of decision variables is given by $n = m + k - 1$, where m is the number of objectives and k was set to 5 for DTLZ1, to 10 for DTLZ2 to DTLZ6 and to 20 for DTLZ7. The number of decision variables in the WFG test problems was set to 24, and the position-related parameter was set to $m - 1$. Instances with two and three objectives were adopted.

The parameters of each MOEA used in our study were chosen in such a way that we could do a fair comparison among them. The distribution indexes for the SBX and polynomial-based mutation operators [85] 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 $p_m = 1/L$, where L is the number of decision variables. In the SMS-EMOA-HyPE, the number of samples was set to 50,000. The number of generations of the LS technique was set to 50 for the DTLZ test problems and to 80 for the WFG test problems, where at each generation, the LS is applied for each reference point. The control parameter F was set to 0.5 for the differential evolution operator. The total number of function evaluations was set in such a way that it did not exceed 30,000 for the DTLZ test problems and 50,000 for the WFG test suite. All the implementations were tested on the same computer which has the following

an extension of C that allows the development of GPU routines called *kernels*. Each kernel defines instructions that are executed on the GPU by many threads at the same time.

Problem	m	SMS-EMOA	SMS-EMOA-HYPE	IGD+-MA	IGD+-MA(GPU)
	Test Suite 1				
DTLZ1	2	**0.8732805**	0.8725481	0.8726563	0.8709863
	3	**0.974249**	0.9666142	0.9737887	0.9731913
DTLZ2	2	3.2109678	3.2095071	**3.2109715**	3.2109601
	3	**7.4313536**	7.4260692	7.4312298	7.4312795
DTLZ3	2	1.9302655	2.7552999	2.8205047	**2.8444797**
	3	6.8129142	5.0821156	**7.0065842**	6.9233977
DTLZ4	2	**2.9687737**	2.8466417	2.9082368	2.9478005
	3	6.927273	6.9031298	6.9659859	**7.0188906**
DTLZ5	2	3.2109635	3.2095599	3.2109646	**3.210965**
	3	**6.1052922**	6.1009119	6.1050065	6.1050063
DTLZ6	2	**3.0727714**	3.0898	2.9075032	2.8973674
	3	**5.6964296**	5.2659912	5.2550039	5.2817297
DTLZ7	2	**4.4180206**	4.3527739	4.4174787	4.417529
	3	**12.8437627**	12.7603802	7.878233	7.5641662
	Test Suite 2				
WFG1	2	7.0150395	6.6915866	**7.4293168**	7.3778004
	3	**62.566032**	53.1739159	62.4667318	62.4431766
WFG2	2	11.4297746	11.4126833	**11.4304887**	11.429895
	3	100.9053244	100.3897934	**100.9423556**	100.8915152
WFG3	2	10.9301202	10.8957265	**10.9346344**	10.9320503
	3	76.0218553	74.3412533	76.0301204	**76.0650755**
WFG4	2	**8.6759874**	8.6474796	8.6749735	8.6751788
	3	**77.3490714**	76.1356581	77.2287144	77.236047
WFG5	2	8.2444335	8.2422967	8.2653013	**8.2702657**
	3	**74.1569177**	73.3959324	74.1328251	74.1289772
WFG6	2	**8.3786401**	8.3522619	8.3785062	8.3762176
	3	74.5010368	73.4821868	74.5165829	**74.6646198**
WFG7	2	8.685331	8.6549507	**8.6863782**	8.6863691
	3	**77.6304566**	76.4916201	77.5775899	77.5752613
WFG8	2	8.3184115	8.2791368	**8.3251368**	8.3208976
	3	**73.6151505**	72.5266533	73.5156236	73.5167815
WFG9	2	**8.5957132**	8.4786182	8.5693595	8.5555043
	3	76.279433	73.9882086	76.3432385	**76.3733424**

Table 7.1: Comparison of results for each test suite, using the average hypervolume indicator.

characteristics: an Intel Core i7-3930k CPU running at 3.20 GHz, with 8GB of RAM 1600 MHz DDR3. Our GPU was a Geforce GTX 680, and we ran our experiments in Fedora 18 (64-bit version).

7.3.4 Numerical Results

Table 7.1 provides the average hypervolume over the 30 independent executions of each approach for each test suite. Additionally, we show the average time, which was measured in minutes, needed to perform the maximum number of function evaluations in each case and the speedup achieved (in parentheses). The best results are presented in **boldface** and the grey-colored cells indicate the second best results.

It is clear that the winner in this experimental study is our GPU-based MOMA

Problem	**m**	**SMS-EMOA**	**SMS-EMOA-HYPE**	**IGD+-MA**	**IGD+-MA(GPU)**
		Test Suite 1			
DTLZ1	2	0.2353 (2.22x)	1.1378 (10.74x)	0.1571 (1.48x)	**0.1059**
	3	1.3434 (2.54x)	0.7481 (1.41x)	0.9396 (1.78x)	**0.5290**
DTLZ2	2	0.3145 (2.36x)	5.1541 (38.69x)	0.2720 (2.04x)	**0.1332**
	3	3.5239 (2.26x)	14.3901 (9.21x)	2.9234 (1.87x)	**1.5616**
DTLZ3	2	0.1349 (1.33x)	0.2886 (2.85x)	0.1488 (1.47x)	**0.1014**
	3	1.2725 (1.95x)	1.2460 (1.91x)	0.8001 (1.22x)	**0.6534**
DTLZ4	2	0.2803 (2.11x)	3.6041 (27.11x)	0.2221 (1.67x)	**0.1329**
	3	2.9593 (2.67x)	10.6125 (9.59x)	2.0486 (1.85x)	**1.1070**
DTLZ5	2	0.3151 (2.38x)	5.1250 (38.66x)	0.2710 (2.04x)	**0.1325**
	3	2.2238 (2.4x)	10.8695 (11.71x)	1.4279 (1.54x)	**0.9283**
DTLZ6	2	0.1501 (1.56x)	0.6505 (6.76x)	0.1114 (1.16x)	**0.0962**
	3	1.7579 (2.51x)	4.2780 (6.12x)	1.2497 (1.79x)	**0.6993**
DTLZ7	2	0.3111 (2.66x)	3.5508 (30.4x)	0.2026 (1.74x)	**0.1167**
	3	2.8511 (3.31x)	11.2552 (13.07x)	1.6282 (1.89x)	**0.8611**
		Test Suite 2			
WFG1	2	0.4365 (2.03x)	1.9709 (9.17x)	0.3661 (1.7x)	**0.2149**
	3	6.1682 (3.25x)	17.4731 (9.2x)	4.5742 (2.41x)	**1.8995**
WFG2	2	0.6315 (2.64x)	3.8164 (15.94x)	0.4441 (1.86x)	**0.2393**
	3	7.1067 (3.46x)	4.9679 (2.42x)	5.1835 (2.52x)	**2.0555**
WFG3	2	0.6760 (2.51x)	4.9836 (18.51x)	0.6065 (2.25x)	**0.2692**
	3	6.4021 (2.75x)	17.4964 (7.53x)	5.7263 (2.46x)	**2.3238**
WFG4	2	0.7430 (2.78x)	7.4647 (27.92x)	0.5865 (2.19x)	**0.2673**
	3	8.5252 (3.22x)	13.9963 (5.29x)	5.8787 (2.22x)	**2.6452**
WFG5	2	0.7245 (2.49x)	9.0330 (31.02x)	0.7122 (2.45x)	**0.2912**
	3	8.1814 (3.15x)	14.4043 (5.55x)	5.9147 (2.28x)	**2.5967**
WFG6	2	0.5864 (2.31x)	6.1397 (24.17x)	0.5381 (2.12x)	**0.2540**
	3	6.0747 (2.8x)	12.14762 (5.6x)	5.1019 (2.35x)	**2.1701**
WFG7	2	1.1146 (3.32x)	12.5224 (37.31x)	0.8211 (2.45x)	**0.3356**
	3	8.4301 (2.53x)	19.5875 (5.89x)	7.6072 (2.29x)	**3.3255**
WFG8	2	0.5485 (2.43x)	3.9599 (17.56x)	0.4466 (1.98x)	**0.2255**
	3	4.6612 (2.69x)	8.9829 (5.18x)	4.6498 (2.68x)	**1.7358**
WFG9	2	0.9392 (2.91x)	10.4772 (32.51x)	0.7952 (2.47x)	**0.3222**
	3	8.8769 (2.67x)	18.8293 (5.67x)	7.8688 (2.37x)	**3.3232**

Table 7.2: Computational time (measured in minutes) required by each execution of the MOEAs compared. In the parentheses we show the speedup achieved.

in terms of CPU time. We are also able to obtain the same results as the sequential version, which verifies that our parallel implementation is working as expected (see Table 7.2). We can see that our MOMA is able to converge faster than SMS-EMOA on some test problems (e.g., in the multi-frontal problems) and it outperforms SMS-EMOA-HYPE in all instances. This confirms that our proposed IGD$^+$-based LS is an effective way to solve MOPs. It is worth noting, however, that for DTLZ5, DTLZ6 and DTLZ7, SMS-EMOA performs better than our MOMA. The reason is probably that the true Pareto front of these problems is linear and disconnected, which makes the approximations produced by our approach to converge to a single region of the search space.

7.4 Final Remarks

We have proposed a new Multi-Objective Memetic Algorithm which has an IGD$^+$-based local search engine. The core idea of our proposed algorithm is to combine properties of two different performance indicators. Our proposal includes a GPU-based implementation which makes it possible to launch multiple local search processes at the same time. We showed that the used of GPUs helps to improve the performance of MOMA and avoids to select on which solutions we need to launch the local search engine. Our results indicate that it is possible to improve the convergence of a hypervolume-based approach in multi-frontal problems. Our proposed GPU-based multi-objective memetic algorithm is able to achieve a significant speedup (of up to 38x) with respect to SMS-EMOA. The proposed approach has a few drawbacks on some problems, because computing the IGD$^+$ indicator requires a reference point set. Thus, the behavior of our proposed approach depends on the approximation of the reference point set. For this reason, we aim to overcome this limitation and we will provide more details on how to do it in the next chapter.

Chapter 8

An Improved Version of a Reference-Based Multi-Objective Evolutionary Algorithm based on IGD$^+$

In recent years, the design of new selection mechanisms has become a popular trend in the development of Multi-Objective Evolutionary Algorithms (MOEAs). This trend has been motivated by the aim of maintaining a good balance between convergence and diversity of the solutions. Reference-based selection is, with no doubt, one of the most promising schemes in this area. However, reference-based MOEAs are known to have difficulties for solving multi-objective problems with complicated Pareto fronts (i.e., multi-objective problems with irregular Pareto front shapes), mainly because they rely on the consistency between the Pareto front shape and the distribution of the reference weight vectors. In this chapter, we propose an improved version of a reference-based MOEA, which uses the modified Inverted Generational Distance (IGD$^+$) indicator. The proposed approach adopts a novel method for approximating the reference set, based on an hypercube-based method. Our reference-based method is able to sample uniformly any Pareto front shape, which is incorporated in the IGD$^+$-EMOA. We addressed the drawbacks of the original IGD$^+$-EMOA (see Chapter 5 for more details), and our results indicate that our proposed approach is able to obtain solutions of a similar quality to those obtained by RVEA [42], MOEA/DD [41], NSGA-III [40] and MOMBI-II [49] in several test problems traditionally adopted in the specialized literature. Our proposed approach is able to outperform these other algorithms in more than 50 percent of the 18 benchmark problems adopted.

This chapter is organized as follows. Our proposed approach is described in Section 8.2. Then, Section 8.3 shows our experimental study. Finally, Section 8.5 provides our final remarks.

8.1 Introduction

As mentioned before, for several years, MOEAs adopted selection mechanisms based on Pareto optimality. However, it has been found that Pareto-based MOEAs can not properly solve many-objective problems (problems with more than three objectives) [35]. This has motivated the development of new strategies for dealing with many-objective problems such as reference-based MOEAs [42, 40, 98, 41]. Reference-based MOEAs can be classified into two main groups: (1) decomposition-based MOEAs and (2) indicator-based MOEAs which rely on the use of reference sets. Decomposition-based MOEAs transform a MOP into a group of sub-problems, in such a way that each sub-problem is defined by a reference weight point. Then, all these sub-problems are simultaneously solved using a single-objective optimizer [39]. They are able to solve MOPs efficiently. However, the main disadvantage of decomposition-based MOEAs is that the diversity of its selection mechanism is led explicitly by the reference weight vectors (normally the weight vectors are distributed in a unit simplex). This makes them unable to properly solve MOPs with complicated Pareto fronts. On the other hand, MOEAs based on the hybridization of a reference set and a performance indicator have shown to be promising schemes for solving many-objective optimization problems [48, 49, 99]. When compared to hypervolume-based MOEAs[1] [38, 100], reference-based MOEAs have a significantly lower computational cost and are able to obtain approximations of a similar quality to hypervolume-based MOEAs. Although effective and suitable for many-objective optimization, reference-based MOEAs in general require the generation of a set of reference weight vectors, analogously to decomposition-based MOEAs. In general, if the set of weight vectors and the Pareto front of a MOP share the same distribution, it is possible to obtain well-distributed approximations. There is, however, experimental evidence that indicates that the weight vectors most commonly used by these MOEAs adopt a simplex-like shape. This sort of scheme works well for Pareto fronts with regular shapes (e.g., a triangle

[1]The main drawback of hypervolume-based MOEAs is the high computational cost associated with the computation of the exact hypervolume contributions, which becomes unaffordable when trying to solve many-objective optimization problems.

or a sphere). Unfortunately, this scheme doesn't work properly with some complicated Pareto fronts (e.g., disconnected, degenerate, inverted simplex-like or badly-scaled). Empirical studies have shown that decomposition-based MOEAs and some indicator-based MOEAs have difficulties to solve these MOPs with complicated Pareto fronts, such as MOEA/D, IGD$^+$-EMOA, Δ_p-EMOA and MOMBI just to name a few. This motivated the work reported here, in which we propose a novel MOEA which uses an adaptive method for building the reference point set. This method is based on the creation of hypercubes. We show that the resulting MOEA has a competitive performance with respect to state-of-the-art MOEAs, and that is able to properly deal with MOPs having complicated Pareto fronts. Next, we give more details about how our approach works.

8.2 Our Proposed Approach

8.2.1 General Framework

Our approach adopts the same structure of the original IGD$^+$-EMOA, but we include some improvements in order to solve MOPs with complicated Pareto fronts. Our approach has the following features: (1) An archiving process for preserving candidate solutions which will form the reference set; (2) a method for adapting the reference set in order to sample uniformly the Pareto front; and (3) a rule for updating the reference set.

The general framework of the MOEA starts with a population $\mathcal{P}_0$ which contains N randomly generated individuals. A new offspring is created by choosing two different parents from $\mathcal{P}$. The parents are recombined using evolutionary operators.[2] After that, the resulting offspring are added to the new set. This process is executed until having a total of λ offspring. Thereafter, the algorithm combines the parents and the offspring populations to form the so-called $\mathcal{A}$ set. In order to select the next population, we apply our LAP-based selection mechanism.

8.2.2 Selection Mechanism

Since we intend to use the IGD$^+$ indicator in the selection mechanism of our MOEA, we adopted the same selection mechanism proposed by IGD$^+$-EMOA [99] (for more details see Chapters 5 and 6).

[2]In our implementation, we adopted SBX (Simulated Binary Crossover) and Polynomial-based Mutation [85].

This selection mechanism transforms the environmental selection mechanism into a Linear Assignment Problem (LAP). In terms of MOP the LAP is created using a cost matrix, where each element $C_{i,j} = C(a_i, t_j)$ represents the relationship between a_i and t_j. Thus, the cost matrix is defined as:

$$C_{i,j} = d^+(a_i, z_j), \quad i = 1, \ldots, |\mathcal{A}|, \quad j = 1, \ldots, |\mathcal{Z}|. \tag{8.1}$$

where $\vec{a}_i \in \mathcal{A}$ is the i^{th} vector point from the population $\mathcal{A}$, $\vec{z}_j \in \mathcal{Z}$ is the reference point and d^+ is the modified Euclidean distance.

In order to build the reference point set, the algorithm consists of two main procedures: (1) A procedure to maintain non-dominated solutions into an archive; and (2) A mechanism to remove non-candidate solutions with a poor distribution from the archive. Next, we provide more details about these two procedures.

8.2.3 Archiving Process

The archive has a pre-set capacity to store the non-dominated solutions, and the maximum number of solutions that are allowed in the archive is defined by a specific p value. When the archive reaches its maximum capacity, the approximation reference algorithm is executed for selecting candidate solutions (these candidate solutions will form the so-called *reference set*). After that, the archive is cleared and the archiving process continues until reaching a maximum number of generations. It is worth mentioning that the archiving process is applied at each generation of the MOEA.

8.2.4 Building the Reference Set

In IGD$^+$-EMOA, we aim to select the best reference points whose directions are promising (i.e., directions with good distribution and spread). In order to do that, we adopted a greedy algorithm based on the hypercube contributions to select a certain number of reference points from the archive. This greedy algorithm executes a density estimator for computing the hypercubes. Its pseudo-code is shown in Algorithm 8. The algorithm is organized as three consecutive loops, and is invoked with a set of non-dominated candidate points (called $\mathcal{A}$ set) and the maximum number of reference points that we aim to find. In the first loop, we create a set of initial candidate solutions to form the so-called $\mathcal{Q}$ set. Thus, the solutions from $\mathcal{A}$ that form part of $\mathcal{Q}$, will be removed from $\mathcal{A}$. After that, the greedy algorithm starts to find the best candidate solutions which will form the reference set $\mathcal{Z}$. In order to find the candidate reference points, the selection mechanism computes the hypercube contributions of the current reference set $\mathcal{Q}$.

Algorithm 8 ComputeReferenceSet($\mathcal{A}, z_{size}$)

```
Input: A current non-dominated set $\mathcal{A} \subset \mathbb{R}^m$ and maximum number of reference
  points $z_{size}$.
Output: Reference point set $\mathcal{Z} \subset \mathbb{R}^m$ with $|\mathcal{Z}| = z_{size}$
  $y_{ref} \leftarrow FindMaxValue(\mathcal{A}) + \epsilon$;
  $\mathcal{Q} \leftarrow \{\}$;
  while $|\mathcal{Q}| < (z_{size} + 1)$ do
    $\vec{a} \leftarrow pop(\mathcal{A})$;
    $\mathcal{Q} \bigcup \{\vec{a}\}$ ;
  end while
  while $\mathcal{A}! = \{\}$ do
    $maxHypercube \leftarrow HCB(\mathcal{Q}, y_{ref})$;
    for each $\vec{q_i} \in \mathcal{Q}$ do
      $ContHyperCube[i] \leftarrow maxHypercube - HCB(\mathcal{Q}\backslash\{\vec{q_i}\}, y_{ref})$;
    end for
    $i_{min} \leftarrow \text{argmin}\, ContHyperCube$;
    $\mathcal{Q}\backslash\{q_{i_{min}}\}$;
    $\vec{a} \leftarrow pop(\mathcal{A})$;
    $\mathcal{Q} \bigcup \{\vec{a}\}$;
  end while
  $\mathcal{Z} \leftarrow \{\}$;
  for each $\vec{q} \in \mathcal{Q}$ do
    $\mathcal{Z} \bigcup \{\vec{q} * \epsilon - \vec{l}\}$;
  end for
  return $\mathcal{Z}$;
```

Once this is done, we remove the i^{th} solution that minimizes the hypercube value and we add a new candidate solution from $\mathcal{A}$ to $\mathcal{Q}$. This process is executed until the cardinality of $\mathcal{A}$ is equal to zero. In the last loop, we apply the *expand* and *translate* operations. These operations transform the surface for spreading the reference set along objective function space. Figure 8.1 shows a graphical example of how Algorithm 8 works, where the the bigger points, from the $\mathcal{Q}$ set, are selected for the steady-state algorithm.

A hypercube is generated by the union of all the maximum volumes covered by a reference point. The i^{th} maximum volume is described as "the maximum volume generated by a set of candidate points" (these candidate points are obtained from the archive and a reference point y_{ref}). The hypercube is computed using Algorithm 9.

In the first part of Algorithm 9, we validate if $\mathcal{Q}$ contains one element. If that

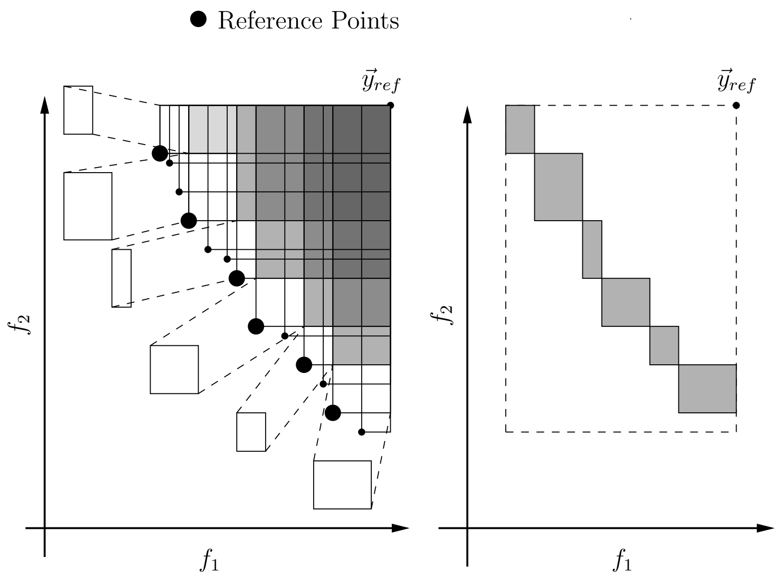


Figure 8.1: Applying the hypercube-based selection mechanism.

Algorithm 9 HCB($\mathcal{Q}, y_{ref}$)

```
Input: A current set $\mathcal{Q} \subset \mathbb{R}^m$ and a reference point $y_{ref}$
Output: Hypercube value
  if $|\mathcal{Q}| = 1$ then
    return vol($\mathcal{Q}, y_{ref}$);
  end if
  $VolList \leftarrow \{\}$;
  for each $\vec{p} \in \mathcal{Q}'$ do
    $VolList \bigcup \{vol(\vec{p}, y_{ref})\}$;
  end for
  $i_{max} \leftarrow \text{argmax}\, VolList$;
  $\vec{q}_{max} \leftarrow \mathcal{Q}[i_{max}]$ ;
  $\mathcal{Y} \leftarrow$ SplitReferencePoint $(\vec{q}_{max}, y_{ref})$;
  $\mathcal{Q} \backslash \{\vec{q}_{max}\}$;
  $hypercube \leftarrow 0$;
  for each $\vec{y}_{new} \in \mathcal{Y}$ do
    $\mathcal{Q}_{new} \leftarrow$ CoverPoints $(\mathcal{Q}, \vec{y}_{new})$;
    $hypercube \leftarrow hypercube + HCB(\mathcal{Q}_{new}, \vec{y}_{new})$;
  end for
  return $hypercube + max(VolList)$;
```

is the case, we compute the volume generated by y_{ref} and $\vec{q} \in \mathcal{Q}$. Otherwise, we compute the union of all the maximum hypercubes. In order to apply this procedure, we find the vector $\vec{q}_{max}$ that maximizes the hypercube. Once this is done, we create m reference points which will form the so-called $\mathcal{Y}$. In order to create the set $\mathcal{Y}$, we combine the current reference point y_{ref} and the point $\vec{q}_{max}$. For each reference point from $\mathcal{Y}$, we reduce the set $\mathcal{Q}$ into a small subset in order to form the set $\mathcal{Q}_{new}$. Thus, we have to split any point from $\mathcal{Q}$, whose components are not covered by the $\vec{y}_{new}$. In order to split the ith point from set $\mathcal{Q}$, we invoke the method "CoverPoints", which is described by Algorithm 11. Once this is done, we proceed to compute recursively the hypercube value of the new set formed by the subset $\mathcal{Q}_{new}$ and the new reference point y_{new}. Algorithm 9 provides an efficient way of estimating the hypercube value. It is worth noting that this value allows to measure the relationship among each element of a non-dominated set. The hypercube is not considered as hypervolume value, however it is considered as another sort of estimation.

Algorithm 10 SplitReferencePoint($\vec{q}_{max}, y_{ref}$)

```
Input: A vector $\vec{q}_{max}$, which contains the maximum volume from the set $\mathcal{Q}$ and a reference point $y_{ref}$
Output: $\mathcal{Y}$
  $\mathcal{Y} \leftarrow \{\}$;
  for each $i \in m$ do
    $y_{new} \leftarrow \{\}$;
    for each $j \in m$ do
      if $i == j$ then
        $y_{new}$.append($y_{max}[j]$);
      else
        $y_{new}$.append($ref[j]$);
      end if
    end for
    $\mathcal{Y}$.append($y_{new}$);
  end for
  return $\mathcal{Y}$;
```

8.2.5 Update Frequency

The timing and frequency of updating the reference set plays an important role in this MOEA. The generated reference point set does not always contribute in a good way because a frequent updating can significantly affect the performance of the algorithm. Therefore, we propose two additional mechanisms for updating the

Algorithm 11 CoverPoints $(\mathcal{Q}, \vec{y}_{new}, m_{\_current})$

```
Input: A current set $\mathcal{Q} \subset \mathbb{R}^m$, a current vector $y_{new}$ and the current dimension for
  slicing $m_{\_current}$
Output: $\mathcal{Q}_{new}$
  $\mathcal{Q}_{new} \leftarrow \{\}$;
  for each $\vec{q} \in \mathcal{Q}$ do
    if $q[m_{\_current}] < y_{new}[m_{\_current}]$ then
      $\mathcal{Q}_{new}$.append($\vec{q}$);
    end if
  end for
  return $\mathcal{Q}_{new}$;
```

reference set. The first one consists in updating the reference set if the variance of the hypercube contribution of the new reference set is lower than the variance of the previous reference set. In the second mechanism, if the hypercube value of the previous reference set is less than the hypercube value of the new reference set, then the new reference set is replaced by the previous one. It is worth indicating that these two mechanisms are adopted in our proposed approach.

8.3 Experimental Study

We compare the performance of our proposed IGD$^+$-EMOA with respect to that of four state-of-the-art MOEAs: NSGA-III [40], RVEA [42], MOMBI-II [49] and MOEA/DD [41]. These MOEAs had been found to be competitive on MOPs with a variety of Pareto front shapes.

8.3.1 Test problems

We aimed to study the performance of our proposed approach when solving MOPs with complicated Pareto front shapes. For this reason, we selected 18 test problems with a variety of representative Pareto front shapes from some well-known and recently proposed test suites (i.e., the DTLZ test suite [92], the WFG test suite [93], the MAF test suite [101] and the VNT test suite [28]). Based on the properties of their Pareto fronts, we categorized the test problems adopted into different groups: convex, concave, inverted simplex-like, disconnected, degenerate and badly-scaled (see Table 8.1).

Properties	Problems
Linear	DTLZ1
Convex and Concave	DTLZ2-3, MAF2-5, WFG1
Inverted Simplex-like	MAF1
Disconnected	DTLZ7, WFG2
Degenerate	DTLZ5-6, VNT2-3, WFG3
Badly-scaled	MAF4-5

Table 8.1: Main properties of the 18 test problems adopted

Problem	Reference point	Problem	Reference point
DTLZ1	(1,1,1)	VNT1	(5, 6, 5)
DTLZ2-6	(2,2,2)	VNT2	(5, -15, -11)
DTLZ7	(2, 2, 7)	VNT3	(9, 18, 5)
MAF1-3	(2,2,2)	WFG1	(3, 5, 7)
MAF4	(3,5, 9)	WFG2	(2, 4, 7)
MAF5	(9, 5, 3)	WFG3	(2, 3, 7)

Table 8.2: Reference points used for the hypervolume indicator

8.3.2 Methodology

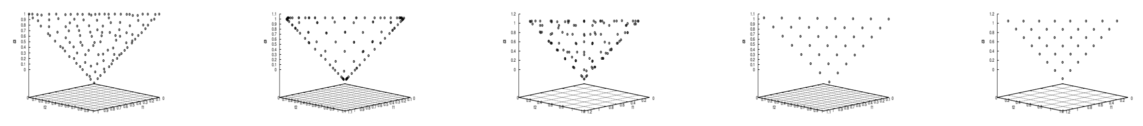
For our comparative study, we decided to adopt the hypervolume indicator, which assesses both convergence and maximum spread along the Pareto front. To compute I_H, we used the reference points shown in Table 8.2.

8.3.3 Parameterization

In the DTLZ and MAF test suites, the total number of decision variables is given by $n = m + k - 1$, where m is the number of objectives and k was set to 5 for DTLZ1 and MAF1, and to 10 for DTLZ2-6, and MAF2-5. The number of decision variables in the WFG test problems was set to 24, and the position-related parameter was set to $m - 1$. The parameters of each MOEA adopted in our study were chosen in such a way that we could do a fair comparison among them. The distribution indexes for the SBX and polynomial-based mutation operators [85], used by all algorithms, were set to: $\eta_c = 20$ and $\eta_m = 20$, respectively. The crossover probability was set to $p_c = 0.9$ and the mutation probability was set to $p_m = 1/L$, where L is the number of decision variables. The total number of function evaluations was set in such a way that it did not exceed 60,000. In MOEA/DD, MOMBI-II and NSGA-III, the number of weight vectors was set to the same value as the population size. The population size N is dependent on H which specifies the number of divisions in objective space. H was set in such a way that N took a value not greater than 120. In RVEA, the rate of changing the penalty function and the frequency to conduct the reference vector adaptation were set to 2 and 0.1, respectively. The main characteristics of the hardware used for the experiments were the following: an Intel Core i7-3930k CPU running at 2.30 GHz, with 8GB of RAM.

8.4 Discussion of Results

Table 8.3 provides the average hypervolume over the 30 independent executions of each compared MOEA for each instance of the DTLZ, WFG, VNT and MAF test suites. The best results are shown in **boldface** and grey-colored cells show the second best results. The variance is shown in parentheses. The Wilcoxon rank sum test was adopted to compare the results obtained by IGD$^+$-EMOA and its competitors at a significance level of 0.05, where the symbol "+" indicates that the compared algorithm is significantly outperformed by IGD$^+$-EMOA. On the other hand, the symbol "-" means that our approach is significantly outperformed by its competitor. Finally, "$\approx$" means that there is no statistically significant difference between the results obtained by IGD$^+$-EMOA and the compared algorithm. It is clear that the winner in this experimental study is our proposed IGD$^+$-EMOA since it was able to outperform MOEA/DD, RVEA, MOMBI-II and NSGA-III in ten cases and in a few more, it obtained very similar results to those of the best performer. Figures 8.2, 8.3, 8.4 and 8.5 present a graphical representation of the approximations to the Pareto front obtained by each MOEA in some instances of the MAF and VNT test problems adopted with 3 objectives. On the MOPs with inverted Simplex-like Pareto fronts, IGD$^+$-EMOA showed a clear advantage over its competitors (see Figure 8.2). Figures 8.2.a to 8.2.e show that the solutions produced by all the MOEAs adopted have a good coverage of the Pareto front. However, the solutions of MOMBI-II and NSGA-III are not distributed very uniformly, while the solutions of RVEA and MOEA/DD are distributed uniformly but their number is apparently less than their population size. On MOPs with degenerate Pareto fronts, our proposed approach had also a good performance. Table 8.3 indicates that IGD$^+$-EMOA was able to outperform its competitors in this type of MOPs since its solutions are distributed more uniformly (see Figure 8.4 which shows the results obtained for VNT2). MOMBI-II, RVEA and IGD$^+$-EMOA are able to obtain solutions of a similar quality when they solve MOPs with badly-scaled Pareto fronts, but our approach was not able



(a) IGD$^+$-EMOA (b) MOMBI-II (c) NSGA-III (d) RVEA (e) MOEA/DD

Figure 8.2: Graphical representation of the final set of solutions obtained by each MOEA on MAF1 with 3 objectives

to outperform MOMBI-II in MAF4. However, IGD$^+$-EMOA was better than its competitors when solving MAF5. On the other hand, it is well-known that MOMBI-II, RVEA and NSGA-III can solve efficiently MOPs with simplex-like Pareto fronts. In this regard, it is worth mentioning that in these MOPs, our proposed IGD$^+$-EMOA was able to obtain approximations of a similar quality to those obtained by its competitors. For DTLZ7, IGD$^+$-EMOA did not perform better than the other MOEAs. The reason is probably that the Pareto front shape of this problem is disconnected, which makes the approximations produced by our approach to converge to a single region. We can see in Table 8.3 that the variance obtained by IGD$^+$-EMOA significantly increases in this MOP. We can conclude that the construction of our reference point set is very sensitive to this sort of scenarios, which is a clear weakness of our proposed approach.

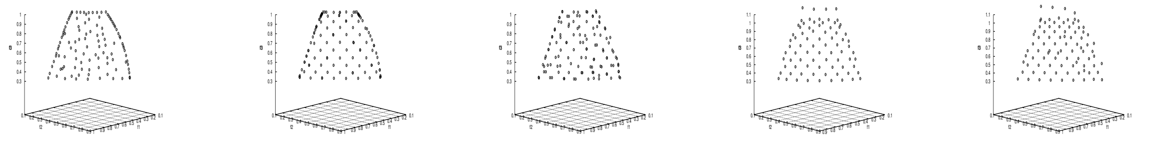
Table 8.4 shows a preliminary study on degenerate many-objective problems by considering DTLZ5 with 3 up to 10 objectives. We can see that our proposed approach significantly outperformed its competitors, since MOMBI-II, RVEA, MOEA/DD and NSGA-III were not able to converge to the true Pareto front as the number of objectives was increased. We can see in Table 8.4 that the performance of MOMBI-II, RVEA, MOEA/DD and NSGA-III is not consistent when the number of objectives is greater than 8. Figure 8.7 presents a graphical representation (using parallel coordinates plots) of the approximations of the Pareto front obtained by each MOEA solving DTLZ5 with 10 objectives. We can see that our proposed IGD$^+$-EMOA was able to obtain the best results in terms of the hypervolume indicator, which means that our approach can solve MOPs with degenerate shapes even in many-objective instances. In general, we can see that our hypercube-based method for building the reference set is able to properly deal with complicated Pareto fronts in high-dimensional objective function spaces.

