Large-scale Multi-objective Optimisation: New Approaches and a Classification of the State-of-the-Art

DISSERTATION

zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

angenommen durch die Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg

von M.Sc. Heiner Zille

geb. am 28.06.1988 in Cuxhaven

Gutachterinnen/Gutachter

Prof. Dr.-Ing. habil. Sanaz Mostaghim
Prof. Dr. Hisao Ishibuchi
Prof. Dr. Juergen Branke

Magdeburg, den 11.12.2019

Abstract

Many problems occurring in nature or technical applications can be formulated as optimisation problems with multiple, conflicting goals that need to be optimised simultaneously. Solving such problems requires a search for the optimal input parameters to the problem, called decision variables. This kind of problems is often solved with metaheuristic approaches such as evolutionary algorithms. In this field of multi-objective optimisation, the topic of solving large-scale problems has become increasingly popular in recent years. Large-scale optimisation in general deals with the optimisation of problems that contain large numbers of decision variables, objective functions or both. The performance of classical algorithms in the optimisation area often deteriorates when faced with large-scale problems. The topic of this thesis is the optimisation of such large-scale problems, with a focus on high-dimensional search spaces, i.e. problems that contain multiple hundreds or thousands of decision variables.

Several approaches have been proposed in the literature which use different strategies, in many cases to reduce the dimensionality of the problem and thus make traditional algorithms applicable to such high-dimensional problems. These approaches are theoretically analysed and compared, and a classification scheme is proposed based on the different techniques used in the related literature. Moreover, many of the related mechanisms require the division of variables into groups. Several mechanisms to do so are described in this thesis, and these are formally categorised based on three proposed classes of grouping methods.

The algorithmic contributions of this thesis include three proposed optimisation techniques for large-scale multi-objective optimisation. Each of these three methods is designed to be used with arbitrary metaheuristics from the literature, and to enable existing algorithms to search efficiently in high-dimensional decision spaces. The proposed mechanisms are theoretically described and analysed. They are further compared to each other and categories based on the proposed classification scheme.

Finally, this thesis provides an extensive experimental evaluation, including the proposed approaches as well as various methods from the literature. Several interesting advantages and disadvantages of the tested algorithms are described and compared, including the dependency on variable groups, and the performances in terms of convergence behaviour and final solution quality. The results show that the proposed approaches are able to heavily increase the performance of existing algorithms for large-scale problems, and that they are competitive and in many cases superior to the state-of-the-art approaches in this field.

Zusammenfassung

Viele Probleme in realen Anwendungen lassen sich mathematisch als Optimierungsprobleme mit mehreren, in Konflikt stehenden Zielen formulieren. Die Lösung solcher Probleme erfordert die Suche nach den optimalen Kombinationen der Designparameter des Problems, sogenannte Entscheidungsvariablen. Neben exakten Methoden werden hierfür in der Praxis oft sogenannte Metaheuristiken wie etwa evolutionäre Algorithmen angewendet. In diesem Forschungsfeld der mehrkriteriellen Optimierung hat das Lösen von hochdimensionalen, sogenannten "large-scale" Optimierungsproblemen in den letzten Jahren immer mehr an Bedeutung gewonnen. Large-scale Optimierung befasst sich mit der Optimierung von Problemen mit mehreren Zielfunktionen und hunderten bis tausenden von Entscheidungsvariablen. Klassische Algorithmen aus dem Bereich der mehrkriteriellen Optimierung sind oft ungeeignet für die Optimierung in solch hochdimensionalen Suchräumen. Die vorliegende Dissertation befasst sich mit der Optimierung von solch hochdimensionalen Problemen. Existierende Methoden werden in der Arbeit theoretisch untersucht, neue Methoden vorgestellt, und eine experimentelle Evaluation durchgeführt.

Verschiedene Ansätze sind in der Literatur zu diesem Thema publiziert worden, oftmals mit dem Ziel durch verschiedene Techniken den Suchraum zu verkleinern, um so klassische Algorithmen anwendbar zu machen. Die Methoden werden theoretisch analysiert und verglichen, und basierend auf den Eigenschaften der Algorithmen aus der Literatur wird ein Klassifizierungsschema vorgestellt. Viele der verwandten Methoden verwenden außerdem einen Mechanismus, um die Variablen in Gruppen einzuteilen. Verschiedene dieser Techniken werden in dieser Arbeit beschrieben und in drei vorgeschlagene Kategorien eingeordnet. Die algorithmischen Beiträge dieser Arbeit umfassen drei vom Autor entwickelte Optimierungstechniken für hochdimensionale mehrkriterielle Probleme. Jede dieser drei Methoden ist so gestaltet, dass beliebige existierende Algorithmen in die Lage versetzt werden, hochdimensionale Suchräume effizient zu durchsuchen. Die drei entwickelten Methoden werden analysiert, verglichen und schließlich in die vorgestellten Klassen von Algorithmen eingeordnet. Die experimentelle Evaluation dieser Arbeit umfasst die entwickelten Algorithmen sowie verschiedene der verwandten Methoden aus der Literatur. In der Analyse werden verschiedene Vor- und Nachteile der Algorithmen beschrieben und verglichen, wie beispielsweise die Abhängigkeit von Variablengruppen und die Leistung in Bezug auf Konvergenzgeschwindigkeit und finale Lösungsqualität. Die Ergebnisse zeigen, dass die entwickelten Methoden in der Lage sind, die Leistung existierender Algorithmen für hochdimensionale Probleme stark zu verbessern. Weiterhin zeigen die vorgestellten Techniken eine vergleichbare und in vielen Fällen überlegene Lösungsqualität im Vergleich mit anderen aktuellen large-scale Methoden.

Contents

Chapter 1

Introduction

In the field of multi-objective optimisation, the topic of solving of large-scale problems has become increasingly popular in the recent years. Large-scale Optimisation (LSO) in general deals with the optimisation of problems that contain large numbers of decision variables, objective functions or both. Using classic metaheuristic algorithms for such problems often leads to a decreased performance when the dimensionality of the problem increases. When large numbers of variables are involved, algorithms are faced with very high-dimensional search spaces that are difficult to explore with limited computational resources [1]. This dissertation thesis deals with such kind of problems and specifically with problems that contain large amounts of decision variables. The thesis introduces basic knowledge in this area and present recent advances and proposed approaches in solving multi-objective large-scale optimisation problems.

1.1 Motivation

Many problems in the real world can be formulated and solved as optimisation problems. This includes classical logistic or combinatorial problems like the Travelling Salesman Problem or optimal resource allocation in a factory setting. For many of these problems, analytical descriptions of the underlying goal are hard or impossible to obtain. Therefore, classical optimisation techniques that make use of gradient information are often not applicable. Moreover, complex problems like the aforementioned Travelling Salesman Problem are NP-hard, which means a deterministic and optimal solution to these combinatorial problems is bound to an expected large computational budget.

To overcome these issues, research has come up with a variety of metaheuristic algorithms, which are often based on natural inspiration from biology or physics. While some local search mechanisms like Simulated Annealing are based on the laws of physics, most metaheuristic methods for global optimisation have their source in biological systems, the most prominent ones being swarm behaviour and the theory of evolution. From these, many algorithms in the area of Particle Swarm Optimisation (PSO) and Evolutionary

Algorithms (EA) have been developed over the years and turned out to be useful when solving optimisation problems.

A challenge in real world applications is that the processes to optimise can have multiple goals that need to be reached simultaneously, and may potentially be in conflict. A simple textbook example is the construction of a car which can be designed to be fast, but at a higher price, or cheap, which might decrease its maximum speed. Speed and price can therefore be regarded as conflicting objectives of this optimisation problem. A car's design can in such a case be represented as a set of chosen parameters, such as the configuration of the engine, the materials used, the shape of the car, and so forth. Constructing a car - or more precisely: finding a parameter configuration to be used in the construction process - that fulfils both of the mentioned objectives as well as possible is a task covered by the area of Multi-objective Optimisation (MOO). Since the beginning of the century, the research in this area has increased, and with increased computational power researchers and engineers are able to optimise problems with higher complexity. However, research showed that established multi-objective algorithms become less effective for high-dimensional problems. As a consequence, an increased dimensionality and complexity of the problem not only requires additional computational resources, but also specialised algorithms.

The core challenge of the present thesis is to examine and propose optimisation strategies for multi-objective problems that contain large amounts of design parameters. A basic example for such a problem is, for instance, the shape optimisation of objects like aircrafts, trains or cars. A prominent example of this kind of problem is the new *Shinkansen* high-speed train in Japan, where the front of the train was optimised using evolutionary computation [2]. These problems contain multiple objectives like the costs or the aerodynamic properties, and are often modelled by a mesh grid of the shape to optimise. Depending on how fine or coarse this grid is modelled, the problem might allow more or less detailed changes in the objective functions, on the cost of increasing the dimensionality of the search space (as each control point of the grid needs to have "optimal" coordinates).

Regarding the scope of this thesis, the large-scale optimisation area can roughly be divided into those challenges associated with large numbers of objective functions and those with large numbers of decision variables. While the former, the so-called many-objective optimisation, has drawn increased attention over the last decade, it must be noted here that the main focus of this thesis lies on large search spaces. The definition of "large" in this context varies in the literature, as is discussed in later chapters, but is usually associated with multiple hundreds to multiple thousands of variables. Challenges and related methods on many-objective optimisation are of influence in different parts of this thesis, but the proposed methods and considerations are primarily focused on problems with large numbers of decision variables.

The area of single-objective large-scale optimisation has been the subject of many research activities in the recent years, while, in contrast, the area of multi-objective problems with large numbers of variables has received less attention in the scientific community so far. Up until the year 2013, to the best of the author's knowledge, no dedicated large-scale multi-objective metaheuristic algorithm existed in the scientific community. The methods developed in the last years, the theoretic backgrounds and challenges and the optimisation techniques proposed by the author in the process are the topics of this dissertation thesis.

1.2 Research Goals and Contributions

The aim of this thesis is the proposal and examination of several ways to optimise large-scale multi-objective optimisation methods, containing large numbers of decision variables, and the challenges and properties of such kind of problems. To reach these goals, the thesis pursues to answer four research objectives which are listed and explained in further detail below.

The contributions of this thesis to the scientific field of large-scale multi-objective optimisation include several proposed methods, the classification of existing and proposed large-scale algorithms and grouping mechanisms, and an extensive experimental evaluation. Some of these contributions and their detailed descriptions are based on the respective publications made by the author in the last years, which is highlighted in each respective section where applicable. Further contributions include a detailed comparison of related literature and a summary and comparison of the respective method's properties.

Objective 1: Analysis and Classification of the State-of-the-Art

To tackle the challenges of large-scale optimisation it is necessary to understand its underlying theory. Concepts like exploration and exploitation that are well understood in metaheuristic optimisation might form a new challenge when the dimension of the search space is large. In addition, the single-objective literature as well as the recent advances in multi-objective large-scale optimisation has concentrated on a variety of different mechanisms that turned out to be helpful in the exploration of high-dimensional spaces like coevolution, grouping mechanisms or problem transformation.

Research Objective 1 is meant to examine the challenges of large search spaces and the approaches developed for multi-objective large-scale optimisation in the recent years. To fulfil this objective, the thesis first describes the existing approaches and compares them theoretically. A comparison of their methodologies is carried out and properties, advantages and disadvantages of the different methods are pointed out. Based on this, the algorithms are decomposed into a number of building blocks, and the similarities and differences in terms of building blocks are examined. All algorithms are further categorised based on a proposed classification scheme, using a set of categories regarding criteria

such as dimensionality reduction, diversity management, many-objective capabilities, and more. These contributions can be found as follows: Basic principles and challenges of large-scale optimisation as well as related methods are described in Chapters 2 and 3. The classification and overview of the existing state-of-the-art and its components is proposed in Chapter 4.

Objective 2: Examination and Classification of Grouping Mechanisms

One of the most important building blocks is the separation of the design variables into groups. Most, but not all of the current large-scale methods require such a mechanism to divide the variables, although they differ in the way these groups are used in the optimisation process. Creation of groups can be done in many different ways, from simple random groups to sophisticated methods that analyse the correlations of variables, but in exchange often come with a high computational cost. It is, however, not known how beneficial especially interaction-based groups are for the performance of existing algorithms, i.e. whether computational overhead in finding "good" interaction-based groups pays off in terms of solution quality.

Research Objective 2 is hence to examine how existing grouping methods for variables can work and how important different groups are for the results of the optimisation with such group-based algorithms. To reach this objective, a description of existing methods is given in Section 3.3 and a classification into different categories and analysis of their properties is presented in Section 4.2. An empirical analysis of the influence of groups on the performance of some current large-scale algorithms is carried out in Section 6.7.

Objective 3: Proposal of new Algorithms

Large-scale optimisation in the single-objective area has gained popularity in the last decade. However, the efficient approximation of large-scale multi-objective problems was in large parts unexplored until the year 2013. Partly, this might be attributed to limited computational resources. Especially the area on many-objective optimisation has gained popularity and a number of new algorithms have been proposed in the recent years. On the other hand, research on large search spaces in multi-objective problems had been, to the best of the author's knowledge, widely non-existent prior to the year 2013. This changed with the proposal of the CCGDE3 [3], and in the time since an increasing number of methods have been proposed to tackle these problems.

Research Objective 3 of this thesis is to propose new methods to solve such problems and to improve the search abilities of current algorithms for large-scale multi-objective problems. In the course of this thesis' scientific process, three new search algorithms have been developed and published. These are

1. The *Weigthed Optimisation Framework* (WOF) that uses weight variables and problem transformation for dimensionality reduction.

2. The incorporation of variable groups into traditional genetic mutation operators.

3. The dimensionality reduction through search in a subspace spanned by linear combinations of solution candidates.

All of these techniques are described and analysed in detail. Their methodologies, advantages and drawbacks are examined in Sections 5.1 to 5.3 respectively. These methods and their building blocks are also classified and analysed based on the proposed categories and criteria in Chapter 4. An experimental evaluation of the algorithms' properties and performances is included in Chapter 6.

Objective 4: Experimental Evaluation

Research Objective 4 is to compare the proposed large-scale approaches as well as some of the existing methods from the literature with each other. The evaluation in Chapter 6 covers the comparison between the algorithms and the analysis of their strengths and weaknesses regarding different criteria. It is visible in the literature that some methods favour convergence towards good solutions while others are able to maintain a better diversity of the solution set. In addition, some methods have a computational overhead for finding suitable groups of design variables before the optimisation process starts. Another important factor may be the necessity of "suitable" variable groups. Therefore, the evaluation will compare methods with different performance measures such as the final solutions produced by the methods, their convergence speed and their computational budgets.

To be able to give as best as an overview over the capabilities of the algorithms, a variety of test functions from the literature is used, which are described briefly in Section 2.6. These come from different benchmark families and represent a variety of different properties. Under the assumption that these benchmarks represent properties of real-world applications, the analysis focusses on the different algorithms and the building blocks of which they consist. By that, we aim to not only identify which algorithms perform superior on certain problems, but also which building blocks in general seem to be favourable for the development of future algorithms in this area.

1.3 Structure of the Thesis

This dissertation thesis is structured in the following way. Chapter 2 introduces the basic concepts that are needed for the rest of the thesis. It covers the foundations of multi-objective optimisation, Pareto-optimality and evolutionary algorithms. Chapter 2 further deals with the challenges of large-scale optimisation. It explains some of the changes that occur when the number of decision variables is increased and briefly covers basics of many-objective optimisation. The notion of grouping mechanisms is formally defined in Section 2.4 and the concept of *Cooperative Coevolution* is explained in detail

in Section 2.5, as it forms the basis and inspiration for most of the large-scale methods. Section 2.6 describes the basic concepts of the most common benchmark families and their properties, which are used for the later evaluation. Finally, Section 2.7 introduces some of the most common indicators to measure the performance of multi-objective algorithms.

Chapter 3 deals with related work from the area of multi-objective large-scale optimisation. Large-scale methods from the literature are introduced and briefly discussed in Section 3.2. A selection of different grouping mechanisms for variables for single- and multi-objective optimisation is then presented in Section 3.3.