Problems	MOMBI-II	RVEA	MOEA/DD	IGD$^+$-EMOA	NSGA-III
DTLZ1	0.96622 (0.000001) +	0.66911 (0.000152) +	0.97379 (0.000000) ≈	**0.97381 (0.000000)**	0.96256 (0.001064) +
DTLZ2	7.36755 (0.000028) +	7.42224 (0.000000) +	7.42225 (0.000000) +	**7.42736 (0.000016)**	7.41893 (0.000000) +
DTLZ3	7.38843 (0.000084) -	7.40582 (0.000084) -	**7.4118 (0.000047)** -	7.2211 (0.006505)	7.38048 (0.000258) -
DTLZ4	7.3593 (0.036144) -	**7.42226 (0.000000)** -	7.42224 (0.000000) -	6.76942 (0.236529)	7.10506 (0.227356) ≈
DTLZ5	6.00978 (0.000000) +	5.9632 (0.000369) +	6.02456 (0.000062) +	**6.1033 (0.000000)**	5.84002 (0.05518) +
DTLZ6	5.79608 (0.00523) +	5.13815 (0.016264) +	5.6037 (0.006442) +	**5.92189 (0.005444)**	5.49135 (0.023354) +
DTLZ7	**13.37473 (0.000091)** -	13.0605 (1.283746) ≈	12.99409 (0.015542) ≈	12.37989 (2.217964)	13.32733 (0.002554) ≈
VIE1	61.44939 (0.000533) +	60.51323 (0.011862) +	60.55111 (0.021176) +	**61.98616 (0.000514)**	61.19214 (0.011932) +
VIE2	7.79702 (0.000001) +	7.7712 (0.000368) +	7.80468 (0.000037) +	**7.84583 (0.000012)**	7.77446 (0.000935) +
VIE3	15.11767 (0.000262) +	15.03082 (0.000422)+	15.06016 (0.000114) +	**15.16248 (0.000258)**	15.12629 (0.000502) +
MAF1	5.44926 (0.000019) +	5.37408 (0.000659) +	5.37139 (0.00009) +	**5.50322 (0.000193)**	5.4129 (0.000875) +
MAF2	5.08952 (0.000056) +	**5.1583 (0.000058)** -	5.11373 (0.000003) +	5.13305 (0.000019)	5.09758 (0.000043) +
MAF3	7.90637 (0.000043) -	**7.91154 (0.004847)** -	7.64261 (1.915744) +	7.79256 (0.020172)	7.89441 (0.00452) -
MAF4	**84.87316 (0.151259)** ≈	83.53436 (29.511151) ≈	51.80943 (1120.296924) +	84.46979 (2.762969)	83.73257 (1.377427) ≈
MAF5	95.97704 (52.294491) +	96.66782 (53.122845) +	96.95207 (0.017991) +	**98.27521 (0.000549)**	88.72762 (237.475764) +
WFG1	50.38691 (7.353216) ≈	**51.68413 (5.001739)** ≈	41.77398 (7.334821) +	48.54235 (50.414084)	44.95726 (10.36034) +
WFG2	48.72516 (12.06217) -	**51.14414 (0.045119)** -	44.23925 (3.146579) +	45.58634 (7.395813)	48.14747 (12.622738) ≈
WFG3	24.28138 (0.007298) ≈	22.12339 (0.086504) +	21.04349 (0.178677) +	**24.34142 (0.015209)**	23.54542 (0.037132) +

Table 8.3: Performance comparison among several MOEAs using the average hypervolume indicator obtained from 30 independent executions solving 18 benchmark problems.

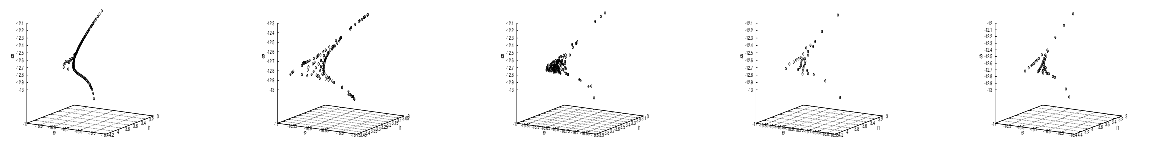
m	MOMBI-II	RVEA	MOEA/DD	IGD$^+$-EMOA	NSGA-III
3	6.00978 (0.000000) +	5.9632 (0.000369) +	6.02456 (0.000062) +	**6.1033 (0.000000)**	5.84002 (0.05518) +
5	21.79297 (0.001846) +	23.30876 (0.003765) +	20.09359 (0.702607) +	**23.48784 (0.006147)**	20.73826 (0.862548) +
8	167.75169 (1.3931) +	166.28627 (311.052362) +	144.38434 (32.923411) +	**180.36716 (24.078975)**	108.75498 (416.448268) +
10	686.39987 (203.794132) +	551.2682 (9825.611555) +	489.97082 (805.137545) +	**718.77776 (8.701444)**	358.05289 (19458.130567) +

Table 8.4: Performance comparison among several MOEAs using the average hypervolume indicator obtained from 30 independent executions solving DTLZ5 with 3 up to 10 objectives.



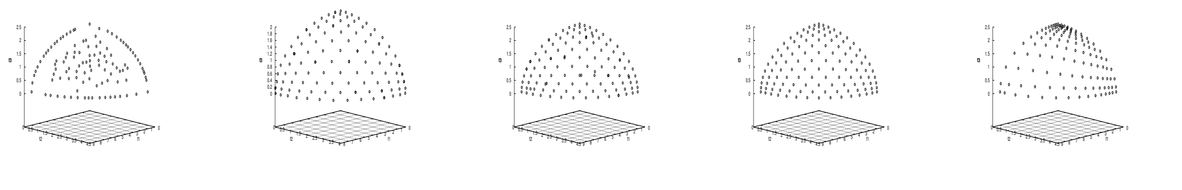
(a) IGD$^+$-EMOA (b) MOMBI-II (c) NSGA-III (d) RVEA (e) MOEA/DD

Figure 8.3: Graphical representation of the final set of solutions obtained by each MOEA on MAF2 with 3 objectives



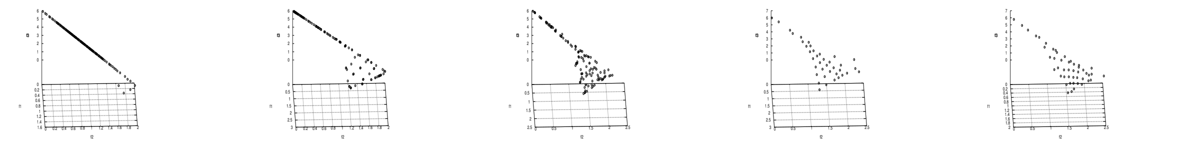
(a) IGD$^+$-EMOA (b) MOMBI-II (c) NSGA-III (d) RVEA (e) MOEA/DD

Figure 8.4: Graphical representation of the final set of solutions obtained by each MOEA on VNT2 with 3 objectives



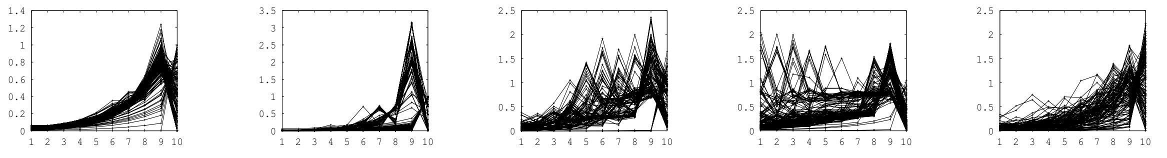
(a) IGD^{+}-EMOA (b) MOMBI-II (c) NSGA-III (d) RVEA (e) MOEA/DD

Figure 8.5: Graphical representation of the final set of solutions obtained by each MOEA on MAF5 with 3 objectives



(a) IGD^{+}-EMOA (b) MOMBI-II (c) NSGA-III (d) RVEA (e) MOEA/DD

Figure 8.6: Graphical representation of the final set of solutions obtained by each MOEA on WFG3 with 3 objectives



(a) IGD^{+}-EMOA (b) MOMBI-II (c) NSGA-III (d) RVEA (e) MOEAD/DD

Figure 8.7: Graphical representation of the final set of solutions obtained by the five MOEAs used in our study on DTLZ5 with 10 objectives.

8.5 Final Remarks

We have proposed a reference-based MOEA for solving many-objective problems with a particular emphasis on those having complicated Pareto front shapes. The core idea of our proposed approach is to adopt the IGD$^+$ performance indicator in its selection mechanism. Additionally, our proposal introduces a novel method for building the reference set which is based on the use of hypercubes. Our results indicate that the use of hypercubes significantly improves the performance of IGD$^+$-EMOA. As can be observed, the reference set is of utmost importance since our approach guides its search process using a set of reference points. Our results indicate that IGD$^+$-EMOA is very competitive with respect to MOMBI-II, RVEA, MOEA/DD and NSGA-III, being able to outperform them in more than 50 percent of the 18 benchmark problems adopted. Based on such results, we claim that our proposed approach is a competitive alternative to deal with MOPs having complicated Pareto front shapes, even in high-dimensional objective spaces. For this reason, we conclude that incorporating our hypercube-based method in order to sample the Pareto front shape of any MOP is a viable alternative to improve the performance of a MOEA.

Chapter 9

An Improved Version of a Multi-Objective Memetic Algorithm

Multi-Objective Memetic Algorithms (MOMAs) have shown to have a good performance when solving multi-objective optimization problems. They are able to drive the search process towards the Pareto front more effectively and efficiently than Multi-objective Evolutionary Algorithms (MOEAs), because they are constituted by the hybridization of a local search engine with a global optimizer. The main advantage of adopting this sort of hybridization is to speed up convergence towards the Pareto front. However, the use of MOMAs introduces new issues, such as how to select the solutions to which the local search will be applied and for how long to run the local search engine, since its use has an extra computational cost. In Chapter 7, we showed that it is possible to combine the hypervolume indicator with the IGD$^+$ indicator in order to build a new MOMA. Our MOMA was able to solve efficiently multi-objective optimization problems. However, the main disadvantage of our proposed MOMA is that the diversity of its selection mechanism is led explicitly by the hypervolume indicator (usually, a hypervolume-based selection mechanism distributes solutions around the knee of the Pareto front). The use of a hypervolume-based selection mechanism causes that our proposed MOMA cannot properly find well-distributed solutions along the Pareto front when solving MOPs with complicated Pareto fronts (i.e., Pareto fronts with irregular shapes). However, the original MOMA is appropriate for solving MOPs with regular Pareto fronts (i.e., those sharing the same shape of a unit simplex).

Here, we propose a new Multi-Objective Memetic Algorithm which uses a Local

Search technique (LS) based on the modified inverted generational distance (IGD$^+$) combined with a hypervolume-based global optimizer. However, the global optimizer uses a modified hypervolume-based selection mechanism, which is able to select well-distributed solutions. The selection mechanism adopts the use of an angle distance for choosing the worst candidate solutions to apply the hypervolume-based estimator. This improvement makes the hypervolume-based selection mechanism to be more efficient than its original version. We show that the resulting MOMA has a competitive performance with respect to its original version in several test problems traditionally adopted in the specialized literature. Our MOMA is able to properly deal with MOPs having complicated Pareto fronts since it can sample uniformly any Pareto front shape. We hypothesized that an appropriate combination could improve the performance of a MOMA and the experiments reported in this chapter validate our hypothesis. We also study the performance of our proposed MOMA when it tries to solve a structural optimization problem (i.e., the structural optimization of the frontal structure of a vehicle for crash-worthiness [102]). With this example, we show that our proposed MOMA is a suitable candidate for solving real-world multi-objective optimization problems.

This chapter is organized as follows. Our proposed approach is described in Section 9.1. Then, Section 9.2 shows our experimental study. Our experimental study using a real-world MOP is presented in the Section 9.3. Section 9.4 presents an Analysis of Variance (ANOVA) of our proposed approach. Finally, Section 9.5 provides our final remarks.

9.1 Our Proposed Approach

Our MOMA consists of two different approaches. The first one is a global optimizer which is based on a modified hypervolume-based selection mechanism. A global optimizer is meant to explore the whole search space for finding different regions, and eventually, the global optimum (or a good approximation of it). The second method is our local search (LS) technique which uses an IGD$^+$-based search technique. A local search method can improve a candidate solution very quickly, but aims to find only local optima (i.e., solutions which are optimal in a specific region of the objective space close to the candidate solution from which the local search is launched). Our approach adopts the same structure of our original MOMA (see Chapter 7), but we include some improvements to solve MOPs with complicated Pareto fronts. Our approach has the following features: (1) a new hypervolume-based selection mechanism, which adopts the angle distance for decreasing the computation of the exact hypervolume contribution; (2) an archiving process for preserving candidate solutions to sample the Pareto front shape; (3) a method for splitting the objective space into different regions for building the reference vectors, which are used by the local search engine; and (3) a rule for updating the reference set.

9.1.1 Our Global Optimizer

Our global optimizer uses the same structure of SMS-EMOA [38], which starts with an initial population of N individuals. Each new individual is created through the use of evolutionary operators (i.e., SBX and polynomial-based mutation). The new offspring will become a member of the population, but one individual is discarded from the population in order to maintain the same population size. The individual which minimizes the hypervolume contribution is eliminated. Here, the hypervolume contribution is computed by estimating the angle distance for each solution from the population in order to select solutions on which we apply the computation. To compute the hypervolume contribution, we propose to find the k nearest solutions in terms of angle distance to the i^{th} candidate individual. This process is repeated for each generation of the evolutionary algorithm, and each new candidate solution is kept into the archive to sample the Pareto front shape.

The archive stores candidate solutions, up to a maximum number of solutions defined by the "ArchiveSize" value (p_{size}). The reference set is computed by finding the best uniform distributed solutions which sample the Pareto front shape of the MOP. Our LS technique starts when a certain percentage of the total number of generations is reached using the reference vectors found so far. Next, we will provide more details of the way in which our LS engine works. The pseudo-code of our proposed MOMA is presented in Algorithm 12.

9.1.2 Modified Hypervolume-based Selection Mechanism

Our proposed algorithm improves the exact computation of the hypervolume contribution, and has the following features: (1) it identifies the n candidate solutions which have the nearest angle distance; and (2) for each candidate solution, it applies the hypervolume contribution with l elements (in this case, we propose to select five solutions). The main idea is to reduce the running time of its original version. Algorithm 13 provides the pseudo-code of the algorithm to compute the hypervolume contributions which is invoked with a set of candidate points (called $\mathcal{D}$ set) and a

Algorithm 12 General Framework of our proposed MOMA

Input: A MOP, a stopping criterion and a uniform spread of N reference vectors
Output: Approximation of the MOP

```
 1: $P_0 \leftarrow$ init();
 2: $t \leftarrow 0$;
 3: $\mathcal{A} \leftarrow \{\}$;
 4: **while** $t < gen_{max}$ **do**
 5:    $Q_{t+1} \leftarrow$ generate($P_t$); /*Generate an offspring using evolutionary operators*/
 6:    **if** $PL > P_{percent}$ **then**
 7:       **if** $|A| < p_{size}$ **then**
 8:          $\mathcal{A} \cup Q_{t+1}$;
 9:       **else**
10:          /*Applying Local Search Engine*/
11:          $P_{t+1} \leftarrow$ LS Engine($P_{t+1}$);
12:          $\mathcal{A} \leftarrow \{\}$;
13:       **end if**
14:    **end if**
15:    $\mathcal{D} \leftarrow P_t \cup \{Q_{t+1}\}$;
16:    /*Modified hypervolume-based selection mechanism for finding the $N$ best
       individuals*/
17:    $P_{t+1} \leftarrow$ reduce($\mathcal{D}$);
18:    $t \leftarrow t + 1$;
19: **end while**
20: return $P_t$;
```

reference point $\vec{y}_{ref}$.

The algorithm is organized into two main parts. In the first loop, we compute the angle distance between solutions $\vec{d_i}$ and $\vec{d_j}$ from the set $\mathcal{D}$, where $\vec{d_i}$ and $\vec{d_j}$ should not be equal (i.e., $i \neq j$). Then, we find the k solutions with the worst angle distance value, which will form a subset $\mathcal{B}$. After that, for each solution from the subset $\mathcal{B}$ we proceed to compute the exact hypervolume contribution with l nearest solutions for the i^{th} element. In this case, we propose to apply the hypervolume contribution, whose cardinality of the set must contain five solutions. Once this is done, we assign a maximum value for the solutions which are not candidates for computing the hypervolume contribution.

Algorithm 13 AngleDistanceHypervolumeContribution($\mathcal{D}, \vec{y}_{ref}$)

```
Input: A current non-dominated set $\mathcal{D} \subset \mathbb{R}^m$ and a reference vector $\vec{y}_{ref}$.
Output: Set of hypervolume Contribution values $HvC$.
1: DistanceMatrix ← init();
2: for $\vec{d_i} \in \mathcal{D}$ do
3:     for $\vec{d_j} \in \mathcal{D}$ do
4:         if $\vec{d_i} \neq \vec{d_j}$ then
5:             DistanceMatrix[$i$][$j$] ← AngleDistance($\vec{d_i}, \vec{d_j}$);
6:         else
7:             DistanceMatrix[$i$][$j$] ← maxValue;
8:         end if
9:     end for
10: end for
11: $\mathcal{F}$ ← FindWorstPoints(DistanceMatrix); /*Find the $l$ points with the worst angle
    distance*/
12: for $\vec{f_i} \in \mathcal{F}$ do
13:     sort(DistanceMatrix[i]);
14:     $\mathcal{B}$ ← FindNearestPoints($\vec{f_i}, DistanceMatrix[i]$);
15:     $HvC[\vec{f_i}]$ ← HVContribution($\mathcal{B}, \vec{y}_{ref}$);
16: end for
17: for $\vec{d_i} \in \mathcal{D}$ do
18:     if $\vec{d_i}$not in$\mathcal{F}$ then
19:         $HvC[\vec{d_i}]$ ← maximunValue;
20:     end if
21: end for
22: return $HvC$;
```

Algorithm 13 shows an efficient way to approximate the hypervolume contribution given a set $\mathcal{D}$ with non-dominated points. This algorithm allows to apply the hypervolume computation to specific solutions (i.e., solutions whose angle distance is minimal).

9.1.3 Our Local Search Engine

As mentioned before, the local search engine uses a reference set for leading the optimization process. The reference set establishes the regions on which local search will explore the solutions. To create the reference set, we propose to sample the Pareto front shape using an archive. Once the archive is created, we proceed to find the best candidate points whose directions are promising. Our proposed approach creates different neighborhoods for each point in the reference set. The i^{th} neighborhood is created by N vector points from the archive. Such points have the nearest distance with respect to the i^{th} reference point in terms of the $d+$ distance (see equation (3.11)). Our local search technique starts with a population $\mathcal{P}$ which contains n individuals.

The new i^{th} offspring is created by choosing three different parents from its neighborhood. The parents are recombined using the differential evolution operator, where the first parent is selected by the nearest distance in terms of the $d+$ distance and the rest of the parents are randomly chosen. The second step is to combine the parents and the offspring of each neighborhood. The new population at generation $t+1$ is generated by finding the nearest point from the population for each z reference point in $\mathcal{Z}$. This process is repeated until the stopping criterion is satisfied.

9.1.4 Archiving Process

As mentioned before, the archive stores non-dominated solutions, up to a maximum number of solutions defined by the "ArchiveSize" value (or p_{size}). When the archive reaches its maximum capacity, the approximation reference algorithm is executed for selecting candidate solutions (these candidate solutions will form the so-called *candidate reference set*). After that, the archive is cleaned, and the archiving process continues until reaching a maximum number of updates. The archiving process is applied after 60% of the total number of generations.

9.1.5 Reference Set

In our approach, we aim to select the best candidate points whose directions are promising (these candidate solutions will sample the Pareto front as uniformly as possible). For this reason, we propose to use a steady-state algorithm based on the hypercube contributions to select a certain number of reference points from the archive. Algorithm 14 provides the pseudo-code of an approach that is invoked with a set of non-dominated candidate points (called $\mathcal{A}$ set) and the maximum number of reference points that we aim to find. The algorithm is organized into two main parts. In the first loop, we create a set of initial candidate solutions to form the so-called $\mathcal{Q}$ set. Thus, the solutions from $\mathcal{A}$ that constitute part of $\mathcal{Q}$ will be removed from $\mathcal{A}$. After that, the algorithm starts to find the best candidate solutions which will form the reference set $\mathcal{Z}$. To find the candidate reference points, the selection mechanism computes the hypercube contributions of the current reference set $\mathcal{Q}$. Once this is done, we remove the i^{th} solution that minimizes the hypercube value, and we add a new candidate solution from $\mathcal{A}$ to $\mathcal{Q}$. This process is executed until the cardinality of $\mathcal{A}$ is equal to zero. In the line 20 of Algorithm 14, we apply the *expand* and *translate* operations. A hypercube is generated by the union of all the maximum volumes covered by a reference point. The i^{th} maximum volume is described as "the maximum volume generated by a set of candidate points" (these candidate points are obtained from the archive using a reference point y_{ref}).

9.1.6 Update Frequency

The timing and frequency of updating the reference set play an important role in this approach. The generated reference point set does not always contribute in a good way because a frequent updating can significantly affect the performance of the algorithm. Therefore, we propose two additional mechanisms for updating the reference set. The first one consists in updating the reference set if the variance of the hypercube contribution of the new reference set is lower than the variance of the previous reference set. In the second mechanism, if the hypercube value of the previous reference set is less than the hypercube value of the new reference set, then the new reference set is replaced by the previous one. It is worth indicating that these two mechanisms are adopted in our proposed approach.

9.2 Experimental Study I

We compared the performance of our improved multi-objective memetic algorithm (MOMA-II) with respect to its original version (for more details see Chapter 7). We evaluate the running time of each version (i.e., a sequential and a parallel

Algorithm 14 ComputeReferenceSet($\mathcal{A}, z_{size}$)

Input: A current non-dominated set $\mathcal{A} \subset \mathbb{R}^m$ and maximum number of reference points z_{size}.

Output: Reference point set $\mathcal{Z} \subset \mathbb{R}^m$ with $|\mathcal{Z}| = z_{size}$

```
1: $y_{ref} \leftarrow FindMaxValue(\mathcal{A}) + \epsilon$;
2: $\mathcal{Q} \leftarrow \{\}$;
3: **while** $|\mathcal{Q}| < (z_{size} + 1)$ **do**
4:     $\vec{a} \leftarrow pop(\mathcal{A})$;
5:     $\mathcal{Q} \bigcup \{\vec{a}\}$ ;
6: **end while**
7: **while** $\mathcal{A}! = \{\}$ **do**
8:     $i \leftarrow 0$;
9:     **for** each $\vec{q} \in \mathcal{Q}$ **do**
10:        $ContHyperCube[i] \leftarrow$ AngleDistanceHypervolumeContribution($\mathcal{Q} \backslash \{\vec{q}\}, y_{ref}$);

11:        $i \leftarrow i + 1$;
12:    **end for**
13:    $i_{min} \leftarrow \text{argmin}\, ContHyperCube$;
14:    $\mathcal{Q} \backslash \{q_{i_{min}}\}$;
15:    $\vec{a} \leftarrow pop(\mathcal{A})$;
16:    $\mathcal{Q} \bigcup \{\vec{a}\}$;
17: **end while**
18: $\mathcal{Z} \leftarrow \{\}$;
19: **for** each $\vec{q} \in \mathcal{Q}$ **do**
20:     $\mathcal{Z} \bigcup \{\vec{q} * \epsilon - \vec{l}\}$;
21: **end for**
22: return $\mathcal{Z}$;
```

Properties	Problems
Linear	DTLZ1
Convex and Concave	DTLZ2-3, MAF2-5, WFG1
Inverted Simplex-like	MAF1
Disconnected	DTLZ7, WFG2
Degenerate	DTLZ5-6, VNT2-3, WFG3
Badly-scaled	MAF4-5

Table 9.1: Main properties of the 18 test problems adopted

Problem	Reference point	Problem	Reference point
DTLZ1	(1,1,1)	VNT1	(5, 6, 5)
DTLZ2-6	(2,2,2)	VNT2	(5, -15, -11)
DTLZ7	(2, 2, 7)	VNT3	(9, 18, 5)
MAF1-3	(2,2,2)	WFG1	(3, 5, 7)
MAF4	(3,5, 9)	WFG2	(2, 4, 7)
MAF5	(9, 5, 3)	WFG3	(2, 3, 7)

Table 9.2: Reference points used for the hypervolume indicator

implementation[1]).

9.2.1 Test problems

We aimed to study the performance of our proposed approach when solving MOPs with complex Pareto front shapes (i.e., convex, concave, inverted simplex-like, disconnected, degenerate and badly-scaled Pareto fronts). For this reason, we selected some test problems with a variety of representative Pareto front shapes from some well-known and recently proposed test suites. To be more specific, we adopted the DTLZ test suite [92], the WFG test suite [93], the MAF test suite [101] and the VNT test suite [28] (see Table 9.1).

9.2.2 Methodology

For our comparative study, we decided to adopt the hypervolume indicator, which assesses both convergence and maximum spread along the Pareto front. To compute I_H, we used the reference points shown in Table 9.2.

Likewise, we propose to measure the distribution of the approximation set. For

[1]The parallel implementation is the same that we proposed in the Chapter 7. However, in this approach, we include a parallel algorithm for computing the hypervolume contribution.

this reason, we decided to include the use of the spacing indicator. The spacing indicator suggested by Schott [83] is computed as a relative distance measure between consecutive solutions in the obtained non-dominated set. The definition of this indicator is as follows:

$$s = \sqrt{\frac{1}{|\mathcal{Q}|}\sum_{i=0}^{|\mathcal{Q}|}(d_i - \bar{d})^2} \tag{9.1}$$

where, $d_i = \min_{k\in\mathcal{Q}\bigwedge k\neq i}\left(\sum_{i=1}^{m}|f_m^i - f_m^k|\right)$ and $\bar{d}$ is the average value of the above distance measure.

We also include the modified spacing indicator, which adopts the angle distance for measuring the range between each non-dominated solution from the set $\mathcal{Q}$. We intend to measure the variance of each minimal angle distance. The angle distance is described as:

$$ad(\vec{q_i}, \vec{q_j}) = \arccos\left(\frac{\vec{q_i}\cdot\vec{q_j}}{\|\vec{q_i}\|\|\vec{q_j}\|}\right) \tag{9.2}$$

Additionally, we also compared the running time of each MOEA, which was measured in minutes. We also incorporate the speed up for comparing the parallel implementation of our MOMA with respect to each of its competitors.

9.2.3 Parameterization

In the MAF and DTLZ test suites, the total number of decision variables is given by $n = m + k - 1$, where m is the number of objectives and k was set to 5 for DTLZ1 and MAF1, and to 10 for DTLZ2-6, and MAF2-5. The number of decision variables in the WFG test suite was set to 24, and the position-related parameter was set to $m - 1$. The parameters of each MOMA adopted in our study were chosen in such a way that we could do a fair comparison among them. The distribution indexes for the Simulated Binary crossover and the polynomial-based mutation operators [85] adopted by all algorithms, were set to: $\eta_c = 20$ and $\eta_m = 20$, respectively. The crossover probability was set to $p_c = 0.9$ and the mutation probability was set to $p_m = 1/L$, where L is the number of decision variables. The total number of function evaluations for each memetic algorithm was set in such a way that it did not exceed 60,000 and the population size was set to 120 for each MOMA. The maximum number of solution for the archive was set to 500, for MOMA-II. All the implementations were tested on the same computer which has the following characteristics: an Intel Core i7-3930k CPU running at 3.20 GHz, with 8GB of RAM 1600 MHz DDR3. Our GPU was a Geforce GTX 680, and we ran our experiments in Fedora 18 (64-bit version).

9.2.4 Discussion of Results

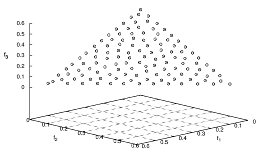
Table 9.3 provides the average hypervolume over the 30 independent executions of each compared MOMA for each instance of the DTLZ, WFG, VNT, and MAF test suites. Likewise, Tables 9.4 and 9.5 provide the average spacing and modified spacing respectively over the 30 independent executions. The best results are shown in **boldface** and grey-colored cells show the second best results. The variance is given in parentheses.

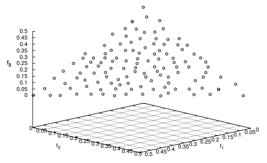
It is clear that the winner in this experimental study is our proposed MOMA-I since it was able to outperform MOMA-II in more than fifty percent in terms of the hypervolume indicator, but MOMA-II obtained very similar results to MOMA-I. MOMA-II is able to outperform its original version in terms of distribution since MOMA-II and MOMA-II/Parallel can generate solutions which are well-distributed along the Pareto front. It is worth noting that MOMA-I generates solutions edges and the knee regions of the Pareto front. MOMA-I is not able to spread solutions along the Pareto front although MOMA-I surpasses its competitors regarding the hypervolume indicator. Figures 9.1-9.15 present a graphical representation of the approximations to the Pareto front obtained by each MOMA in some instances of the DTLZ, MAF, and VNT test problems adopted with three objectives.

On the MOPs with regular (i.e., convex, concave and linear shapes) Pareto fronts, MOMA-II shows a clear advantage over its competitors, since MOMA-II generates well-distributed solutions and it avoids to keep solutions in the knee regions. However, MOMA-I is able to create well-distributed solutions in the edge and in the knee regions and it outperforms MOMA-II in terms of the hypervolume indicator (see Figures 9.1, 9.2, 9.3, 9.4). Tables 9.4 and 9.5 show that the obtained distribution by MOMA-II outperforms MOMA-I in these problems.

On the MOPs with inverted Simplex-like Pareto fronts, MOMA-I performs better than MOMA-II regarding the hypervolume indicator. Its parallel version works very similarly to our CPU-based MOMA (see Figure 9.11). Figure 9.11 shows that the solutions produced by the adopted MOMAs have good coverage of the Pareto front.

On MOPs with degenerate Pareto fronts (i.e., DTLZ5, DTLZ6, VNT2, and VNT3), our proposed approach outperforms its original version. Table 9.3 indicates that MOMA-II was able to drive the optimization process using the angle distance,


(a) MOMA-I


(b) MOMA-II

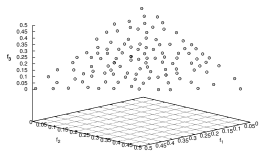

(c) MOMA-II/Parallel

Figure 9.1: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on DTLZ1 with 3 objectives

since its solutions are distributed more uniformly (see Figures 9.5, 9.5, 9.9 and 9.10). MOMA-I and MOMA-II can obtain solutions of similar quality when they solve MOPs with badly-scaled Pareto fronts, but our approach outperforms its original version in MAF4 and MAF5. On the other hand, as mentioned before, MOMA-I can solve MOPs with simplex-like Pareto fronts in an efficient manner. In this regard, it is worth mentioning that in these MOPs, our proposed MOMA was able to obtain approximations of a similar quality to those obtained by its competitors. For DTLZ7, MOMA-II performs better than its competitor. Our proposed approach is able to sample the Pareto front shape as uniformly as possible using the angle distance. The construction of our reference point set works very well in this sort of scenarios (i.e., irregular Pareto front shapes). On the other hand, MOMA-I generates well-distributed solutions in the edge regions, and it is not able to spread solutions along the Pareto front (see Tables 9.4 and 9.5). It is worth mentioning that this sort of problem is very difficult to solve since the Pareto front shape is disconnected and irregular. However, our proposed approach has a good convergence in spite of this difficulty.

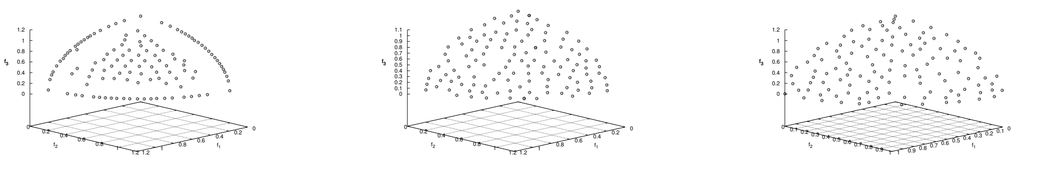
Table 9.6 shows a study of the running time of each MOMA. It is clear that the winner of this experimental study is our MOMA-II/Parallel regarding CPU time. We are also able to obtain the same results as the CPU-based MOMA, which verifies that our parallel implementation is working as expected. Our MOMA-II/Parallel is able to obtain solutions of a similar quality as MOMA-I. On the other hand, this study confirms that our proposed IGD$^+$-based LS is an effective way to solve complicated MOPs, but the LS depends on the definition of well-distributed reference points.

9.3 Solving a real-world optimization problem

In this section, we evaluate our MOMA-II on a multi-objective optimization problem that models crash safety design of vehicles. This MOP aims at the design of the frontal

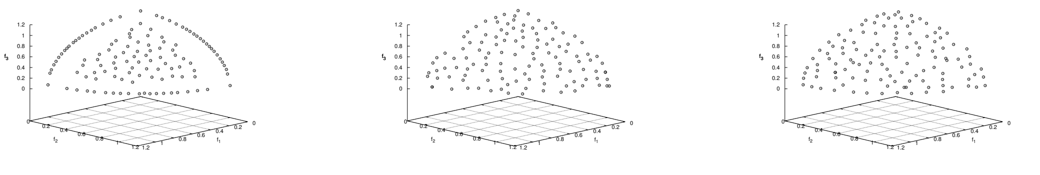
Problems	MOMA-I	MOMA-II	MOMA-II/Parallel
DTLZ1	**0.974498(0.0)**	0.967744(5e-06)	0.965165(6e-05)
DTLZ2	**7.43048(0.0)**	7.407862(0.000153)	7.401935(0.000261)
DTLZ3	6.488775(3.053799)	7.331207(0.015623)	**7.34676(0.001596)**
DTLZ4	6.909481(0.25347)	**7.292107(0.387131)**	7.16539(0.743282)
DTLZ5	5.008962(0.065273)	6.09001(7.3e-05)	**6.095326(6.7e-05)**
DTLZ6	5.276411(0.475358)	5.89638(0.003954)	**5.916629(0.005026)**
DTLZ7	10.944332(4.124945)	12.72887(1.531952)	**13.065952(0.887248)**
MAF1	**5.531737(2e-06)**	5.400789(0.001865)	5.385073(0.00225)
MAF2	**5.13774(0.0)**	5.054947(0.000822)	5.049649(0.00109)
MAF3	**7.953254(0.0)**	7.908287(0.00125)	7.904027(0.003435)
MAF4	83.604422(1.49467)	85.032317(1.148134)	**85.261421(0.258035)**
MAF5	86.342127(295.925351)	86.320354(813.912931)	**94.732153(227.092937)**
WFG1	**71.958616(2.72051)**	54.74326(4.565736)	53.314301(3.953018)
WFG2	**101.136803(0.005187)**	99.813961(0.068134)	99.967004(0.03222)
WFG3	**76.278619(0.002793)**	73.198446(0.313928)	72.950653(0.312567)
WFG4	**77.488835(0.001871)**	75.759379(0.033253)	75.892545(0.04884)
WFG5	**74.183962(1.1e-05)**	72.640281(0.215921)	72.428069(0.237723)
WFG6	**74.731767(0.145603)**	73.377549(0.224466)	73.320232(0.180422)
WFG7	**77.664878(2.5e-05)**	76.209886(0.036435)	76.256588(0.059257)
WFG8	**73.968437(0.007002)**	72.197654(0.093873)	72.23113(0.10217)
WFG9	**76.654169(0.08226)**	72.644709(1.375751)	72.551441(1.680141)
VIE1	**61.903108(0.036651)**	61.168247(0.019782)	61.156076(0.017319)
VIE2	**7.851557(0.0)**	7.826619(2.8e-05)	7.824563(6.2e-05)
VIE3	**15.180549(1.6e-05)**	15.126093(0.001393)	15.135716(0.000326)

Table 9.3: Performance comparison among several MOMAs using the average hypervolume indicator obtained from 30 independent executions solving 24 benchmark problems.



(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.2: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on DTLZ2 with 3 objectives

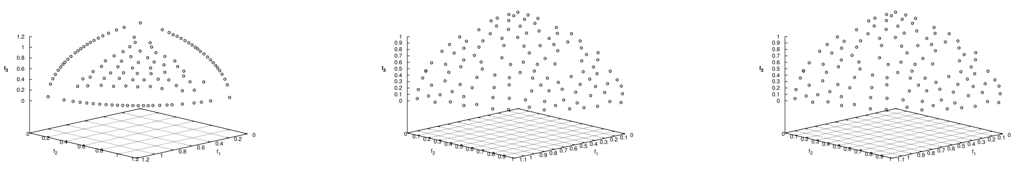


(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.3: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on DTLZ3 with 3 objectives

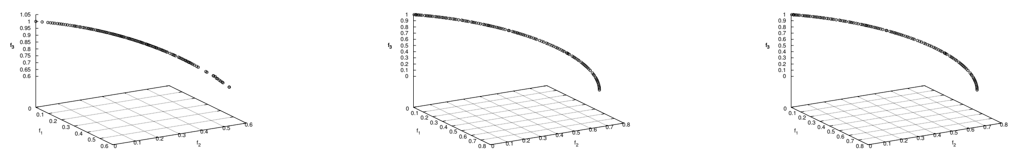
Problems	**MOMA-I**	**MOMA-II**	**MOMA-II/Parallel**
DTLZ1	**0.00549(0.0)**	0.029316(0.004157)	0.043394(0.009272)
DTLZ2	0.042087(3e-06)	0.032824(1.9e-05)	**0.031573(3.3e-05)**
DTLZ3	1.088404(18.70655)	0.720126(12.206421)	**0.095582(0.050677)**
DTLZ4	**0.023307(0.000301)**	0.032229(8.6e-05)	0.031795(0.000101)
DTLZ5	0.006532(9e-06)	0.006278(1e-06)	**0.005966(1e-06)**
DTLZ6	**0.01089(5e-05)**	0.057337(0.001525)	0.043129(0.000447)
DTLZ7	0.058786(0.000185)	0.047437(0.000228)	**0.048478(8.5e-05)**
MAF1	**0.01435(1e-06)**	0.025076(1.2e-05)	0.025462(7e-06)
MAF2	0.021896(2e-06)	0.017055(9e-06)	**0.016873(1.1e-05)**
MAF3	**0.014218(2e-06)**	0.686561(3.607834)	0.49651(4.407834)
MAF4	3.696129(348.04145)	1.365616(26.095513)	**0.218978(0.000516)**
MAF5	**0.124541(0.003563)**	0.1663(0.004588)	0.177499(0.001266)
WFG1	0.137122(0.003864)	0.079512(0.000595)	**0.073165(0.00043)**
WFG2	**0.084556(0.000526)**	0.163593(0.000852)	0.169295(0.000503)
WFG3	0.121439(6.6e-05)	0.091632(0.000204)	**0.092781(0.000281)**
WFG4	0.149812(4.5e-05)	0.156329(0.000307)	**0.148194(0.000265)**
WFG5	0.149747(3.4e-05)	0.143176(0.000224)	**0.140333(0.000274)**
WFG6	**0.146962(6.1e-05)**	0.158453(0.000285)	0.151972(0.000278)
WFG7	**0.148852(5.3e-05)**	0.15137(0.000211)	0.15396(0.000451)
WFG8	**0.157233(8.7e-05)**	0.157762(0.000661)	0.158087(0.00043)
WFG9	0.151543(8.1e-05)	0.152307(0.00042)	**0.150543(0.000393)**
VIE1	**0.089613(0.000446)**	0.151537(0.001044)	0.151101(0.000701)
VIE2	**0.024476(8.9e-05)**	0.077188(0.000497)	0.072312(0.00047)
VIE3	**0.020918(5.4e-05)**	0.071089(0.000331)	0.068556(4.9e-05)

Table 9.4: Performance comparison among several MOMAs using the average spacing indicator obtained from 30 independent executions solving 24 benchmark problems.



(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.4: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on DTLZ4 with 3 objectives

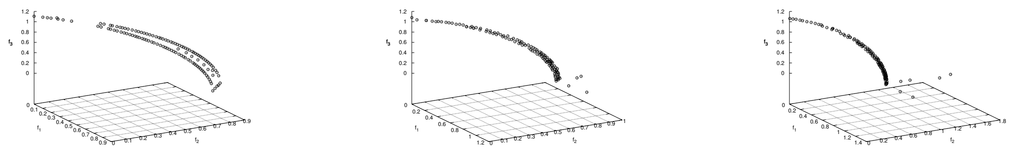


(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.5: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on DTLZ5 with 3 objectives

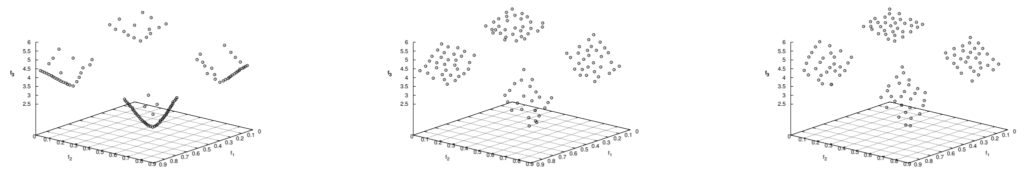
Problems	MOMA-I	MOMA-II	MOMA-II/Parallel
DTLZ1	**0.000414(0.0)**	0.000826(0.0)	0.001223(2e-06)
DTLZ2	0.001121(0.0)	0.000799(0.0)	**0.000742(0.0)**
DTLZ3	0.001548(1e-06)	**0.00102(0.0)**	0.001082(0.0)
DTLZ4	**0.000598(0.0)**	0.000775(0.0)	0.000867(0.0)
DTLZ5	0.000155(0.0)	8.9e-05(0.0)	**8.8e-05(0.0)**
DTLZ6	0.000183(0.0)	0.000817(1e-06)	**0.000526(0.0)**
DTLZ7	**1.8e-05(0.0)**	5.4e-05(0.0)	5.4e-05(0.0)
MAF1	**0.00022(0.0)**	0.000293(0.0)	0.000293(0.0)
MAF2	0.000227(0.0)	**0.000225(0.0)**	0.000227(0.0)
MAF3	**0.000847(0.0)**	0.001085(0.0)	0.001673(4e-06)
MAF4	**0.000297(0.0)**	0.000947(0.0)	0.000781(0.0)
MAF5	**0.001658(1e-06)**	0.002076(1e-06)	0.002592(1e-06)
WFG1	0.000531(0.0)	0.000547(0.0)	**0.00037(0.0)**
WFG2	**0.00082(0.0)**	0.001499(0.0)	0.001489(0.0)
WFG3	0.000588(0.0)	**0.000419(0.0)**	0.000432(0.0)
WFG4	0.002137(0.0)	0.001647(0.0)	**0.001552(0.0)**
WFG5	0.00185(0.0)	**0.001533(0.0)**	0.001602(0.0)
WFG6	0.002206(1e-06)	0.001834(1e-06)	**0.001637(0.0)**
WFG7	0.002002(0.0)	**0.001714(0.0)**	0.001996(1e-06)
WFG8	0.002774(1e-06)	**0.001649(0.0)**	0.001803(0.0)
WFG9	0.005365(3e-06)	0.002229(1e-06)	**0.001914(1e-06)**
VIE1	**0.000116(0.0)**	0.000288(0.0)	0.000295(0.0)
VIE2	**3e-06(0.0)**	8e-06(0.0)	5e-06(0.0)
VIE3	**4e-06(0.0)**	2.4e-05(0.0)	2.5e-05(0.0)

Table 9.5: Performance comparison among several MOMAs using the average of the modified spacing indicator obtained from 30 independent executions solving 24 benchmark problems.



(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.6: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on DTLZ6 with 3 objectives

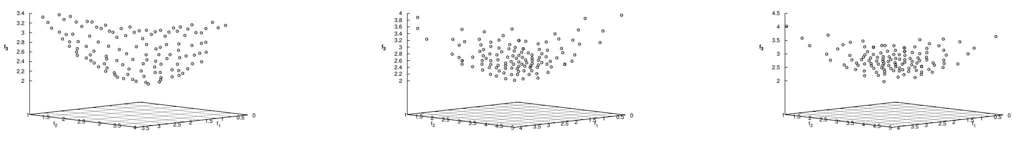


(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.7: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on DTLZ7 with 3 objectives

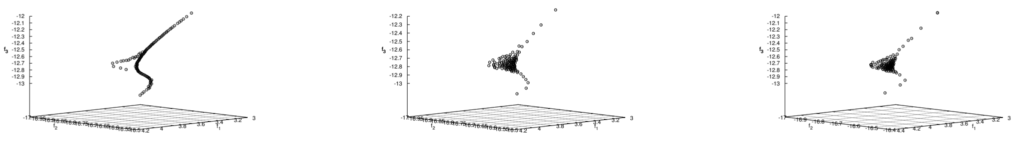
Problems	**MOMA-I**	**MOMA-II**	**MOMA-II/Parallel**
DTLZ1	9.4274 (12.263x)	1.899331 (2.471x)	0.76874
DTLZ2	10.892649 (15.211x)	2.132161 (2.977x)	0.716115
DTLZ3	1.670008 (2.588x)	1.020812 (1.582x)	0.645277
DTLZ4	10.557177 (15.846x)	2.058528 (3.09x)	0.666255
DTLZ5	8.849651 (16.201x)	1.550911 (2.839x)	0.54625
DTLZ6	8.271462 (13.772x)	1.35217 (2.251x)	0.600583
DTLZ7	11.192558 (13.297x)	2.433561 (2.891x)	0.841717
MAF1	14.924303 (16.702x)	2.426999 (2.716x)	0.893588
MAF2	20.617702 (21.457x)	2.826771 (2.942x)	0.960879
MAF3	11.381151 (14.976x)	1.623884 (2.137x)	0.759954
MAF4	7.029891 (10.172x)	1.202022 (1.739x)	0.691107
MAF5	15.33621 (18.216x)	2.415801 (2.87x)	0.841887
WFG1	14.044311 (15.133x)	2.081011 (2.242x)	0.928058
WFG2	15.851603 (16.685x)	2.562334 (2.697x)	0.950071
WFG3	19.39901 (19.17x)	3.566729 (3.525x)	1.011963
WFG4	22.195684 (21.653x)	2.793638 (2.725x)	1.025083
WFG5	22.459885 (22.003x)	2.752031 (2.696x)	1.020747
WFG6	20.409212 (21.086x)	2.357306 (2.436x)	0.967892
WFG7	25.308688 (22.018x)	3.389747 (2.949x)	1.149461
WFG8	15.601784 (16.779x)	2.137104 (2.298x)	0.929813
WFG9	25.111543 (22.232x)	3.995938 (3.538x)	1.129505
VIE1	6.408723 (8.226x)	2.977246 (3.821x)	0.779104
VIE2	5.030915 (7.143x)	2.37711 (3.375x)	0.70427
VIE3	4.021957 (7.209x)	1.600258 (2.868x)	0.557919

Table 9.6: Performance comparison among several MOMAs using the average time obtained from 30 independent executions solving 24 benchmark problems.



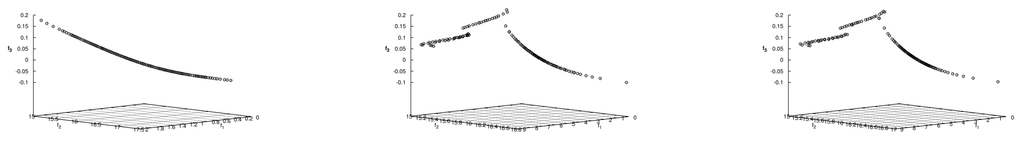
(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.8: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on VNT1 with 3 objectives



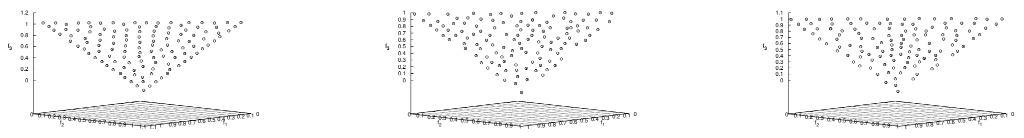
(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.9: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on VNT2 with 3 objectives



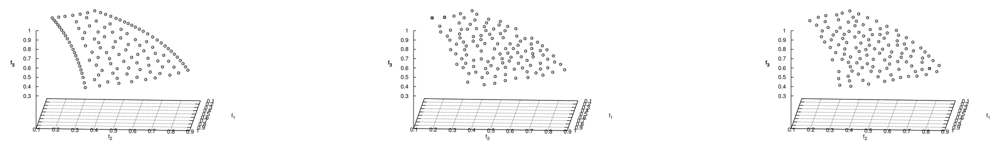
(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.10: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on VNT3 with 3 objectives



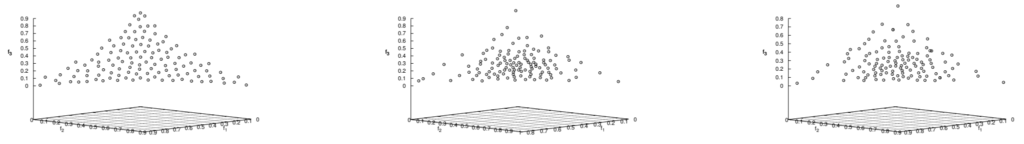
(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.11: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on MAF1 with 3 objectives



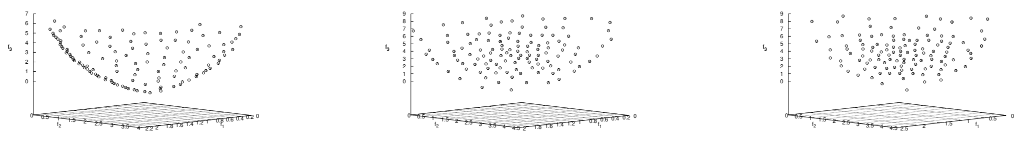
(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.12: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on MAF2 with 3 objectives



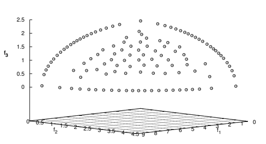
(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.13: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on MAF3 with 3 objectives

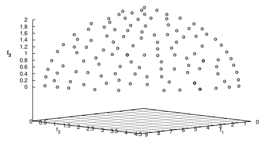


(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

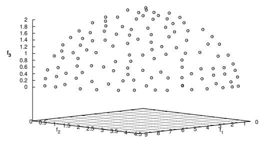
Figure 9.14: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on MAF4 with 3 objectives



(a) MOMA-I



(b) MOMA-II



(c) MOMA-II/Parallel

Figure 9.15: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on MAF5 with 3 objectives

structure of vehicle for crash-worthiness [102]. It considers five design variables, which are defined by the thickness of each member (i.e., structural pieces of the vehicle) around the frontal structure. This optimization problem involves minimizing the mass of the vehicle, deceleration during the full frontal crash (which is proportional to biomechanical injuries caused to the occupants) and toe board intrusion in the offset-frontal crash (which accounts for the structural integrity of the vehicle).

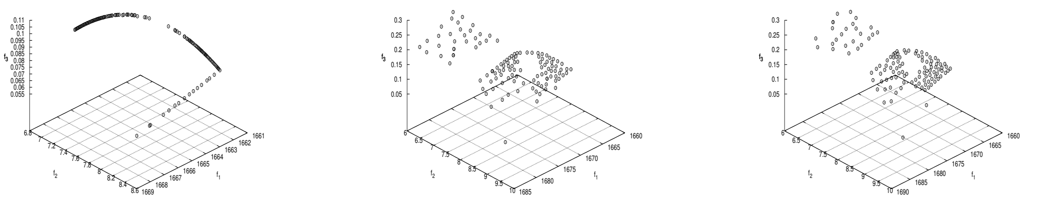
This real-world multi-objective optimization problem is formulated as follows:

$$
\begin{aligned}
&\text{Given} \\
&\quad \vec{x} = \{x_1, x_2, x_3, x_4, x_5\} \\
&\text{Minimize} \\
&\quad f_1(\vec{x}) = 1640.2823 + 2.3573285x_1 + 2.3220035x_2 \\
&\qquad + 4.5688768x_3 + 7.7213633x_4 + 4.4559504x_5 \\
&\quad f_2(\vec{x}) = 6.5856 + 1.15x_1 - 1.0424x_2 \\
&\qquad + 0.9738x_3 + 0.8364x_4 - 0.3695x_1x_4 + \\
&\qquad 0.0861x_1x_5 + 0.3628x_2x_4 - 0.1106x_1^2 \\
&\qquad - 0.3437x_3^2 + 0.1764x_4^2 \\
&\quad f_3(\vec{x}) = -0.0551 + 0.0181x_1 + 0.1024x_2 \\
&\qquad 0.0421x_3 - 0.0073x_1x_2 + 0.024x_2x_3 \\
&\qquad 0.0118x_2x_4 - 0.0204x_3x_4 - 0.008x_3x_5 \\
&\qquad - 0.0241x_2^2 + 0.0109x_4^2 \\
&\text{Where} \\
&\quad f_1 = \text{Mass} \\
&\quad f_2 = A_{in} \\
&\quad f_3 = \text{Intrusion} \\
&\text{Subject to} \\
&\qquad 1 \leq x_i \leq 3, \forall i = 1, 2, \ldots, 5.
\end{aligned} \tag{9.3}
$$

In order to evaluate the performance of our MOMA in this scenario, the total number of function evaluations for our MOMA was set in such a way that it did not exceed 50,000. The population size was set as 120 individuals. To validate the performance of our MOMA, we adopted the hypervolume indicator. Table 9.7 shows that our selection mechanism based on angle distance works as we expected because MOMA-II outperforms its original version in terms of the hypervolume indicator. Figure 9.16 presents a graphical representation of the approximations to the Pareto front obtained by each MOMA in the real-world MOP, from which we can conclude that MOMA-II is able to sample as uniformly as possible the solutions along the Pareto front. Meanwhile, MOMA-I leads the search towards the edge of the Pareto front.

	Spacing	**Modified Spacing**	**Hypervolume**
MOMA-I	**0.15472(0.002832)**	**6e-06(0.0)**	132.269572(0.209582)
MOMA-II	0.26061(0.006171)	2e-05(0.0)	**139.794453(0.078125)**
MOMA-II/Parallel	0.355658(0.045455)	3.3e-05(0.0)	139.152724(0.410714)

Table 9.7: Performance comparison among several MOMAs using the average of modified spacing, spacing and the hypervolume indicators obtained from 30 independent executions when solving a real-world multi-objective optimization problem.



(a) MOMA-I (b) MOMA-II (c) MOMA-II/Parallel

Figure 9.16: Graphical representation of the final set of solutions obtained by each Multi-Objective Memetic Algorithm on a real-world multi-objective optimization problem.

9.4 Experiment Study II: Analysis of Variance

The one-way s (ANOVA) [103, 104] was used to validate the sensitivity of our MOMA-II to its parameters (p_{size} and l factor) as well as to analyze the way in which these parameters affect the value of the hypervolume indicator. As mentioned before, the parameter p_{size} defines the size of the archive. The archive stores the solutions to sample the Pareto front shape of the MOP (these solutions will form the so-called reference point set). On the other hand, the l factor controls the selection of the n candidate solutions with the nearest angle distance. This factor affects the distribution of solutions for the reference set. In this context, we intend to observe the secondary effects of the selection value for each parameter. In this experimental study, we selected a set of MOPs where our proposed MOMA-II had a good, regular and bad performance. We adopted DTLZ1, DTLZ2, DTLZ5, DTLZ6, MAF1, MAF2, MAF5, WFG2 and WFG4 (these problems are complicated to solve and have different Pareto front shapes). We intend to analyze the parameters settings of our MOMA-II. For this reason, the ANOVA requires to establish different levels for each parameter. These levels were selected based on prior knowledge, which was determined by the previous experimental study (see Section 9.2). The design of the ANOVA is described as follows:

Combination	p_{size}	l	Combination	p_{size}	l
C1	300	2	**C10**	500	20
C2	300	5	**C11**	500	50
C3	300	10	**C12**	500	120
C4	300	20	**C13**	700	2
C5	300	50	**C14**	700	5
C6	300	120	**C15**	700	10
C7	500	2	**C16**	700	20
C8	500	5	**C17**	700	50
C9	500	10	**C18**	700	120

Table 9.8: Resulting combinations for each pair of parameters (p_{size} and l).

- Levels for each parameter are defined as:
  - $p = \{300, 500, 700\}$
  - $l = \{2, 5, 10, 20, 50, 120\}$

Base on these levels, we created 18 combinations for each pair of parameters. Each pair is shown in Table 9.8. To test our MOMA-II, we used a significance evidence level $\alpha = 0.05$ (i.e., the confidence level is 95%). The hypervolume value was computed over 30 independent executions performed for each MOP. For the rest of the parameter values of our MOMA, we adopted the same values indicated before. The hypotheses used by the ANOVA are the following:

- Null hypothesis (H_0): The means among all groups are equal (i.e., $\mu_1 = \mu_2 = , \ldots, = \mu_n$). In other words, the H_0 hypothesis implies that there is not enough evidence to claim that the mean of the hypervolume is different from another.
- Alternative hypothesis (H_A): There are some statistically significant differences between the means of each group (or population). This means that there are one or more combinations that are different regarding the hypervolume indicator.

Likewise, it is important to realize that the one-way ANOVA is a statistical test, which cannot tell that specific populations are significantly different from each other. ANOVA only indicates whether at least two or more groups are different. If the ANOVA leads to the conclusion that there is evidence that populations are not equal, we need to use a post hoc test, to determine which specific groups differed from each other. The Tukey multiple comparison test is one of the several post hoc tests that can be used to determine which group differs from among the rest [104]. This analysis

Problem	F**-value**	P**-value**	**Null Hypothesis**
DTLZ1	32.916522	2.079494e-71	False
DTLZ2	1.412057	1.248502e-01	True
DTLZ5	17.936218	4.537211e-42	False
DTLZ6	1.962303	1.195673e-02	False
DTLZ7	0.882281	5.955787e-01	True
WFG2	2.115254	5.732081e-03	False
WFG4	19.988917	1.540276e-46	False
MAF1	3.229930	1.454899e-05	False
MAF2	2.767108	1.926622e-04	False
MAF5	3.041358	4.223093e-05	False

Table 9.9: Table for describing the ANOVA results for each MOP.

compares the difference between each pair (i.e., adopting two combinations for each level) of means with appropriate adjustment for the multiple testing. The obtained results are presented using a confidence interval graph, in which each confidence interval corresponds to each pair. In this graph, if the confidence interval intersects with zero then, there is no statistically significant difference between each of the compared populations. On the other hand, if the confidence interval does not cross with zero, there is a statistically significant difference between the means of each compared group.

The obtained results reject the null hypothesis (H_0) for DTLZ1, DTLZ5, DTLZ6, WFG2, WFG4, MAF1, MAF2 and MAF5, but H_0 is only true for DTLZ2 and DTLZ7 (see Table 9.9). Our results indicate that our MOMA-II does not have perturbations when the problem is concave and unimodal. Although DTLZ7 is considered a complicated MOP (i.e., it has a disconnected Pareto front shape), the null hypothesis is true. We want to analyze the results when ANOVA rejects the null hypothesis (i.e., the alternative hypothesis). In this case, we adopt the use of box plots and interval plots to see the effects generated into our MOMA-II by each combination of parameter values. Figures 9.17, 9.18 and 9.19 show an example of box plot and interval plot for DTLZ5. The rest of Figures of this experimental study are shown in Appendix B. The description for each test where the ANOVA rejects the null hypothesis is the following:

DTLZ test suite : In this test suite, the performance of our MOMA-II did not show sensitivity to its parameters. However, we can see that some values of l affect the performance of our proposed approach (i.e., when $l > 50$). In this experimental study we showed that the size of the archive does not significantly

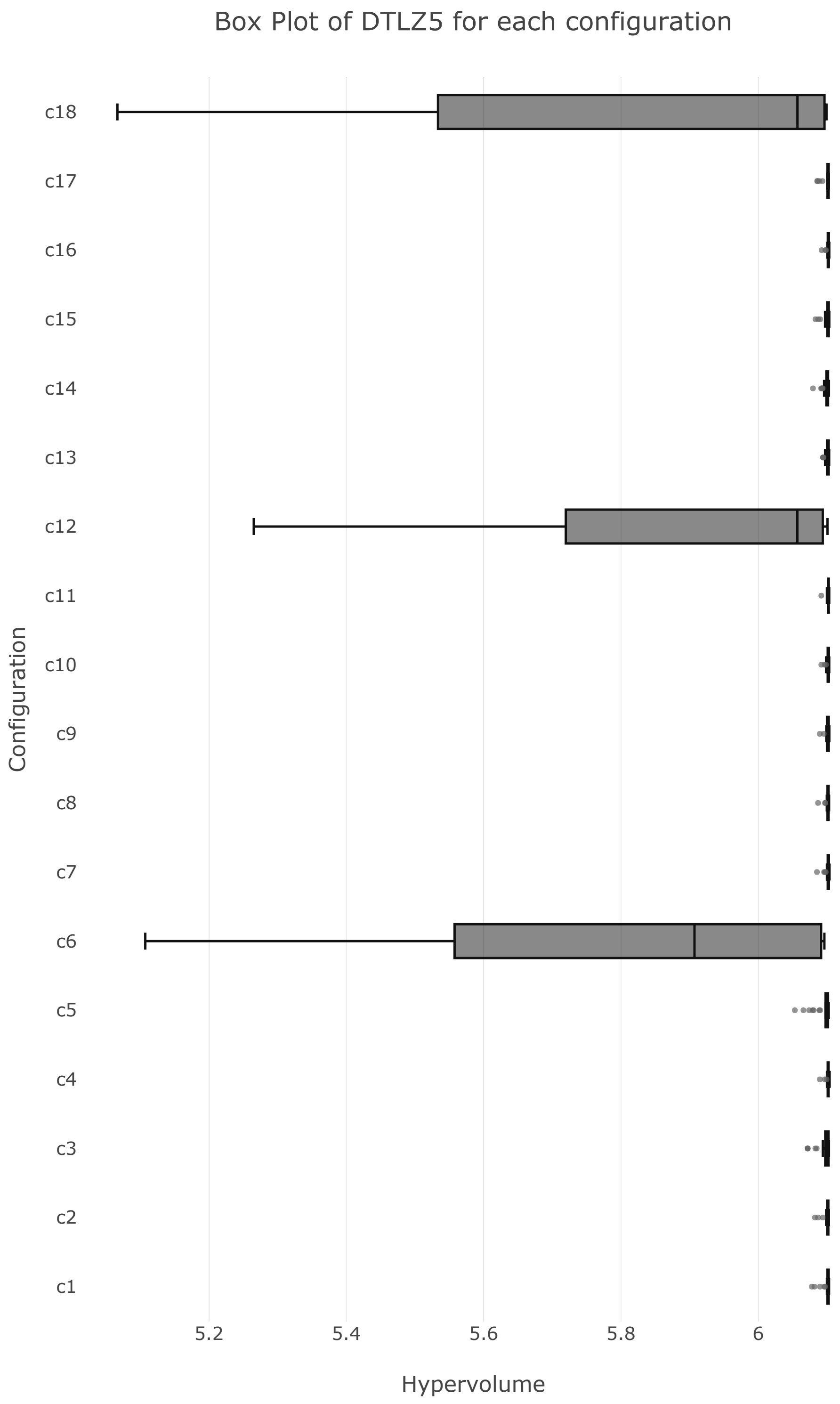


Figure 9.17: Descriptive statistics for MOMA-II solving DTLZ5 with each combination for each pair of parameters.

affect the search process.

- For DTLZ5, the best performance of our MOMA-II is in the combinations C5, C11, and C17, since the variance has the lowest value and the Tukey test verifies that there is a statistically significant difference. However, in C6, C12, and C18, our approach is affected by the number of candidate solutions when $l = 120$. For the rest of combinations, the performance of our MOMA-II is quite similar.
- For DTLZ6, although this problem is unimodal and degenerated, the performance of our approach is similar for all cases. The size of the archive and the number of candidate solutions do not affect the search process.

WFG test suite : In this test suite, we found that the performance of our algorithm depends on the type of MOP (i.e., if a MOP is unimodal or is multimodal). The Tukey test shows that the relation between l and p_{size} does not exist, since the l parameter considerably affects the behavior of our MOMA.

- For WFG2, the obtained results by our MOMA-II did not change since for each paired combination (C1-C18) of parameter values, the performance of our proposed MOMA is not affected.
- For WFG4, although this problem is multimodal, the best performance of our algorithm appears with C1, C7, and C13, since the hypervolume value increases in these scenarios. We can see that our algorithm has a regular performance when $2 \leq l \leq 50$. However, in C6, C12, and C18, our approach is affected since the variance our approach is considerably increased (i.e., when $l \geq 50$).

MAF test suite : In this test suite, we found that some combinations do not have any effect on performance, and the best performance of our MOMA is obtained when the parameters are $2 \leq l \leq 50$ and $300 \leq p_{size} \leq 700$.

- For MAF1, C12 is the best combination found to solve MOPs with inverted simplex-like Pareto front shapes. It is worth noting that the performance of our proposed algorithm is very similar for each combination (i.e., C1-C11 and C13-C18). In this case, we suggest to use $p_{size} = 500$ and $l = 120$.
- In the experimental study for MAF2, we found that the combination C6 affects the performance of our MOMA, since the variance increases when

$l = 120$. On the other hand, when we adopt C1-C9, our approach is able to obtain similar results. The Tukey test shows that there is no evidence that any combination of parameter values can produce a better approximations.

- For MAF5, the Tukey test shows that the C6 combination, as mentioned before, is the worst configuration to use in our MOMA. We can see that p_{size} does not affect the performance of the proposed approach. However, if $l > 50$, the performance of our algorithm is not steady (i.e., the variance of the hypervolume presents a considerable increase).

9.5 Final Remarks

We have proposed a new Multi-Objective Memetic Algorithm which has an IGD$^+$-based local search engine. Our algorithm adopts a novel method for computing the hypervolume contribution, which is based on the used of angle distances. MOMA-II avoids keeping solutions in the knee regions, and it is able to spread the solutions along the Pareto front as uniformly as possible. We tested our MOMA on a multi-objective optimization problem for the crash safety design of vehicles, where we showed that the modified hypervolume-based approach is able to properly sample the Pareto front shape. Based on such results, we claim that our proposed approach is a competitive alternative to deal with MOPs having irregular Pareto front shapes, even in real-world MOPs. Our results indicate that it is possible to improve the behavior of its original version (i.e., MOMA-I) for solving MOPs with complicated Pareto front shapes. Our proposal includes a parallel implementation which makes it possible to launch multiple local search processes at the same time. We showed that the use of multiple processors helps to improve the performance of MOMA and avoids to select the solutions on which we need to launch the local search engine. It is worth emphasizing that our proposed Parallel multi-objective memetic algorithm is able to achieve a significant speedup (of up to 22x) with respect to its original version.

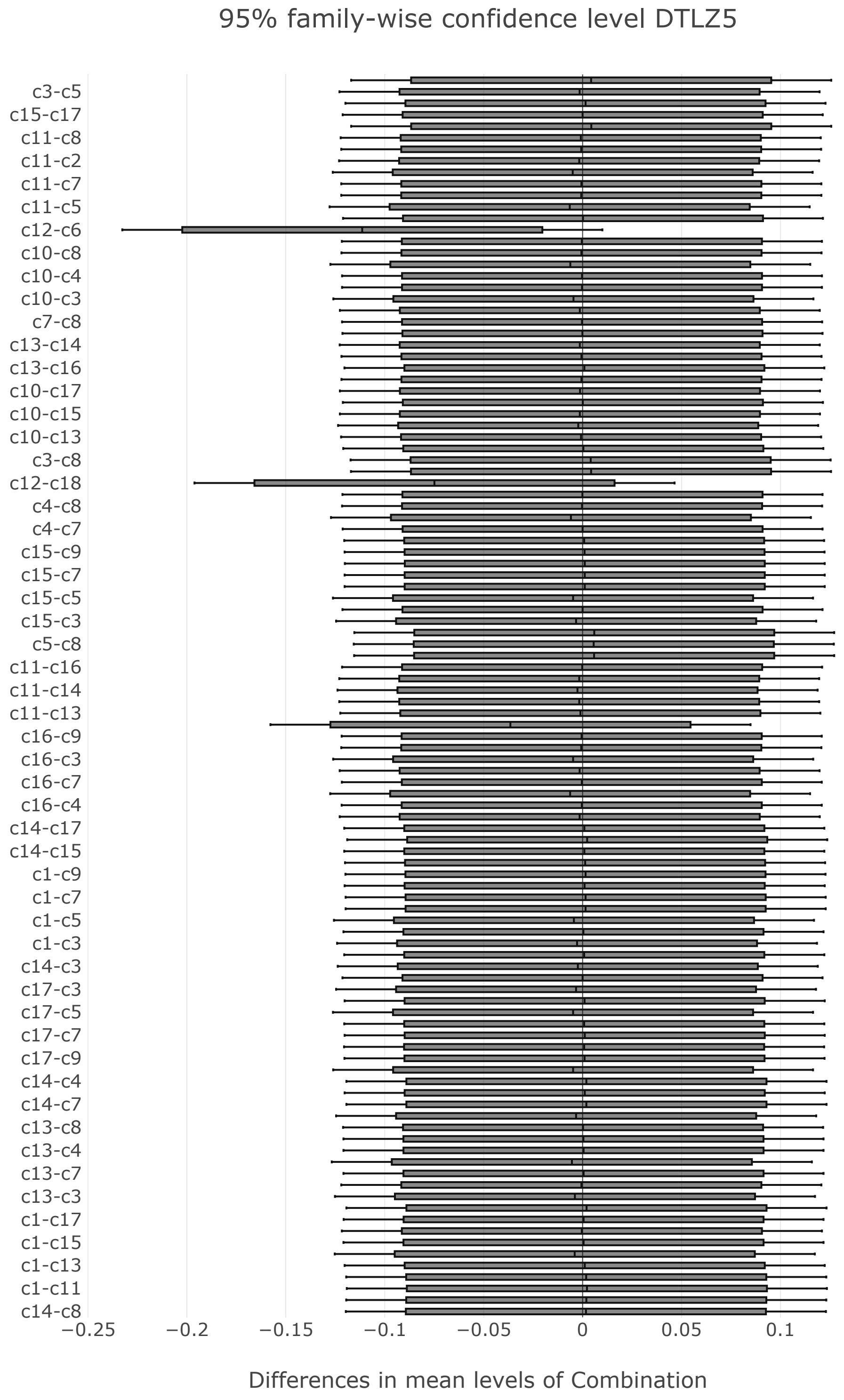


Figure 9.18: Graphical representation of each confidence interval obtained by the Tukey test for DTLZ5, where each confidence interval intersects with zero.