Chapter 4 compares and classifies the previously described algorithms and grouping mechanisms. This enables a more general view of large-scale methods and helps to identify interchangeable components. The existing works are compared based on a variety of criteria, which include, among others, their methods of dimensionality reduction, the necessary computational budget or the ability to work in many-objective scenarios. The classification of large-scale optimisation algorithms and the analysis of their components form a new contribution in this thesis and has not been published before.

Chapter 5 then describes the new methods for optimising large-scale problems that are developed in the course of this thesis and its preceding publications. These are the Weighted Optimisation Framework (WOF) (Section 5.1), the usage of variable groups in mutation operators (Section 5.2), the dimensionality reduction through linear combinations (Section 5.3). Each of these methods is analysed in detail, with a deeper focus on the effects, possibilities and limitations of the problem transformation techniques in the WOF method. The findings and analyses are based on the contributions of the author made in [4, 5, 1, 6, 7], but are extended in depth and detail in this dissertation. In the end of the chapter, the proposed methods are analysed with respect to the classification criteria used in Chapter 4.

Chapter 6 contains the experimental evaluation of the proposed and related methods as well as the evaluation of the influence of different variable groups. First, each of the three proposed search mechanisms is evaluated individually in different configurations in Sections 6.2 to 6.4. The proposed methods are compared to each other in Section 6.5. Afterwards, in Section 6.6 the proposed methods and several of the latest large-scale approaches from the literature are compared in terms of multiple performance criteria on a variety of benchmark functions. Section 6.7 examines the influence and effectiveness of interaction-based variable groups on the performance of large-scale algorithms. Finally, the evaluations are summarised and discussed in Section 6.8.

The dissertation thesis is summarised and concluded in Chapter 7 and an outlook on future research topics in this area is given.

Chapter 2

Basic Principles and Large-scale Optimisation

In this chapter, basic concepts in the area of multi-objective and large-scale optimisation are introduced. The following sections describe in detail the general terms of multi-objective problems (Section 2.1), Evolutionary Algorithms (Section 2.2) and aspects of large-scale optimisation (Section 2.3). Afterwards, two important concepts generally used in the process of solving large-scale problems are explained. Section 2.4 gives a formal definition of variable grouping mechanisms, which is used throughout this thesis in most of the related and the proposed approaches. The second concept is Cooperative Coevolution (Section 2.5), which initially motivated the use of variable groups in the optimisation area. Some of the most common benchmark suites for multi-objective and large-scale optimisation are introduced briefly in Section 2.6, and evaluation metrics that are used to measure the performance of algorithms in the experimental evaluation are described in Section 2.7. The last section of this chapter provides a short summary of the basic concepts.

2.1 Multi-objective Optimisation

As described above, real-world applications in nature and science often contain multiple conflicting objectives or goals. Such a problem is called a multi-objective problem (MOP). Mathematically, it can be formulated as shown in Eq. (2.1).

$$\begin{aligned} Z: \quad & min \quad \vec{f}(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), ..., f_m(\vec{x}))^T \\ & s.t. \quad \vec{x} \in \Omega \end{aligned} \tag{2.1}$$

This kind of MOP maps the decision space, also called search space, Ω of dimension n to the objective space $\mathcal{M}$ of dimension m, as exemplarily depicted in Fig. 2.1. It consists of m objective functions, sometimes also called fitness functions, which have to

be minimised or maximised simultaneously. In the remainder of this thesis, the terms *objective function* and *fitness function* are used synonymously. Furthermore, we assume without loss of generality that all objective functions have to be minimised. The decision space Ω is defined by the encoding of the optimisation task. In the case of a Travelling Salesman Problem (TSP), the search space might consist of all permutations of cities to define different orders of visiting them. Many problems are also modelled as integer or binary problems. In the context of this thesis, it is without loss of generality assumed that the search space is real and a subspace of the $\mathbb{R}^n$, constrained by a number of inequalities, i.e. $\Omega = \{\vec{x} \in \mathbb{R}^n | \, \vec{g}(\vec{x}) \leq 0\} \subseteq \mathbb{R}^n$. Moreover, most algorithms designed for multi-objective optimisation work with simple "box-constraints", which merely define a domain for each design variable in the form $x_{i,min} \leq x_i \leq x_{i,max}$, $\forall i \in \{1, .., n\}$. There are many constraint handling techniques in the literature which can be applied to existing algorithms to tackle more complex linear and non-linear constraints. These methods of constraint handling are, however, not the scope of this thesis. The interested reader is referred to [8, 9] for further information.

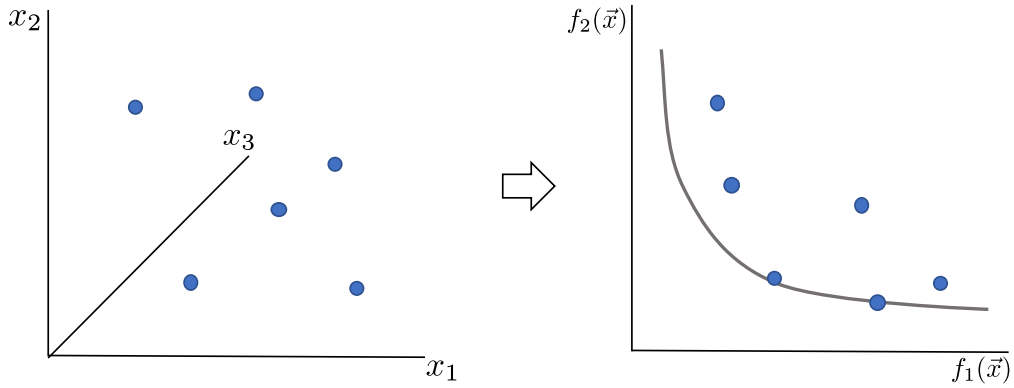


Figure 2.1: Exemplary visualisation of decision space and objective space. The solutions in the decision space (left) are evaluated into points in the objective space of the problem (right). A hypothetical optimal front is shown in as a grey line in the objective space.

There are a few additional challenges in MOPs in comparison to single-objective problems. First of all, due to the conflicting objectives, it is no longer possible to determine a single optimal solution as the algorithm's output. Instead, so-called Pareto-optimal solutions need to be found. The term Pareto-optimality in general refers to a situation where for an allocation of values to a set of variables, one can not improve the value of one of the variables without making at least one of the others worse. The term originates from the field of microeconomic theory and is used to describe certain allocations of goods. A Pareto-optimal allocation of goods to subjects is one where it is not possible to improve the utility, or wealth, of one subject without making another one worse [10].

This concept from economic theory has since been used in the case of multi-objective optimisation to induce a partial ordering on the objective function values [8, 11]. This is achieved by defining a domination criteria among solutions, the so-called Pareto-

dominance. Each solution in the objective space consists of a value for each of the objectives to be optimised, i.e. $\vec{f}(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), ..., f_m(\vec{x}))^T$ as denoted in Eq. (2.1). Based on this, a formal definition of Pareto-dominance and Pareto-optimality is given in Definitions 2.1 and 2.2, where it is assumed that - without loss of generality - all objective functions should be minimised.

Definition 2.1 (Pareto-dominance) *A solution $\vec{x} \in \Omega$ dominates another solution $\vec{y} \in \Omega$ in the Pareto-sense, if and only if the following two conditions hold: (1) for all objective functions, the image $\vec{f}(\vec{x})$ is at least as good as the image $\vec{f}(\vec{y})$ and (2) there exists one objective function where $\vec{f}(\vec{x})$ is better than $\vec{f}(\vec{y})$.*

$$\vec{x} \preceq \vec{y} \Leftrightarrow \forall j \mid f_j(\vec{x}) \leq f_j(\vec{y}) \wedge \exists j \mid f_j(\vec{x}) < f_j(\vec{y}), \quad j \in \{1, .., m\} \tag{2.2}$$

Definition 2.2 (Pareto-optimality) *Pareto-optimal solutions are all solutions $\vec{x} \in \Omega$, which can not be dominated by any other solution in the search space.*

$$\vec{x} \in \Omega \text{ is pareto-optimal} \Leftrightarrow \nexists\, \vec{y} \in \Omega \mid \vec{y} \preceq \vec{x} \tag{2.3}$$

The optimal solutions in the search space Ω form a so-called *Pareto-optimal set*, also called *Pareto-set* (PS), in which the different solutions of the problem represent different trade-offs between the objective functions. This is formally defined in Definition 2.3. Solutions in the PS do not dominate each other, and can not be dominated by any other solutions in Ω. Between the solutions in this set, no order is defined, as solutions might be better in terms of one objective, but worse in terms of another. The corresponding solutions in the objective space $\mathcal{M}$ are called the Pareto-front (PF), as defined in Definition 2.4.

Definition 2.3 (Pareto-set) *The Pareto-set PS of a multi-objective optimisation problem is the set of all Pareto-optimal solutions of that problem, i.e. all solutions that can not be dominated by other solutions.*

$$PS := \{\vec{x} \mid \vec{x} \text{ is Pareto-optimal}\} \tag{2.4}$$

Definition 2.4 (Pareto-front) *The Pareto-front PF of a multi-objective optimisation problem is the image of the Pareto-set of that problem, i.e. the set of all points in the objective space which are obtained by applying $\vec{f}(\cdot)$ to the solutions in PF.*

$$PF := \{\vec{f}(\vec{x}) \mid \vec{x} \in PS\} \tag{2.5}$$

As a result of this trade-off property of multi-objective problems, decision makers are likely to be interested in all different kinds of trade-off solutions that can be regarded as optimal for this problem. This enables them to make an educated choice of which solution

to their problem is implemented in the real environment. Therefore, most metaheuristic approaches aim to find a good approximation of the Pareto-optimal set of solutions. Usually, this set should be as close as possible to true Pareto-optimal solutions, while at the same time being as distributed as possible along the Pareto-front. In the context of multi-objective optimisation, these properties are often referred to as *convergence* and *diversity* of a solution set, and are explained in further detail below.

2.2 Population-based Metaheuristics

Even though some optimisation problems can be expressed in analytical form and solved by mathematical or exact means, this is in general not always possible or practical. Some problems can be formulated mathematically, but require exponential time to be solved with exact methods, while other problems might involve complex simulations, which can not be described analytically. For such problems, metaheuristic optimisation provides a way to obtain solutions with suitable solution quality and tractable computation time. Solutions computed with a metaheuristic are not guaranteed to be optimal, but may require a lot less computational resources.

In the area of metaheuristic optimisation, a variety of methods has been developed over the years, and some of them have been inspired by biological or physical processes. Some of the most prominent techniques might be Hill Climbing [12], Simulated Annealing [12], Ant Colony Optimisation [13, 14, 12], Evolutionary Algorithms [8, 12] and Particle Swarm Optimisation [15, 16, 12]. While the former two of these are representatives of local search mechanisms, especially the latter two methods are of interest in this thesis, as they are more suitable for global optimisation and belong to a group of so-called population-based metaheuristics. To generate solutions and find better ones, these techniques retain a set of solutions, called a population. Evolutionary Algorithms (EAs) are inspired by the biological evolution of species, and utilise adapted versions of natural selection, recombination and mutation of solutions to optimise problems. Particle Swarm Optimisation (PSO) algorithms, on the other hand, are inspired by the movement of swarms in nature, for instance swarms of birds. PSO treats the real-valued solutions $\vec{x}$ as coordinates of particles which move through the decision space. By utilising concepts like velocity and inertia of particles as well as attraction to other particles, new solutions for the problem are generated.

Evolutionary Algorithms approximate the Pareto-optimal set by gradually improving on the current solutions they maintain in their population. For that, basic principles of the evolutionary improvement processes in nature are adopted into the algorithm. The main idea is to construct new solution candidates by altering and recombining existing solutions, as nature does with species to adapt them to the environment. The key concept that makes improvement possible is the natural selection mechanism, also often called "survival of the fittest" based on the theory of evolution, published by Charles Darwin in 1859 [17].

In a first step of an EA, an initial random population is initialised. The solutions in this population are evaluated based on fitness function as described in Section 2.1. After that, a number of operations is carried out in a loop until a termination criterion is reached (usually referring to a certain solution quality or a computational budget). The basic function of an EA is outlined in Algorithm 1. To create new solutions, an Evolutionary Algorithm uses recombination and mutation operators to combine existing solutions. After these are evaluated, an environmental selection procedure determines which solutions are taken over into the next generation, i.e. the next iteration of the main loop of the EA. Since better solutions are favoured, promising parameter combinations in the solutions are expected to increase within the population over generations. Gradually the algorithm approaches better solutions until the optimal, or in the case of multiple objectives, Pareto-optimal solutions are reached.

Algorithm 1 Basic outline of an evolutionary algorithm

Input: Optimisation Problem Z
Output: Solution population P

```
1: P ← initial random population
2: evaluate(P)
3: while termination criterion not reached do
4:     P' ← matingSelection(P)
5:     Q ← recombination(P')
6:     Q ← mutation(Q)
7:     evaluate(Q)
8:     P ← environmentalSelection(P, Q)
9: end while
10: return P
```

A requirement for a suitable performance of an EA is the existence of a continuous fitness landscape. Recombinations (sometimes also referred to as crossover) of solutions and mutations are based on the assumption that solutions with similar fitness are also similar in their representation. In other words, two solutions with similar objective function values should be represented by similar combinations of the underlying decision variables. As a result, modelling optimisation problems for EAs should be done with this consideration in mind.

2.3 Large-scale Optimisation

The term large-scale optimisation usually refers to optimisation problems where different aspects of the optimisation problems are increased in dimensionality. It is used in different ways sometimes in the literature on metaheuristic optimisation, as an optimisation problem can have different “large-scale” aspects. Most commonly, the term “large-scale” refers to a large numbers of decision variables, while the actual number in the literature varies. Another area that has drawn increasing attention in the last years is the so-called “many-objective” optimisation. This term usually refers to multi-objective optimisation

problems with more than 3 objective functions, while bi- and tri-objective problems fall into the usual category of multi-objective optimisation. In this work, the term *large-scale* usually refers to a multi-objective problem with a large number of decision variables if not stated otherwise. Both kinds of high-dimensional problems are described shortly in the next two subsections.

2.3.1 Many-variable Optimisation

As mentioned above, the most common type of problem under the term *large-scale* are the ones with a large number of variables. In the literature, problems with different numbers of variables were considered as large-scale. Most often, any problem with more than around 100 variables can be called *large-scale*, and numbers between 100 and 5000 have been most common (refer to Section 4.1). In this section some challenges that arise in this kind of problem are described.

If the number of variables is increased, it becomes much more challenging for metaheuristic methods to search this high-dimensional space with a limited population size. Starting with the initialisation of the population, the limited population size only allows an exploration of a limited area of the search space. With increasing dimensionality, only a small portion of the search space can be explored by the algorithm. At the same time one might argue that the influence of genetic operators becomes smaller, especially the mutation. A standard parameter setting for the mutation rate in many algorithms is $1/n$, with n being the number of variables. Thus, only one variable is mutated at a given time in the expected case. The influence this has on the solution as a whole becomes smaller if there are, for instance, $n = 1000$ variables compared to a problem with only $n = 10$ variables.

The biggest challenge for large-scale algorithms is to explore the high-dimensional space with limited computational resources. The amount of solutions, and therefore function evaluations needed to thoroughly search a decision space, increases exponentially with the dimension of that space. However, it is not possible to increase the computational resources available for solving in the same way. Therefore, algorithms for such problems need to be able to find promising areas of the search space and exploit them with a small amount of function evaluations. A central aspect of almost all large-scale algorithms is therefore the reduction of the dimensionality in certain ways.

A concept used widely in single-objective large-scale optimisation is *Cooperative Co-evolution*, which divides the search space into multiple independent groups of variables and retains independent populations for each of them. This concept is explained in further detail in Section 2.5. An overview of existing multi-objective algorithms in this many-variable area is given in Chapter 3.