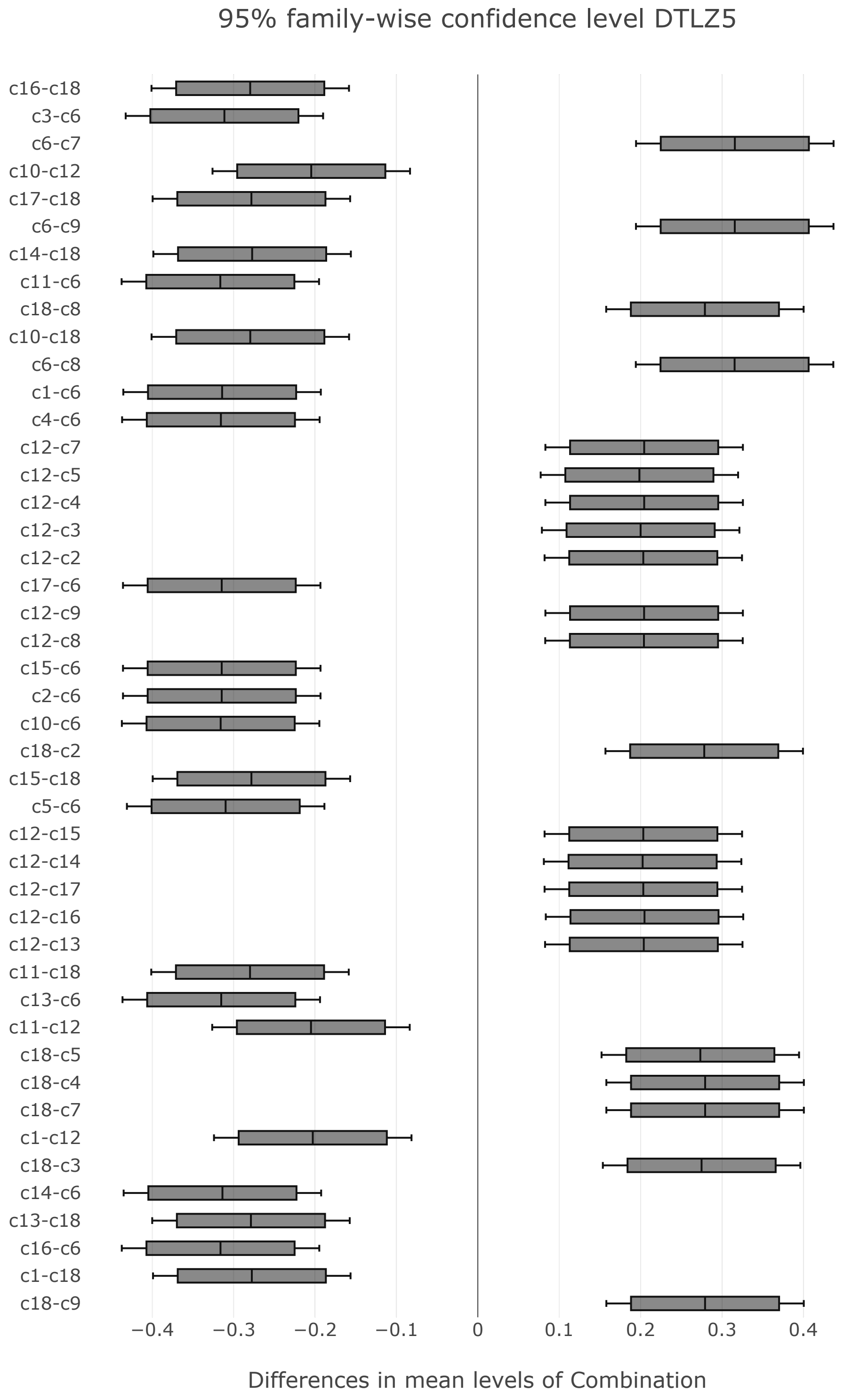


Figure 9.19: Graphical representation of each confidence interval obtained by the Tukey test for DTLZ5, where each confidence interval does not intersect with zero.

Chapter 10

Conclusions and Future Work

In this work, we have presented the contributions developed so far for improving Multi-Objective Evolutionary Algorithms by using a local search technique based on the IGD$^+$ indicator. Such contributions follow two main goals:

- To approximate a good representation of the Pareto front ($\mathcal{PF}$).
- Decrease the running time of the Multi-Objective Memetic Algorithm.

Throughout this work, we showed the capabilities of the IGD$^+$ indicator when was used for driving the optimization process. Furthermore, we introduced different strategies to hybridize the IGD$^+$ indicator with a Multi-Objective Memetic Algorithm. In the following, we present the final remarks derived from in this study.

10.1 Conclusions

We have proposed a new indicator-based approach for solving many-objective problems. This method indicates that the IGD$^+$ indicator constitutes a good choice to approximate the True Pareto Front. As can be observed in Chapter 5, the Pareto compliant property between two-objective vectors is of utmost importance and improves the performance of the selection mechanism of a MOEA. Our study showed that an IGD$^+$-based selection mechanism is an effective way to lead the search towards a single region of the objective space. Likewise, we have proposed a new Multi-Objective Memetic Algorithm which has an IGD$^+$-based local search engine (see Chapter 7). The core idea of our proposed algorithm is to combine properties of two different performance indicators (the hypervolume indicator and the IGD$^+$ indicator). For this reason, we created a local search engine based on the IGD$^+$

indicator and a global optimizer based on hypervolume indicator. We proposed a GPU-based MOMA which makes it possible to launch multiple local search processes at the same time. We showed that the use of GPUs helps to improve the performance of the MOMA and avoids the problem of deciding on which solutions we need to launch the local search engine. Our results indicate that it is possible to improve the convergence of a MOEA if the global optimizer incorporates an indicator-based local search engine.

In Chapter 9, we proposed an improvement to our MOMA. MOMA-II adopts a novel method for computing the hypervolume contribution, which can lead the optimization process to sample the solutions as uniformly as possible. MOMA-II avoids to keep solutions in knee regions, and it is able to provide a good spread of solutions along the Pareto front. Our results indicate that it is possible to improve the behavior of its original version (i.e., MOMA-I) in MOPs with complex Pareto front shapes. We tested our MOMA on a MOP that models the crash safety design of vehicles, where we showed that the modified hypervolume-based approach is able to properly sample the Pareto front shape of a real-world problem. We can conclude that our proposed approach is a competitive alternative to deal with MOPs having irregular Pareto front shapes, and even complicated real-world MOPs.

10.2 Future Work

As part of our future work, we would like to validate our improved MOMA in many-objective optimization problems, in order to observe its performance. Additionally, we would like to test our GPU-based MOMA in more real-world MOPs. We would also like to study the use of other performance indicators (or combinations of them) to design local search engines. Some possible options are δ_p and $R2$.

Appendix A

Test problems

In this Appendix, we review some test suites, which are commonly used in many-objective optimization. Here, we describe the DTLZ [92], WFG [93], MAF [101] and VNT [28] test suites, which are defined for real-valued and unconstrained problems. These test suites proposed different Pareto front shapes, which are attractive to use for assessing the performance of MOEAs. These test suites are based on the properties of their Pareto fronts. We categorized the test problems adopted into different groups: convex, concave, inverted simplex-like, disconnected, degenerate and badly-scaled. Here, we describe the definition of each test problem used in this thesis.

A.1 Deb-Thiele-Laumanns-Zitzler Test Suite

The Deb-Thiele-Laumanns-Zitzler (DTLZ) Test Suite [92] includes nine representative test problems for comparing multi-objective optimizers, which are scalable to any number of decision variables and objectives. In the following, we present the seven unconstrained problems of the DTLZ test suite. In this test suite, the number of objectives is represented by m. The total number of decision variables is given by $n = m + k - 1$, where k is the number of distance parameters.

A.1.1 DTLZ1

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

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

All objective function values lie on the linear hyper-plane $\sum_{i=1}^{m} f_i = 0.5$. The difficulty in this problem is to converge to the hyper-plane, since the search space contains $(11^k - 1)$ local Pareto optimal fronts. The authors suggest to use $k = 5$. The Pareto optimal front is shown in Figure A.1.

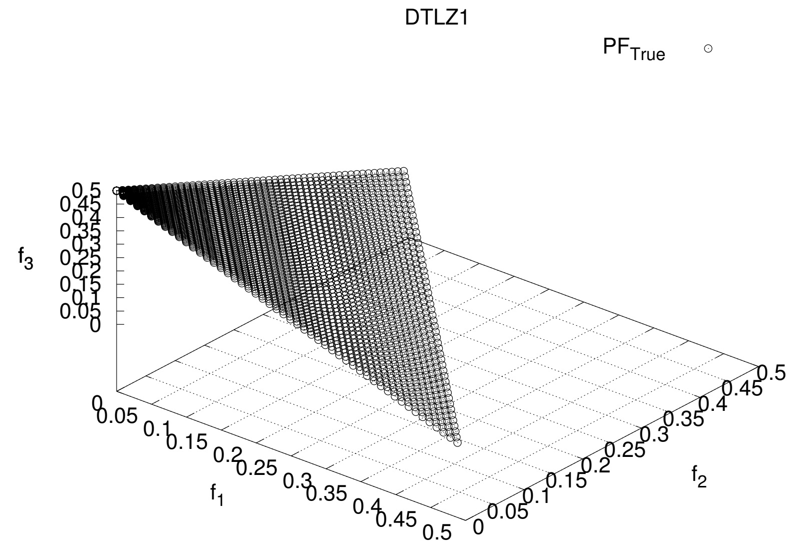


Figure A.1: Pareto optimal front with three objective functions corresponding to DTLZ1

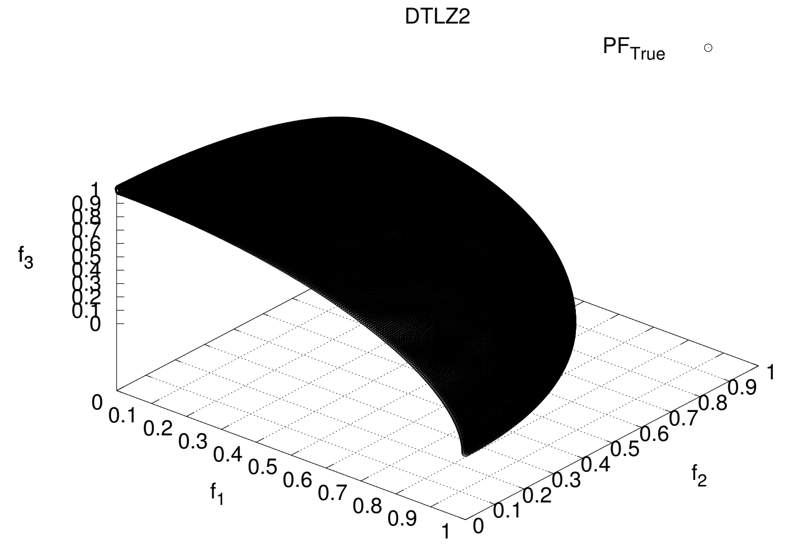


Figure A.2: Pareto optimal front with three objective functions corresponding to DTLZ2

A.1.2 DTLZ2

This problem is separable and unimodal. The geometry of its Pareto optimal front is concave. DTLZ2 is defined as follows:

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

The Pareto optimal solutions are produced when $\vec{y} = (0.5, 0.5, \dots)^T$ and all objective functions values must satisfy that $\sum_{i=1}^{m}(f_i)^2 = 1$. The authors of this problem suggest to use $k = 10$ (see Figure A.2).

A.1.3 DTLZ3

This problem is defined as DTLZ2 but it includes a new g function, that makes it multifrontal. The definition is given as follows:

$$
\begin{aligned}
&\text{Given}\\
&\qquad \vec{x} = \{x_1, \dots, x_{m-1}, x_m, \dots, x_n\}\\
&\text{Minimize}\\
&\qquad f_1(\vec{x}) = (1+g(\vec{y}))\prod_{i=1}^{m-1}\cos\left(\frac{x_i\pi}{2}\right)\\
&\qquad f_{j=2:m-1}(\vec{x}) = (1+g(\vec{y}))\left(\prod_{i=1}^{m-j}\cos\left(\frac{x_i\pi}{2}\right)\right)\sin\left(\frac{x_{m-j+1}\pi}{2}\right)\\
&\qquad f_m(\vec{x}) = (1+g(\vec{y}))\sin\left(\frac{x_1\pi}{2}\right)\\
&\text{Where}\\
&\qquad y_{i=1:k} = \{x_m, x_{m+1}, \dots, x_n\}\\
&\qquad g(\vec{y}) = 100\left[k + \sum_{i=1}^{k}(y_i-0.5)^2 - \cos(20\pi(y_i-05))\right]\\
&\text{Subject to}\\
&\qquad 0 \le x_i \le 1, \forall i = 1, 2, \dots, n.
\end{aligned}
\tag{A.3}
$$

The above function g introduces $(3^k - 1)$ local Pareto optimal fronts, and one global Pareto optimal front. Each of these local Pareto fronts are parallel to the global Pareto optimal front. This makes optimizers to converge towards local Pareto optimal fronts. The global Pareto optimal front corresponds to $\vec{y} = (0.5, 0.5, \dots)^T$ (see Figure A.3).

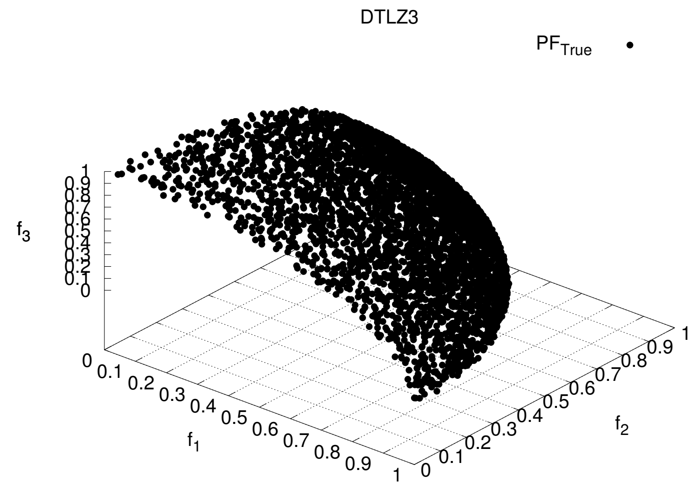


Figure A.3: Pareto optimal front with three objective functions corresponding to DTLZ3

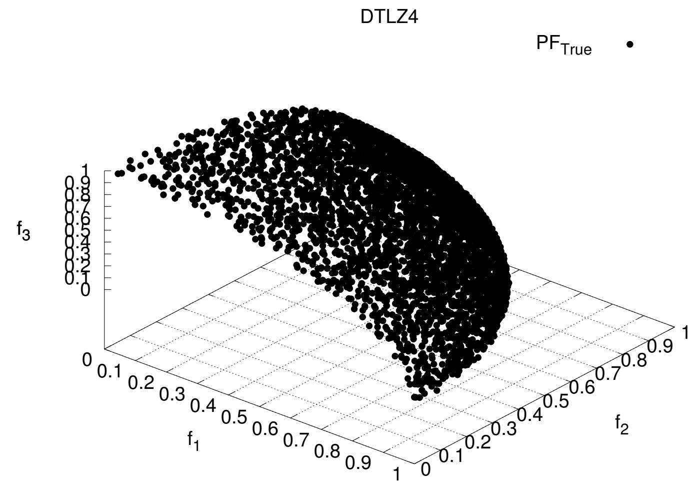


Figure A.4: Pareto optimal front with three objective functions corresponding to DTLZ4.

A.1.4 DTLZ4

This problem is concave, separable and unimodal, and it 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

$$\begin{aligned}
y_{i=1:k} &= \{x_m, x_{m+1}, \ldots, x_n\} \\
g(\vec{y}) &= 100\left[k + \sum_{i=1}^{k} (y_i - 0.5)^2 - \cos(20\pi(y_i - 05))\right]
\end{aligned}$$

Subject to $\quad 0 \leq x_i \leq 1, \forall i = 1, 2, \ldots, n.$

(A.4)

The parameters $\alpha = 100$ and $k = 10$ are suggested here. This problem allows a dense set of solutions to exist near the $f_m - f_1$ plane. Figure A.4 shows the true Pareto front shape with three objective functions.

A.1.5 DTLZ5

This problem is unimodal and degenerated. DTLZ5 is defined as:

Given

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

Minimize

$$\begin{aligned}
f_1(\vec{x}) &= (1 + g(\vec{y})) \prod_{i=1}^{m-1} \cos\left(\frac{\theta_i \pi}{2}\right) \\
f_{j=2:m-1}(\vec{x}) &= (1 + g(\vec{y})) \left(\prod_{i=1}^{m-j} \cos\left(\frac{\theta_i \pi}{2}\right)\right) \sin\left(\frac{\theta_{m-j+1} \pi}{2}\right) \\
f_m(\vec{x}) &= (1 + g(\vec{y})) \sin\left(\frac{\theta_1 \pi}{2}\right)
\end{aligned}$$

Where

$$\begin{aligned}
y_{i=1:k} &= \{x_m, x_{m+1}, \dots, x_n\} \\
\theta_i &= \begin{cases} x_i & i = 1 \\ \frac{1+2g(\vec{y})}{2(1+g(\vec{y}))} x_i & \forall i \in \{2, 3, \dots, m-1\} \end{cases} \\
g(\vec{y}) &= \sum_{i=1}^{k} (y_i - 0.5)^2
\end{aligned}$$

Subject to $\quad 0 \leq x_i \leq 1, \forall i = 1, 2, \dots, n.$

(A.5)

The g function with $k = 10$ variables is suggested. The Pareto optimal front corresponds to $\vec{y} = (0.5, \dots, 0.5)^T$, and all objective function values must satisfy $\sum_{i=1}^{m} (f_i)^2 = 1$ (see Figure A.5).

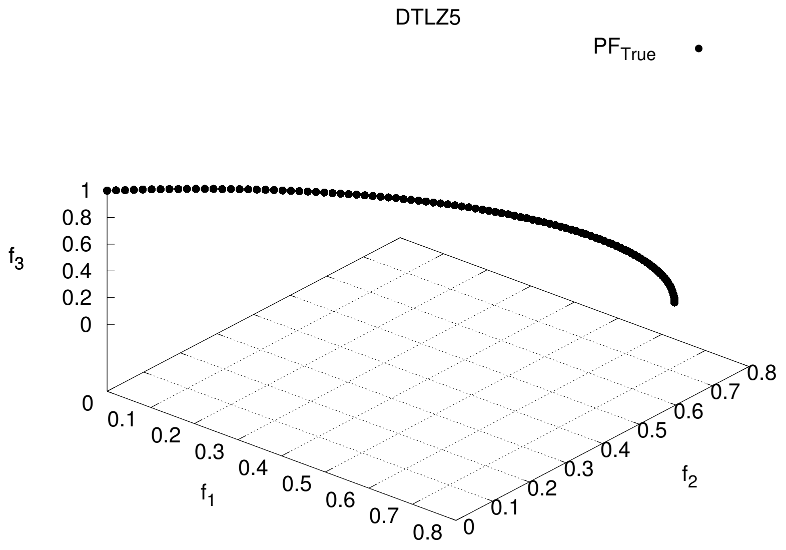


Figure A.5: Pareto optimal front with three objective functions corresponding to DTLZ5

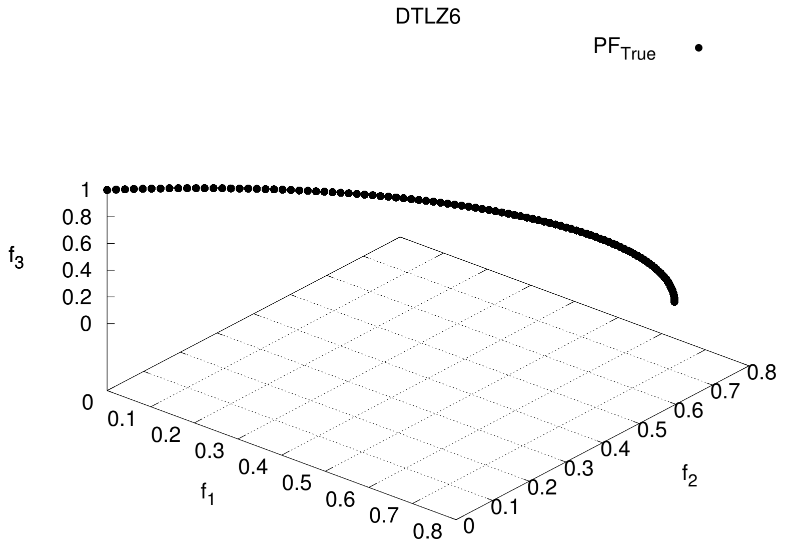


Figure A.6: Pareto optimal front with three objective functions corresponding to DTLZ6

A.1.6 DTLZ6

This problem is a modified version of DTLZ5. The resulting problem is unimodal and degenerated and it is defined as follows:

Given

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

Minimize

$$\begin{aligned}
f_1(\vec{x}) &= (1 + g(\vec{y})) \prod_{i=1}^{m-1} \cos\left(\frac{\theta_i \pi}{2}\right) \\
f_{j=2:m-1}(\vec{x}) &= (1 + g(\vec{y})) \left(\prod_{i=1}^{m-j} \cos\left(\frac{\theta_i \pi}{2}\right) \right) \sin\left(\frac{\theta_{m-j+1} \pi}{2}\right) \\
f_m(\vec{x}) &= (1 + g(\vec{y})) \sin\left(\frac{\theta_1 \pi}{2}\right)
\end{aligned}$$

Where

$$\begin{aligned}
y_{i=1:k} &= \{x_m, x_{m+1}, \ldots, x_n\} \\
\theta_i &= \begin{cases} x_i & i = 1 \\ \frac{1+2g(\vec{y})}{2(1+g(\vec{y}))} x_i & \forall i \in \{2, 3, \ldots, m-1\} \end{cases} \\
g(\vec{y}) &= \sum_{i=1}^{k} y_i^{0.1}
\end{aligned}$$

Subject to $\quad 0 \leq x_i \leq 1, \forall i = 1, 2, \ldots, n.$

(A.6)

The Pareto optimal front corresponds to $\vec{y} = (0, \ldots, 0)^T$. The value of k is chosen as 10. Figure A.6 shows the true Pareto front shape of this problem.

A.1.7 DTLZ7

This problem has a disconnected Pareto front shape consisting of 2^{m-1} Pareto optimal regions. This problem is a good example for simulating a multi-objective real-world problem. DTLZ7 is defined as follows:

Given

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

Minimize

$$f_{j=1:m-1}(\vec{x}) = f_i$$
$$f_m(\vec{x}) = (1 + g(\vec{y}))\left(m - \sum_{i=1}^{m-1}\left(\frac{f_i}{1+g(\vec{y})}(1 + \sin(3\pi f_i))\right)\right)$$

Where

$$y_{i=1:k} = \{x_m, x_{m+1}, \ldots, x_n\}$$
$$g(\vec{y}) = 1 + \frac{9}{k}\sum_{i=1}^{k} y_i$$

Subject to

$$0 \leq x_i \leq 1, \forall i = 1, 2, \ldots, n. \tag{A.7}$$

The function g requires $k = 10$. The Pareto optimal solutions correspond to $\vec{y} = (0, \ldots, 0)^T$. Figure A.7 shows the true Pareto front shape of DTLZ7.

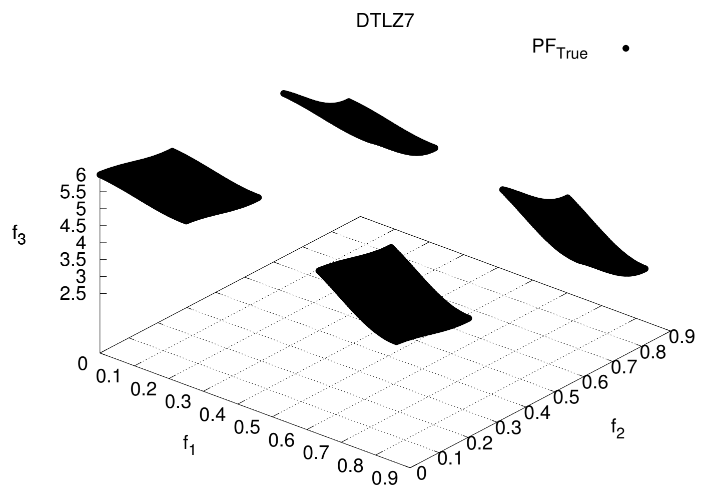


Figure A.7: Pareto optimal front with three objective functions corresponding to DTLZ7

A.2 Walking Fish Group Test Suite (WFG)

The Walking-Fish-Group test suite was published in 2005 by Huband et al. [93]. This test suite suggests nine multi-objective test problems (WFG1 to WFG9), which are scalable with respect to both the number of decision variables and the number of objectives. These problems include a wide variety of Pareto front shapes. Moreover, characteristics such as bias, multi-modality, and non-separability, are defined by a set of transformations. In the following, we present this test suite. Here, m represents the number of objectives, and each problem is defined in terms of a decision vector $\vec{x} \in \mathbb{R}^n$. All $x_i \in \vec{x}$ will have a domain $[0, 1]$. In this test suite, 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 from a vector of working parameters $\vec{z} \in \mathbb{R}^n$. The domain of all $z_i \in \vec{z}$ is $[0, 2i]$. It is worth nothing, that the number of variables n is defined as $n \leq m$ and $n = k+l$, where $k \in \{m-1, 2(m-1), 3(m-1), \dots\}$ is the position related parameter, and l is known as the distance related parameter. Each transition vector adds complexity to the underlying problem. The multi-objective optimizer directly manipulates the $\vec{z}$ vector, through which $\vec{x}$ is indirectly manipulated.

A.2.1 WFG1

WFG1 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 to solve (see Figure A.8). Its definition is given as:

Given

$$\vec{z} = \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\}$$

Minimize

$$\begin{aligned}
f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}\left(1-\cos\left(\frac{x_i\pi}{2}\right)\right)\\
f_{j=2:m-1}(\vec{x}) &= x_m + 2j\left(\prod_{i=1}^{m-j}\left(1-\cos\left(\frac{x_i\pi}{2}\right)\right)\right)\left(1-\sin\left(\frac{x_{m-j+1}\pi}{2}\right)\right)\\
f_m(\vec{x}) &= x_m + 2m\left(1 - x_1 - \frac{\cos\left(\frac{10\pi x_1}{2}\right)}{10\pi}\right)
\end{aligned}$$

Where

$$\begin{aligned}
x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \left\{\frac{2(i-1)k}{(m-1)}+1, \ldots, \frac{2ik}{(m-1)}\right\}\right)\\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_n\}, \{2(k+1), \ldots, 2n\}\right)\\
y_{i=1:n} &= \text{b\_poly}(y'_i, 0.02)\\
y'_{i=1:k} &= y''_i\\
y'_{i=k+1:n} &= \text{b\_flat}(y''_i, 0.8, 0.75, 0.85)\\
y''_{i=1:k} &= \frac{z_i}{2i}\\
y''_{i=k+1:n} &= \text{s\_linear}\left(\frac{z_i}{2i}, 0.35\right)
\end{aligned} \tag{A.8}$$

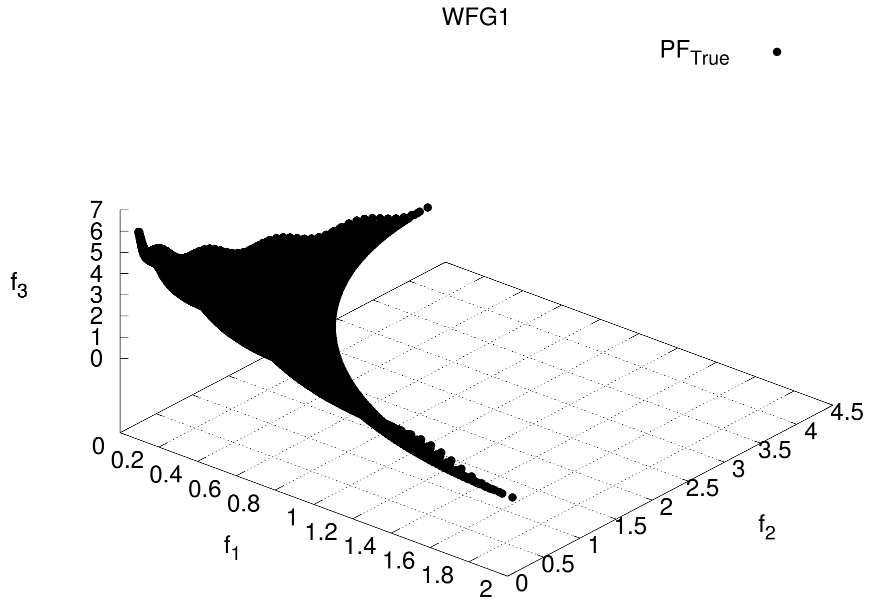


Figure A.8: Pareto optimal front with three objective functions corresponding to WFG1

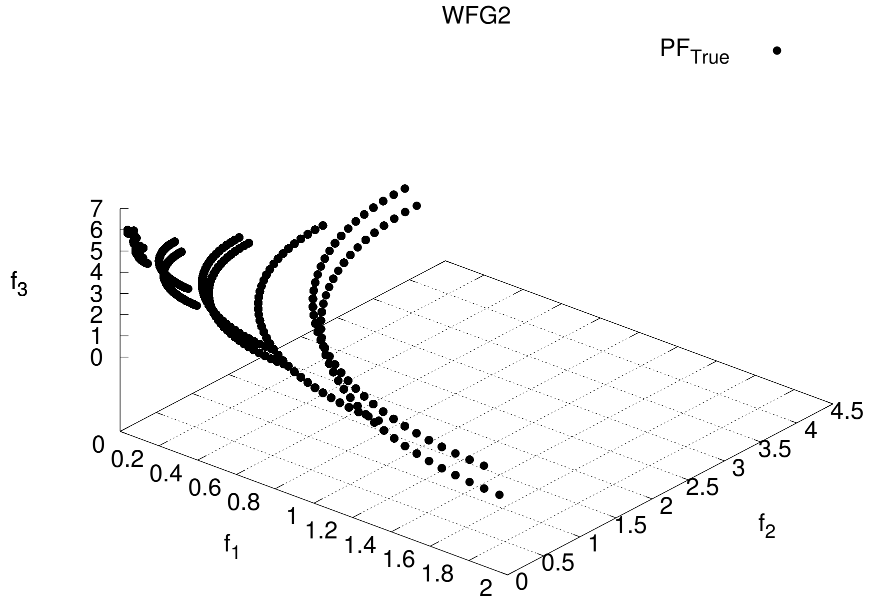


Figure A.9: Pareto optimal front for three objectives corresponding to WFG2

A.2.2 WFG2

This problem is non-separable and multimodal. The Pareto optimal front shape is disconnected (see Figure A.9). It is given by the following expression:

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

Where

$$
\begin{aligned}
x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \{1, \ldots, 1\}\right) \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_{k+l/2}\}, \{1, \ldots, 1\}\right) \\
y'_{i=1:k} &= y'_i \\
y'_{i=k+1:k+l/2} &= \text{r\_nonsep}(\{y'_{k+2(i-k)-1}, y'_{k+2(i-k)}\}, 2) \\
y'_{i=1:k} &= \frac{z_i}{2i} \\
y'_{i=k+1:n} &= \text{s\_linear}\left(\frac{z_i}{2i}, 0.35\right)
\end{aligned}
\tag{A.9}
$$

A.2.3 WFG3

This problem is non-separable but unimodal. It has a linear and degenerate Pareto front shape (see Figure A.10). It is given by the following expression:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{z} = \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
&\text{Minimize} \\
f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1}(x_i) \\
f_{j=2:m-1}(\vec{x}) &= x_m + 2j\left(\prod_{i=1}^{m-j} x_i\right)(1 - x_{m-j+1}) \\
f_m(\vec{x}) &= x_m + 2m(1 - x_1) \\
\text{Where } x_{i=1} &= u_i \\
x_{i=2:m-1} &= x_m(u_i - 0.5) + 0.5 \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_{k+l/2}\}, \{1, \ldots, 1\}\right) \\
u_i &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \{1, \ldots, 1\}\right) \\
y'_{i=1:k} &= y'_i \\
y'_{i=k+1:k+l/2} &= \text{r\_nonsep}(\{y'_{k+2(i-k)-1}, y'_{k+2(i-k)}\}, 2) \\
y'_{i=1:k} &= \frac{z_i}{2i} \\
y'_{i=k+1:n} &= \text{s\_linear}\left(\frac{z_i}{2i}, 0.35\right)
\end{aligned}
\tag{A.10}
$$