2.3.2 Many-objective Optimisation

The area of many-objective optimisation has drawn increasing attention in the last years. When the number of objectives is increased, most classical methods in multi-objective optimisation do not perform well. Due to the large number of non-dominated solutions in the population, the concept of Pareto-dominance, which is used in many algorithms, suffers from the fact that with increasing dimensions, all solutions in the population most likely are non-dominated to each other from a very early point on in the search process [18, 19]. Therefore, selection criteria like non-dominated sorting (e.g. in NSGA-II [11]) fail to create selection pressure towards better solutions and make the search less effective. To overcome this effect, most many-objective algorithms rely on reference directions and similar concepts which first appeared in the MOEA/D algorithm in the year 2007 [20]. Multiple algorithms build upon this concept, including algorithms like RVEA [21], MOEA/DD [22] and NSGA-III [23], and the research in the last years has led to a variety of many-objective optimisation methods. The area of many-objective optimisation which can perform well with 5, 10 or 15 objective functions simultaneously has been an ongoing research topic in recent years.

2.3.3 Variable Interaction and Problem Separability

For the reduction of dimensionality of the search space, a common technique is to divide the variables into groups. One way to perform this division is to use the interaction between decision variables. Given a MOP as defined in Eq. (2.1), the interaction between variables according to [24] and [25] is described as follows.

Definition 2.5 (Variable Interaction) *For each objective function $f_k(\vec{x})$, an interaction between two decision variables x_i and x_j is assumed if values a_1, a_2, b_1, b_2 exist, so that*

$$f_k(\vec{x})|_{x_i=a_1,x_j=b_1} < f_k(\vec{x})|_{x_i=a_2,x_j=b_1} \tag{2.6a}$$

and

$$f_k(\vec{x})|_{x_i=a_1,x_j=b_2} > f_k(\vec{x})|_{x_i=a_2,x_j=b_2} \tag{2.6b}$$

where

$$f_k(\vec{x})|_{x_i=a,x_j=b} = f_k(x_1, ..., x_{i-1}, a, x_{i+1}, ..., x_{j-1}, b, x_{j+1}, ..., x_n)$$

This formalises the idea that, for variables that do not interact, the order between two values $f_k(\vec{x})|_{x_i=a_1}$ and $f_k(\vec{x})|_{x_i=a_2}$ is independent of the value of the variable x_j [26, 27]. In other words, the influence of the variable x_i on the fitness function f does not depend on the choice for the value of x_j. If $f_k(\vec{x})|_{x_i=a_1}$ is smaller than $f_k(\vec{x})|_{x_i=a_2}$ for a certain value of $x_j = b_1$ (Eq. (2.6a)), but larger for another value of $x_j = b_2$ (Eq. (2.6b)), an

interaction between these variables exists, meaning that the value x_j influences which values of x_i are obtaining smaller (larger) fitness function values.

Objective functions which contain no interacting variables are called "separable" problems. For such problems, the optimal values for each variable do not depend on any other variables' values, making it possible to optimise the problem one variable at a time to obtain the global optimum. Therefore, instead of solving a (single-objective) n-dimensional problem, one could solve n 1-dimensional problems, making the task significantly easier. It can be of advantage to know the variable interactions of an optimisation problem beforehand to include this information when using coevolution or other variable-group-based methods. In real applications, such information can potentially be obtained through expert knowledge for a specific application. However, since this may not always be possible, some methods in the literature aim to identify interacting variables through a problem analysis step. Some of these are described in Section 3.3.3.

2.3.4 Variable Contribution

In contrast to the interaction of variables, which is a property of each single objective function, another topic arises when multiple objectives are concerned, which is the question of the contribution of a variable. As decribed, solutions to a multi-objective problem should not only be as close to Pareto-optimal solutions, but also cover the whole Pareto-set as completely and evenly as possible.

In the context of this thesis, the term "variable contribution" refers to the influence of a decision variable on the convergence and diversity of a solution set. More precisely, the question is whether the change of a variable value changes the corresponding objective function vector in a way that the solution moves closer to the Pareto-front, or in a way that it represent different areas of the Pareto-front.

To explain this concept, consider the following optimisation problem with $m = 2$ objective functions and $n = 2$ variables:

$$\begin{aligned} min \quad & f_1(\vec{x}) = x_1 + x_2 \\ & f_2(\vec{x}) = 1 - x_1 + x_2 \\ s.t. \quad & \vec{x} \in [0,1]^2 \end{aligned} \tag{2.7}$$

The Pareto-set of this problem consists of all solutions where $x_2 = 0$, as x_2 increases both objective functions simultaneously. Therefore, changing the value of x_2 results in a solution being closer or further away from any Pareto-optimal solution. This situation is depicted in Fig. 2.2, where it can be seen that a change in x_2 results in a decreased or increased distance to the optimal solutions. In contrast, for a good approximation of the whole Pareto-set, the values of x_1 need to be as diverse as possible throughout a solution population. We can see in Eq. (2.7) that changing the value of x_1 of a solution does not increase the closeness of this solution to the optimal ones, but rather discovers new

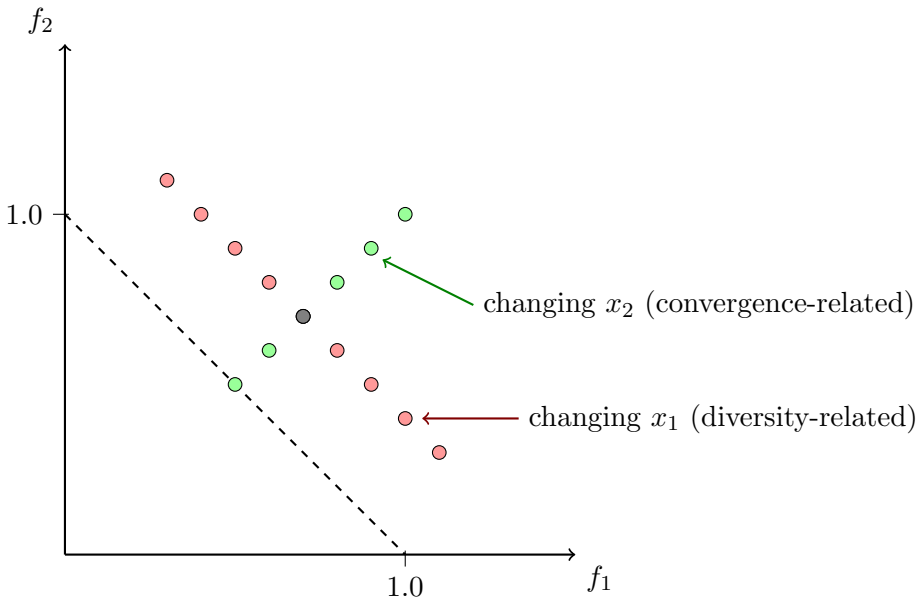


Figure 2.2: Visualisation of variable contributions. x_2 is convergence-related as changing it, while keeping x_1 fixed, results in solutions closer or further away from the true Pareto-front (shown as a dotted line). Similarly, x_1 is diversity-related.

solutions equally far away but representing different trade-offs between the objectives. This is depicted in Fig. 2.2, where changing the variable x_1 results in the created solutions moving "sideways", i.e. along the Pareto-optimal front, therefore creating more diverse solutions. In the case where $x_2 = 0$, changing x_1 results in the discovery of new optimal solutions. In this example, x_1 would be considered as a *diversity-related* variable, sometimes also called a "position-variable" in the literature. The variable x_2 is called a *convergence-related* variable, sometimes also called a "distance-variable", as it mainly contributes to the closeness to the Pareto-optimal areas of the objective space.

To test the ability of algorithms in terms of achieving good diversity and convergence, some test problems like, for instance, the WFG benchmark functions [28] provide a customisable parameter, with which the number of position- and distance-variables in the problems can be controlled. In large-scale optimisation, this concept of variable contribution was used in some related algorithms (e.g. MOEA/DVA [24] and LMEA[25]) that exploit these properties of variables in their search mechanic to create variable groups. These methods and their contribution detection mechanisms are examined in further detail in Sections 3.2 and 3.3.

2.4 Variable Grouping Mechanisms

A key feature of almost all large-scale algorithms is the division of the variables into a certain number of so-called *groups*. Creating variable groups, i.e. splitting the set

of variables of a problem into several smaller subsets, is most often motivated by the biological principle of Cooperative Coevolution (Section 2.5). Dividing the variables and applying optimisation to certain groups of them independently of other variables might also be seen as a divide-and-conquer approach.

The application of variable groups in optimisation plays an important role in this thesis and is formally described as follows:

Definition 2.6 (Variable Grouping Mechanism) *A Variable Grouping Mechanism* Γ *performs a segregation of decision variables* $x_1, .., x_n$ *of an optimisation problem* Z *into a number* γ *of groups* $G_1, .., G_\gamma$*. Formally,* Γ *provides a function* g *that assigns each variable index* $i \in \{1, .., n\}$ *to a corresponding group index* $j \in \{1, .., \gamma\}$*.*

$$\begin{array}{rcl} g : \{1, .., n\} & \rightarrow & \{1, .., \gamma\} \\ i & \mapsto & j \end{array} \tag{2.8}$$

As a result, the groups G_j *are defined as follows:*

$$G_j := \{i \,|\, g(i) = j\} \quad \forall j \in \{1, .., \gamma\} \tag{2.9}$$

The assignment of the variable indices to group indices can be done by an arbitrary mechanism, such as randomly, based on an analysis of the variables or based on the problem's properties (Section 3.3). This assignment may be associated with computational effort and involve statistical analysis or the creation of new solution candidates with corresponding function evaluations. In order to save computational resources, this computation of groups is usually precomputed and afterwards accessed via the function $g(\cdot)$ and the sets $G_1, .., G_\gamma$ respectively. Let $\hat{G}$ be the set of all created groups: $\hat{G} := \{G_j\}_{j=1,..,\gamma}$. The following notation is used to calculate the assignment and create the groups.

$$\{g, \hat{G}\} = \Gamma(Z, P) \tag{2.10}$$

The grouping mechanism Γ receives an optimisation problem Z and a set of solutions P as an input and provides the function g and the set of groups $\hat{G}$. Note that in some grouping mechanisms, such as random grouping, P can be an empty set, since no information about existing solutions is required.

2.5 Cooperative Coevolution

One of the most popular concepts for large-scale optimisation is *Cooperative Coevolution* (CC), which was first introduced into the area of optimisation by Potter and De Jong in

1994 [29]. Like evolutionary algorithms, CC is a nature-inspired method. Coevolution in evolutionary biology refers to a situation, where different species of individuals co-exist together in an environment, and the existence and actions of one species have an influence on the evolution of the other species. This can be expressed as "the change of a biological object triggered by the change of a related object" [30].

This principle of Cooperative Coevolution was adapted into the area of optimisation in the following way. A solution $\vec{x} = (x_1, ..., x_n)$ is seen as the state of a whole ecological system, i.e. as a combination of different species' individuals. Given a segregation of the variables into suitable groups, it is possible to create independent populations for each of these variable groups, each containing only values for the variables which belong to that group. This situation is exemplarily depicted in Fig. 2.3. The different groups of variables are also sometimes referred to as *species* or *subcomponents*, the populations for the groups sometimes as *subpopulations*.

Instead of optimising all decision variables using one population of solutions, a CC-based metaheuristic optimises one independent population for each variable group. The populations are usually optimised in turns, and genetic operators like crossover and mutation are only used on the currently optimised population, while the variables in remaining populations remain unchanged. The advantage of this approach is that the smaller groups of variables have a smaller search space than the whole problem originally had, which can be beneficial for the exploration within this group.

However, a solution from one of these populations can not be evaluated on its own, since the fitness functions of the problem can only be evaluated for complete solutions (i.e. which contain values for all variables). Therefore, it is necessary to combine the variable values from different populations to perform the function evaluation in the optimisation (hence *Cooperative* Coevolution). This cooperative function evaluation is depicted as an example in Fig. 2.4. After generating new solutions within the population of the first variable group, containing the variables x_1, x_2, x_3 and x_4, their values are combined with the values of other solutions from the population of group 2. The resulting solutions for the problem are evaluated and their objective function values are assigned to the created solutions in the population of group 1.

The concept of CC was used in a variety of large-scale algorithms, mostly in the single-objective area [31, 32, 33, 34, 35, 36, 37]. In later chapters of this thesis, we explore several algorithms which make use of this concept in the area of multi-objective optimisation (Chapters 3 and 4). A CC-based version of the basic EA from Algorithm 1 looks like shown in Algorithm 2. The main difference lies in the creation of multiple populations (Line 2), as well as the cooperative function evaluations in Lines 4 and 11. The different populations P_1 to P_γ, where γ is the number of groups, are optimised in turns, while the solutions in the respective populations of the other groups are left unchanged. For each function evaluation, variable values from each of the other $\gamma - 1$ groups are taken

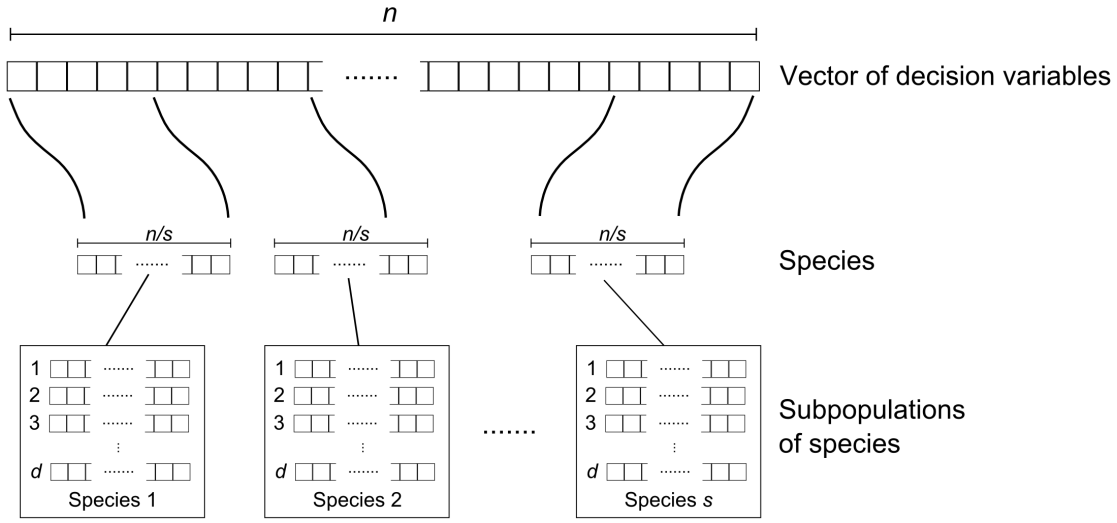


Figure 2.3: Creation of the species and their subpopulations out of the n decision variables in Cooperative Coevolution. Each of the independent populations contains d solutions, which only consist of values for the variables of the respective group. Illustration based on [3].

to form an individual and evaluate its fitness. One further change is that the output of the algorithm should ideally consist of a population of complete solutions to the problem instead of multiple smaller populations. This could be achieved either by an archive that stores the best evaluated solution combinations during the search, or by combining the subpopulations in a separate step after the optimisation is finished.

Algorithm 2 Basic outline of a Cooperative Coevolution-based evolutionary algorithm.

Input: Optimisation Problem Z
Output: Solution populations for each variable group $\{P_1, .., P_\gamma\}$

1: $\{g, \hat{G}\} = \Gamma(Z, \varnothing)$
2: $\{P_1, .., P_\gamma\} \leftarrow \gamma$ initial random populations with variables in G_γ respectively
3: **for all** P_j, $j \in \{1, .., \gamma\}$ **do**
4: $\quad evaluate(P_j \,|\, P_1, .., P_{j-1}, P_{j+1}, .., P_\gamma)$
5: **end for**
6: **while** termination criterion not reached **do**
7: $\quad$ **for all** P_j, $j \in \{1, .., \gamma\}$ **do**
8: $\quad\quad P'_j \leftarrow matingSelection(P_j)$
9: $\quad\quad Q_j \leftarrow recombination(P'_j)$
10: $\quad\quad Q_j \leftarrow mutation(Q_j)$
11: $\quad\quad evaluate(Q_j \,|\, P_1, .., P_{j-1}, P_{j+1}, .., P_\gamma)$
12: $\quad\quad P_j \leftarrow environmentalSelection(P_j, Q_j)$
13: $\quad$ **end for**
14: **end while**
15: **return** $\{P_1, .., P_\gamma\}$

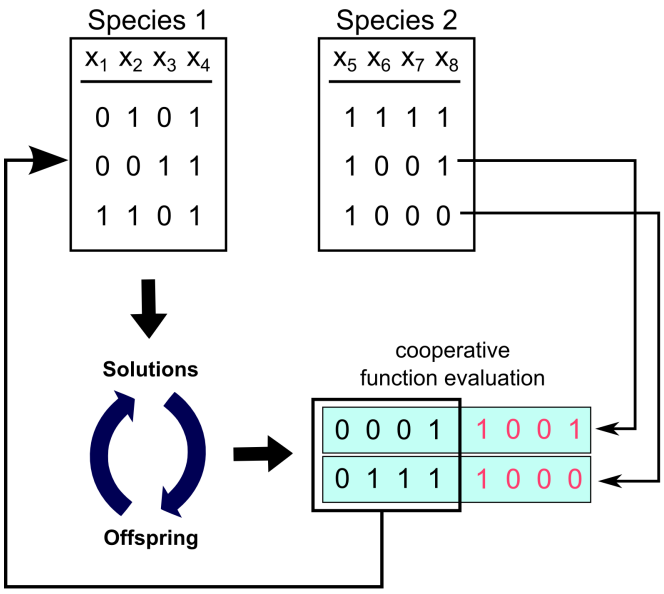


Figure 2.4: Principle of Cooperative Coevolutionary fitness evaluation. Graphic taken from the author's contribution in [38].

The main challenge that arises is the question on how to find suitable "partners" from the other populations for the evaluation of a solution. Since the fitness values of an incomplete solution in one of the populations may vary depending on which values it is combined with, this choice has an effect on the search process. If a more robust function evaluation is preferred, a solution could be evaluated multiple times with different partners to obtain a better understanding of the average fitness of this specific parameter combination in the current population. However, this results in increased computational effort.

In single-objective optimisation, since there is only one fitness function, it can be promising to choose the respective best or worst individuals from the other populations. However, in multi-objective problems, since there is no total order on the solutions, the appropriate choice of evaluation-partners is more difficult. This is especially visible if we consider that a multi-objective algorithm aims to achieve high diversity as well. Two different combinations with variables from groups that contain diversity-related variables might result in completely different fitness function values for all of the objective functions. This makes traditional CC harder to apply for multi-objective problems.

2.6 Benchmark Problems for Multi-Objective Optimisation

This section gives a brief overview of existing multi-objective benchmark problems, including the ones used in the evaluation of this thesis. Most of these benchmarks have

been widely used in the multi-objective literature for many years. The problems include a variety of different characteristics, which allows testing algorithm performance for several different types of problems. In the literature, a variety of scalable test problems has been introduced to evaluate the performance of multi-objective metaheuristic algorithms. Some of these are scalable in terms of the number of variables (like for instance the ZDT problems [39]) while others like the DTLZ [40] or WFG [28] are scalable both in terms of the number of variables and objectives [41].

The detailed mathematical description of each problem family is not shown here. Instead we aim to briefly summarise the different benchmark families in terms of scalability and other interesting characteristics. The DTLZ, WFG, UF and LSMOP suites are also used in the evaluation of this thesis (Chapter 6) due to their scalability and complexity. For a detailed overview and analysis of shortcomings and properties of current benchmarks, the reader is referred to [42, 28, 43, 44].

Some of the oldest benchmark problems in the multi-objective community are the ZDT benchmark problems [39], which were proposed in the year 2000. Six different benchmarks were proposed, each with 2 objective functions and, in their original form, 30 decision variables. While ZDT1-4 and ZDT6 are continuous problems, ZDT5 is a binary optimisation problem, which is less often used in the literature. The ZDT problems possess different properties, including convex, concave and disconnected Pareto-fronts. However, their structure is rather simple compared with current state-of-the-art test problems. The problems contain one variable whose values decide about the distribution of solutions along the Pareto-front, while the rest of the variables are used to determine the closeness to the front. In modern terms, as an analysis in [25] showed, all but one variable of the ZDT problems are convergence-related variables. A major drawback of the ZDT functions are also the inability to scale the number of objectives. In recent years, with the increase of many-objective literature, these problems are not used often any more, because the development of modern evolutionary algorithms lead to sophisticated methods, and the complexity of the ZDT functions does not pose a challenge to these algorithms any more.

One of the most common benchmark families which is still used in a variety of works are the DTLZ functions [40]. These problems were proposed to overcome certain shortcomings of the previous ZDT suite. The DTLZ problems, named after the initials of their four creators, improve on some of the shortcomings of the ZDT problems. They are scalable in the number of variables and objective functions, and provide more complex structures. They are still used often in recent studies for multi- and also many-objective algorithm design. Modifications of the DTLZ functions also serve as the basis for some of the MaF benchmark problems, which are used in the many-objective competitions of the IEEE Conference of Evolutionary Computation (CEC) in recent years [45, 46].

Table 2.1: Properties of the nine LSMOP benchmarks as listed in [44].

	Modality	Separability
LSMOP 1	Unimodal	Fully Separable
LSMOP 2	Mixed	Partially Separable
LSMOP 3	Multi-modal	Mixed
LSMOP 4	Mixed	Mixed
LSMOP 5	Unimodal	Fully Separable
LSMOP 6	Mixed	Partially Separable
LSMOP 7	Multi-modal	Mixed
LSMOP 8	Mixed	Mixed
LSMOP 9	Mixed	Fully Separable

Huband et al. proposed the *Walking Fish Group* benchmark suite [28]. The WFG benchmarks are the first problems to possess a parameter to set how many of the total variables are related to convergence and how many are related to diversity. This property has rarely been used in the literature, as they were mostly used in their standard configuration with a very low number of diversity-related variables. Some studies, however, made use of this scalability and studied the WFG functions with high numbers of diversity-related variables [47, 4, 1, 5, 26]. The WFG benchmarks also possess a variety of features including degenerated fronts in WFG3, disconnected fronts in WFG2, convex, concave and mixed Pareto-front shapes and others.

The UF problem suite was proposed as part of the competition on multi-objective optimisation at the IEEE Conference of Evolutionary Computation 2009 [48]. It consists of 10 problems, were the UF1-7 problems are bi-objective, the WF8-10 are tri-objective problems. Originally, these problems were specified with 30 decision variables, but are scalable in the number of variables freely. A recent study in 2018 [47] showed that these problems are especially hard in terms of achieving a good diversity along the optimal front.

Distance Minimisation Problems (DMPs) have been introduced in the literature as scalable test problems which can be easily visualised in the objective space [49, 50, 51, 52, 18, 53]. In a DMP, several predefined objective points are defined in the decision space. Each one of these points corresponds to one of the objective functions, i.e. the number of objective points is the same as the number of objectives. The goal is to find the solutions in the decision space which have the minimum (Euclidean) distances to all the objective points [41]. Although the problems are relatively simple and all its objective functions are separable, a big advantage is that these problems can, to a certain degree, be seen as related to real application in logistics and location planning [54, 55]. DMPs can easily be visualised even when many-objective instances are used, and were used in many variations to include multi-modality [56], constraints [57] or dynamics in the problem [41, 58]. Other work has also focused on changing the complexity through the introduction of Manhattan

distances, although the Pareto-optimal areas of these problems are no longer easy to compute [38, 59, 60].

In [61], an optimisation problem based on a real world application was introduced. The multi-objective version of this problem was used in [62] and [63] with up to 4864 decision variables. A drawback of this problem is that despite its large number of variables, the optimal solutions are not known, which makes it difficult to compare the performance of algorithms with certain indicators. Furthermore, the related work in [62] solved the problem using derivative information from the objective functions, which is usually not applicable when using metaheuristic algorithms for black-box problems.

Recently, a set of special problems, called LSMOP (Large-scale many-objective problems), was introduced [44], which are specifically designed to test the search abilities of algorithms in large-scale and many-objective optimisation. The LSMOP benchmarks are scalable both in terms of objective functions and decision variables. The suite further enables to specify the separation into groups and the interactions between the groups beforehand, which was not possible in some of the previous benchmark functions [6]. The nine LSMOP benchmarks proposed in the work are used in many of the recent large-scale publications in the multi-objective area. Some of its properties are summarised in Table 2.1.

2.7 Evaluation Metrics

Compared to single-objective optimisation, the results of multi-objective algorithms can not be evaluated by simply comparing the achieved fitness values. The challenge of multi-objective problems is to find solution sets which propose a trade-off between the objective functions and lie as close to the Pareto-optimal solutions as possible. Therefore, the obtained solution sets of algorithms are often compared with certain metrics, also called performance indicators, which map a set of solutions to a single number that allows a comparison between algorithms. Different indicators can be used to measure the diversity and convergence of a solution set. This section briefly describes a selection of evaluation metrics for multi-objective optimisation algorithms. The focus lies mainly on the Hypervolume indicator and the IGD indicator, which are both used later in the experimental evaluation of this thesis (Chapter 6).

An indicator that is able to measure the convergence of a solution set is the *Generational Distance* (GD) [64]. The GD requires a reference set, usually assumed to be a sample of the true Pareto-front. It computes, in the objective space, the average of the shortest Euclidean distances from each point in the obtained solution set to its closest point in the Pareto-front sample. In this way, the GD can provide information on how close the obtained solutions are to the optimal ones, but the GD can not make any statement about the distribution of the solutions. In the extreme case, all solutions could be concentrated on a very small part of the PF of the problem, and still achieve GD values close to zero.

In order to measure the diversity of a solution set, the *Inverted Generational Distance* (IGD) indicator is often used in the literature [65]. As for GD, the IGD metric requires a reference set, i.e. a sample of the true PF. It computes the average of the distances from each point in this PF sample to the closest obtained solution. This is defined formally in Definition 2.7.

Definition 2.7 (IGD) *Let P be a set of Pareto-optimal points in objective space and S be a set of solutions obtained by an optimisation algorithm. The IGD indicator of the set S with respect to P is defined as*

$$IGD(S,P) = \frac{1}{|P|}\left(\sum_{\vec{p}\in P}\min_{\vec{s}\in S} d(\vec{p},\vec{s})^q\right)^{1/q} \tag{2.11}$$

where $d(\vec{p},\vec{s})$ is the Euclidean distance between the two points $\vec{p}$ and $\vec{s}$.

In the remainder of this thesis, the value for q in the IGD definition is set to 1, following the common setting in the literature [66].

Even though the IGD is intended to serve as a diversity-related indicator, the IGD value of a set of solutions can provide information about convergence and diversity. In order to obtain a small IGD value, the solutions need to be distributed along the PF. However, a well-distributed set of solutions far away from the PF still results in large IGD values. Therefore, even though small IGD values can result from good convergence or good diversity, the best IGD values close to zero are only obtainable through a converged and well-distributed set. For this reason, the literature in the large-scale and many-objective area has often used the IGD indicator to compare the overall performance of algorithms [67, 68, 69, 24, 25, 70, 71].

Different extensions to the IGD have been proposed in the literature, for instance the IGD+ metric [72] or the average Hausdorff distance [73]. Studies on the IGD in comparison with its variants can, for instance, be found in [66, 74].

The Hypervolume (HV) indicator [75, 76] is one of the most frequently used metrics in the literature, and has, next to the IGD metric, been used in a variety of large-scale publications [68, 69, 77, 78, 79, 3, 47]. The HV can measure the diversity as well as the convergence of a solution set, and does so with respect to a reference point in the objective space as defined in the following.

Definition 2.8 (Hypervolume) *Let S be a set of solutions obtained by an optimisation algorithm and $\vec{r}\in\mathbb{R}^m$ be a reference point in the objective space. Let $S'\subseteq S$ be the set of solutions which dominate $\vec{r}$ in the Pareto-sense. The Hypervolume (HV) indicator of*

the set S with respect to $\vec{r}$ is defined as

$$HV(S, \vec{r}) := \bigcup_{\vec{s} \in S'} vol(\vec{s}, \vec{r}) \tag{2.12}$$

where $vol(\vec{s}, \vec{r})$ denotes the volume in the objective space spanned between $\vec{s}$ and $\vec{r}$.

Assuming that all objectives need to be minimised, the reference point is usually required to be greater than the found solutions in each dimension of the objective space. The closer the solutions are to the true PF, the further away they are from the reference point and thus the HV value increases. The correct choice of the reference point is not trivial, since a reference point too close to the PF excludes some solutions from contributing to the HV. On the other hand, a reference point too far away from the PF strongly emphasises the role of convergence of solutions, since a large HV value can be achieved by just one single optimal solution. In the literature, reference points are often obtained by using the worst values in each dimension (nadir point) of a given solution set, for instance a sample of the true PF or the set of obtained solutions.

An advantage of the HV compared to GD or IGD is that it does not require a sample of the true PF, and can therefore also be used when such optimal solutions are not available, for instance in real applications. Due to that property, the HV was also used inside of optimisation algorithms, so-called indicator-based algorithms. Since the maximisation of the HV results in solutions close to and distributed along the PF, this indicator can be used as an objective of the optimisation process to find good solutions. However, the computational complexity of calculating the exact HV rises with increasing numbers of objectives.

The scale of the HV indicator depends on the scale of the objective functions and the choice of the reference point. To solve this issue, some implementations normalise the objective functions values in each dimension (using the minima and maxima for each objective) before calculating the HV. In the remainder of this thesis, the term *Hypervolume* refers to this normalised HV version. Furthermore, to obtain values which are easier to compare, some work in the literature uses a relative version of this indicator, called the *relative Hypervolume* or the *Hypervolume rate*. In this version, the computed HV value is divided by the maximally achievable Hypervolume, given the chosen reference point.

2.8 Summary

This chapter presents basic principles of multi-objective optimisation and introduces several concepts that are required in the remainder of this thesis. A brief overview on multi-objective optimisation and its formal definition, along with Pareto-optimality and the related concepts, are given. The principles of population-based metaheuristics are given, and the functionality of evolutionary algorithms is explained.

The following sections deal with the special properties and challenges of large-scale multi-objective optimisation. The terminology of large scale and many-objective optimisation is introduced and the concepts of variable groups and the different roles of variables in terms of interaction, convergence and diversity are explained. Cooperative Coevolution, which is commonly used in many large-scale methods, is introduced.

After that, a brief overview is given about the existing benchmark suites which exist in the literature and which are commonly used in the scientific community for designing and comparing algorithms. Many of them are scalable in the number of objectives and variables. Finally, a brief description of different evaluation metrics from the literature is given, and the used metrics for the later experimental evaluation (HV and IGD) are formally defined.

Chapter 3

Related Work

In the recent years, large-scale optimisation has drawn increased attention in the scientific community. While the large-scale single-objective literature has grown in the last decade, the amount of research and algorithms developed exclusively for multi-objective large-scale problems remains sparse, with the majority of large-scale algorithms published since the year 2016.

This chapter gives an overview of the related literature on large-scale multi-objective optimisation. In the following, Section 3.1 gives an overview of the literature on large-scale algorithms that have been developed in recent years. In Section 3.2, the related large-scale multi-objective algorithms, which are the focus of this thesis are used in subsequent chapters, are explained in detail. Section 3.3 presents a selection of related grouping mechanisms, which have been developed in the single- and the multi-objective area. Finally, Section 3.4 provides a short summary of the described algorithms and grouping mechanisms of this chapter.

3.1 Overview of the State of the Art

The aim of this section is to provide a brief overview of existing methods to solve large-scale problems, both in single- and in multi-objective optimisation. Even though a variety of large-scale algorithms for single-objective optimisation have been developed in recent years, the focus of this thesis lies on the multi-objective area. Therefore, only a brief summary of single-objective methods is given in this section. Further information about large-scale optimisation algorithms in the existing single-objective literature is, for instance, found in [80]. Large-scale optimisation has also drawn interest in the area of exact methods, like for instance in [81], however, the focus of this thesis lies on metaheuristic approaches. In the following we give a short review on related single-objective works (based on the author's article in [1]) that have been influential to the multi-objective area, followed by a brief overview of multi-objective large-scale approaches.