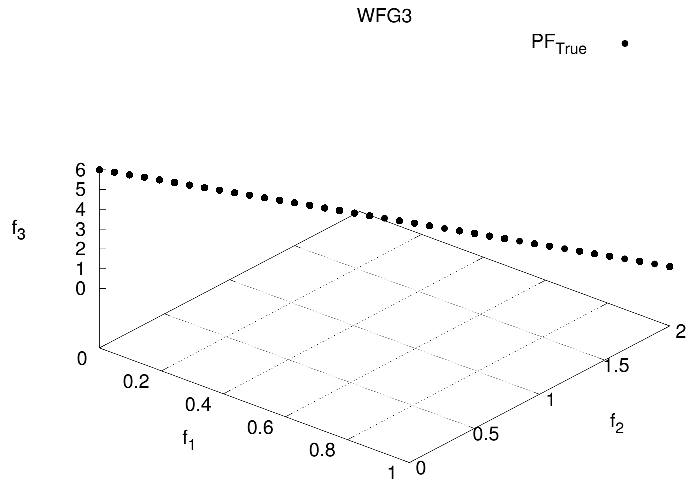


Figure A.10: Pareto optimal front for three objectives corresponding to WFG3

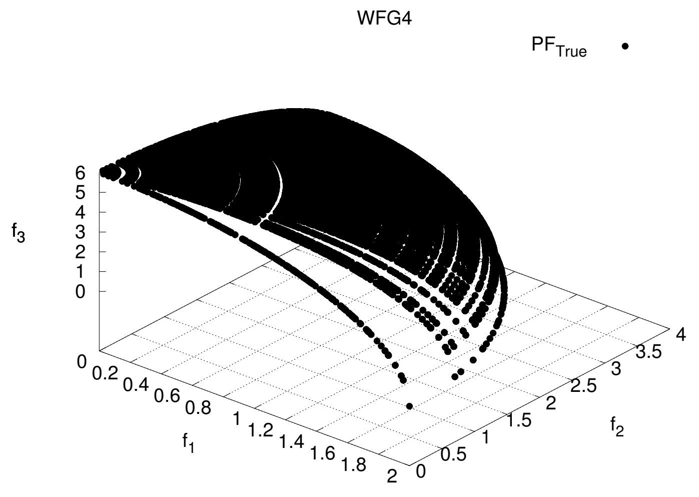


Figure A.11: Pareto optimal front for three objectives corresponding to WFG4

A.2.4 WFG4

WFG4 is separable, but highly multimodal. Its Pareto front shape is concave (see Figure A.11) and is defined as follows:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{z} = \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
&\text{Minimize} \\
f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1} \sin(x_i\pi/2) \\
f_{j=2:m-1}(\vec{x}) &= x_m + 2j\left(\prod_{i=1}^{m-j} \sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
f_m(\vec{x}) &= x_m + 2m\cos(x_1\pi/2) \\
&\text{Where} \\
x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_{ik/(m-1)}\}, \{1, \ldots, 1\}\right) \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_n\}, \{1, \ldots, 1\}\right) \\
y_{i=1:n} &= \text{s\_multi}\left(\frac{z_i}{2i}, 30, 10, 0.35\right)
\end{aligned}
\tag{A.11}
$$

A.2.5 WFG5

This problem is deceptive and separable, and its Pareto front shape is concave (see Figure A.12). WFG5 is defined by the following expression:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{z} = \{z_1, \dots, z_k, z_{k+1}, \dots, z_n\} \\
&\text{Minimize} \\
&\qquad f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1} \sin(x_i\pi/2) \\
&\qquad f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j} \sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
&\qquad f_m(\vec{x}) = x_m + 2m\cos(x_1\pi/2) \\
&\text{Where} \\
&\qquad x_{i=1:m-1} = \text{r\_sum}\left(\{y_{k+1}, \dots, y_{ik/(m-1)}\}, \{1, \dots, 1\}\right) \\
&\qquad x_m = \text{r\_sum}\left(\{y_{k+1}, \dots, y_n\}, \{1, \dots, 1\}\right) \\
&\qquad y_{i=1:n} = \text{s\_decept}\left(\frac{z_i}{2i}, 0.35, 0.001, 0.05\right)
\end{aligned}
\tag{A.12}
$$

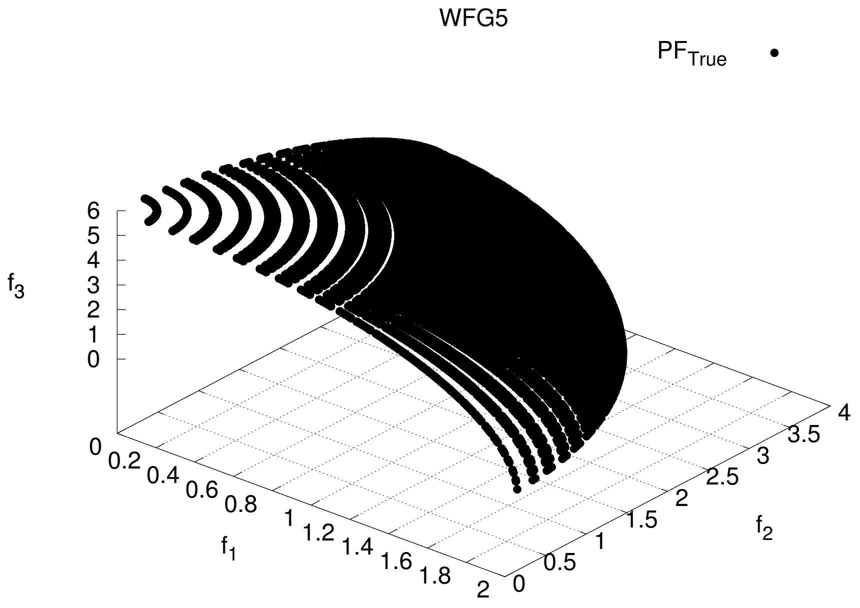


Figure A.12: Pareto optimal front for three objectives corresponding to WFG5

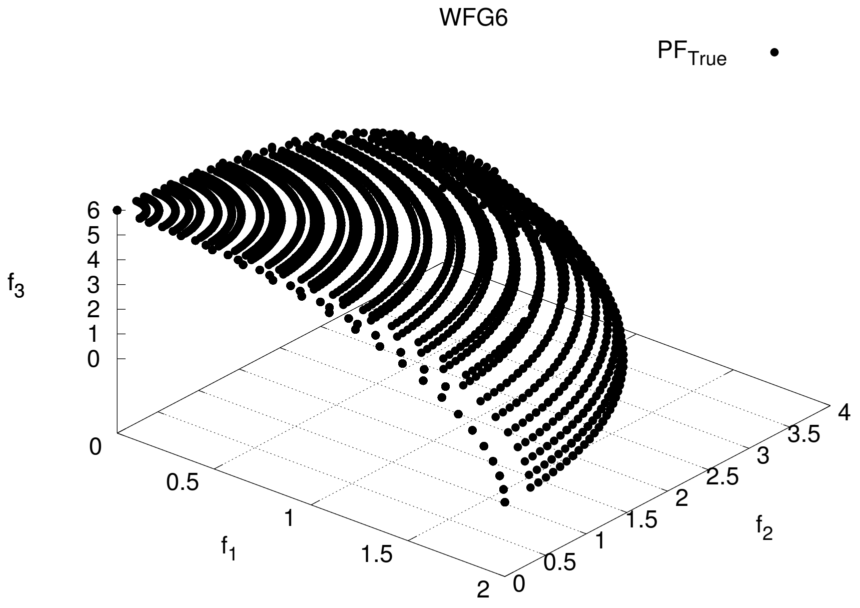


Figure A.13: Pareto optimal front for three objectives corresponding to WFG6

A.2.6 WFG6

This problem is deceptive and separable, and its Pareto front shape is concave (see Figure A.13). It is defined by the following expression:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{z} = \{z_1, \dots, z_k, z_{k+1}, \dots, z_n\} \\
&\text{Minimize} \\
&\qquad f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1} \sin(x_i\pi/2) \\
&\qquad f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j} \sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
&\qquad f_m(\vec{x}) = x_m + 2m\cos(x_1\pi/2) \\
&\text{Where} \\
&\qquad x_{i=1:m-1} = \text{r\_nonsep}\left(\{y_{(i-1)k/(m-1)+1}, \dots, y_{ik/(m-1)}\}, \{1, \dots, 1\}\right) \\
&\qquad x_m = \text{r\_nonsep}\left(\{y_{k+1}, \dots, y_n\}, l\right) \\
&\qquad y_{i=1:k} = \frac{z_i}{2i} \\
&\qquad y_{i=k+1:n} = \text{s\_decept}\left(\frac{z_i}{2i}, 0.35\right)
\end{aligned}
\tag{A.13}
$$

A.2.7 WFG7

This problem is separable and unimodal. It has a concave Pareto optimal front which is shown in Figure A.14. It is defined as follows:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{z} = \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
&\text{Minimize} \\
f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1} \sin\left(x_i\pi/2\right) \\
f_{j=2:m-1}(\vec{x}) &= x_m + 2j\left(\prod_{i=1}^{m-j} \sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
f_m(\vec{x}) &= x_m + 2m\cos(x_1\pi/2) \\
&\text{Where} \\
x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \{1, \ldots, 1\}\right) \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_n\}, \{1, \ldots, 1\}\right) \\
y_{i=1:k} &= y'_i \\
y_{i=k+1:n} &= \text{s\_linear}(y'_i, 0.35) \\
y'_{i=1:k} &= \text{b\_param}(z_i/(2i), \text{r\_sum}(\{z_{i+1}/(2(i+1)), \ldots, z_n/2n\}, \\
&\quad \{1, \ldots, 1\}), \frac{0.98}{49.98}, 0.02, 50) \\
y_{i=k+1:n} &= \frac{z_i}{2i}
\end{aligned}
\tag{A.14}
$$

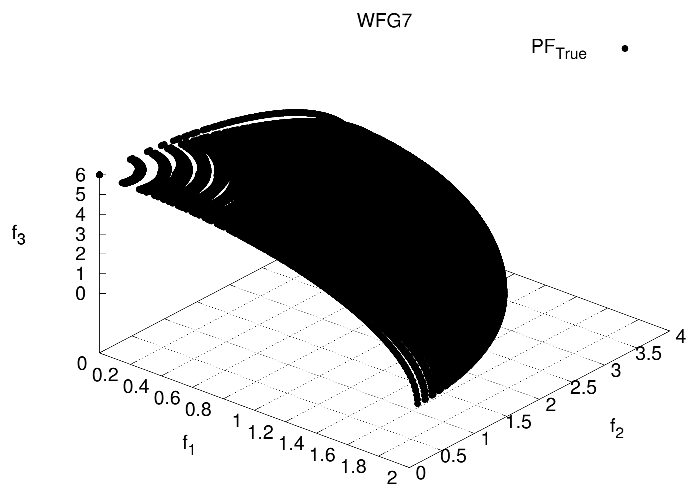


Figure A.14: Pareto optimal front for three objectives corresponding to WFG7

A.2.8 WFG8

This problem has a parameter dependent bias, and is non-separable and unimodal. The Pareto optimal front shape is concave (see Figure A.15), and is defined as:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{z} = \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
&\text{Minimize} \\
f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1} \sin(x_i\pi/2) \\
f_{j=2:m-1}(\vec{x}) &= x_m + 2j\left(\prod_{i=1}^{m-j} \sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
f_m(\vec{x}) &= x_m + 2m\cos(x_1\pi/2) \\
&\text{Where} \\
x_{i=1:m-1} &= \text{r\_sum}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, \{1, \ldots, 1\}\right) \\
x_m &= \text{r\_sum}\left(\{y_{k+1}, \ldots, y_n\}, \{1, \ldots, 1\}\right) \\
y_{i=1:k} &= y'_i \\
y_{i=k+1:n} &= \text{s\_linear}(y'_i, 0.35) \\
y_{i=1:k} &= \frac{z_i}{2i} \\
y'_{i=k+1:n} &= \text{b\_param}(z_i/(2i), \text{r\_sum}(\{z_1/2, \ldots, z_{i-1}/(2(i-1))\}, \\
&\quad \{1, \ldots, 1\}), \frac{0.98}{49.98}, 0.02, 50)
\end{aligned}
\tag{A.15}
$$

A.2.9 WFG9

WFG9 is non-separable, multimodal, deceptive, and has a parameter dependent bias. These features make it a very difficult problem. The Pareto optimal front shape is concave, but in the edge of the shape is flat (see Figure A.16). This problem is defined as:

$$\begin{aligned}
&\text{Given} \\
&\qquad \vec{z} = \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
&\text{Minimize} \\
f_1(\vec{x}) &= x_m + 2\prod_{i=1}^{m-1} \sin(x_i\pi/2) \\
f_{j=2:m-1}(\vec{x}) &= x_m + 2j\left(\prod_{i=1}^{m-j} \sin(x_i\pi/2)\right)\cos(x_{m-j+1}\pi/2) \\
f_m(\vec{x}) &= x_m + 2m\cos(x_1\pi/2) \\
&\text{Where} \\
x_{i=1:m-1} &= \text{r\_nonsep}\left(\{y_{(i-1)k/(m-1)+1}, \ldots, y_{ik/(m-1)}\}, k/(m-1)\right) \\
x_m &= \text{r\_nonsep}\left(\{y_{k+1}, \ldots, y_n\}, l\right) \\
y_{i=1:k} &= \text{s\_decept}(y'_i, 0.35, 0.001, 0.05) \\
y_{i=k+1:n} &= \text{s\_multi}(y'_i, 30, 95, 0.35) \\
y'_{i=1:n-1} &= \text{b\_param}(z_i/(2i), \text{r\_sum}(\{z_{i+1}/(2(i-1), \ldots, z_n/2n)\}, \\
&\quad \{1, \ldots, 1\}), \frac{0.98}{49.98}, 0.02, 50) \\
y'_n &= \frac{z_n}{2n}
\end{aligned} \tag{A.16}$$

The previous problems are defined in terms of a set of transformation functions, which map parameters with domain $[0, 1]$ onto the range $[0, 2i]$. There are three types of transformation functions: bias, shift and reduction functions. For more details about of these functions, see [93].

A.3 MAF test suite

Benchmark problems play an important role in order in understanding the weaknesses and strengths of multi-objective evolutionary algorithms. As mentioned before,

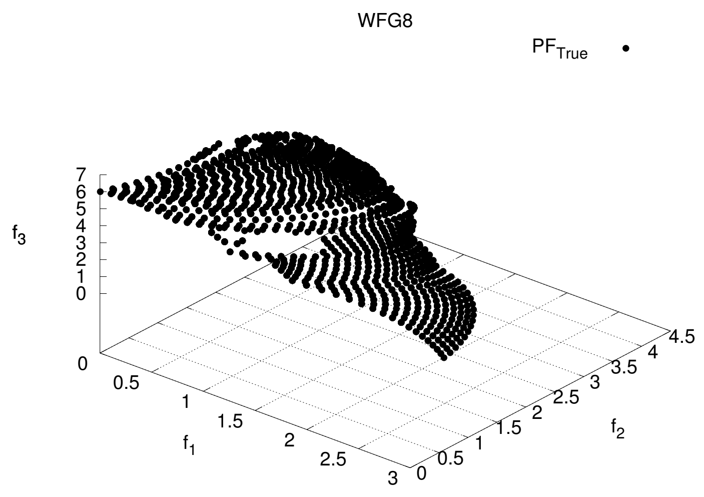


Figure A.15: Pareto optimal front for three objectives corresponding to WFG8

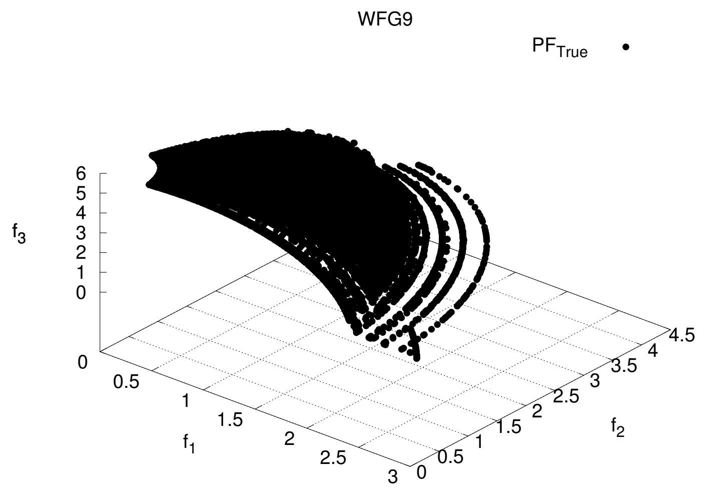


Figure A.16: Pareto optimal front for three objectives corresponding to WFG9

several scalable continuous test problems, such as those in the DTLZ test suite [92] and the WFG test suite [93], have been commonly used. However, all of these test suites only represent one or several aspects of real-world scenarios, but they typically have a "regular" Pareto front shapes. For instance, the Pareto front of most of the DTLZ and WFG test problems is similar to a simplex-like shape. Cheng et al. [101] proposed 15 benchmark problems. These problems have diverse properties which cover a good representation of various real-world scenarios, such as being multimodal, disconnected, degenerate, non-separable, and "irregular" Pareto front shapes. Here, we only present five of those problems (MAF1 up to MAF5).

A.3.1 MAF1 (modified inverted DTLZ1)

This test problem has an inverted linear Pareto front, it is separable and multifrontal. It is defined by the following expression:

$$\begin{aligned}
&\text{Given} \\
&\qquad \vec{x} = \{x_1, \dots, x_{m-1}, x_m, \dots, x_n\} \\
&\text{Minimize} \\
&\qquad f_1(\vec{x}) = (1 + g(\vec{y})) \prod_{i=1}^{m-1} x_i \\
&\qquad f_{j=2:m-1}(\vec{x}) = (1 + g(\vec{y}))(1 - x_{m-j+1}) \prod_{i=1}^{m-j} x_i \\
&\qquad f_m(\vec{x}) = (1 + g(\vec{y}))(x_1) \\
&\text{where} \\
&\qquad g(\vec{x}) = \sum_{i=1}^{k} (x_i - 0.5)^2 \\
&\text{Subject to} \quad 0 \leq x_i \leq 1, \text{ para } i = 1, 2, \dots, n.
\end{aligned} \tag{A.17}$$

The number of decision variables is given by $n = m + k - 1$, where k denotes the distance parameters and it should be set as $k = 5$. This test problem has an linear inverted Pareto Front shape, while the Pareto optimal set is relatively simple. In spite of this problem being linear, some multi-objective evolutionary algorithms are not able to solve it.

A.3.2 MAF2 (Badly-scaled DTLZ2)

This test problem is a modified version of DTLZ2, which increases the difficulty of its original version. MAF2 is separable and unimodal, and its Pareto front shape is badly-scaled concave. The authors of this test problem suggest to use $k = 10$ for the distance parameter. It is defined as follows:

$$\begin{aligned}
\text{Given}& \\
\vec{x} &= \{x_1, \ldots, x_{m-1}, x_m, \ldots, x_n\} \\
\text{Minimize}& \\
f_1(\vec{x}) &= (1 + g(\vec{y})) \prod_{i=1}^{m-1} \cos(\theta_i) \\
f_{j=2:m-1}(\vec{x}) &= (1 + g(\vec{y})) \left(\prod_{i=1}^{m-j} \cos(\theta_i) \right) \sin(\theta_i) \\
f_m(\vec{x}) &= (1 + g(\vec{y})) \sin(\theta_1) \\
\text{where}& \\
\theta_i &= \frac{\pi}{2} \left(\frac{x_i}{2} + \frac{1}{4} \right) \\
g_i(\vec{x}) &= \sum_{j=m+(i-1)\frac{n+m+1}{m}}^{m+i\frac{n+m+1}{m}-1} \left(\frac{x_j}{2} + \frac{1}{4} \right)^2 \\
\text{subject to} \quad & 0 \leq x_i \leq 1, \text{ for all } i = 1, 2, \ldots, n.
\end{aligned} \tag{A.18}$$

A.3.3 MAF3 (convex DTLZ3)

MAF3 is a modified version of DTLZ3. The resulting problem includes a factor p to transform the Pareto front shape. The definition is given as follows:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{x} = \{x_1, \ldots, x_{m-1}, x_m, \ldots, x_n\} \\
&\text{Minimize} \\
&\qquad f_1(\vec{x}) = \left((1+g(\vec{y}))\prod_{i=1}^{m-1}\cos\left(\frac{x_i\pi}{2}\right)\right)^4 \\
&\qquad f_{j=2:m-1}(\vec{x}) = \left((1+g(\vec{y}))\left(\prod_{i=1}^{m-j}\cos\left(\frac{x_i\pi}{2}\right)\right)\sin\left(\frac{x_{m-j+1}\pi}{2}\right)\right)^4 \\
&\qquad f_m(\vec{x}) = \left((1+g(\vec{y}))\sin\left(\frac{x_1\pi}{2}\right)\right)^2 \\
&\text{where} \\
&\qquad y_{i=1:k} = \{x_m, x_{m+1}, \ldots, x_n\} \\
&\qquad g(\vec{y}) = 100\left[k + \sum_{i=1}^{k}(y_i - 0.5)^2 - \cos(20\pi(y_i - 05))\right] \\
&\text{subject to} \\
&\qquad 0 \le x_i \le 1,\ \text{para } i = 1, 2, \ldots, n.
\end{aligned}
\tag{A.19}
$$

The above function g introduces $(3^k - 1)$ local Pareto optimal fronts, and one global Pareto optimal front. All the local Pareto fronts are parallel to the global Pareto optimal front. This makes optimizers to converge towards a local Pareto front.

A.3.4 MAF4 (inverted badly scaled DTLZ3)

This problem is a modified version of DTLZ3, which includes a factor a in order to scale the Pareto front shape. The problem is described as follows:

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

The authors propose to use $a = 2$. The landscape of this test problem is highly multifrontal, containing $(3^k - 1)$ local Pareto optimal fronts.

A.3.5 MAF5 (Convex badly-scaled DTLZ4)

MAF5 adopts the same structure of DTLZ4, which is separable and unimodal. The Pareto front shape is convex and it is described as follows:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{x} = \{x_1, \ldots, x_{m-1}, x_m, \ldots, x_n\} \\
&\text{Minimize} \\
f_1(\vec{x}) &= a^m \left((1 + g(\vec{y})) \left(1 - \prod_{i=1}^{m-1} \cos\left(\frac{\pi x_i^{\alpha}}{2} \right) \right) \right)^4 \\
f_{j=2:m-1}(\vec{x}) &= a^{m-1} \left((1 + g(\vec{y})) \left(\prod_{i=1}^{m-j} \cos\left(\frac{\pi x_i^{\alpha}}{2} \right) \right) \sin\left(\frac{\pi x_{m-j+1}^{\alpha}}{2} \right) \right)^4 \\
f_m(\vec{x}) &= a \left((1 + g(\vec{y})) \left(\sin\left(\frac{\pi x_1^{\alpha}}{2} \right) \right) \right)^4 \\
&\text{where} \\
y_{i=1:k} &= \{x_m, x_{m+1}, \ldots, x_n\} \\
g(\vec{y}) &= \sum_{i=1}^{k} (y_i - 0.5)^2 \\
&\text{subject to} \\
&\qquad 0 \le x_i \le 1, \text{ para } i = 1, 2, \ldots, n.
\end{aligned}
\tag{A.21}
$$

A.4 Viennet Test Suite

A.4.1 Viennet1

Viennet1 is a mutil-objective problem with three objectives [28]. The Pareto optimal set is connected and symmetric, but the Pareto optimal front is a degenerated curved surface (see Figure A.17). It is defined as follows:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{x} = \{x_1, x_2\} \\
&\text{Minimize} \\
&\qquad f_1(\vec{x}) = x_1^2 + (x_2 - 1)^2 \\
&\qquad f_2(\vec{x}) = x^2 + (y + 1)^2 + 1 \\
&\qquad f_3(\vec{x}) = (x_1 - 1)^2 + x_2^2 + 2 \\
&\text{subject to} \\
&\qquad -2 \leq x_i \leq 2, \forall i = 1, 2
\end{aligned}
\tag{A.22}
$$

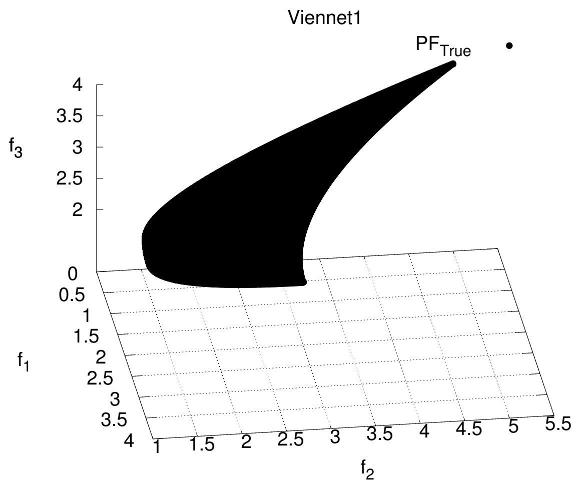


Figure A.17: Graphical representation of Viennet1

A.4.2 Viennet2

Viennet2 has a connected Pareto optimal front shape, but it is degenerated (see Figure A.19) [28]. It is defined as follows:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{x} = \{x_1, x_2\} \\
&\text{Minimize} \\
&f_1(\vec{x}) = \frac{(x_1-1)^2}{2} + \frac{(x_2+1)^2}{13} + 3 \\
&f_2(\vec{x}) = \frac{(x_1+x_2-3)^2}{36} + \frac{(-x_1+x_2+2)^2}{8} - 17 \\
&f_3(\vec{x}) = \frac{(x_1+2x_2-1)^2}{175} + \frac{(-2x_2+x_1)^2}{17} - 13 \\
&\text{subject to} \\
&\qquad -4 \le x_i \le 4, \forall i = 1, 2
\end{aligned}
\tag{A.23}
$$

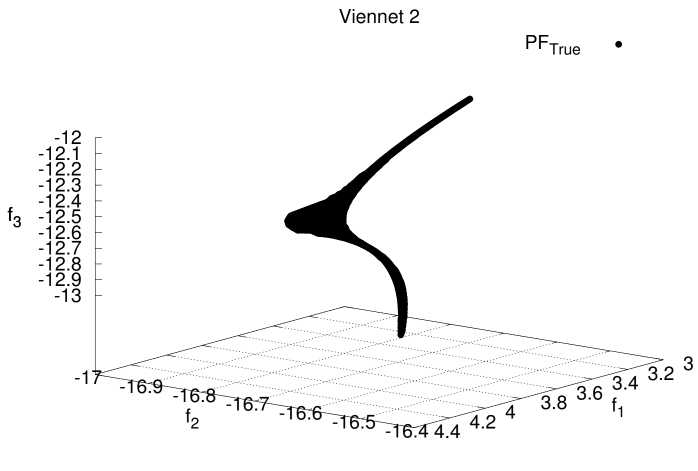


Figure A.18: Graphical representation of Viennet2

A.4.3 Viennet3

The Pareto optimal set is disconnected and un-symmetric, and its Pareto optimal front shape of Viennet3 is degenerated (see Figure A.19) [28]. It is defined as follows:

$$
\begin{aligned}
&\text{Given} \\
&\qquad \vec{x} = \{x_1, x_2\} \\
&\text{Minimize} \\
&\qquad f_1(\vec{x}) = \frac{1}{2}(x_1^2 + x_2^2) + \sin(x_1^2 + x_2^2) \\
&\qquad f_2(\vec{x}) = \frac{(3x_1 2x_2 + 4)^2}{8} + \frac{(x_1 + x_2 + 1)^2}{27} + 15 \\
&\qquad f_3(\vec{x}) = \frac{1}{(x_1^2 + x_2^2 + 1)} - 1.1e^{(-x_1^2 - x_2^2)} \\
&\text{subject to} \\
&\qquad -3 \leq x_i \leq 3, \forall i = 1, 2
\end{aligned}
\tag{A.24}
$$

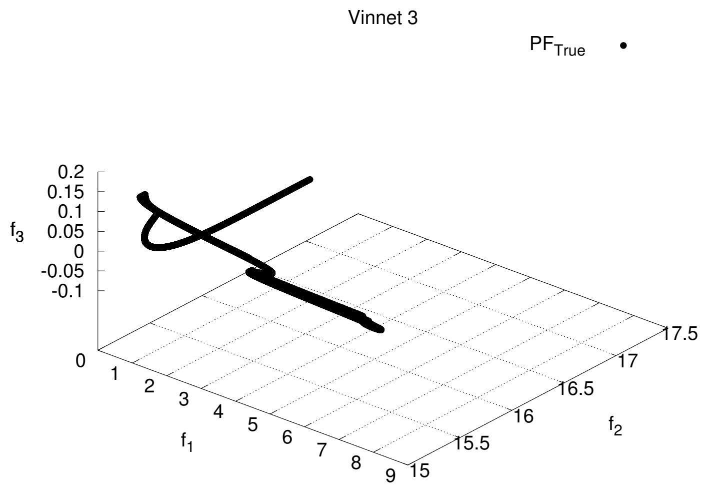


Figure A.19: Graphical representation of Viennet3

Appendix B

Experimental Results of the Analysis of Variance (ANOVA)

In this Appendix, we show the box plots and the interval plots, which were obtained by the Tukey test. This appendix B includes all graphs for each selected MOP.

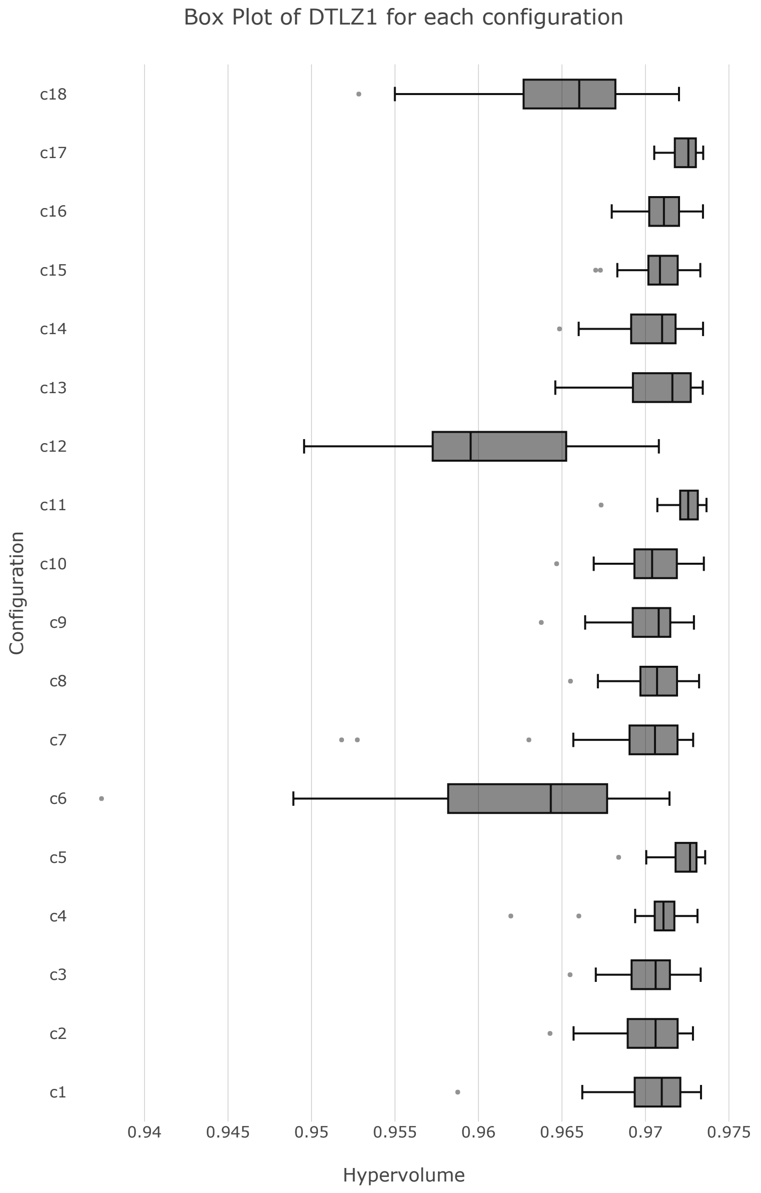


Figure B.1: Descriptive statistics for MOMA-II solving DTLZ1 with each combination for each pair of parameters.

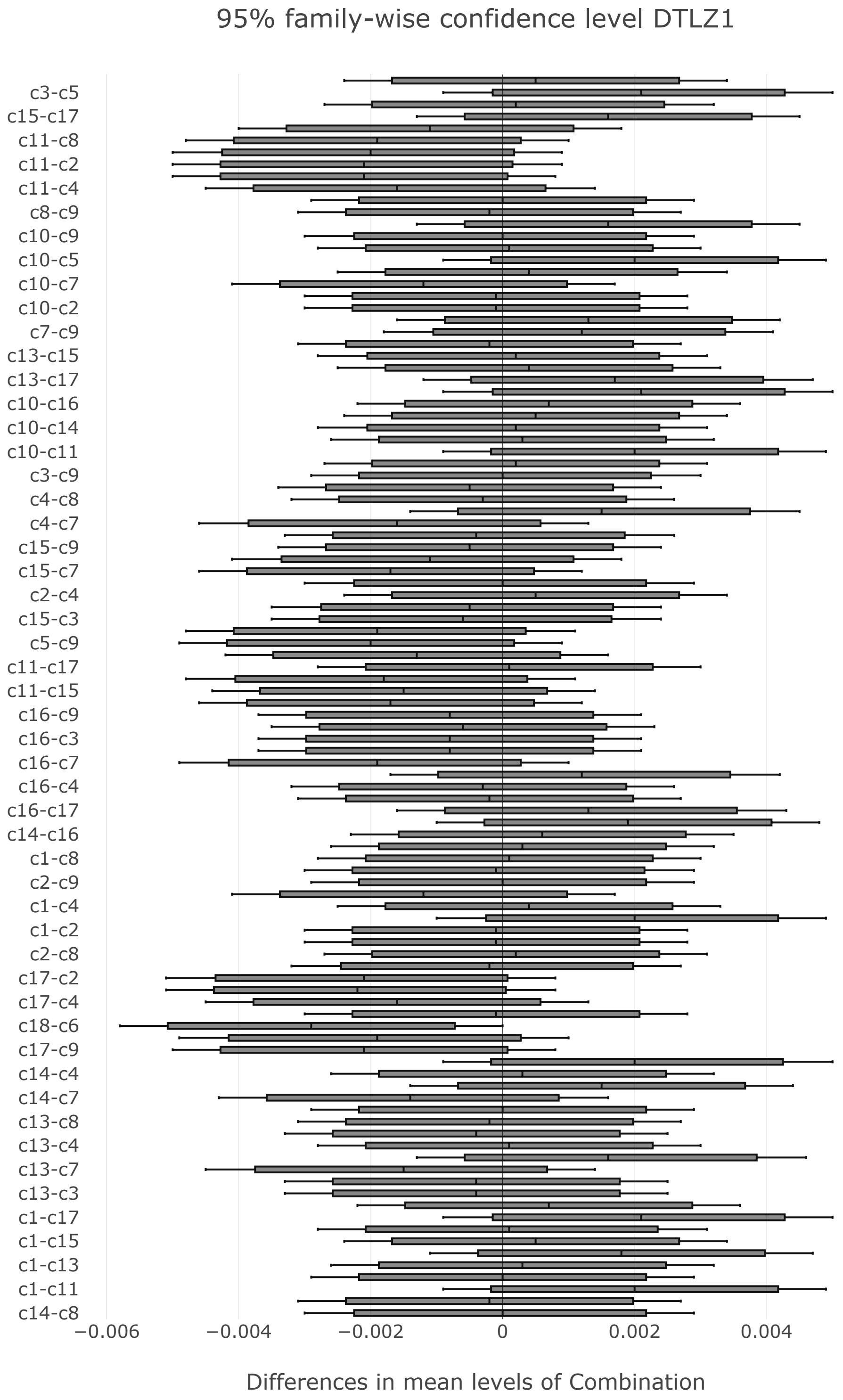


Figure B.2: Graphical representation of each confidence interval obtained by the Tukey test for DTLZ1, where each confidence interval intersects with zero.

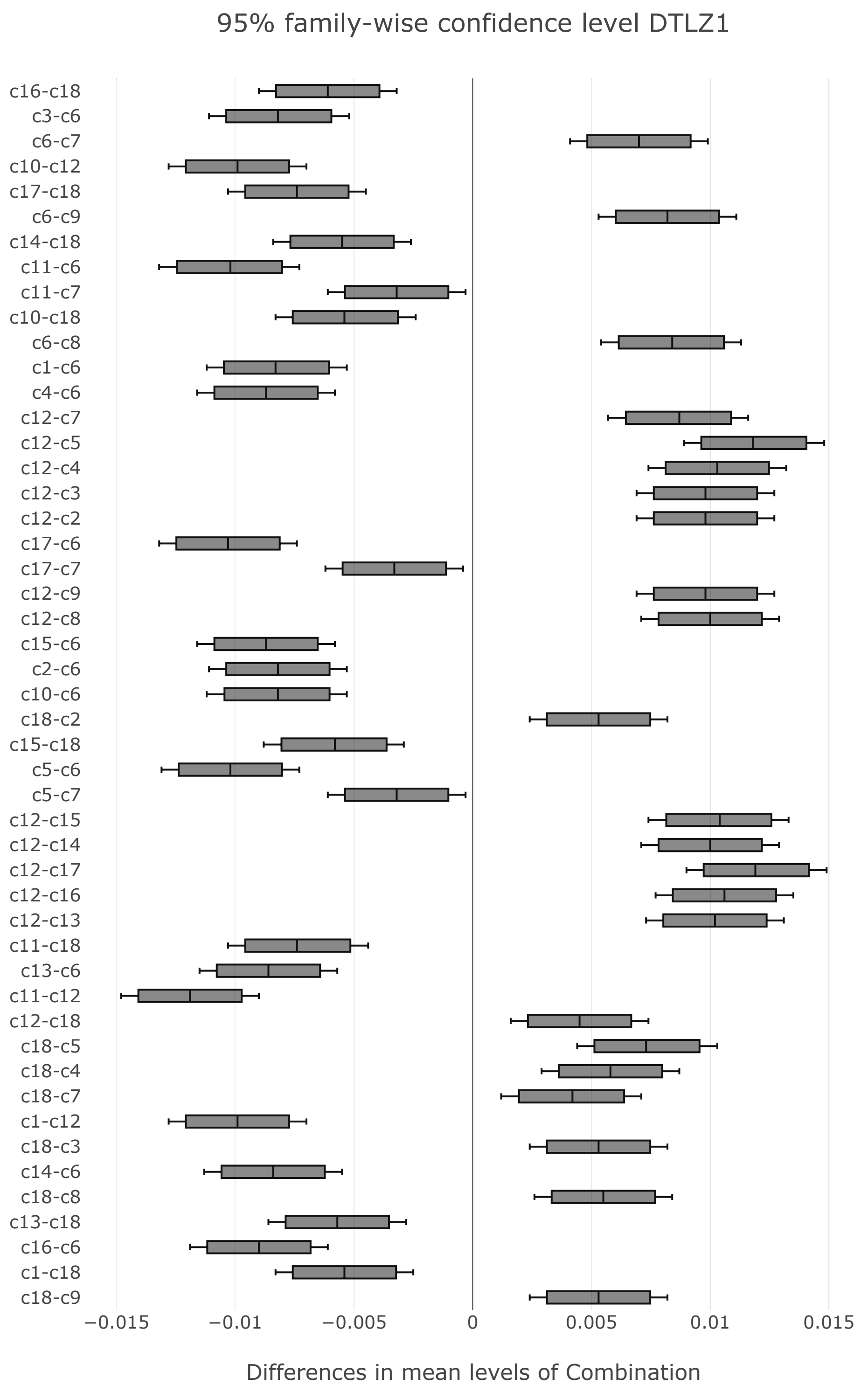


Figure B.3: Graphical representation of each confidence interval obtained by the Tukey test for DTLZ1, where each confidence interval does not intersect with zero.

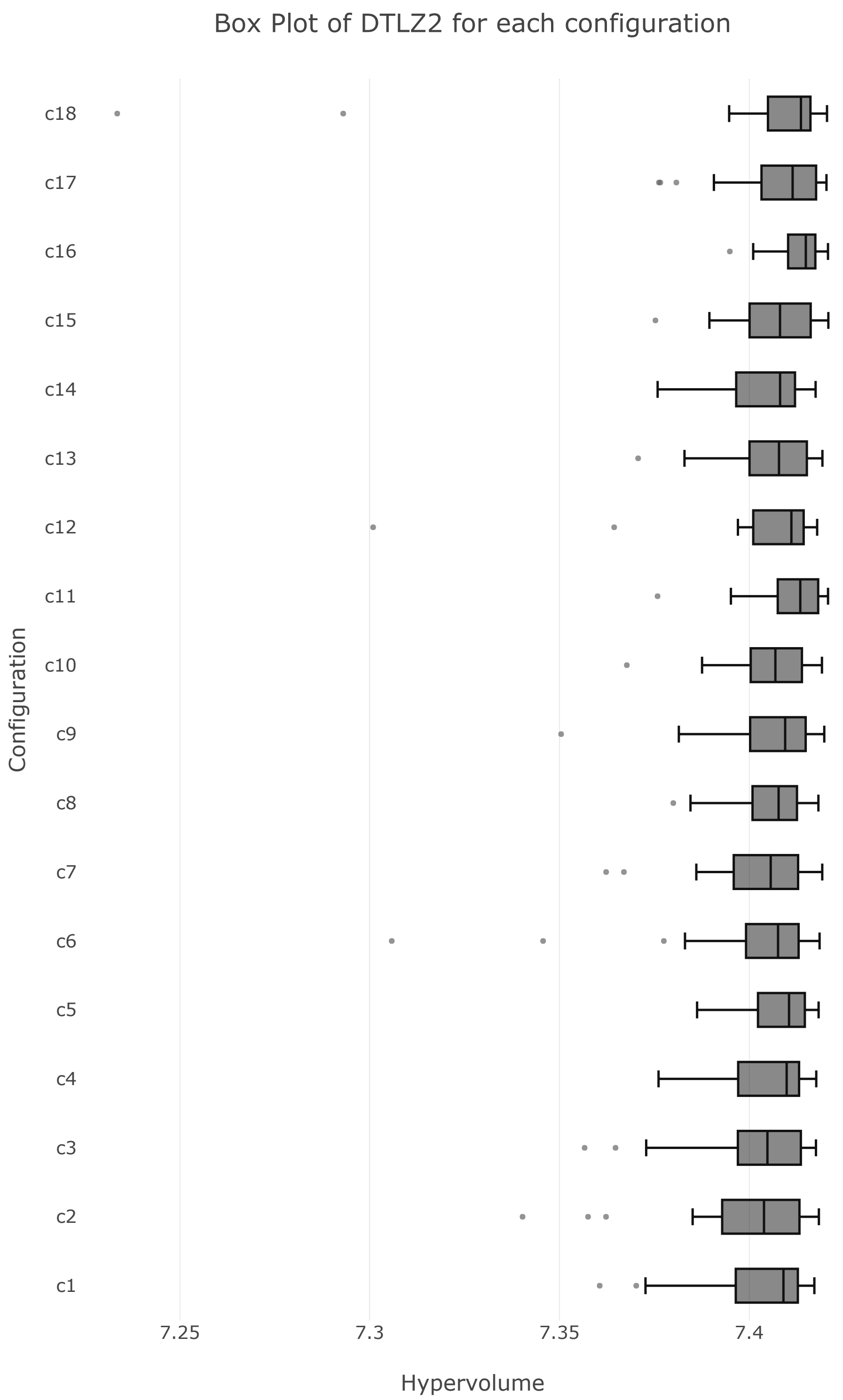


Figure B.4: Descriptive statistics for MOMA-II solving DTLZ2 with each combination for each pair of parameters.

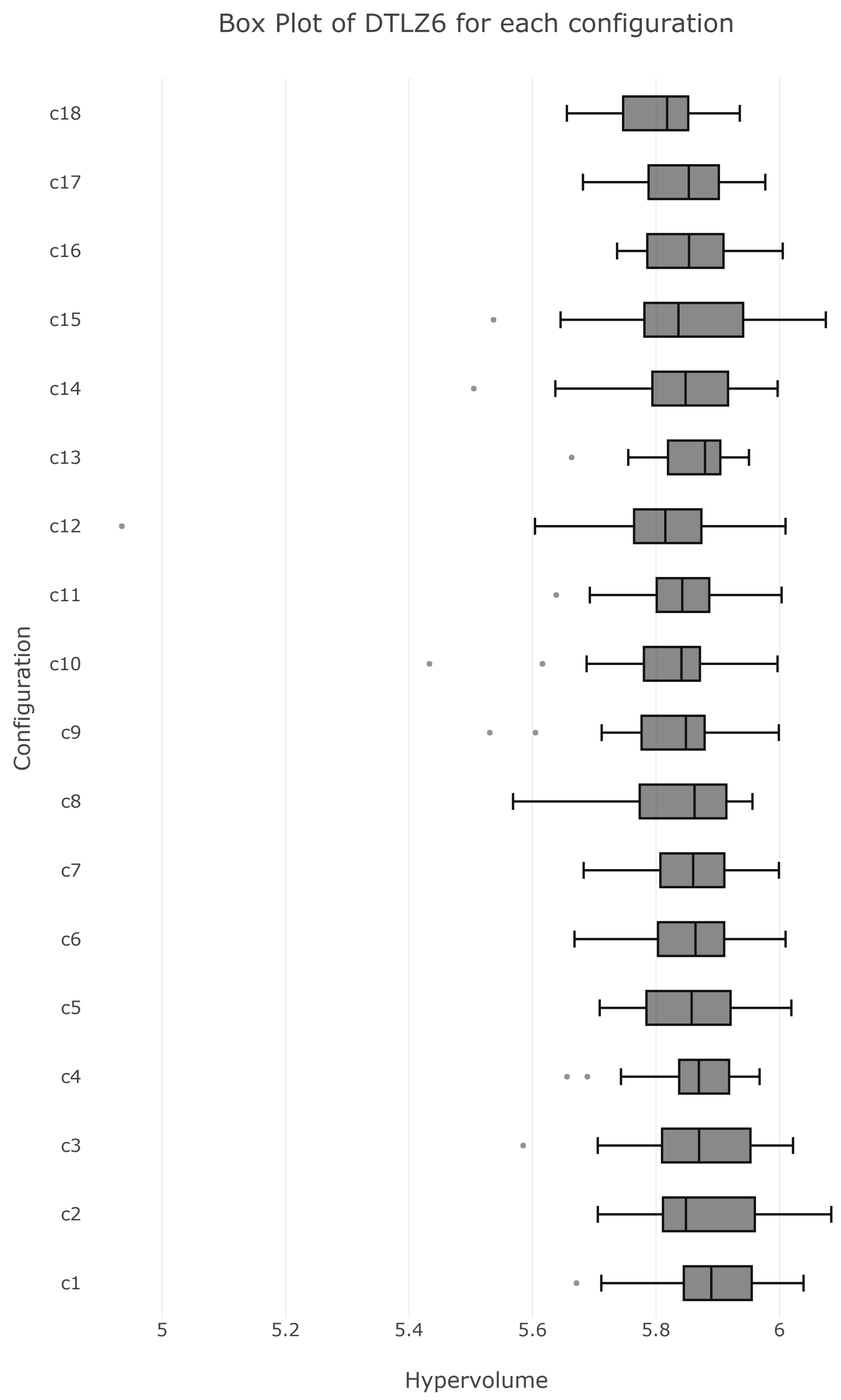


Figure B.5: Descriptive statistics for MOMA-II solving DTLZ6 with each combination for each pair of parameters.

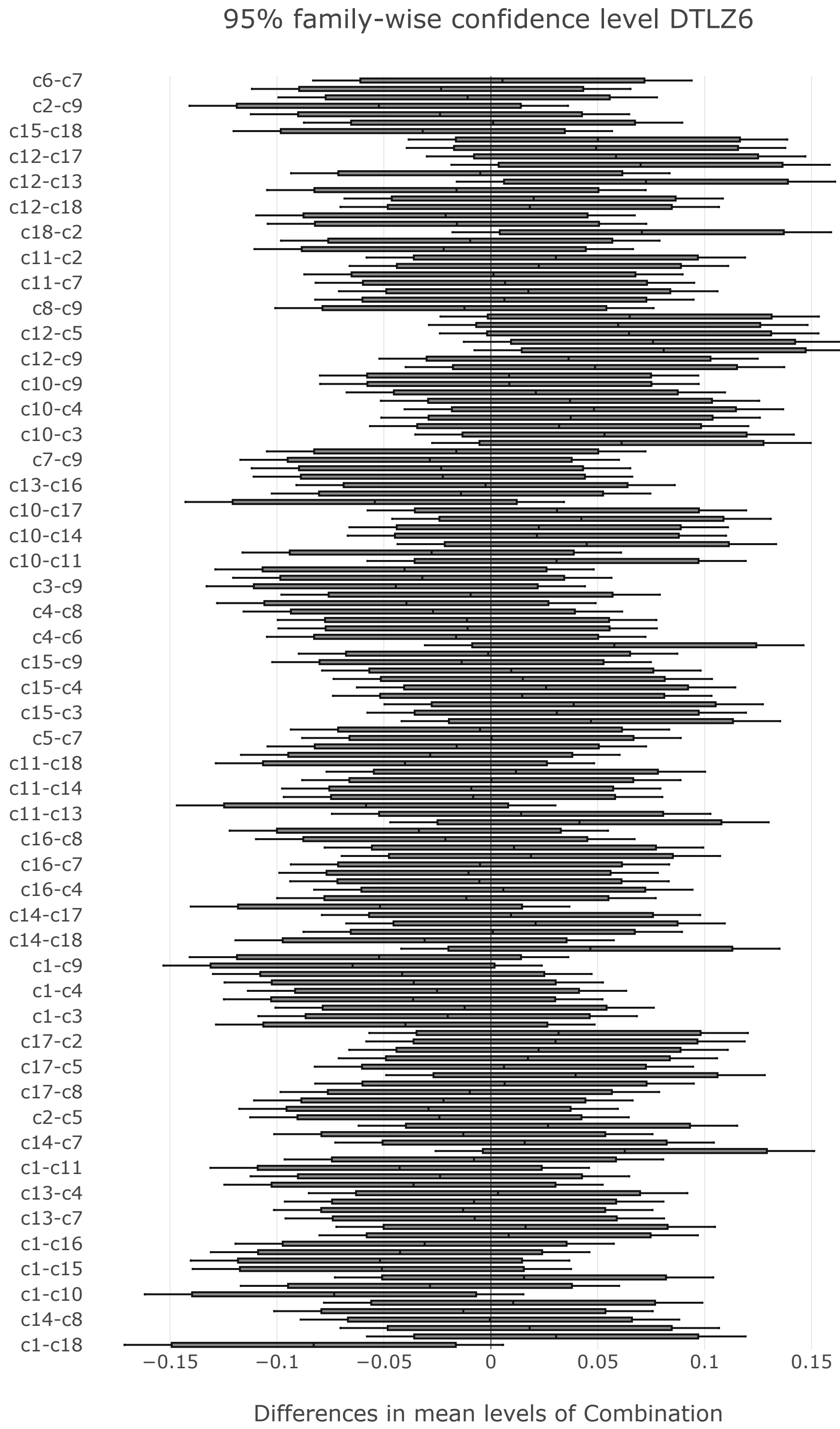


Figure B.6: Graphical representation of each confidence interval obtained by the Tukey test for DTLZ6, where each confidence interval intersects with zero.

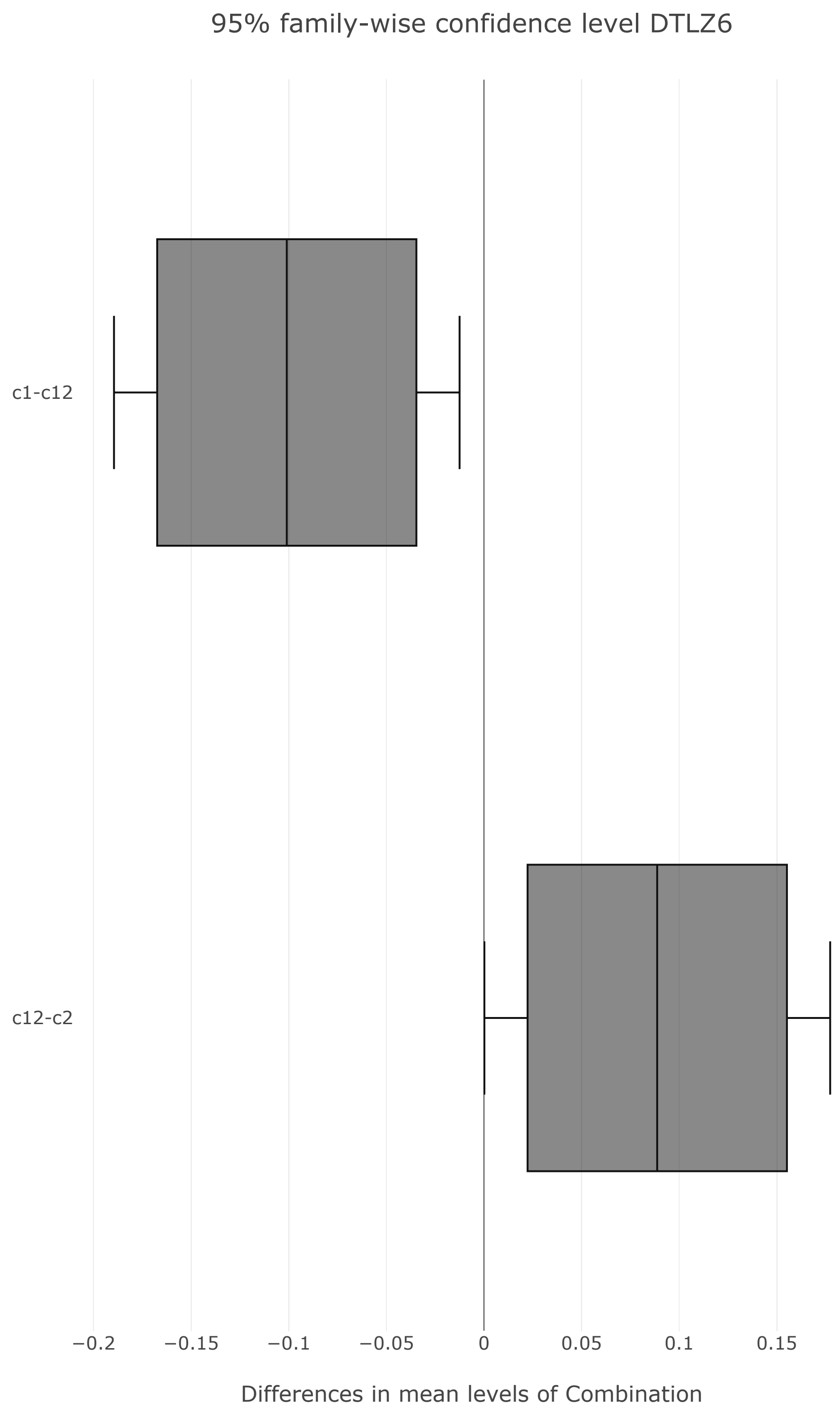


Figure B.7: Graphical representation of each confidence interval obtained by the Tukey test for DTLZ6, where each confidence interval does not intersect with zero.

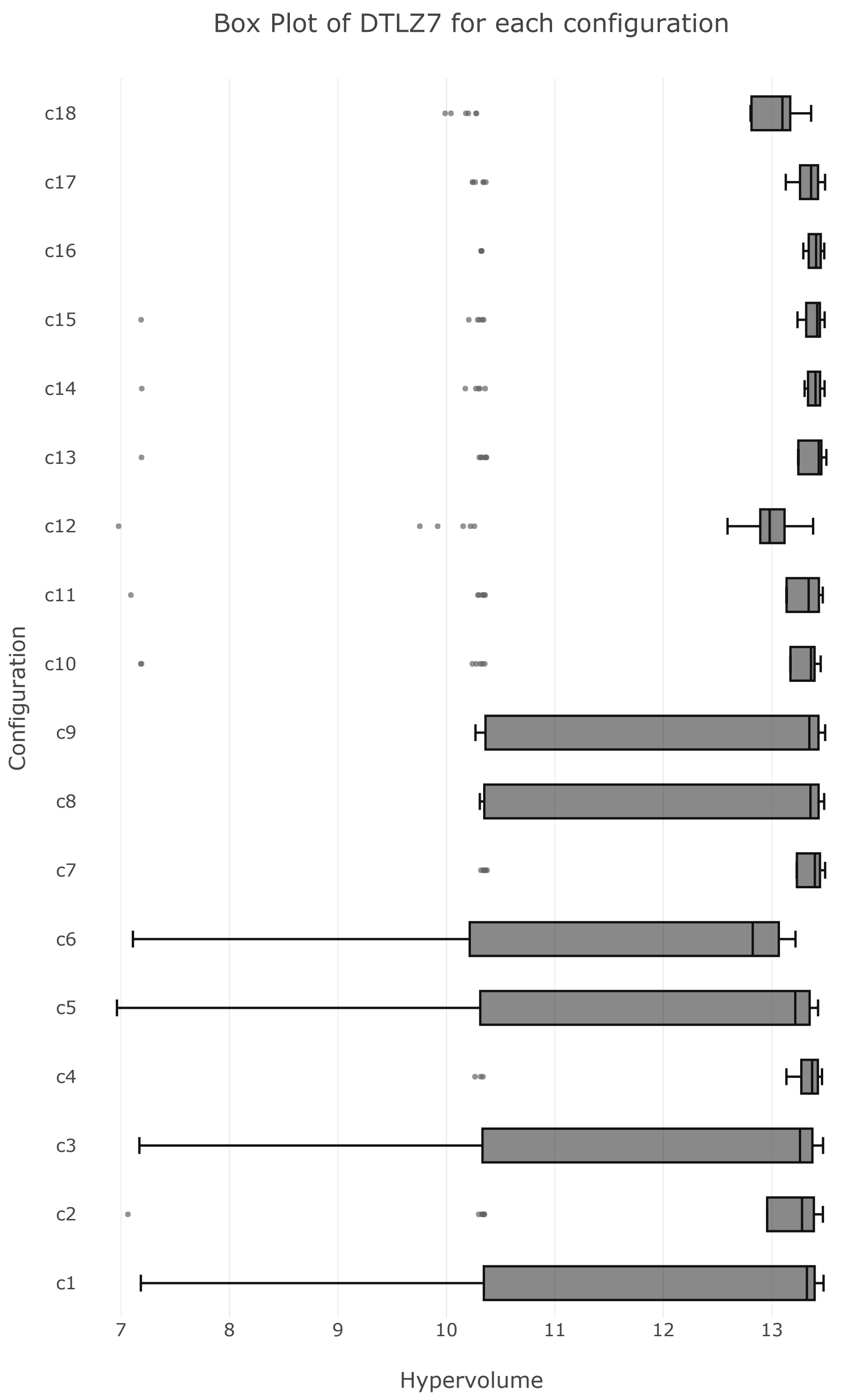


Figure B.8: Descriptive statistics for MOMA-II solving DTLZ7 with each combination for each pair of parameters.

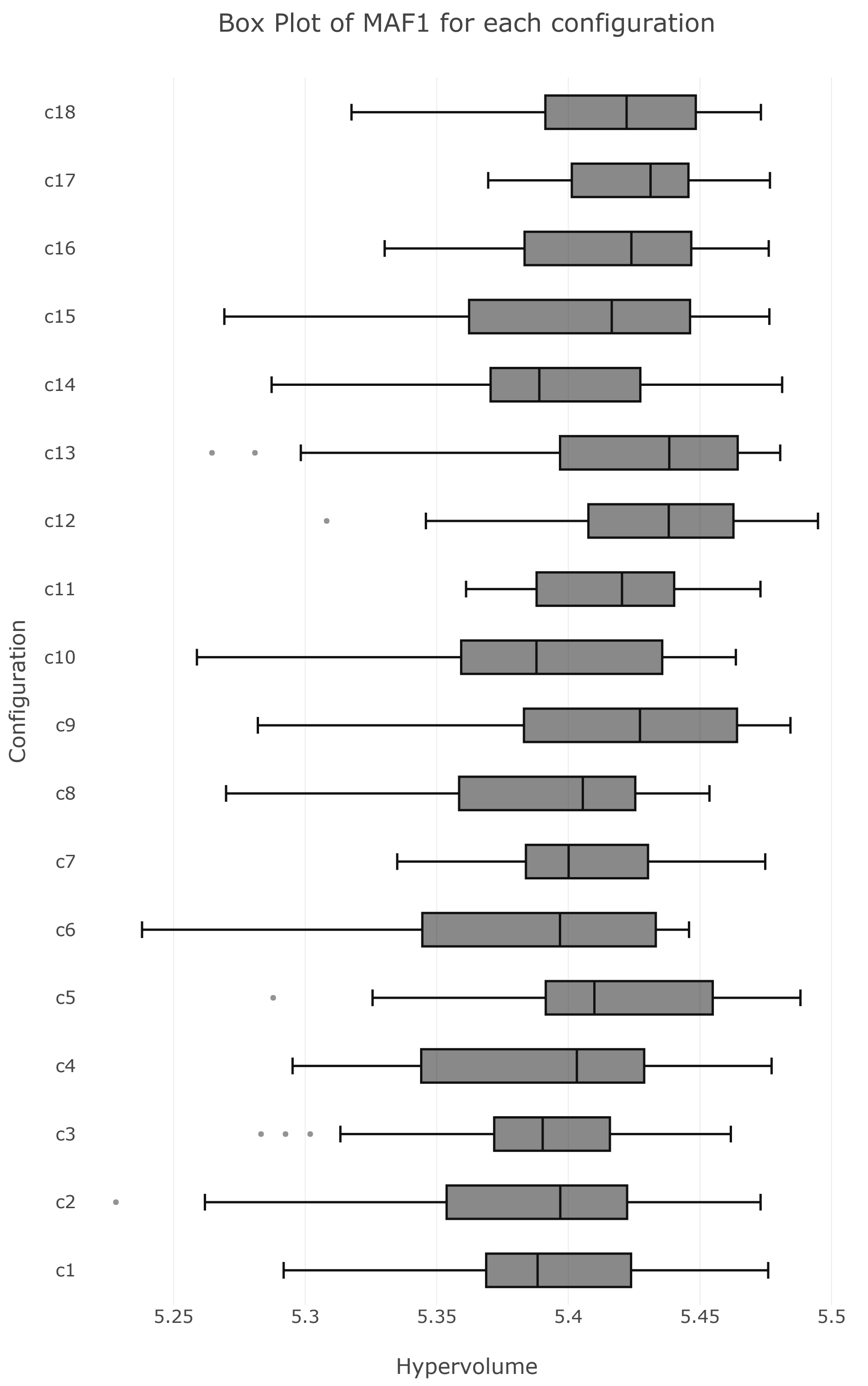


Figure B.9: Descriptive statistics for MOMA-II solving MAF1 with each combination for each pair of parameters.

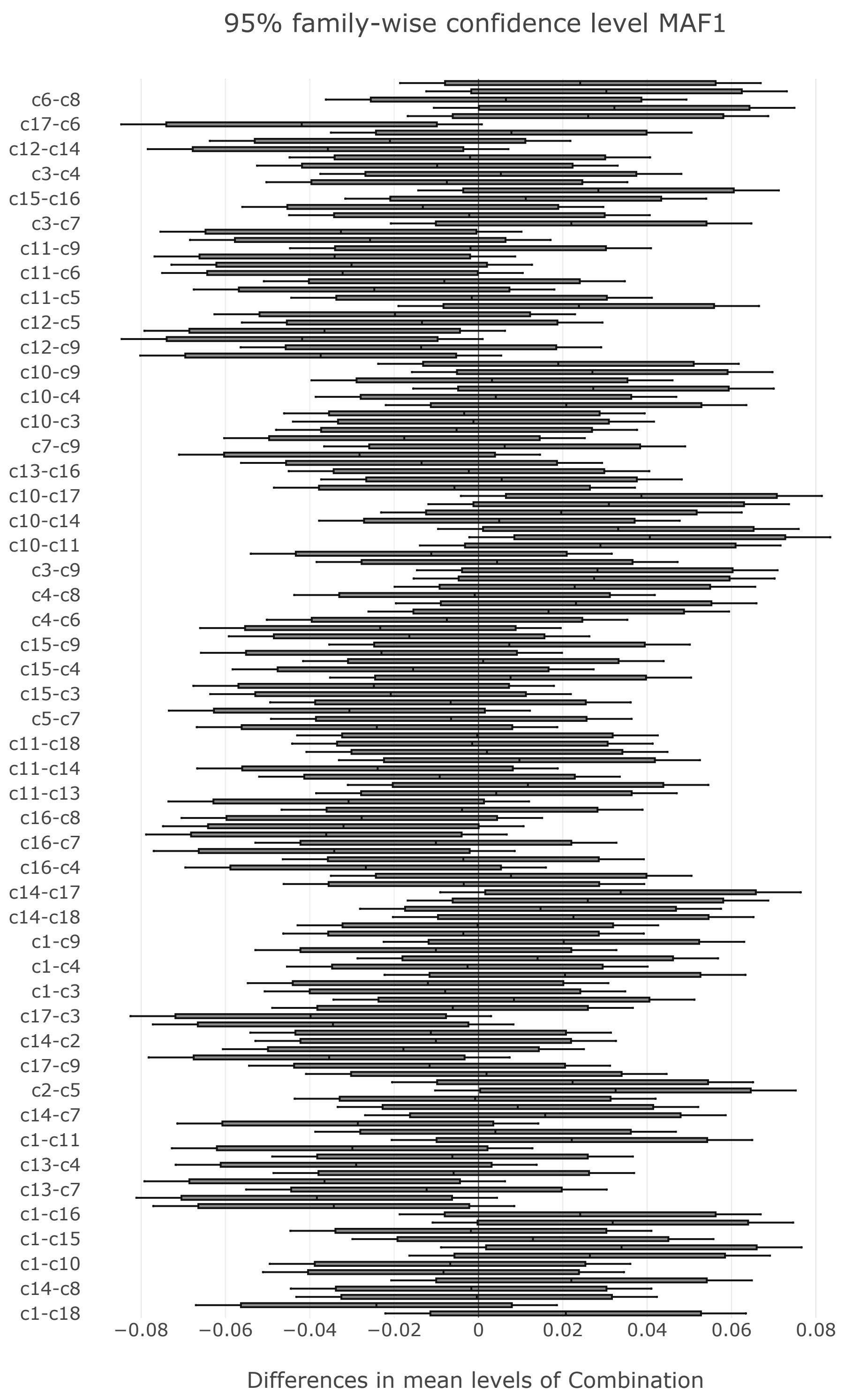


Figure B.10: Graphical representation of each confidence interval obtained by the Tukey test for MAF1, where each confidence interval intersects with zero.

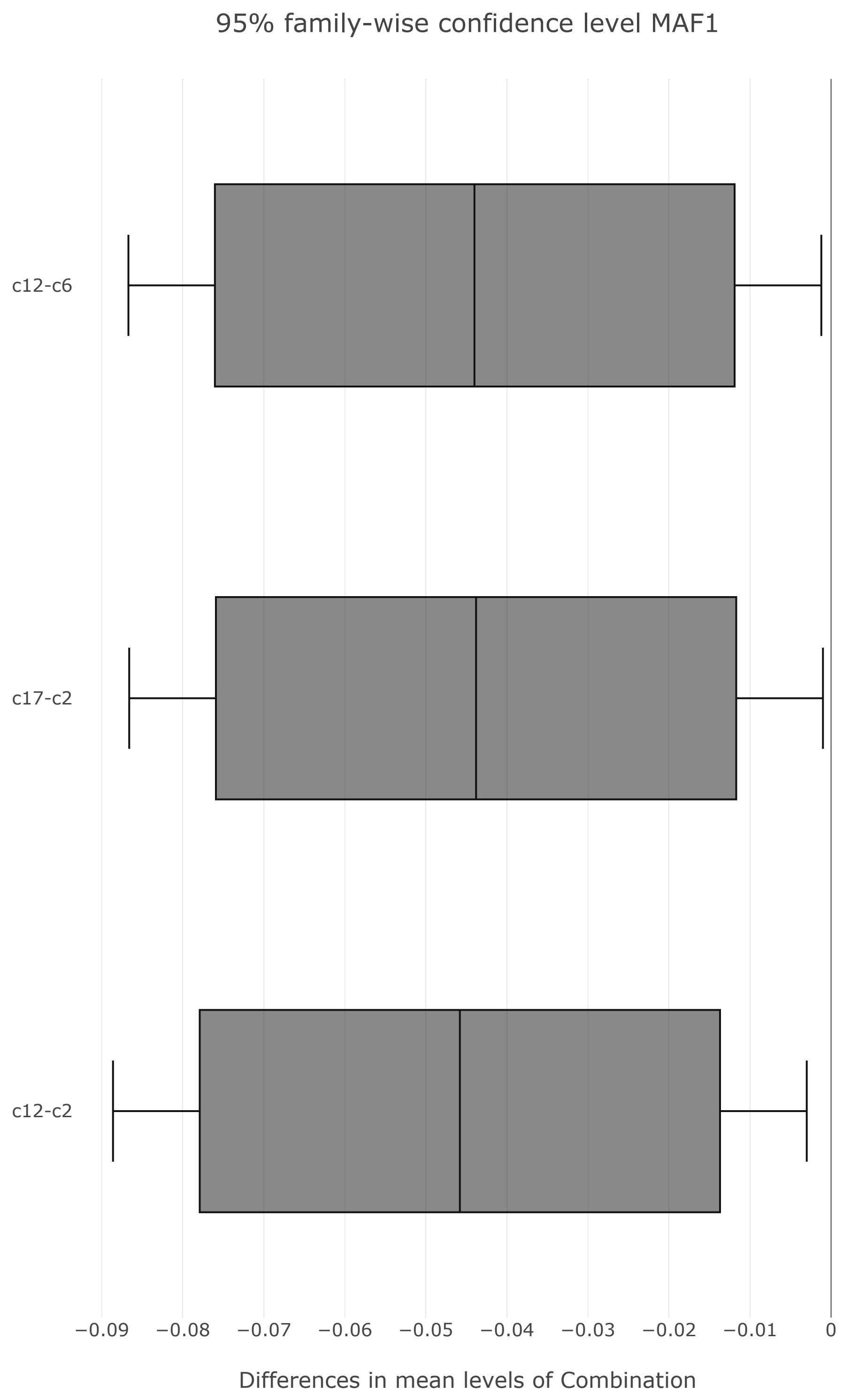


Figure B.11: Graphical representation of each confidence interval obtained by the Tukey test for MAF1, where each confidence interval does not intersect with zero.