Single-objective Large-scale Optimisation

As described in Section 2.5, the most prominent concept that led to the development of large-scale algorithms is Cooperative Coevolution. The idea of a division of the decision space variables in smaller groups of variables, and optimising these independently with an evolutionary algorithm, was originally proposed by Potter and De Jong in 1994 [29] for single-objective optimisation. Later work by Potter et al. further studied a method for dynamically evolving the variable groups needed in the CC framework [31]. The concept of CC was since used in a variety of large-scale single-objective algorithms [31, 32, 33, 34, 35, 36, 37].

In 2010, Chen et al. used a separate step prior to the optimisation process to determine the segregation of the variables into groups for non-separable problems [36]. They took into account the interaction of variables with a learning mechanism for finding the optimal distribution of variables to the subcomponents. They reported good results compared to a naive grouping of the variables, although the additional learning step consumed more computational resources. A study on this tradeoff and impact of Variable Interaction Learning on Cooperative Coevolutionary algorithms was later conducted by the same authors in 2013 [82].

In 2008, Yang et al. [33] proposed a CC method for single-objective problems using two special features. One was a repeated reassignment of the variables into subcomponents in every iteration of the algorithm. The second was a weighting scheme to optimise a so-called "weight" for each subcomponent, i.e. apply a multiplication with a certain value to every variable in the same group. These "weights" were then evolved with a metaheuristic for the best, worst and a random member of the population. The (re-)grouping in this work was done at random, i.e. in each iteration of the algorithm, new random groups were created [1]. This approach later served as an inspiration for the development of the Weighted Optimisation Framework [4, 1], which is one of the proposed methods in this PhD thesis (see Section 5.1).

The same principle of using CC with weights has then been used in other works [34, 83] for single-objective problems, using benchmark functions of up to 1000 decision variables. A mechanism for choosing appropriate lower and upper bounds of weights was proposed in [34]. In a later study, Li et al. stated that using this weighting approach in CC is less effective than improving the frequent (re-)grouping of variables [37, 1].

In 2014, a mechanism called "Differential Grouping" (DG) was introduced to find improved distributions of the variables in single-objective CC algorithms [84]. The aim of DG was to determine variable interactions to base groups not only on random assignments, but on the information which variables should be optimised together in the same group. A more detailed description of DG and its successor, DG2, is given later in Section 3.3. Other concepts and extensions to this approach for variable grouping were proposed [85, 86, 87].

A single-objective competitive PSO algorithm [88] was developed in 2015 which showed good results for large-scale benchmark problems with up to 5000 decision variables [1]. Further, CC was, for instance, applied to a large-scale version of the capacitated arc routing problem, with a special variable grouping mechanism that made use of the information of the routes found the algorithm in previous iterations [89].

Multi-objective Large-scale Optimisation

In contrast to single-objective problems, the popularity of research on multi-objective large-scale optimisation has only increased within the last few years. In the multi-objective case, the additional challenge of exploring a high-dimensional search space is even more challenging when multiple areas of the search space need to be found to cover different parts of the Pareto-optimal front of the problem. A study in 2013, for instance, showed that the performance of existing algorithms deteriorates when the dimension of the search space is increased [90].

A first approach to utilise the concept of CC in multiple objectives was performed by Iorio and Li [32] in 2004. This approach makes use of the NSGA-II algorithm, but was not developed as a dedicated large-scale optimiser, and as such was not tested on any high-dimensional search spaces. The algorithm optimises each decision variable on its own, utilising its own population consisting only out of values for this one variable. The ZDT test problems (refer to Section 2.6) were used with 2 objectives and only a small number of up to 30 decision variables. The results showed that their algorithm was able to compete with the performance of the NSGA-II in most of their experiments.

To the best of the author's knowledge, the first dedicated multi-objective large-scale algorithm was proposed in 2013 by Antonio and Coello Coello, called CCGDE3 [3]. Their work used the concept of CC together with the Generalized Differential Evolution 3 algorithm (GDE3 [91]). In their experiments, using some of the 2-objective ZDT benchmark functions with between 200 and 5000 decision variables, the coevolution-enhanced version of GDE3 outperformend the traditional GDE3 and NSGA-II algorithms. However, to achieve good approximations of the PF, the CCGDE3 method still required a large number of function evaluations, ranging between $150,000$ to $220,000$ for the 1000-variable instances and up to over $5,700,000$ evaluations for 5000-variable problems. Furthermore, although the study showed that the concept of CC can be applied to multiple objectives, it was only tested on the ZDT functions. It remained unclear whether this concept would work with more complicated benchmark problems like the WFG or the DTLZ benchmarks.

3.2 Related Approaches in Large-scale Multi-objective Optimisation

Following the proposal of CCGDE3, a variety of algorithms was proposed to tackle large-scale multi-objective optimisation, most of them published since the year 2016. In this section, the existing algorithms in this area are described in detail. The following list of methods includes all known large-scale multi-objective algorithms that were published until the end of March 2019. For each of the methods, the concept is described and its methodology is visualised using simplified flowcharts. These flowcharts represent abstract workflows of each method, and are composed of certain *building blocks*, which the author identifies for each algorithm and which are used in the next chapter for further analysis of the state-of-the-art. The algorithms in the following are presented in order based on similar characteristics. They are further summarised, compared and categorised in detail in Chapter 4. Further information on the experimental evaluations in the respective publications is presented in Appendix A.

CCGDE3

The CCGDE3 applied the concept of CC to multi-objective optimisation [3]. The basic structure of the algorithm is shown in Fig. 3.1. CCGDE3 first divides the variables randomly into groups and afterwards applies a CC-based optimisation of the large-scale problems. This is done through maintaining independent populations of solutions for the different variable groups in the same way described in Section 2.5. This is done until a termination criterion is met. As described, one major problem when using CC with multiple objectives is to select the "partners" for each individual of the distinct populations to perform the function evaluation step. This is not trivial since the notion of "best" and "worst" individual in the population is not defined as in the single-objective case. To evaluate an individual in one of the populations in CCGDE3, in the first iteration of the algorithm, random partners are selected from each of the other population, and the obtained objective function values are assigned to the individual at hand. In the subsequent iterations, random individuals from the respective first non-dominated fronts of each of the populations are used. The groups in CCGDE3 were created randomly and evenly-sized. A potential disadvantage of the CCGDE3 algorithm is the initial random creation of variable groups. Since the algorithm does not to change the groups during runtime, CCGDE3 is potentially vulnerable to problems with high interactions between variables.

MOEA/D^2

Another approach based on CC was released in 2016 and named MOEA/D^2 [79]. It follows mostly the same ideas and structure as the CCGDE3 algorithm, as shown in Fig. 3.2. It first divides the decision variables randomly into a number of subcomponents / groups in the same way as CCGDE3. However, the MOEA/D^2 does not create completely

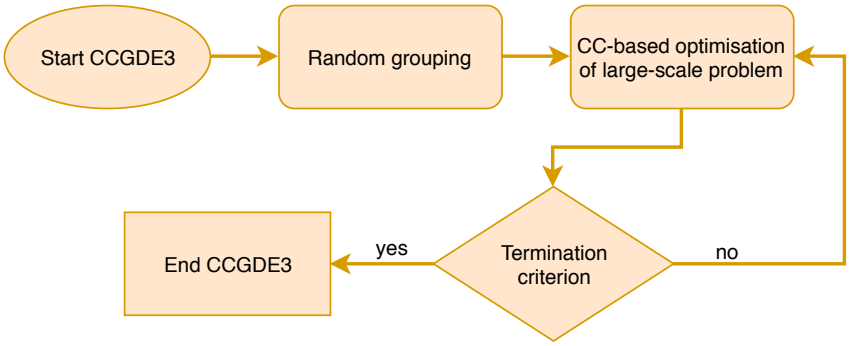


Figure 3.1: Outline of the CCGDE3 algorithm.

independent subpopulation for each of these subsets of variables. Instead, only one population is maintained, where during the optimisation of each group, the genetic operators are only applied to the variables in that specific group. This eliminates the need to find suitable partners for each created solution-part in order to form evaluable solutions. The technique was incorporated into the MOEA/D [20] algorithm, which is an evolutionary algorithm with a decomposition technique in objective space. An external archive of solutions was used based on Pareto-dominance to return the best (i.e. non-dominated) solutions at the end of the search.

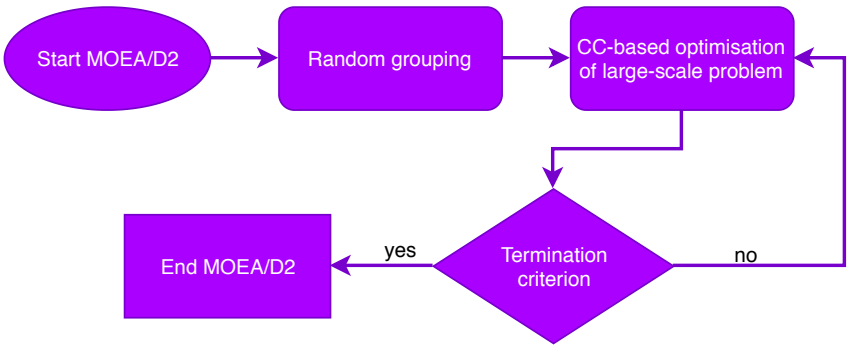


Figure 3.2: Outline of the MOEA/D^2 algorithm.

CCLSM

In 2018, Li and Wei proposed a CC-based method for large-scale multi-objective optimisation called CCLSM [70]. The work basically applies a CC structure with a grouping method that takes into account the interaction between decision variables. It therefore closely resembles CCGDE3 and MOEA/D^2, with the difference of applying an interaction-based grouping strategy from the single-objective literature, as is depicted in Fig. 3.3. For the selection of variables from the other groups' populations, a "representative" solution

is chosen and updated in each population, which values are used to combine solutions in the evaluation steps.

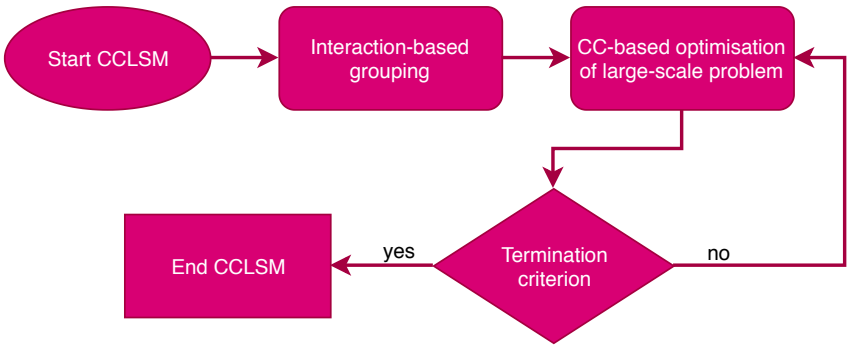


Figure 3.3: Outline of the CCLSM algorithm.

MOEA/D(s&ns)

An approach called MOEA/D(s&ns) was proposed in [92] in 2018 based on CC and the MOEA/D-DE algorithm [93]. In MOEA/D(s&ns), a CC-version was used to divide the population into smaller subpopulations. MOEA/D(s&ns) is included in this list for the sake of completeness, although it must be noted that due to the article's scientific quality, the author of this thesis was not able to fully comprehend the algorithm structure, focus of the work, nor experimental methodology of this article.

MOEA/DVA

In 2015, a large-scale optimisation algorithm called MOEA/DVA (*Multi-objective Evolutionary Algorithm based on Decision Variable Analysis*) was proposed by Ma et al. [24]. Its concept was similar to that of CC, mainly dividing the variables into multiple groups and optimise each of them independently. However, there are some differences to the classical concept of CC as it was in CCGDE3. MOEA/DVA did not choose mating partners at random any more, but instead kept individuals for the original problem together during the optimisation. This means, instead of keeping independent subpopulations, MOEA/DVA uses a variant of CC where only one population exists, and evolutionary operators are only applied to a certain group of variables at a time. Due to this, the need to find partners from other populations became unnecessary, as each individual of the population already contains values for all other variables in other groups (i.e. those variables currently not under optimisation).

A new concept that was first introduced in MOEA/DVA - and subsequently used in other methods - was the additional division of variables into groups based on their contribution to convergence (i.e. distance to the Pareto-front), diversity, or both (see

Section 2.3.4). The details of the used grouping techniques are described later in more detail, see Section 3.3.2.

The basic outline of this algorithm is shown in Fig. 3.4. After the algorithm first separates the variables based on their contribution (contribution-based grouping), MOEA/DVA then further divides the convergence-related variables into groups based on their interactions prior to the start of the optimisation (interaction-based grouping). This step is called *Interdependence Analysis* and is described in further detail in Section 3.3.3. For variables which are classified as mainly contributing to the diversity, a uniform initialisation procedure is used, and their values remain fixed after that, while the convergence-related variables are optimised. The groups of only convergence-related variables are optimised in the described CC-inspired way. This is done until a first termination criteria is reached, which is shown in Fig. 3.4 as "convergence detection". The algorithm measures the progress with a utility value, computed from the sum of objective values over the population. Once it receives a sufficient convergence, a so-called *uniformity optimisation* is carried out, which optimises the original, large-scale problem as a whole without using groups, and also includes the diversity-related variables to obtain a better spread of solutions along the Pareto-front [6].

Although MOEA/DVA takes into account the special requirements of multi-objective optimisation, i.e. the different goals of diversity and convergence during the search, it comes with certain disadvantages. Mainly, the variable interaction analysis to form the groups requires a major share of the available computational budget. In a study in [1], MOEA/DVA needed more than $8,000,000$ function evaluations for computing the groups of a 1000-variable problem [6]. The necessary evaluations vary depending on the used parameters, but rise quadratically with increasing numbers of decision variables. Further details can be found in Sections 3.3.3 and 4.2.

MOEA/D-RDG

A variant of MOEA/DVA was proposed called *MOEA/D with Random-based Dynamic Grouping* (MOEA/D-RDG) [67]. The basic structure is outlined in Fig. 3.5. MOEA/D-RDG uses the same mechanisms as MOEA/DVA to divide the variables based on their contributions and for the CC-based optimisation of the convergence-related variables. The difference between MOEA/DVA and MOEA/D-RDG lies only in the used grouping mechanism to perform the interaction-based grouping, where the newly proposed RDG strategy replaces the Interdependence Analysis in MOEA/DVA. Further details can be found in the description of the grouping mechanisms in Section 3.3.3. The big advantage of RDG compared to the Interdependence Analysis in MOEA/DVA is that no additional function evaluations are needed, therefore freeing more computational budget for the actual optimisation process.

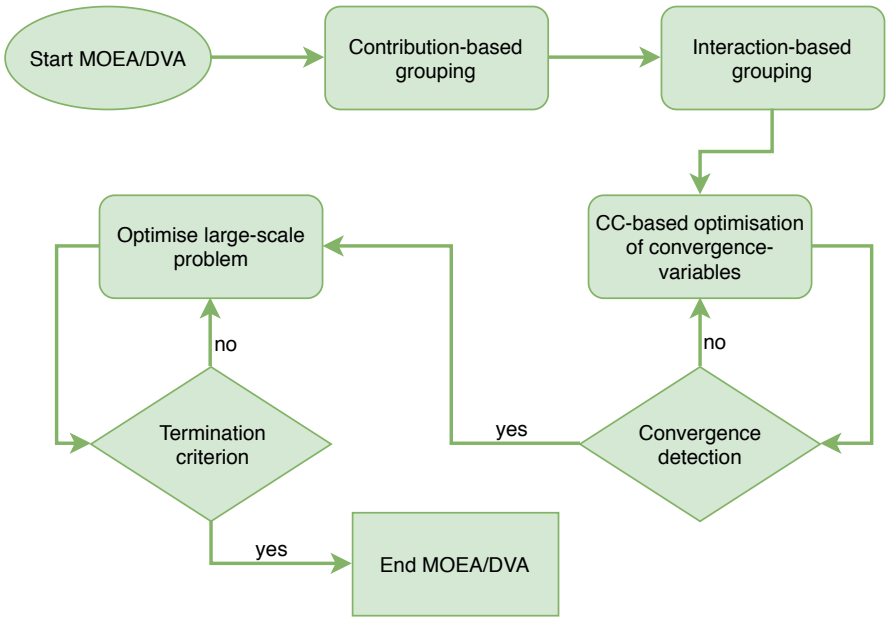


Figure 3.4: Outline of the MOEA/DVA algorithm.