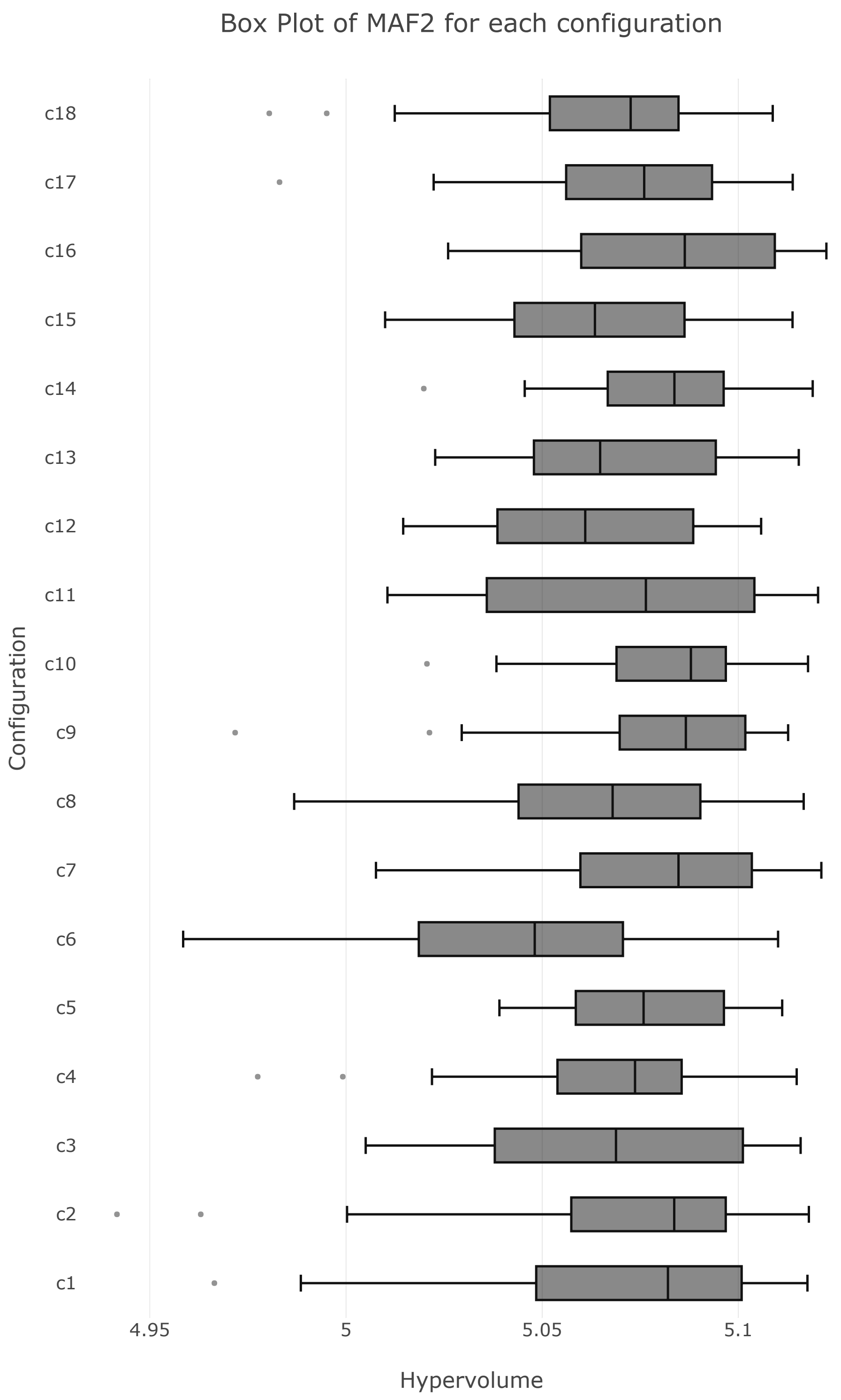


Figure B.12: Descriptive statistics for MOMA-II solving MAF2 with each combination for each pair of parameters.

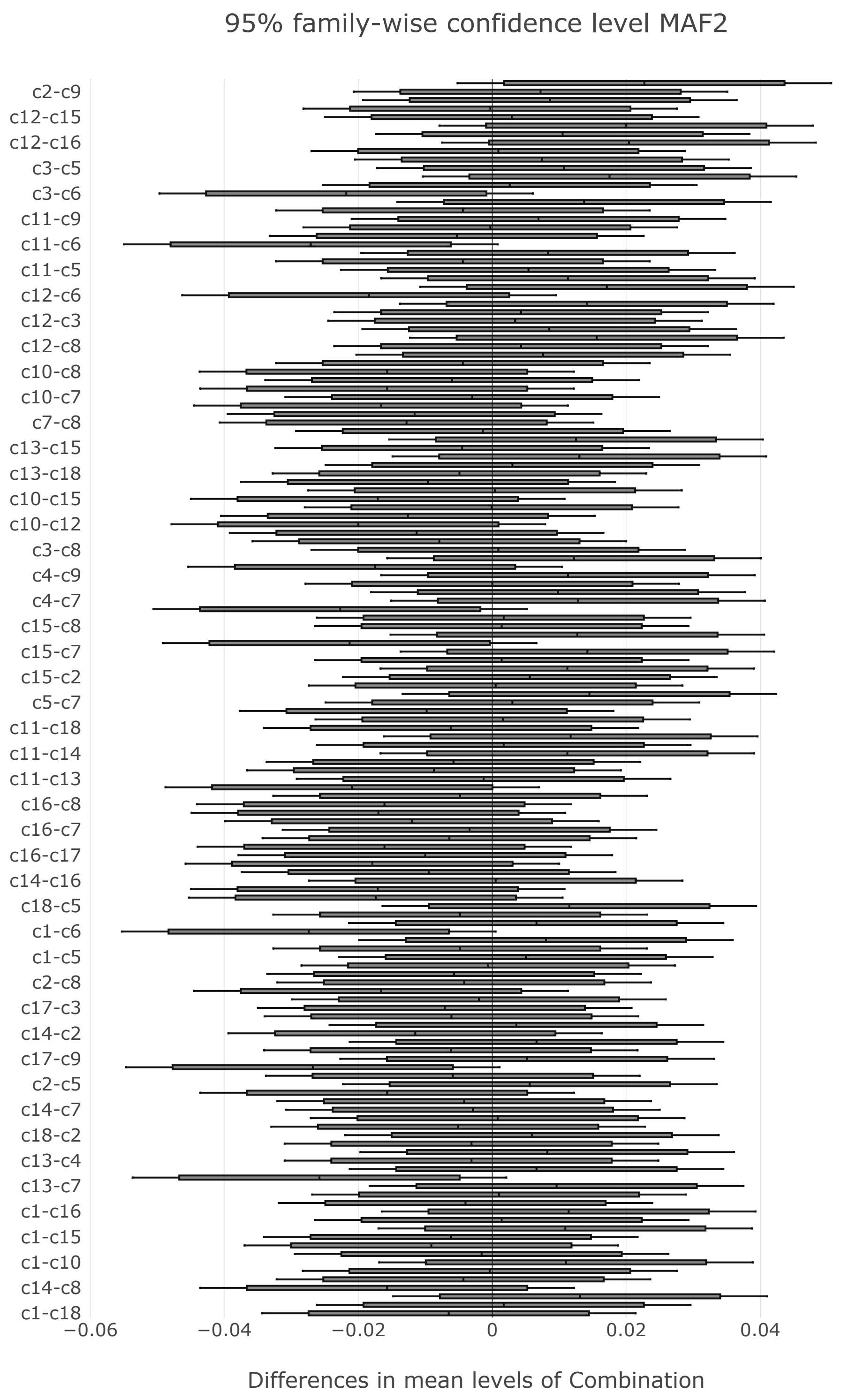


Figure B.13: Graphical representation of each confidence interval obtained by the Tukey test for MAF2, where each confidence interval intersects with zero.

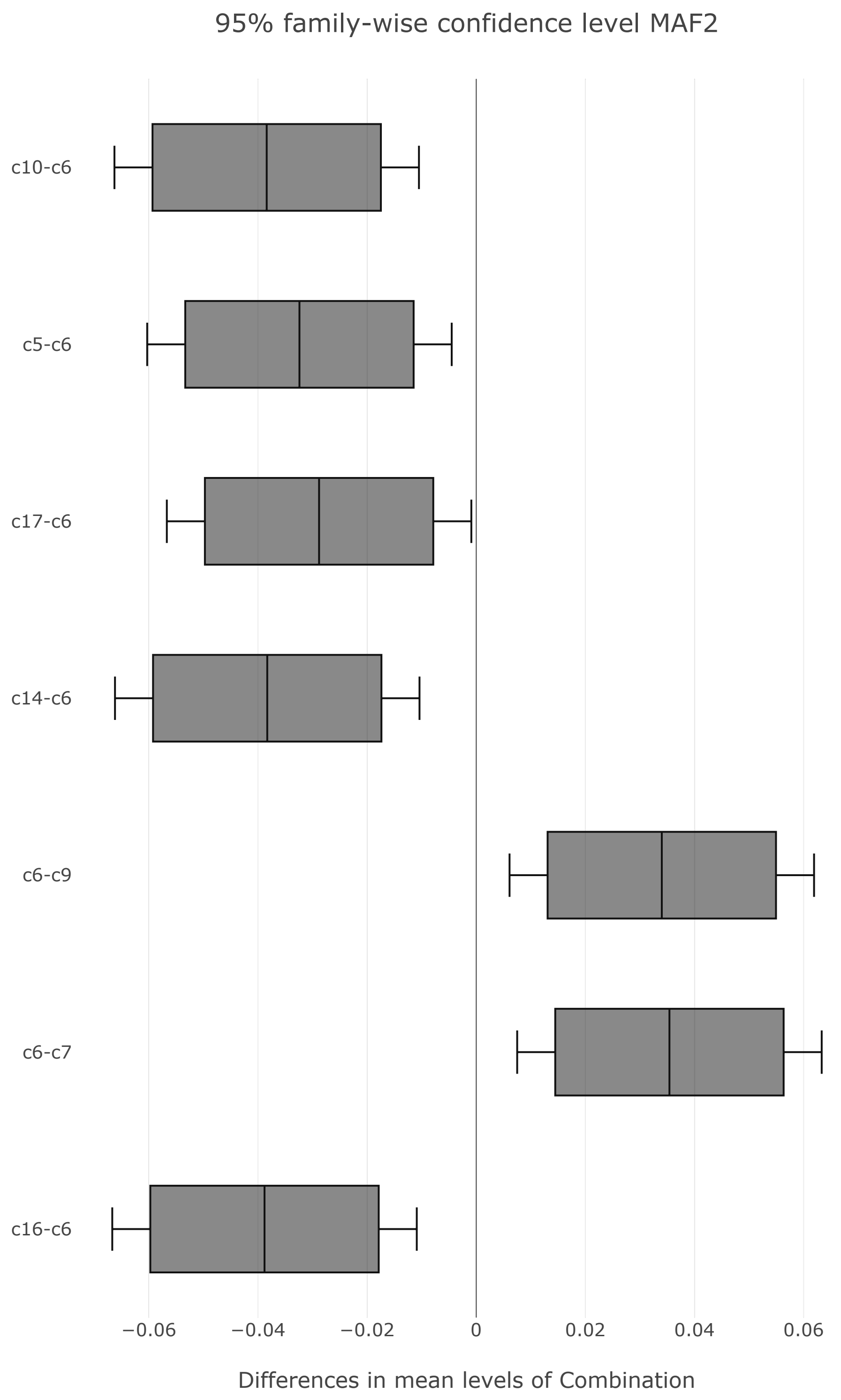


Figure B.14: Graphical representation of each confidence interval obtained by the Tukey test for MAF2, where each confidence interval does not intersect with zero.

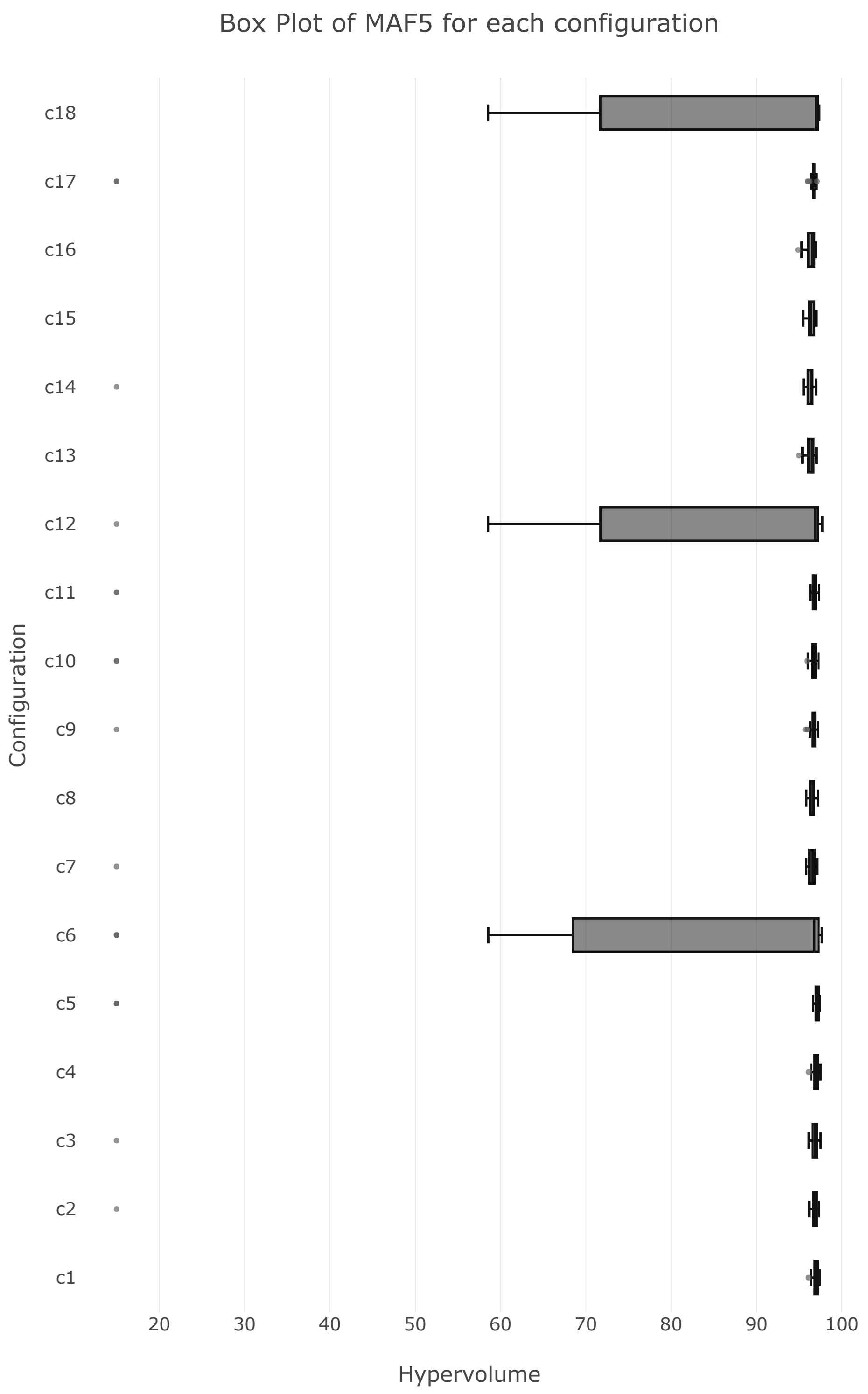


Figure B.15: Descriptive statistics for MOMA-II solving MAF5 with each combination for each pair of parameters.

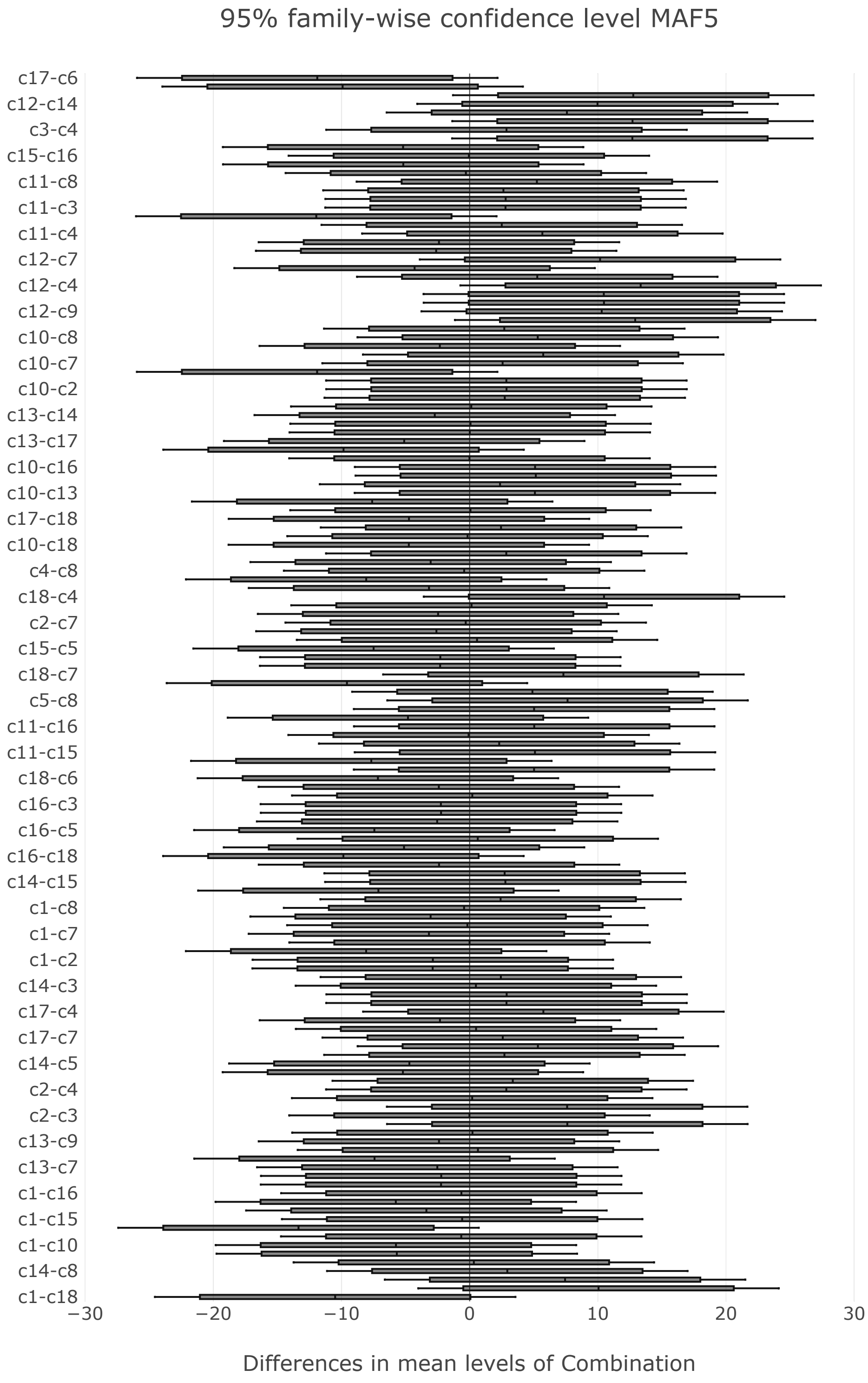


Figure B.16: Graphical representation of each confidence interval obtained by the Tukey test for MAF5, where each confidence interval intersects with zero.

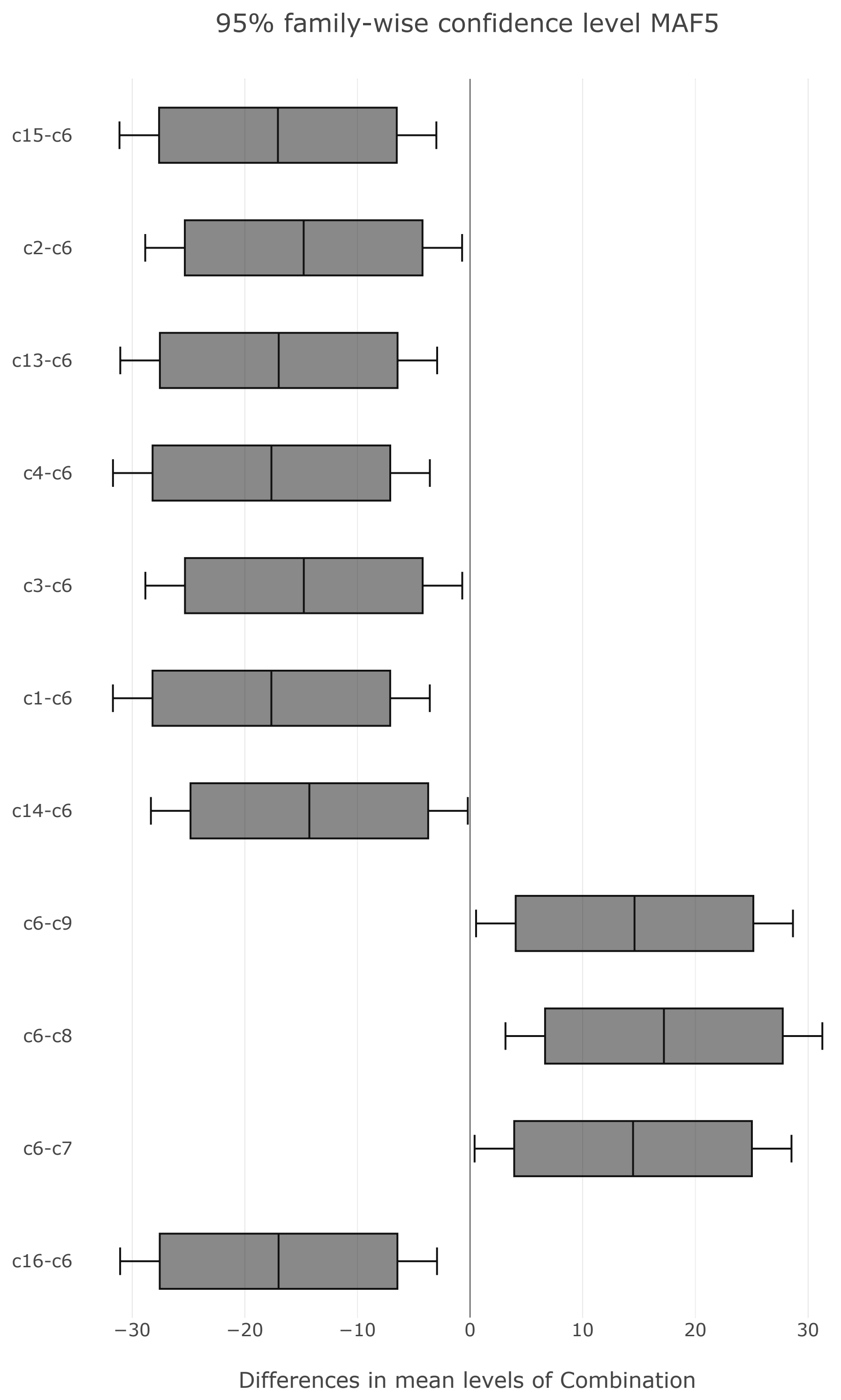


Figure B.17: Graphical representation of each confidence interval obtained by the Tukey test for MAF5, where each confidence interval does not intersect with zero.

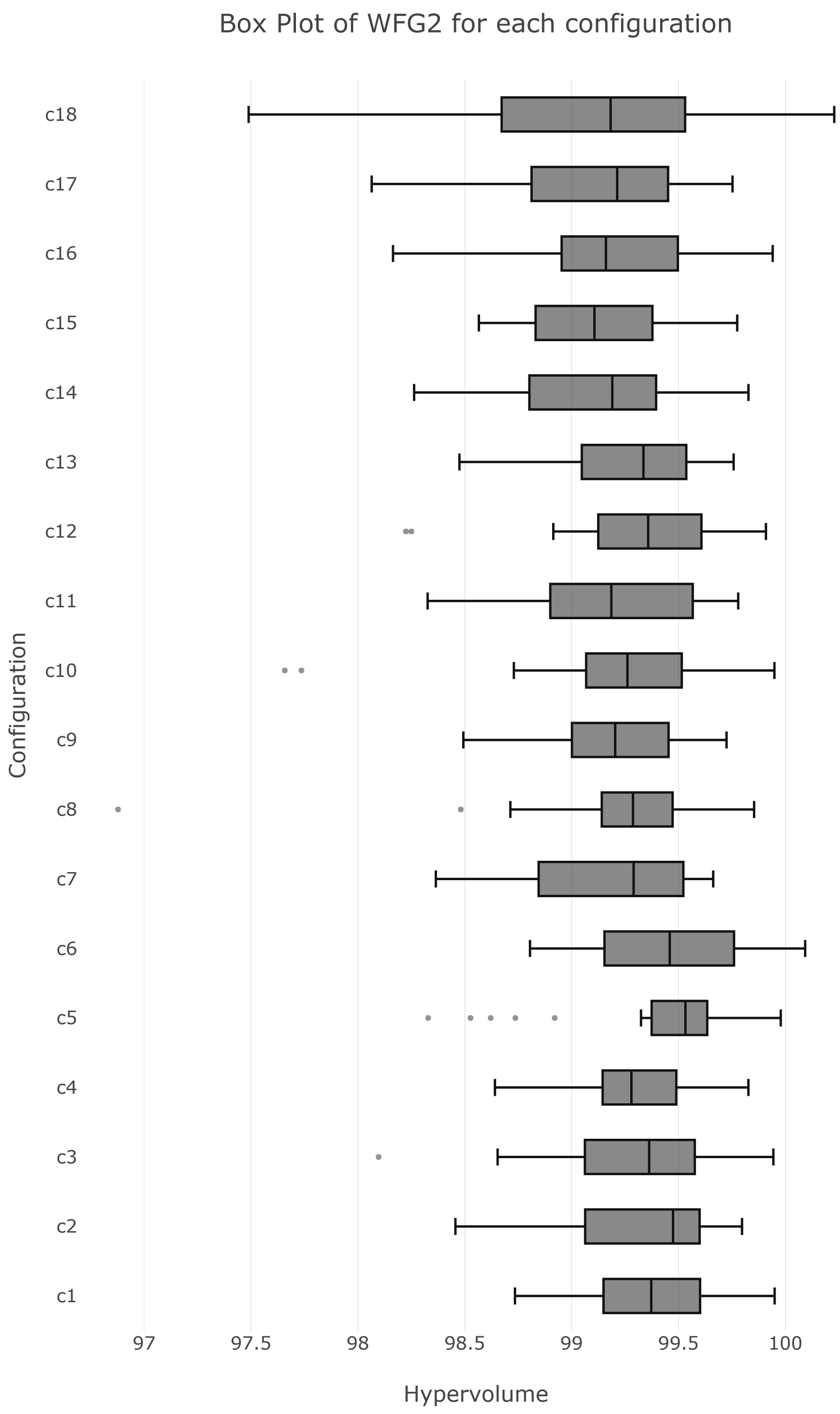


Figure B.18: Descriptive statistics for MOMA-II solving WFG2 with each combination for each pair of parameters.

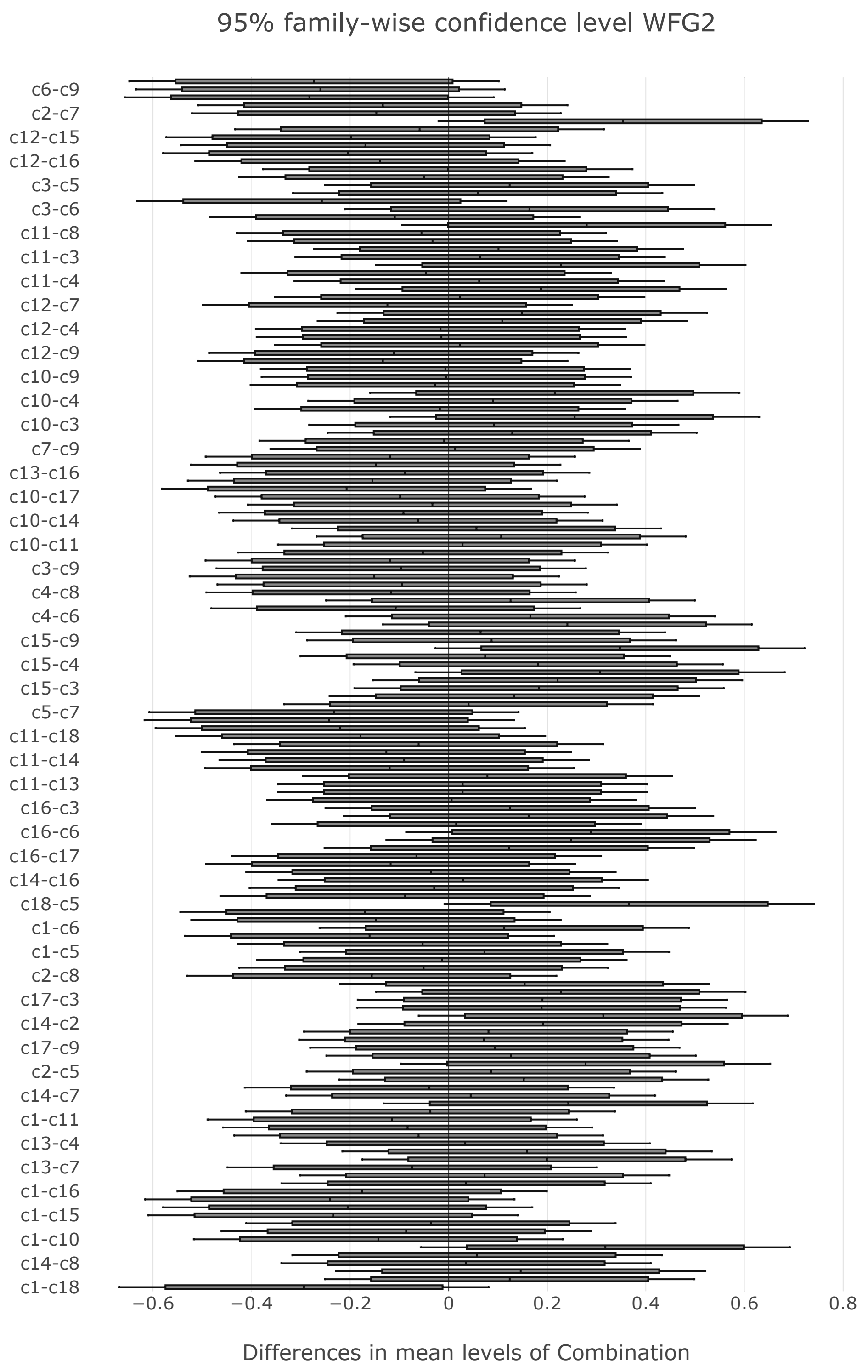


Figure B.19: Graphical representation of each confidence interval obtained by the Tukey test for WFG2, where each confidence interval intersects with zero.

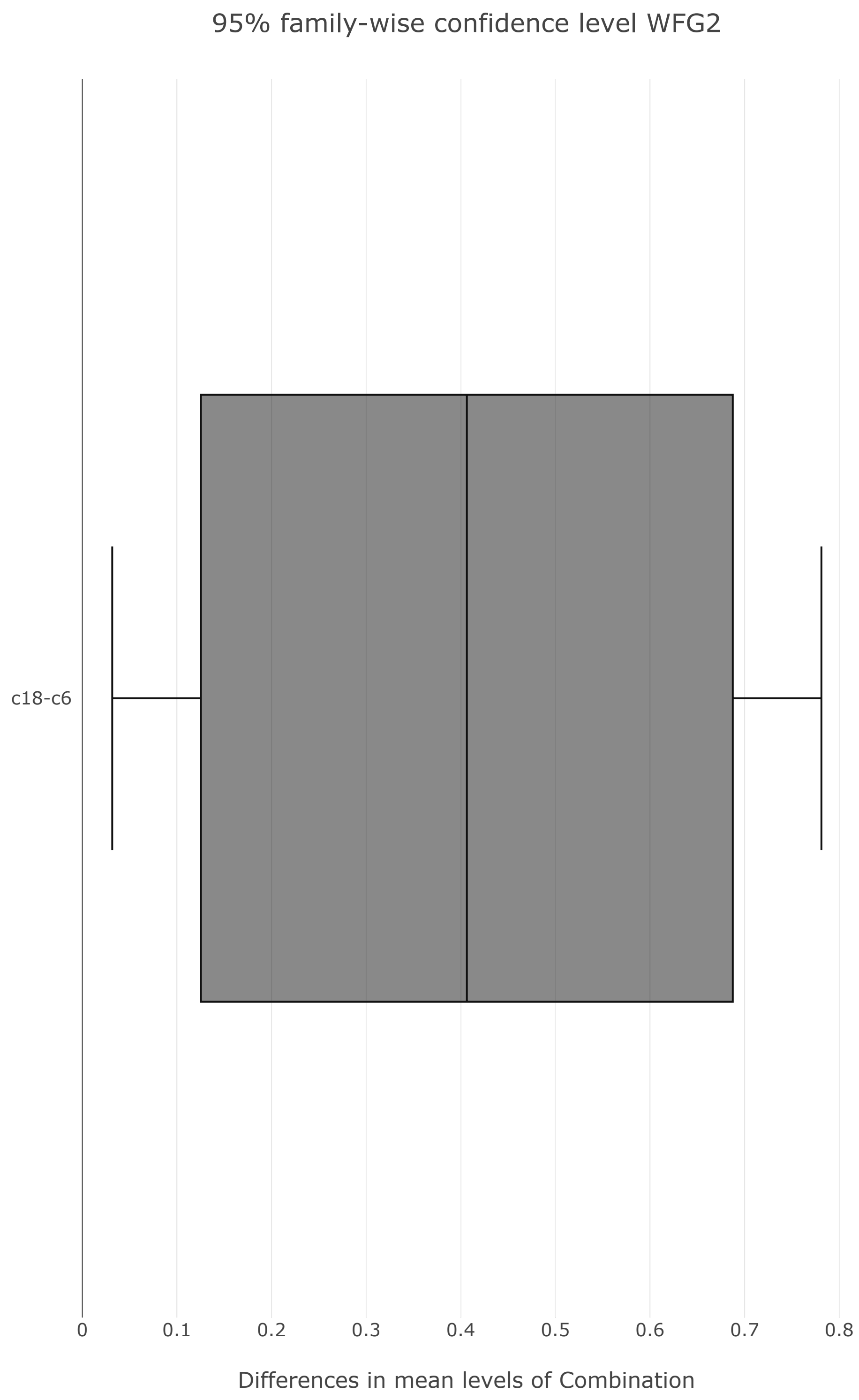


Figure B.20: Graphical representation of each confidence interval obtained by the Tukey test for WFG2, where each confidence interval does not intersect with zero.