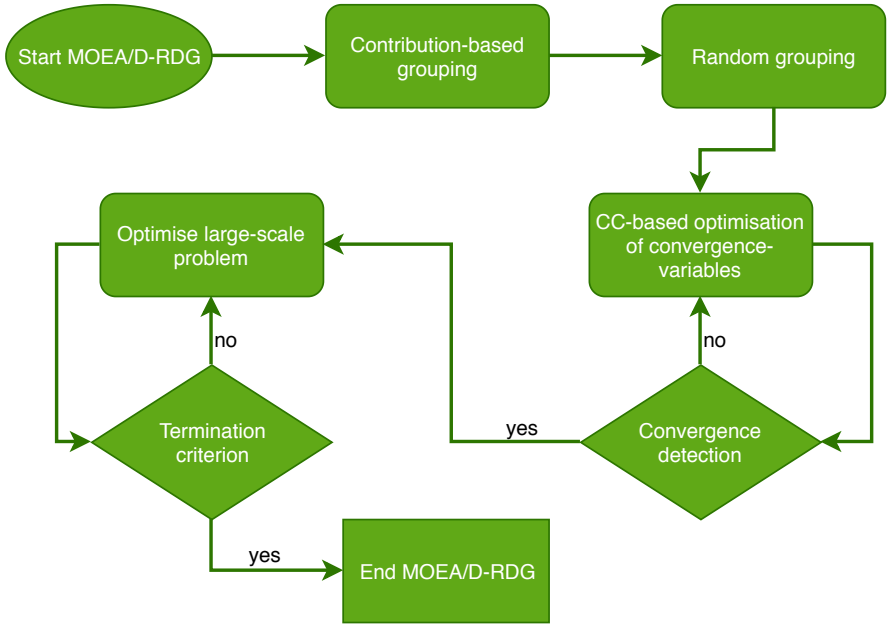


Figure 3.5: Outline of the MOEA/D-RDG algorithm.

LMEA

Another approach that uses a similar idea as MOEA/DVA is the *Large-scale Many-objective Evolutionary Algorithm* (LMEA) [25], published in 2016. As the name suggests, LMEA aims to solve problems with both, many variables and many objectives by using techniques from both areas in one algorithm. The structure of LMEA is depicted in Fig. 3.6. Similar to MOEA/DVA, LMEA works by first performing a contribution-based grouping, dividing the variables into two distinct groups, named convergence-related and diversity-related variables. To do so, it relies on a clustering-based approach (for details see Section 3.3.2). The convergence-related groups are then, similar to MOEA/DVA, divided further into several groups by taking into account the interaction between variables (for details see Section 3.3.3). These convergence-related groups are optimised in a CC-inspired fashion, and in a next step the diversity-related variables are optimised. This is done in turns until a termination criterion is met, using two different optimisation methods for the convergence-related groups and the diversity-related variables. An advantage of this method is that it includes the diversity-related variables during the optimisation more frequently compared MOEA/DVA, which only considers the diversity-related variables towards the end of the optimisation. However, similar to MOEA/DVA, LMEA requires a very large amount of function evaluations to obtain the interaction-based variable groups [6].

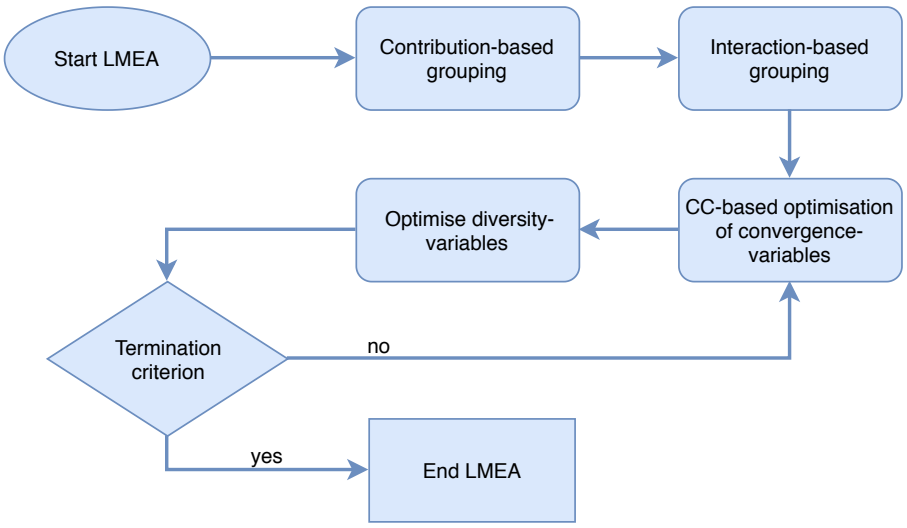


Figure 3.6: Outline of the LMEA algorithm.

DPCCMOEA

Cao et al. in 2017 proposed the *Distributed Parallel Cooperative Coevolutionary Multiobjective Evolutionary Algorithm for Large-Scale Optimisation*, abbreviated as DPCCMOEA [68]. This approach makes use of the CC framework and implements it in a distributed

fashion, so that each subpopulation contains values for all variables, but only those in the respective group are optimised by each separate process.

The basic flow of the algorithm is depicted in Fig. 3.7. For readability, the functional components are shown, and the parallel nature of the process is omitted in the flowchart. An in-depth analysis on the parallelisation of large-scale methods is given in Section 4.1.5. DPCCMOEA makes use of contribution-based groups, and further applies a single-objective graph-based version of Differential Grouping (see Section 3.3.3) to find the interaction-based groups in a preprocessing step, which is implemented in parallel to save computation time. In order to reduce the communication between processes while using the CC framework, each process contains a whole population of individuals, i.e. variable values for all other groups, which are not optimised by the current process, are present. This is denoted in Fig. 3.7 as "create independent populations". Each of the parallel processes optimises one of the convergence-related variable groups, as well as the diversity-related variables, and information is exchanged between in the processes by transferring their respective best individuals to other processes, using a neighbourhood structure. This is repeated until a termination criterion is met.

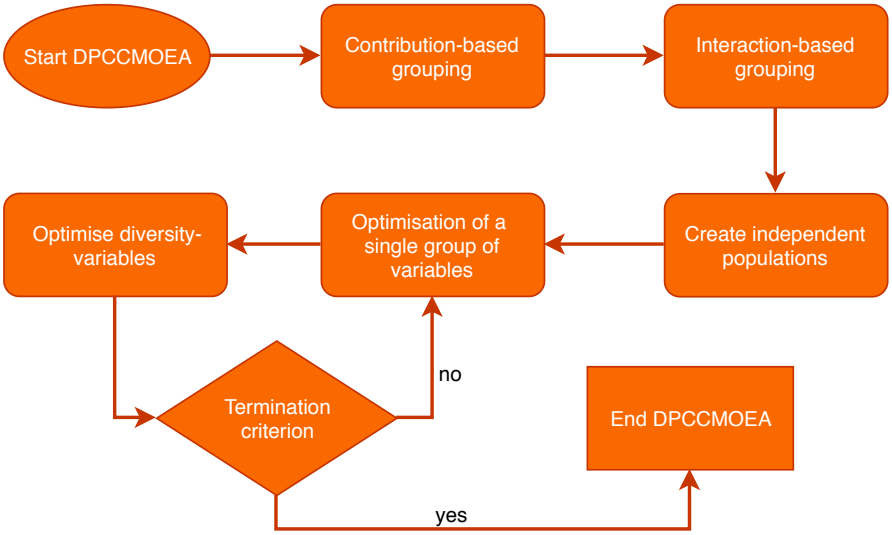


Figure 3.7: Outline of the DPCCMOEA algorithm.

S^3-CMA-ES

In 2018, an approach called S^3-CMA-ES was proposed that makes use of several independent subpopulations [77]. The S^3-CMA-ES is based on the concept of covariance matrix adaptation evolutionary strategies (CMA-ES), and is designed with the aim to use multiple populations to increase the diversity of solutions, where each population is optimised separately with a focus on one area of the search space.

The structure of this method is shown in Fig. 3.8. Similar to MOEA/DVA and LMEA, the variables are first divided into diversity-related and convergence-related variables, and the convergence-variables are further divided into subcomponents based on the variable interactions, where a modified version of Differential Grouping 2 is used (see Section 3.3.3). After that, a number of different populations for the problem are created randomly, with the special property that the diversity-related variables are identical for all solutions within one population. Each of the populations is then optimised using a CC-inspired fashion, i.e. each group of convergence-variables is optimised separately while keeping the variables in the other groups fixed. This process is done in turns without any interaction between the populations, until all populations are considered as converged (measured by a threshold in the improvement of the best solution in the population). Once all populations are considered as converged for their respective set of fixed diversity variables, a step is carried out to improve the diversity of solutions overall through optimising the diversity-related variables. Afterwards, if the termination criterion of the algorithm is not met, new populations are generated for the next iteration of the algorithm.

An interesting property of the S^3-CMA-ES is that it not only divides the variables into groups and optimises them separately. By assigning different diversity-variables and optimising each population until convergence, it also distributes the computational budget dynamically based on how long it takes to converge in a certain area of the search space. However, the S^3-CMA-ES potentially suffers from the same disadvantage as MOEA/DVA and LMEA, meaning the dependency on suitable, interaction-based variable groups which have to be obtained with a large computational budget. In the article, the same grouping strategies as in MOEA/DVA were used for the contribution-based groups, and a version of Differential Grouping 2 [87] (see Section 3.3.3) was used for the interaction-based groups. Therefore, the number of evaluations required for obtaining the groups still rises quadratically.

PEA

The *Parallel Evolutionary Algorithm (PEA)* was proposed by Chen et al. in 2018 [71]. Its aim is, as the name suggests, to parallelise the evolutionary optimisation process in a more efficient way. Specifically, if an EA is parallelised by distributing the creation and evaluation of solutions to multiple processes, the selection operators (both for reproduction and environmental) cause the need for frequent communication between processes, as they usually require knowledge about all created solutions. PEA aims to solve this issue by introducing independent sub-populations which converge independently and exchange information through local and global archives. The outline of PEA is shown in Fig. 3.9.

To increase the performance for large-scale problems, PEA utilises the concept of convergence- and diversity-related variables. Each subpopulation only optimises the convergence-related variables, keeping the diversity-variables fixed. The diversification

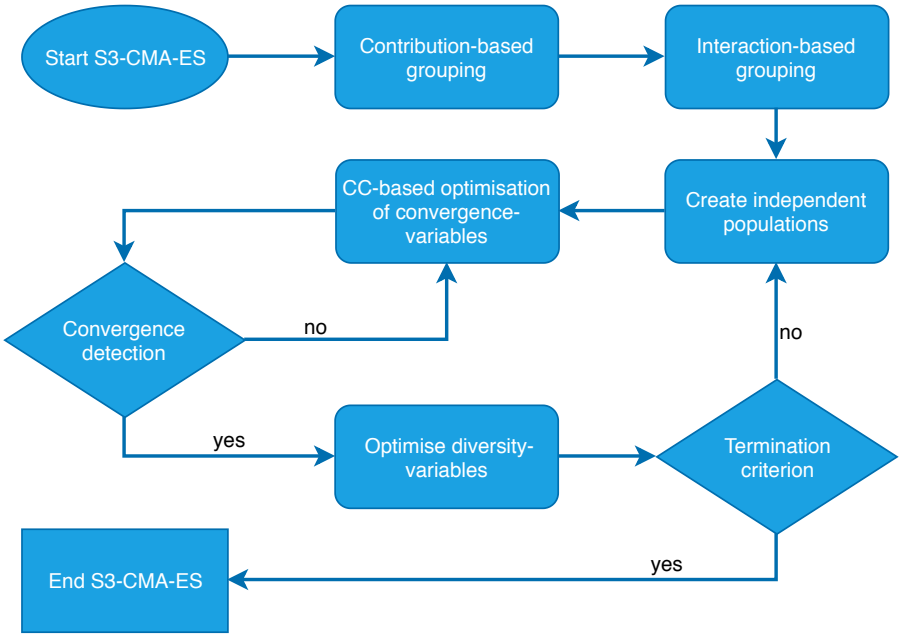


Figure 3.8: Outline of the S^3-CMA-ES algorithm.

is realised through the creating of new subpopulations from a solution archive. Within a subpopulation, the convergence-related variables are divided further into groups in a random way. The solution archive is updated after the convergence of a subpopulation is detected.

It is interesting to mention that like most large-scale approaches, PEA uses a segregation of the decision variables and applies the evolutionary operators only to a group of variables at a time. Since the aim of PEA is to build a parallel EA with little necessary communication between processes, the parallelism is not implemented through the variable groups, but through complete populations which focus solely on convergence. This is an interesting property and shows a similar concept as S^3-CMA-ES, DLS-MOEA (see below) and - in parts - also the WOF method which is proposed by the author of this thesis (Section 5.1). A further analysis of this methodology is given in Section 4.1.

ReMO

In 2017, Qian and Yu proposed a large-scale optimisation method based on random embedding, called ReMO [78]. This algorithm was designed as an approach for high-dimensional problems with low effective dimensions, which refers to problems where not all the variables actually contribute to the objective function values. In ReMO, the original n-dimensional decision space is embedded in a lower-dimensional space of dimension $v \ll n$. The search process then takes place in the v-dimensional space, and solutions are mapped to the original decision space for function evaluation using a

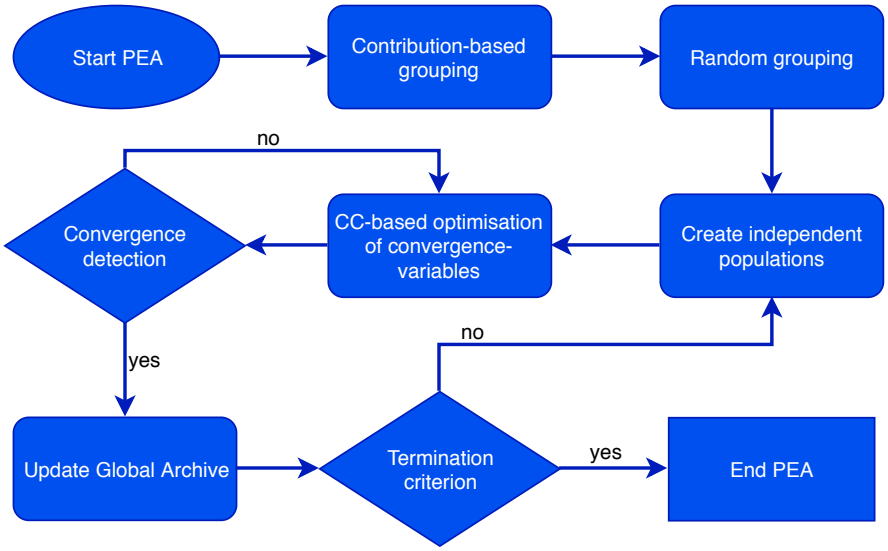


Figure 3.9: Outline of the PEA algorithm.

$v \times n$-matrix as the transformation. The structure of the ReMO method is shown in Fig. 3.10. The transformation matrix for the embedding is chosen before the start of the actual optimisation and is not changed afterwards. This step is shown as "problem transformation" in the flowchart. The actual search process can be done with any metaheuristic approach in the low-dimensional space (i.e. on the transformed problem) until a termination criterion is reached. Two algorithms were used in the article, the NSGA-II and MOEA/D algorithms, resulting in the algorithms Re-NSGA-II and Re-MOEA/D respectively.

An interesting part of this algorithm is the use of random matrices for the embedding-step. Instead of applying a mechanism to detect the effective dimensions and base the embedding on this information, ReMO chooses the elements of the embedding matrix uniformly at random from a Gaussian distribution. It is further noteworthy that ReMO is only intended to work for problems where only a few variables contribute to the objective function values. If all variables contribute evenly to the objective function values, as is the case for many of the current benchmark functions, the linear transformation using a random matrix may not be suitable to find these optimal solutions in the original, high-dimensional space. This is further examined in detail in Section 6.6.1.