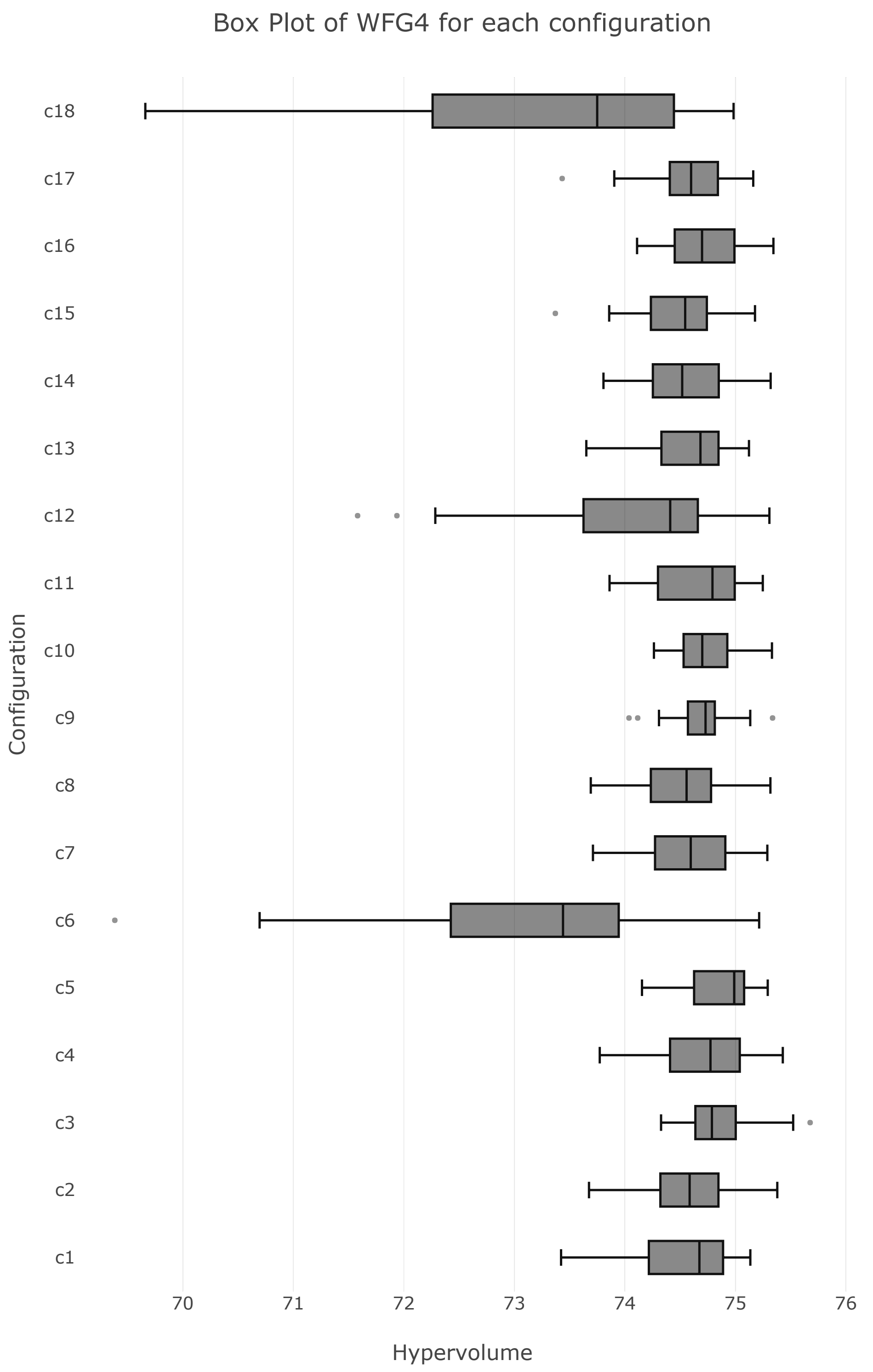


Figure B.21: Descriptive statistics for MOMA-II solving WFG4 with each combination for each pair of parameters.

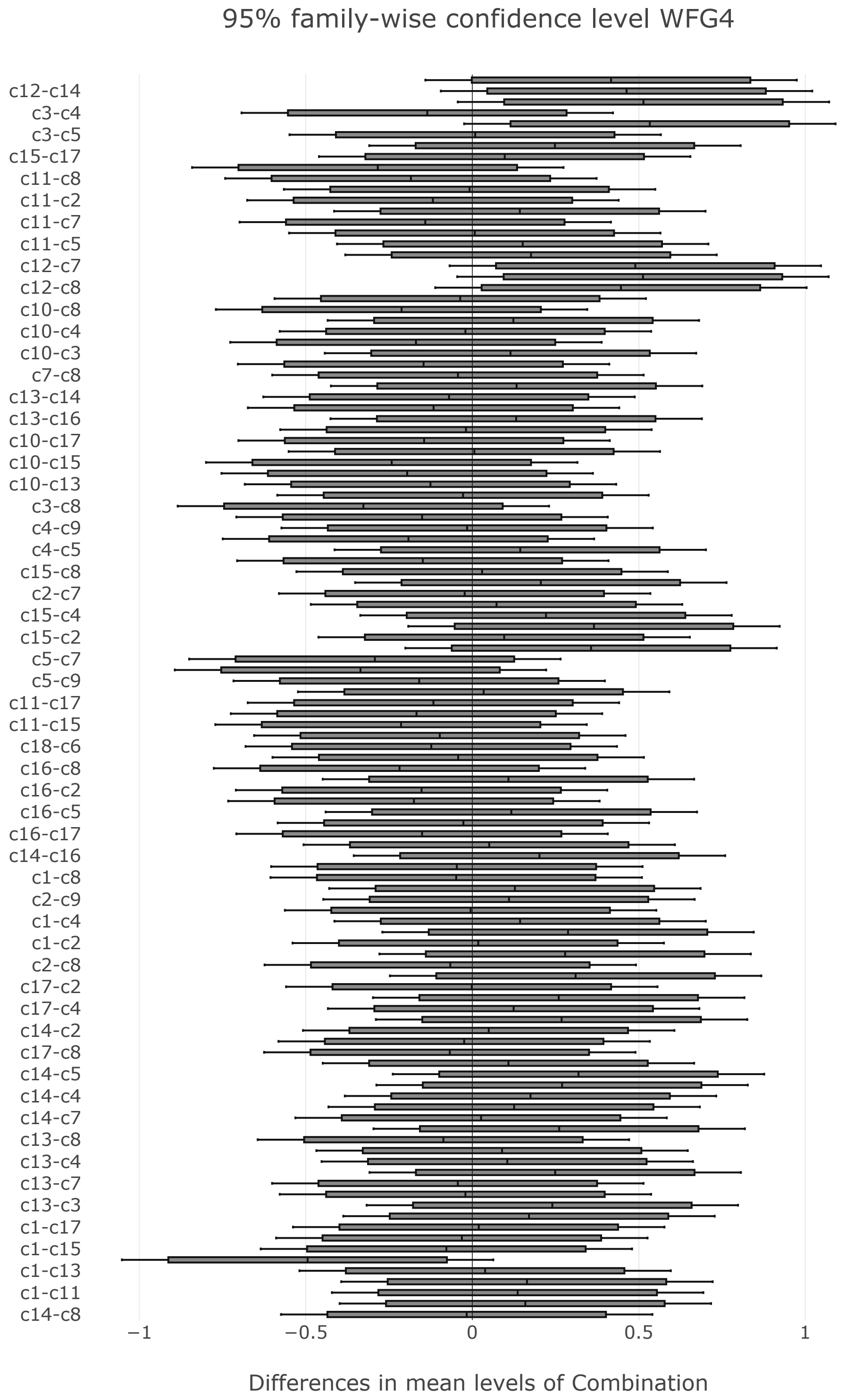


Figure B.22: Graphical representation of each confidence interval obtained by the Tukey test for WFG4, where each confidence interval intersects with zero.

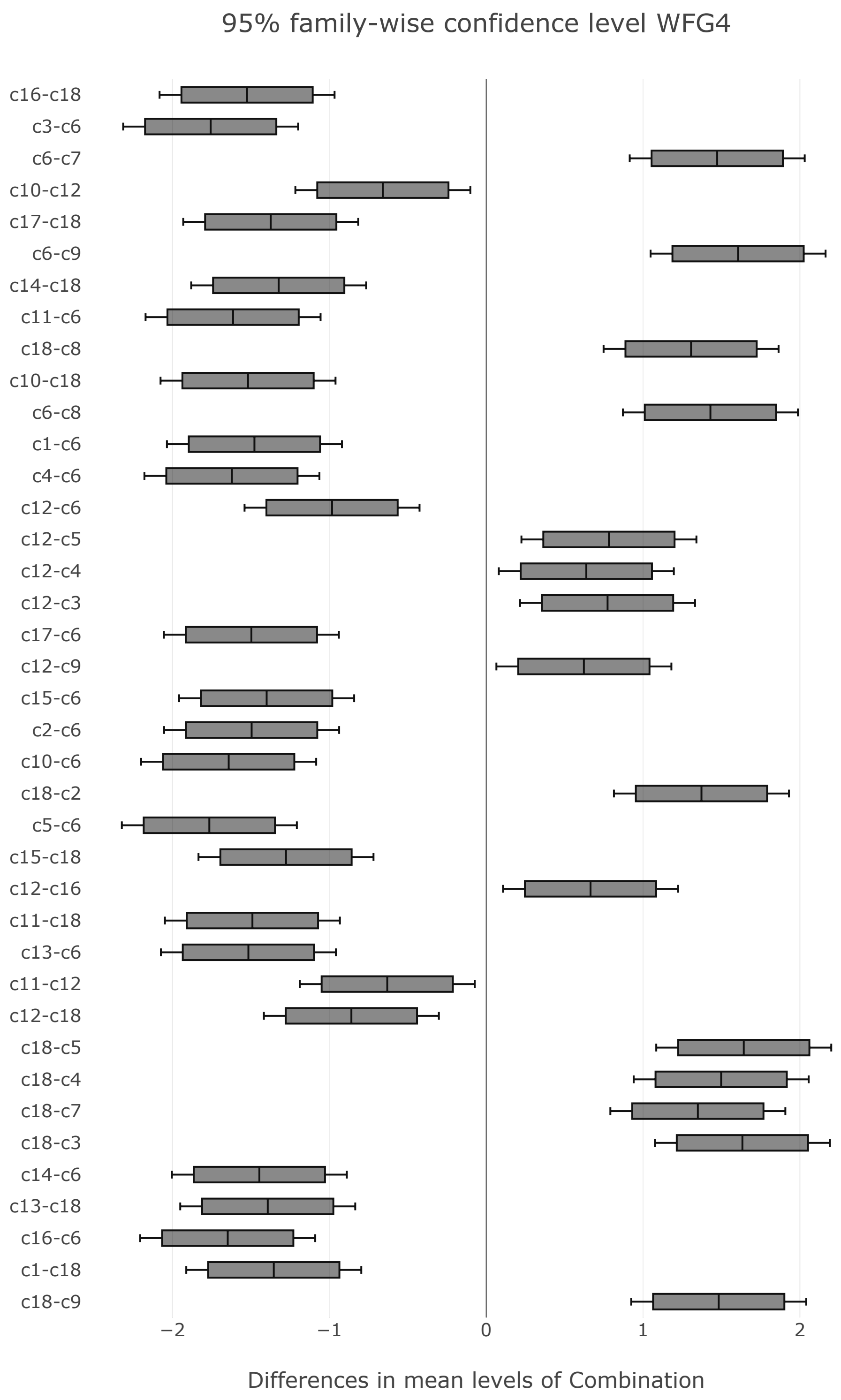


Figure B.23: Graphical representation of each confidence interval obtained by the Tukey test for WFG4, where each confidence interval does not intersect with zero.

Bibliography

[1] Martin Pilát and Roman Neruda. Hypervolume-Based Local Search in Multi-Objective Evolutionary Optimization. In *2014 Genetic and Evolutionary Computation Conference (GECCO 2014)*, pages 637–644, Vancouver, Canada, July 12-16 2014. ACM Press. ISBN 978-1-4503-2662-9.

[2] Yan-Yan Tan, Yong-Chang Jiao, Hong Li, and Xin-Kuan Wang. MOEA/D-SQA: a multi-objective memetic algorithm based on decomposition. *Engineering Optimization*, 44(9):1095–1115, 2012.

[3] 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. ISBN 978-0-387-33254-3.

[4] Edgar Manoatl Lopez, Luis Miguel Antonio, and Carlos A. Coello Coello. A GPU-Based Algorithm for a Faster Hypervolume Contribution Computation. In António Gaspar-Cunha, Carlos Henggeler Antunes, and Carlos Coello Coello, editors, *Evolutionary Multi-Criterion Optimization, 8th International Conference, EMO 2015*, pages 80–94. Springer. Lecture Notes in Computer Science Vol. 9019, Guimarães, Portugal, March 29 - April 1 2015.

[5] Man Leung Wong and Geng Cui. Data Mining Using Parallel Multi-objective Evolutionary Algorithms on Graphics Processing Units. In Shigeyoshi Tsutsui and Pierre Collet, editors, *Massively Parallel Evolutionary Computation on GPGPUs*, pages 287–307. Springer, 2013. ISBN 978-3-642-37958-1.

[6] Fernando Bernardes de Oliveira, Donald Davendra, and Frederico Gadelha Guimar aes. Multi-Objective Differential Evolution on the GPU with C-CUDA. In Václav Snášel, Ajith Abraham, and Emilio S. Corchado, editors, *Soft Computing Models in Industrial and Environmental Applications, 7th International Conference (SOCO'12)*, pages 123–132. Springer. Advances in Intelligent Systems and Computing Vol. 188, Ostrava, Czech Republic, 2013.

[7] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, second edition, 2006.

[8] A. A. Goldstein. Cauchy's method of minimization. *Numerische Mathematik*, 4(1):146–150, Dec 1962.

[9] Ioannis K. Argyros and Daniel González. *Newton's Method for Convex Optimization*, pages 23–56. Springer International Publishing, Cham, 2016.

[10] R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *The Computer Journal*, 7(2):149–154, 1964.

[11] Robert Hooke and T. A. Jeeves. " direct search" solution of numerical and statistical problems. *Journal of the ACM*, 8(2):212–229, April 1961.

[12] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

[13] David R. Oldroyd. Charles darwin's theory of evolution: A review of our present understanding. *Biology and Philosophy*, 1(2):133–168, Jun 1986.

[14] Chang Wook Ahn. *Advances in Evolutionary Algorithms. Theory, Design and Practice*. Springer, 2006. ISBN 3-540-31758-9.

[15] Lawrence J. Fogel. *Intelligence Through Simulated Evolution: Forty Years of Evolutionary Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1999.

[16] Melanie Mitchell and John H. Holland. When will a genetic algorithm outperform hill climbing? In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 647–, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[17] C.M. Fonseca and P.J. Fleming. Multiobjective genetic algorithms. In *Genetic Algorithms in Engineering Systems*, chapter 3, pages 63–78. The Institution of Electrical Engineers. Control Engineering Series 55, Bath, UK, 1997.

[18] Mitsuo Gen and Runwei Cheng. *Genetic Algorithms and Engineering Design*. John Wiley and Sons, Inc., New York, 1997.

[19] Heder S. Bernardino and Helio J. C. Barbosa. Artificial Immune Systems for Optimization. In *Nature-Inspired Algorithms for Optimisation*, pages 389–411. Springer, Berlin, 2009. ISBN 978-3-642-00266-3.

[20] Marco Dorigo and Thomas Stützle. *The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances*, pages 250–285. Springer US, Boston, MA, 2003.

[21] E. Kaya and M. S. Kiran. An improved binary artificial bee colony algorithm. In *2017 15th International Conference on ICT and Knowledge Engineering (ICT KE)*, pages 1–6, Nov 2017.

[22] J. Kennedy and R. Eberhart. Particle swarm optimization. In *IEEE International Conference on Neural Networks, 1995. Proceedings.,*, volume 4, pages 1942–1948 vol.4, November 1995.

[23] 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.

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

[25] Richard S. Rosenberg. Simulation of genetic populations with biochemical properties: II. Selection of crossover probabilities. *Mathematical Biosciences*, 8(1):1 – 37, 1970.

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

[27] 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.

[28] 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, USA, May 1999.

[29] Carlos A. Coello Coello and Margarita Reyes Sierra. A Study of the Parallelization of a Coevolutionary Multi-Objective Evolutionary Algorithm. In Raúl Monroy, Gustavo Arroyo-Figueroa, Luis Enrique Sucar, and Humberto Sossa, editors, *Proceedings of the Third Mexican International Conference on Artificial Intelligence (MICAI'2004)*, pages 688–697. Springer Verlag. Lecture Notes in Artificial Intelligence Vol. 2972, April 2004.

[30] 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.

[31] 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.

[32] Michael Pilegaard Hansen and Andrzej Jaszkiewicz. Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Technical University of Denmark, March 1998.

[33] Hisao Ishibuchi, Hiroyuki Masuda, Yuki Tanigaki, and Yusuke Nojima. Modified Distance Calculation in Generational Distance and Inverted Generational Distance. In António Gaspar-Cunha, Carlos Henggeler Antunes, and Carlos Coello Coello, editors, *Evolutionary Multi-Criterion Optimization, 8th International Conference, EMO 2015*, pages 110–125. Springer. Lecture Notes in Computer Science Vol. 9019, Guimarães, Portugal, March 29 - April 1 2015.

[34] Hisao Ishibuchi, Hiroyuki Masuda, and Yusuke Nojima. A Study on Performance Evaluation Ability of a Modified Inverted Generational Distance Indicator. In *2015 Genetic and Evolutionary Computation Conference (GECCO 2015)*, pages 695–702, Madrid, Spain, July 11-15 2015. ACM Press. ISBN 978-1-4503-3472-3.

[35] Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. Evolutionary many-objective optimization: A short review. In *2008 Congress on Evolutionary*

Computation (CEC'2008), pages 2424–2431, Hong Kong, June 2008. IEEE Service Center.

[36] 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.

[37] Nicola Beume. S-Metric Calculation by Considering Dominated Hypervolume as Klee's Measure Problem. *Evolutionary Computation*, 17(4):477–492, Winter 2009.

[38] 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.

[39] 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.

[40] Kalyanmoy Deb and Himanshu 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, August 2014.

[41] Ke Li, Kalyanmoy Deb, Qingfu Zhang, and Sam Kwong. An Evolutionary Many-Objective Optimization Algorithm Based on Dominance and Decomposition. *IEEE Transactions on Evolutionary Computation*, 19(5):694–716, October 2015.

[42] Ran Cheng, Yaochu Jin, Markus Olhofer, and Bernhard Sendhoff. A Reference Vector Guided Evolutionary Algorithm for Many-Objective Optimization. *IEEE Transactions on Evolutionary Computation*, 20(5):773–791, October 2016.

[43] 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.

[44] K. Gerstl, G. Rudolph, O. Schütze, and H. Trautmann. Finding Evenly Spaced Fronts for Multiobjective Control via Averaging Hausdorff-Measure. In *The 2011 8th International Conference on Electrical Engineering, Computer Science and Automatic Control (CCE'2011)*, pages 975–980, Mérida, Yucatán, México, October 2011. IEEE Press.

[45] 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.

[46] Saúl Zapotecas Martínez, Víctor A. Sosa Hernández, Hernán Aguirre, Kiyoshi Tanaka, and Carlos A. Coello Coello. Using a Family of Curves to Approximate the Pareto Front of a Multi-Objective Optimization Problem. In Thomas Bartz-Beielstein, Jürgen Branke, Bogdan Filipič, and Jim Smith, editors, *Parallel Problem Solving from Nature - PPSN XIII, 13th International Conference*, pages 682–691. Springer. Lecture Notes in Computer Science Vol. 8672, Ljubljana, Slovenia, September 13-17 2014.

[47] Adriana Menchaca-Mendez, Carlos Hernández, and Carlos A. Coello Coello. Δ_p-MOEA: A New Multi-Objective Evolutionary Algorithm Based on the Δ_p Indicator. In *2016 IEEE Congress on Evolutionary Computation (CEC'2016)*, pages 3753–3760, Vancouver, Canada, 24-29 July 2016. IEEE Press. ISBN 978-1-5090-0623-9.

[48] Raquel Hernández Gómez and Carlos A. Coello Coello. MOMBI: A New Metaheuristic for Many-Objective Optimization Based on the R2 Indicator. In *2013 IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 2488–2495, Cancún, México, 20-23 June 2013. IEEE Press. ISBN 978-1-4799-0454-9.

[49] Raquel Hernández Gómez and Carlos A. Coello Coello. Improved Metaheuristic Based on the R2 Indicator for Many-Objective Optimization. In *2015 Genetic and Evolutionary Computation Conference (GECCO 2015)*, pages 679–686, Madrid, Spain, July 11-15 2015. ACM Press. ISBN 978-1-4503-3472-3.

[50] Pablo Moscato. Memetic algorithms: A short introduction. pages 219–234. McGraw-Hill Ltd., UK, Maidenhead, UK, England, 1999.

[51] Chi-Keong Goh. *Multi-Objective Memetic Algorithms.* Springer, Berlin, Germany, 2009. ISBN 978-3-540-88050-9.

[52] S.F. Adra, I.A. Griffin, and P.J. Fleming. An Adaptive Memetic Algorithm for Enhanced Diversity. In *Adaptive Computing in Design and Manufacture 2006. Proceedings of the Seventh International Conference*, pages 251–254, Bristol, UK, April 2006. The Institute for People-centred Computation.

[53] J. H. Ang, K. C. Tan, and A. A. Mamum. An Evolutionary Memetic Algorithm for Rule Extraction. *Expert Systems with Applications*, 37(2):1302–1315, March 2010.

[54] Joshua D. Knowles and David W. Corne. A Comparison of Diverse Approaches to Memetic Multiobjective Combinatorial Optimization. In *Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program*, pages 103–108, Las Vegas, Nevada, July 2000.

[55] Julian Molina, Manuel Laguna, Rafael Marti, and Rafael Caballero. SSPMO: A Scatter Tabu Search Procedure for Non-Linear Multiobjective Optimization. *INFORMS Journal on Computing*, 19:91–100, 02 2007.

[56] Gabor J. Szekely and Maria L. Rizzo. Hierarchical clustering via joint between-within distances: Extending ward's minimum variance method. *Journal of Classification*, 22(2):151–183, Sep 2005.

[57] Carlos A. Coello Coello and Maximino Salazar Lechuga. MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization. In *Congress on Evolutionary Computation (CEC'2002)*, volume 2, pages 1051–1056, Piscataway, New Jersey, May 2002. IEEE Service Center.

[58] Patrick Koch, Oliver Kramer, Günter Rudolph, and Nicola Beume. On the Hybridization of SMS-EMOA and Local Search for Continuous Multiobjective Optimization. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, GECCO '09, pages 603–610, New York, NY, USA, 2009. ACM.

[59] Peter A. N. Bosman and Edwin D. de Jong. Exploiting gradient information in numerical multi–objective evolutionary optimization. In *Proceedings of the 7th*

Annual Conference on Genetic and Evolutionary Computation, GECCO '05, pages 755–762, New York, NY, USA, 2005. ACM.

[60] Peter A. N. Bosman and Edwin D. de Jong. Combining gradient techniques for numerical multi-objective evolutionary optimization. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, GECCO '06, pages 627–634, New York, NY, USA, 2006. ACM.

[61] Pradyumn Kumar Shukla. Gradient based stochastic mutation operators in evolutionary multi-objective optimization. In Bartlomiej Beliczynski, Andrzej Dzielinski, Marcin Iwanowski, and Bernardete Ribeiro, editors, *Adaptive and Natural Computing Algorithms*, pages 58–66, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[62] Adriana Lara, Gustavo Sanchez, Carlos A. Coello Coello, and Oliver Schütze. HCS: A New Local Search Strategy for Memetic Multi-Objective Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation*, 14(1):112–132, February 2010.

[63] Oliver Schutze, Victor Adrian Sosa Hernandez, Heike Trautmann, and Gunter Rudolph. The Hypervolume Based Directed Search Method for Multi-Objective Optimization Problems. *Journal of Heuristics*, 22(3):273–300, June 2016.

[64] Na Wang, Hongfeng Wang, Yaping Fu, and Dingwei Wang. A decomposition based memetic multi-objective algorithm for continuous multi-objective optimization problem. pages 896–900, 07 2015.

[65] Jesus Alejandro Hernandez Mejia, Oliver Schutze, Oliver Cuate, Adriana Lara, and Kalyanmoy Deb. RDS-NSGA-II: A Memetic Algorithm for Reference Point Based Multi-Objective Optimization. *Engineering Optimization*, 49(5):828–845, 2017.

[66] Peter A. N. Bosman and Dirk Thierens. The naive MIDEA: a baseline multi-objective EA. volume 3410, pages 428–442, 03 2005.

[67] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Proceedings of the Parallel Problem Solving from Nature VI*

Conference, pages 849–858, Paris, France, 2000. Springer. Lecture Notes in Computer Science No. 1917.

[68] 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, 2001.

[69] Oliver Schütze, Adanay Martin, Adriana Lara, Sergio Alvarado, Eduardo Salinas, and Carlos A. Coello Coello. The directed search method for multi-objective memetic algorithms. *Computational Optimization and Applications*, 63(2):305–332, March 2016.

[70] Saúl Zapotecas Martínez, Alfredo Arias Montaño, and Carlos A. Coello Coello. A Nonlinear Simplex Search Approach for Multi-Objective Optimization. In *2011 IEEE Congress on Evolutionary Computation (CEC'2011)*, pages 2367–2374, New Orleans, Louisiana, USA, 5-8 June 2011. IEEE Service Center.

[71] Praveen Koduru, Sanjoy Das, Stephen Welch, and Judith L. Roe. Fuzzy Dominance Based Multi-objective GA-Simplex Hybrid Algorithms Applied to Gene Network Models. In Kalyanmoy Deb et al., editor, *Genetic and Evolutionary Computation–GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part I*, pages 356–367, Seattle, Washington, USA, June 2004. Springer-Verlag, Lecture Notes in Computer Science Vol. 3102.

[72] Kleopatra Pirpinia, Tanja Alderliesten, Jan-Jakob Sonke, Marcel van Herk, and Peter A.N. Bosman. Diversifying Multi-Objective Gradient Techniques and Their Role in Hybrid Multi-Objective Evolutionary Algorithms for Deformable Medical Image Registration. In *2015 Genetic and Evolutionary Computation Conference (GECCO 2015)*, pages 1255–1262, Madrid, Spain, July 11-15 2015. ACM Press. ISBN 978-1-4503-3472-3.

[73] Xavier Gandibleux, Nazik Mezdaoui, and Arnaud Fréville. A tabu search procedure to solve multiobjective combinatorial optimization problems. In Rafael Caballero, Francisco Ruiz, and Ralph Steuer, editors, *Advances in Multiple Objective and Goal Programming*, pages 291–300, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

[74] Ching-Shih Tsou, Hsiao-Hua Fang, Hsu-Hwa Chang, and Chia-Hung Kao. An Improved Particle Swarm Pareto Optimizer with Local Search and Clustering. In *Simulated Evolution and Learning, 6th International Conference, SEAL 2006, Proceedings*, pages 400–407, Hefei, China, October 2006. Springer. Lecture Notes in Computer Science Vol. 4247.

[75] Wali Khan Mashwani and Abdellah Salhi. Multiobjective memetic algorithm based on decomposition. *Applied Soft Computing*, 21:221–243, August 2014.

[76] H. Li and Q. Zhang. Multiobjective optimization problems with complicated pareto sets, moea/d and nsga-ii. *IEEE Transactions on Evolutionary Computation*, 13(2):284–302, April 2009.

[77] Qingfu Zhang, Aimin Zhou, Shizheng Zhao, Ponnuthurai Suganthan, Wudong Liu, and Santosh Tiwari. Multiobjective optimization test instances for the cec 2009 special session and competition. *Mechanical Engineering*, 01 2008.

[78] Yutao Qi, Zhanting Hou, He Li, Jianbin Huang, and Xiaodong Li. A decomposition based memetic algorithm for multi-objective vehicle routing problem with time windows. *Computers and Operations Research*, 62, 04 2015.

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

[80] Praveen Koduru, Sanjoy Das, and Stephen M. Welch. Multi-Objective Hybrid PSO Using ϵ-Fuzzy Dominance. In Dirk Thierens, editor, *2007 Genetic and Evolutionary Computation Conference (GECCO'2007)*, volume 1, pages 853–860, London, UK, July 2007. ACM Press.

[81] Riccardo Poli, James Kennedy, and Tim Blackwell. Particle swarm optimization. *Swarm Intelligence*, 1(1):33–57, Jun 2007.

[82] 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. ISBN 978-0-387-33254-3.

[83] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms.* John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.

[84] Karl Bringmann and Tobias Friedrich. Approximating the Least Hypervolume Contributor: NP-Hard in General, But Fast in Practice. In Matthias Ehrgott, Carlos M. Fonseca, Xavier Gandibleux, Jin-Kao Hao, and Marc Sevaux, editors, *Evolutionary Multi-Criterion Optimization. 5th International Conference, EMO 2009*, pages 6–20. Springer. Lecture Notes in Computer Science Vol. 5467, Nantes, France, April 2009.

[85] 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.

[86] José A. Molinet Berenguer and Carlos A. Coello Coello. Evolutionary Many-Objective Optimization Based on Kuhn-Munkres' Algorithm. In António Gaspar-Cunha, Carlos Henggeler Antunes, and Carlos Coello Coello, editors, *Evolutionary Multi-Criterion Optimization, 8th International Conference, EMO 2015*, pages 3–17. Springer. Lecture Notes in Computer Science Vol. 9019, Guimarães, Portugal, March 29 - April 1 2015.

[87] H. W. Kuhn and Bryn Yaw. The hungarian method for the assignment problem. *Naval Research Logistics. Quart*, pages 83–97, 1955.

[88] François Bourgeois and Jean-Claude Lassalle. An extension of the munkres algorithm for the assignment problem to rectangular matrices. *Commun. ACM*, 14(12):802–804, December 1971.

[89] Michael T.M. Emmerich and André H. Deutz. Test Problems Based on Lamé Superspheres. In Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, and Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*, pages 922–936, Matshushima, Japan, March 2007. Springer. Lecture Notes in Computer Science Vol. 4403.

[90] Indraneel Das and J. E. Dennis. Normal-Boundary Intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8(3):631–657, March 1998.

[91] Johannes Bader and Eckart Zitzler. HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization. *Evolutionary Computation*, 19(1):45–76, Spring 2011.

[92] 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, USA, 2005.

[93] 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.

[94] NVIDIA Corporation. CUDA Zone, 2014.

[95] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. Scalable Parallel Programming with CUDA. *Queue*, 6(2):40–53, March 2008.

[96] John E. Stone, David Gohara, and Guochun Shi. OpenCL: A Parallel Programming Standard for Heterogeneous Computing Systems. *IEEE Des. Test*, 12(3):66–73, May 2010.

[97] Michael J. Flynn. Some computer organizations and their effectiveness. *IEEE Transactions Computers*, 21(9):948–960, September 1972.

[98] Edgar Manoatl Lopez and Carlos A. Coello Coello. IGD$^+$-EMOA: A Multi-Objective Evolutionary Algorithm based on IGD$^+$. In *2016 IEEE Congress on Evolutionary Computation (CEC'2016)*, pages 999–1006, Vancouver, Canada, 24-29 July 2016. IEEE Press. ISBN 978-1-5090-0623-9.

[99] Edgar Manoatl Lopez and Carlos A. Coello Coello. Improving the Integration of the IGD+ Indicator into the Selection Mechanism of a Multi-objective Evolutionary Algorithm. In *2017 IEEE Congress on Evolutionary Computation (CEC'2017)*, pages 2683–2690, San Sebastián, Spain, June 5-8 2017. IEEE Press. ISBN 978-1-5090-4601-0.

[100] Eckart Zitzler and Simon Künzli. Indicator-based Selection in Multiobjective Search. In Xin Yao et al., editor, *Parallel Problem Solving from Nature - PPSN VIII*, pages 832–842, Birmingham, UK, September 2004. Springer-Verlag. Lecture Notes in Computer Science Vol. 3242.

[101] Ran Cheng, Miqing Li, Ye Tian, Xingyi Zhang, Shengxiang Yang, Yaochu Jin, and Xin Yao. A benchmark test suite for evolutionary many-objective optimization. *Complex & Intelligent Systems*, 3(1):67–81, Mar 2017.

[102] Xingtao Liao, Qing Li, Xujing Yang, Weigang Zhang, and Wei Li. Multiobjective optimization for crash safety design of vehicles using stepwise regression model. *Structural and Multidisciplinary Optimization*, 35(6):561–569, Jun 2008.

[103] William Scheaffer Mendenhall, Richard L Wackerly, and D Dennis. *Estadística matemática con aplicaciones.* Grupo Editorial Iberoamérica,, 1994.

[104] Ronald E Walpole, Raymond H Myers, and Sharon L Myers. *Probabilidad y estadística para ingenieros.* Pearson Educación, 1999.