Noteworthy is at this point that ReMO is a large-scale algorithm which is not based on problem decomposition. No variable groups are necessary to use ReMO, and therefore no computational budget is necessary to form groups. This is a big difference to previous well-performing methods like MOEA/DVA or LMEA, and raises the concern whether these large budgets to obtain groups are justified.

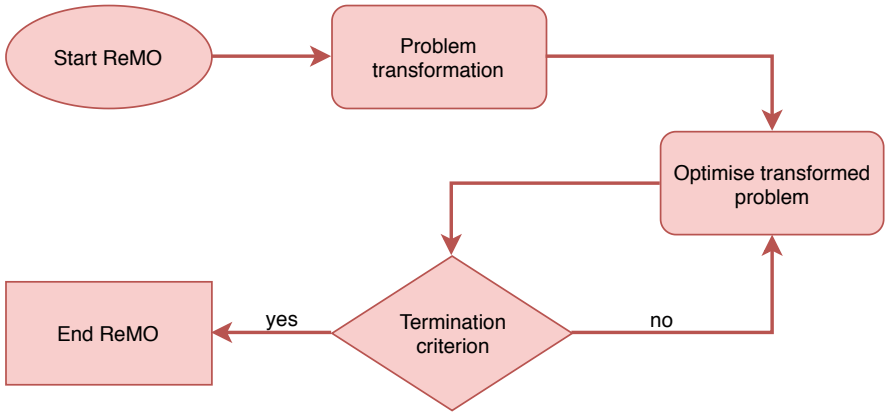


Figure 3.10: Outline of the ReMO algorithm.

DLS-MOEA

The *Multi-Objective Evolutionary Algorithm with Dual Local Search* (DLS-MOEA) is a large-scale algorithm proposed in November 2018, with a special focus on problems where it is difficult to obtain and maintain diversity of solutions [47]. The DLS-MOEA is an indicator-based approach based on the SMS-EMOA algorithm [94, 95], focused on a diversification method based on local search.

DLS-MOEA alternates between two different optimisation strategies, as shown in Fig. 3.11. In the first one, the indicator-based optimisation, solutions are generated via crossover and mutation, and the selection is based on the Hypervolume indicator. In the second strategy, the indicator-based local search, solutions are generated via a local search based on the single-objective SEE method [96]. SEE is an approach that iteratively learns the probabilities that a mutation in a positive or negative direction (for each variable separately) leads to an increase or decrease in solution quality. In DLS-MOEA, created solutions are added to an archive, and the change of quality is measured via an increase or decrease in Hypervolume.

Using these two mechanisms in turns, the DLS-MOEA aims to produce more diverse solution sets, especially in problems where diversity is difficult to obtain. Therefore, an analysis was conducted to divide existing benchmarks into different categories called convergence-focused problems (which were the ZDT, DTLZ and WFG problems), diversity-type I problems (which includes the WFG benchmarks with increased numbers of diversity-related variables) and diversity-type II problems (which were the UF benchmarks).

Like ReMO, DLS-MOEA does not use variable groups, and does therefore not require a computational budget to obtain information about variable contribution or interactions. This can be of advantage for application where only a limited budget is available. A possible disadvantage of the DLS-MOEA might be its dependency on an indicator

that can measure both convergence and diversity of a solution set. In the article, the Hypervolume indicator was used (refer to Section 2.7). Since no experiments with more than 2 objective functions were performed at that time, it is unclear whether the good performance would transfer to higher-dimensional objective spaces. The reason for this is that the Hypervolume indicator becomes more costly to calculate with increased dimensionality. If DLS-MEOA is used to solve problems with more than 3 objectives, its runtime might dramatically increase due to the very frequent calculation of the Hypervolume. On the other hand, it is possible to replace the Hypervolume with another indicator in these scenarios, but the influence on the algorithm's performance is unknown. It is further noteworthy that the DLS-MOEA's good performance was mainly shown for the UF benchmarks. Therefore, its performance for real optimisation problems is linked to the question how well current benchmarks mirror the properties of real applications and which of the UF properties are most beneficial to the performance of DLS-MOEA.

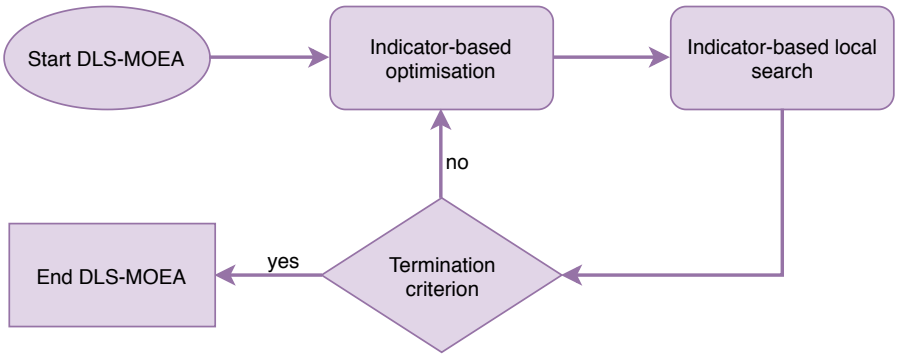


Figure 3.11: Outline of the DLS-MOEA algorithm.

LSMOF

In January 2019, the *Large-scale Multi-objective Framework* (LSMOF) was proposed, which is based on so-called "problem reformulation" [69]. The main idea of LSMOF is strongly based of the transformation strategy of the WOF algorithm, which was proposed by the author of this thesis in 2016 (see Section 5.1). LSMOF utilises a transformation step, which reduces the dimensionality of the problem through so-called weight-variables. These weights are associated with linear search directions in the decision space, starting from certain reference solution candidates. One weight variable is used to change all decision variables simultaneously, and thus creating new solutions by only altering one variable. This concept allows to decrease the search space dimensionality. LSMOF proposes a transformation method that uses two search directions for each reference solution to increase the chance to find the Pareto-optimal set within the lower-dimensional search space spanned by the weight variables.

The second technique which is used in LSMOF is similar to the idea used in DLS-MOEA, namely the usage of single-objective optimisation using a performance indicator

to optimise the formulated problems. By optimising the low-dimensional transformed problem only with respect to one fitness value (like for instance the Hypervolume of the obtained solution set), the large-scale multi-objective problem is transformed into a low-dimensional single-objective problem.

The basic structure of LSMOF can be seen in Fig. 3.12. It uses two stages of optimisation in the same way as the WOF method: first, a stage where the weight variables are used to reformulate the problem (problem transformation), and these transformed problems are optimised using the mentioned single-objective indicator-based optimisation strategy. This is done repeatedly to create solutions close to the Pareto-set until a first termination criterion is reached. In the article, this was set to 50% of the available function evaluations. The second phase of the algorithms consists of the optimisation of the original problem to obtain diversity (using the second half of the available function evaluations).

A closer examination of the properties of such weight-variable-based problem transformation strategies can be found in the following Chapter 4 and Sections 5.1 and 5.4.

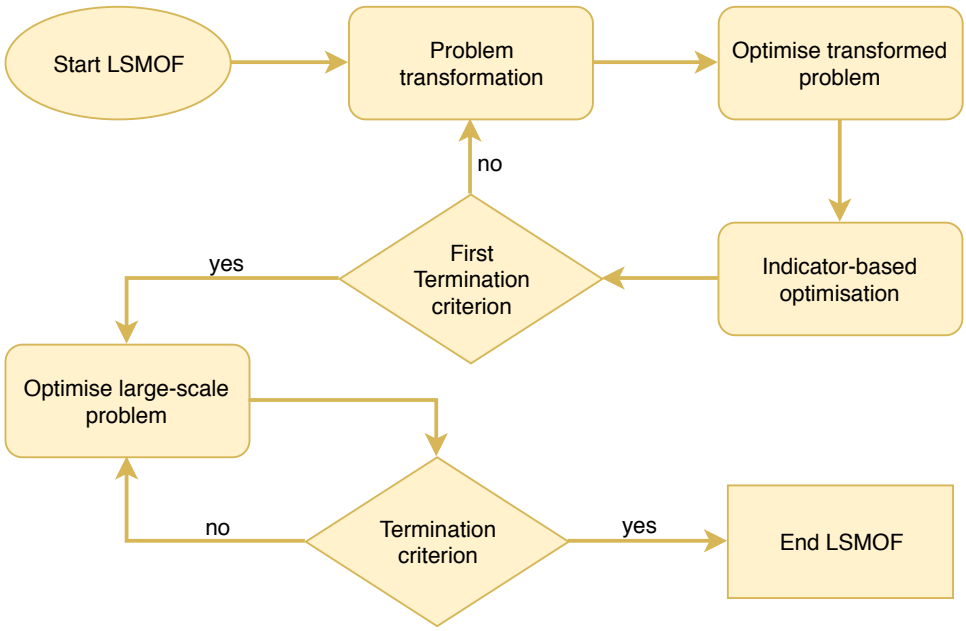


Figure 3.12: Outline of the LSMOF algorithm.

3.3 Related Variable Grouping Methods

Many of the existing large-scale methods involve a kind of grouping mechanism to divide the variables. Formally, this can be described as in Section 2.4, and groups can be based on the contribution of variables to diversity or convergence, on the interaction of variables, or on neither of those. Mostly, as far as interaction-based groups are concerned, these

mechanisms are adapted from single-objective literature. When it comes to contribution-based groups, there is a need for additional mechanisms in the multi-objective community, as these properties (i.e. the distinction between diversity and convergence) only exist in the multi-objective case.

This section takes a look into some common methods from the literature, which were or can be used in some of the related algorithms above, as well as the proposed methods of this thesis. In contrast to the above list of related large-scale multi-objective algorithms, the list of grouping mechanisms is not exhaustive for two reasons. First, the amount of literature on single-objective grouping methods exceeds the limit of this thesis, and second, the scope of the present thesis lies on large-scale optimisation in general, with a focus on the algorithms and their building blocks. A study on the usefulness of interaction-based groups is carried out in the experiment section of this thesis. A methodology to make single-objective grouping methods applicable to multi-objective problems was published in [26].

For real-world applications, it might be intuitive that optimising certain parts of the problem can be done independently of another part, and thus such independent subproblems should be identified and used in (for instance) techniques like CC. For instance, the optimisation of a production chain inside a factory might have none or minimal influence on the optimisation of the delivery logistics that happens when the finished products leave the production. In the best case, they could be treated as two independent optimisation problems. In reality, there might be some variables that contribute to both problems, and recently some research has focused on this kind of *Intervowen Systems* [97]. Another area where large-scale problems can be involved is the shape-optimisation of aircrafts or trains. Also in these cases, variables that contribute to the shape of a wing might be less likely to interact strongly with variables related to the face of an aircraft. Finding these interactions, however, is a difficult task, as especially in this example the objective functions might be complex flow-simulations and there might not exist an analytical form to analyse. This emphasises the need for automated group detection or segregation in other ways to improve the optimisation procedure.

In real applications, grouping mechanisms can further be a good way to include expert knowledge into the search process. In this case, a person familiar with the application might provide the knowledge which variables should be in the same group. This inclusion of additional information can even be useful in low-dimensional problems.

In the area of single-objective large-scale optimisation, a variety of grouping strategies has been proposed so far. An overview of existing grouping methods for single-objective problems was published in the year 2015 in [80]. A simple random grouping mechanism was, for instance, used in [33]. Yang et al. [98] proposed a multilevel Cooperative Coevolutionary method for finding optimal group sizes. The interaction of variables was taken into account in [36] by a learning mechanism for finding the optimal groups of

variables [1]. Of special interest are the single-objective methods Differential Grouping (DG) and its improved version called Differential Grouping 2 (DG2) [84, 87], which were published in 2014 and 2017 respectively. DG2 tries to overcome the drawback of a large computational overhead in finding variable interactions that was present in the original version. Other concepts and extensions [85, 86] to this approach for variable grouping have also been proposed [1].

In the multi-objective area, Coello and Sierra examined the relative importance of certain decision variables and divided the search process into multiple steps, assigning the decision variables to different groups by analysing most promising parts of the Pareto-front [99]. Although their results were competitive to other algorithms such as NSGA-II, the three used test problems only contained two decision variables each. The contribution-based groups are a relatively new phenomenon and have first been introduced in the MOEA/DVA and LMEA algorithms. Their methods of finding the contribution of variables were reused in several of the other related works, and it is worth noting that finding the contribution-based groups is generally less expensive in terms of function evaluations than the interaction-based grouping. The methods used in MOEA/DVA and LMEA are described below in further detail. In the literature, a brief division of grouping methods based on their used information was done in [27], although contribution-based methods were not included in that study.

The remainder of this section is divided into three main categories of grouping methods as follows, and a few representative candidates of each category are described. The descriptions of the simple and the interaction-based approaches are partly based on the descriptions of the author in previous works [26, 1, 6]. The proposed classification of grouping methods using the following three categories is further explained below in Chapter 4.

1. Simple methods, which do not take into account any effect of variables on the objective functions, and therefore do not use any additional function evaluations for obtaining groups.
2. Contribution-based methods, which divide the variables based on their assumed contribution to convergence or diversity of the problem.
3. Interaction-based methods, which follow the classical approach from single-objective optimisation to divide based on the interaction between variables.

3.3.1 Simple Methods

In this thesis, we use the term *Simple Methods* to refer to grouping methods, which do not use any external knowledge about the problem, any information about the influence of variables on the objective function, nor do they require function evaluations to be executed. These mechanisms have, on the one hand, the disadvantage that they are not

aiming to obtain knowledge about the problems, which can make them unsuitable for certain applications. On the other hand, as they are computationally inexpensive, they can be used anew in every generation of the algorithm, i.e. to create different groups, which might be of advantage. Further, as these methods are independent of the actual problem to optimise, they can be applied without further adjustments to single- and multi-objective optimisations. In the following, a few prominent grouping mechanisms are described. One further method that falls into this category, the *Ordered Grouping*, was introduced by the author of this thesis in a previous work [5], and is therefore presented in Chapter 5.

Random Grouping

Random grouping has often been used in the literature [33, 80]. It creates a fixed number of γ groups and distributes the variables randomly to them. The result are mostly evenly-sized groups, each of size $\frac{n}{\gamma}$, but uneven random groups are also possible. In most implementations in the literature, equal-sized groups were used. Random groups do not require a computational budget (i.e. function evaluations). Therefore, they can be obtained fast and without additional overhead for the optimisation algorithm. Due to that, random grouping can also be repeated during the runtime of an algorithm. This can be an advantage, since it may increase the chances that interacting variables end up in a common group [26]. On the other hand, random groups may be of limited use when optimising problems that contain many interacting variables.

Linear Grouping

The so-called *Linear Grouping* works similarly to the random one, and was for instance used in [1, 4]. It assigns all n variables to a fixed number (γ) of groups in "natural" order. This means, the first $\frac{n}{\gamma}$ variables of the optimisation problem are assigned to the first group, the next $\frac{n}{\gamma}$ to the second group and so forth [1]. As in random groups, the number of groups must be specified beforehand. If no specific knowledge about the problem exists, linear groups can be assumed as efficient as random groups in terms of keeping interacting variables together. However, it is to a certain extent more dependent on the problem formulation (as the order of variables and therefore the groups are defined by the problem formulation).

3.3.2 Contribution-based Methods

As described above, with the transition from a single to multiple objective functions, the additional challenge arises to keep a good diversity within the population and as a consequence, along the Pareto-optimal front. To account for this, the algorithms MOEA/DVA and LMEA in 2016 proposed grouping mechanisms with the goal to identify whether a variable is mainly contributing to convergence or diversity of the solution set. This concept was explained above in Section 2.3.4. At the current time, the proposed contribution-detection mechanisms of MOEA/DVA and LMEA are the only ones in

existence, although they have been reused in other algorithms as described above. In the following, both of these methods are described in further detail.

Control Variable Analysis

In MOEA/DVA, a simple mechanism is used to decide the contribution of a variable by changing its value while keeping all other variable values fixed [24]. As in the example given in Section 2.3.4, changing each variable separately might cause the solution to "move" in the objective space into different directions. The mechanism is called *Control Variable Analysis* in the original article and works as follows.

The variables are divided into three categories, called position, mixed and distance variables. For each variable x_i of the problem a separate analysis step is carried out to determine which of these categories the variable belongs to. Each time, a random solution is generated in the beginning of the analysis. A parameter of the mechanism, called NCA, defines how many solutions are sampled in the following procedure. The algorithm samples NCA different values for x_i, where the domain of the variable is divided into NCA parts and a random solution is drawn from each of these intervals. This results in a total of NCA solutions, each containing the same values for all variables except the currently analysed one. I.e., these NCA solutions only differ in their values for x_i. These solutions are then subject to a non-dominated sorting procedure, and the sizes of the non-dominated fronts of this set are used for the subsequent decision of the variable type. If the first non-dominated front contains exactly NCA solutions, this means that the change of variable x_i only resulted in pairwise non-dominating solutions, meaning the variable is likely to only change the position of a solution along the Pareto-front, but does not affect the closeness towards the front. If the sizes of all non-dominated fronts are equal to one, it means that varying x_i results only in dominating or dominated solutions, and therefore x_i is likely to affect only the convergence of solutions. In all other cases, x_i is called a "mixed" variable, which is likely to affect both, convergence and diversity at the same time. However, the authors of MOEA/DVA do not treat mixed and diversity variables differently in the optimisation process, and therefore they add the mixed variables to the diversity-related set of variables.

Since this procedure is carried out for each of the n variables, and NCA solutions are evaluated each time, the computational budget necessary is $n \times NCA$, with n being the number of variables. This budget rises only linear with the number of variables, and is therefore a small computational overhead compared to the interaction-based analysis that the MOEA/DVA algorithm uses afterwards.

Clustering-based Contribution Groups

In contrast to the approach in MOEA/DVA, the contribution-based groups in LMEA are obtained by a clustering approach. One reason to adapt a new mechanism was the fact that the method used in MOEA/DVA is not able to divide mixed variables in a suitable

way, and all mixed variables were considered as diversity-related. Since MOEA/DVA only optimises these diversity- and mixed-variables after a suitable convergence is reached using the (purely) convergence-related variables, this might influence performance when a problem contains a lot of mixed variables and only a small number (or none at all) of purely convergence-related variables.

The approach in LMEA aims to divide the mixed variables better among the two categories, depending on how large the influence of a mixed variable to convergence and to diversity is. The authors further motivate this approach with the fact that the MOEA/DVA contribution analysis is dependent on the concept of Pareto-dominance. LMEA is as a whole algorithm designed to work for many-objective problems, and since it is known that this poses difficulties in many-objective problems due to the increased number of dimensions, the analysis in LMEA is not dependent on dominance.

To determine the contribution of variables, a random initial population of the problem is used. For each variable $i = 1, .., n$, a number of $nSel$ of solutions from the current population is drawn at random. For each of these $nSel$ solutions, the considered variable value x_i is perturbed $nPer$ times to create a number of $nPer$ new solutions. These solutions are then evaluated and a line is fit to the points in the objective space which minimises the mean squared error between the line and the $nPer$ solutions. Next, the angle between this line and the normal of the hyperplane defined by $\sum f_i = 1$ is computed. This normal line or normal vector is used to represent the direction of convergence of the problem. The larger the angle between the fitted line of the solutions and the normal, the more the variable perturbation resulted in a diversity-related change in the objective space.

The perturbation, line-fitting and calculation of angles is done for each variable $nSel$ times, thus each variable is afterwards associated with $nSel$ angles. A k-means clustering approach (where $k = 2$) is then used to cluster the variables based on these angles, i.e. in the $nSel$-dimensional space. As a result, two disjoint sets S_1 and S_2 are obtained which contain the indices of all the variables. After these steps, the sets of convergence- and diversity-related variables are formed as follows.

- The convergence-related variables are the ones which fulfil the following conditions: (1) They belong to the set S_k ($k = \{1, 2\}$) which has the smaller mean value of the computed angles and (2) the mean of the mean squared error of the fitted lines is smaller than 0.01.
- The remaining variables are considered as diversity-related variables.

3.3.3 Interaction-based Methods

The third category of grouping methods regards the interaction based groups. These methods aim to identify interacting variables (see Section 2.3.3) through an analysis,

in most cases prior to the start of the actual optimisation phase of algorithms. As a result, these methods consume function evaluations to examine the effect of changes in the variables on the fitness function(s).

A general difficulty of such interaction-detection mechanisms lies in the balance between the computational budget and the level of detail with which the analysis is carried out. Without prior knowledge about the problem, the decision whether variables are considered interacting has to be based on definitions like the one given above in Section 2.3.3. However, under the assumption that the problems are seen as black boxes, the given equations can only be tested with certain variable value combinations at a time, and not computed analytically and on a global scale. The large computational budget usually results from the fact that a large number of variable combinations need to be tested to determine for each pair of variables whether the interaction conditions are fulfilled. It is further of importance that there is the possibility of wrong judgements when variables interact only in local areas of the search space, and conditions like the ones in Section 2.3.3 are for instance only tested with the lower and upper bounds of variables. This risk can be reduced by testing more combinations of values for each pair of variables, which, on the other hand, then results in higher amounts of used evaluations again.

The definition of variable interaction is based on differences in objective function values as shown in Definition 2.5. Therefore, the single-objective methods to check for these differences are not applicable to multi-objective problems, as the definition of "differences" between the objective function vectors is not clear any more. For instance, it is possible that certain variables interact with respect to one of the m objective functions, but not with respect to the other functions. There are, however, methods to use these definitions in multi-objective optimisation. Specific ways were implemented into MOEA/DVA and LMEA before, and their implementations are outlined below. A more generalised way to apply single-objective grouping methods to multi-objective optimisation was also proposed in [26] by the author of the present thesis. In the following, some selected single- and multi-objective interaction-grouping methods are described in further detail.

Differential Grouping

Differential Grouping (DG) is an interaction-based grouping approach for single-objective optimisation which was introduced in the year 2014 [84]. Its goal is to detect the interaction between variables by comparing the amount of change in a single objective function in reaction to a change of a variable x_i before and after another variable x_j was changed [1]. Formally, the definition of interaction used in DG differs from the commonly used one in other works, which we introduced in Section 2.3.3. DG uses the following equations to determine interaction, where it is noted that they defined interaction for additively separable functions, and therefore DG does not claim to find *any* interactions.

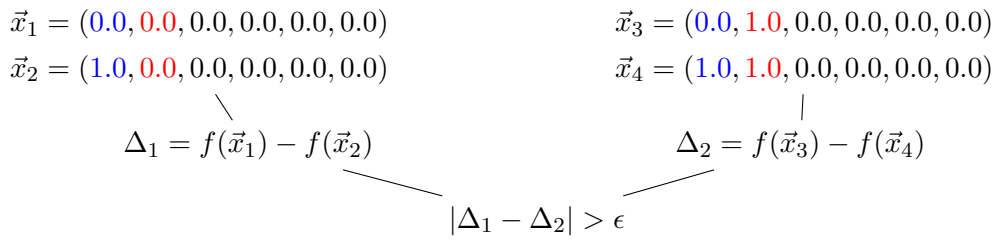


Figure 3.13: Visualisation of Differential Grouping.

Definition 3.1 (Differential Grouping Interaction) *Two decision variables x_i and x_j are interacting, if $\forall a, b_1 \neq b_2, \delta \in \mathbb{R}, \delta \neq 0$ the following condition holds.*

$$\Delta_{\delta,x_i} f(\vec{x})|_{x_i=a,x_j=b_1} \neq \Delta_{\delta,x_i} f(\vec{x})|_{x_i=a,x_j=b_2} \tag{3.1}$$

where

$$\Delta_{\delta,x_i} f(\vec{x}) = f(..., x_i + \delta, ...) - f(..., x_i, ...) \tag{3.2}$$

Using these equations, DG answers the following question [1]: "*When changing the value of x_i, does the amount of change in $f(\vec{x})$ remains the same regardless of the value of another variable x_j?*" If this is the case, the variables x_i and x_j seem not to interact with each other, i.e. the fitness function is influenced by a change in each of them, but the amount of change stays the same. These variables can be separated into different groups, as the optimal values of x_i are supposedly independent of the values of x_j. If the condition is not fulfilled, the two variables seem to interact, so they are assigned to the same group.

In practice, the definition above is of course too restrictive, as it is highly unlikely that these differences are ever equal, and variables which only interact minimally, i.e. the Δ values in Eq. (3.1) are close together but not exactly equal, could be considered as non-interacting. To account for this, a threshold value ϵ is used in DG which controls the maximally allowed amount of variation in fitness so that two variables are not regarded as interacting. The procedure of DG is shown exemplarily in Fig. 3.13. DG creates four different solutions, where the values of two variables are changed according to the above equations. In their implementation in [84], the respective upper and lower bounds of each of the variables were used for the values of a, b_1, b_2. The resulting differences in the function values are compared with the threshold ϵ. The number and sizes of the groups are determined by the algorithm automatically depending on how many variables interact with each other. If $|\Delta_1 - \Delta_2| > \epsilon$, the variables are considered interacting and they are put in the same group. All variables that do not interact with any other variables are gathered in an additional group, which in the end holds all non-interacting variables.

In the literature, DG showed good performance in single-objective optimisation [84], but requires a large computational overhead to find the interactions. Although the algorithm

implementation adds variables iteratively to the groups, and therefore might not need the complete $n \cdot (n-1)$ checks of variable pairs, each single check needs four function evaluations, and the total computational effort is quadratic in the number of decision variables. In [84], it is computed that, assuming that the true interactions of the problem result in a number of $\gamma = \frac{n}{l}$ evenly sized groups with l variables each, the number of function evaluations consumed is in $O(\frac{n^2}{l})$. Based on the analysis in the original article, for a fully separable problem using $n = 2000$ decision variables, DG requires $n \cdot (n-1) + 2 \cdot n = 4,002,000$ function evaluations to perform the grouping, and the effort reduces with increasing numbers of interactions between variables [1]. Since the actual used computational budget is dependent on the interactions in the problem, it is hard to know beforehand how many function evaluations are actually needed until DG terminates its analysis. This might not be desirable since it would then not be clear how much computational budget is left for the actual optimisation of the problem. Some of the shortcomings of DG were addressed in its successor, described in the following section.

Differential Grouping 2

The *Differential Grouping 2* (DG2) was proposed recently in the year 2017 [87] and addresses some of the drawbacks of DG, which are (1) the dependency on a threshold parameter, (2) the inability to deal with so-called "overlapping" components and (3) the high computational overhead. Compared to its predecessor, DG2 obtains a better group quality and requires a lower computational budget. The results of evaluated solutions are stored and reused during the grouping process. In this way, the necessary function evaluations for DG2 are reduced to $\frac{n(n+1)}{2} + 1$. For a 1000-variable problem this results in $500,500$ evaluations. This makes DG2 superior compared to its predecessor, although for real world problems, this amount can still be infeasible depending on the application.

Moreover, DG2 improves on the quality of the found groups. DG had the disadvantage of iterating over the variables and adding variables to groups once the first interaction with another variable was found. Therefore, variables which interact with only one variable in one group but with many variables in a second group might end up in the first of these groups. DG2, in contrast, builds up the complete "graph" of interactions between all variable pairs and forms groups based on this information. For more details on the specific mechanisms of DG2 the reader is further referred to the original publication.

Interdependence Analysis in MOEA/DVA

The basis for the grouping mechanism of MOEA/DVA, called *Interdependence Analysis* in the original article, is the definition in Eq. (2.6). After creating an initial population, the variables are first divided into convergence- and diversity-related variables. The following interaction-based analysis is then applied to all variables, although the division into groups is done for the convergence-variables only in the subsequent step.

For every combination of variables x_i and x_j $(i, j \in 1, .., n)$, one solution is selected by random choice out of the current population of the algorithm. Then, DG2 creates three new individuals by sampling random values for x_i and x_j within their feasible domains, and replacing the variable values in the chosen solution candidate. For each of the m objective functions, the interaction is checked separately, using these four created solutions, according to Definition 2.5 . DG2 possesses a parameter NIA, which determines how often this process is repeated in order to increase the probability of finding the variables' interactions. To utilise the created solutions, the current population is updated after each check using the three newly created solutions. If the variables at hand are convergence-related and dominate the current solution in the population, the new values for x_i and x_j are kept and the solution in the population is replaced.

In the following step, the interaction information for each of the objective functions is used to form the different groups. In order for the algorithm to consider two variables as interacting, they both need to be convergence-related and have an interaction in at least one of the objective functions. In addition, any interaction between two variables leads to the inclusion of all variables that further interact with any of these two into the same groups. The article describes this concept as forming the groups as maximal connected subgraphs of the variable interaction graph. This concept might lead to potentially large groups if the problem contains many interacting variables or the interactions are very different in the different objective functions.

Regarding the computational budget, it is stated in [24] that the interaction analysis needs $\frac{3n(n-1)*NIA}{2}$ function evaluations. This is a larger amount than needed by the DG and DG2 algorithms, and is due to the parameter NIA. The interactions in MOEA/DVA are checked using random values instead of fixed (i.e. upper and lower bounds in DG) ones. Therefore, it can potentially discover more local interactions, but to increase the chances of finding these interactions in all objective functions, the procedure is repeated NIA times. As a result, to analyse a 1000-variable problem, the algorithm needs approximately $NIA \cdot 1,500,000$ function evaluations. Using a value of $NIA = 6$, as was used for instance in the experiments in [25], the algorithm uses almost $9,000,000$ evaluations to analyse the problem, which consumed most of the evaluations used in the experiments in the article.

Interaction Analysis in LMEA

The interaction-based grouping method in LMEA was called *Interaction Analysis* [25], and works similarly to the method used in MOEA/DVA. Interaction is assumend between two variables using the above Definition 2.5. As in MOEA/DVA, an interaction in one of the objective functions is sufficient to regard two variables as interacting.

The difference to the MOEA/DVA method lies in the way that variables are assigned to the created groups. The interaction analysis builds the groups iteratively by adding the variables one after another. Each variable x_i is checked for interactions with all variables

in the already existing groups. If an interaction with at least one of the variables exists, x_i is added to that respective group. This check of interaction is done using the mentioned equations and a number of randomly chosen solutions from the current population. The number of solutions drawn and used for the interaction checks is called $nCor$ in the article and roughly corresponds to the parameter NIA in MOEA/DVA. Groups are formed by iteratively building the union of sets which share an interaction with a variable. Therefore, the groups are formed again in the same way as in MOEA/DVA, meaning that one interacting pair of variables between two groups is sufficient to join these two groups and form one big one, even if many of the variables inside this group may not interact with each other.

In contrast to MOEA/DVA, in LMEA only the convergence-related variables are used in the analysis. For problems with a larger number of diversity-related variables, this may lead to lower computational overhead. However, the general computational costs are similar to the ones of MOEA/DVA. In the experiments in the original article the parameter $nCor$ was set to 6, meaning that the analysis takes again up to $9,000,000$ evaluations to analyse the interactions of a 1000-variable problem.

Random-based Dynamic Grouping

One of the most striking disadvantages of interaction based grouping methods like DG or the ones used in MOEA/DVA and LMEA is the necessity of a large computational budget to find the interacting variables. To address this issue, a *Random-based Dynamic Grouping* strategy called *RDG* was proposed in 2016 [67]. The aim of this RDG strategy is a reduction of the necessary function evaluations when creating the groups based on the interaction information. The RDG method was implemented into the same framework as the MOEA/DVA algorithm, and therefore also only divides the convergence-based variables into groups [6].

The authors of RDG argue that especially in many-objective optimisation, groups that are suitable for all objective functions might be hard to find or non-existent, and that interaction, which might exist only locally between variables, can only be found with a large computational overhead. Therefore, they apply random groups in RDG, where in each iteration of the main loop of MOEA/DVA, new random groups are created and optimised based on the CC-inspired framework. The dynamic property of the method refers to the sizes of groups. All groups are always created randomly, but the probabilities of choosing the size of the groups varies and is updated based on the success of previous usages of the group sizes. In the beginning of the algorithm, all sizes of the random groups have the same probability to be chosen. When a group size was used to create random groups, the different groups are optimised and a performance metric is used afterwards to determine how much of the old population is dominated by the population that was just optimised with the given group size. Through this mechanism, the algorithm iteratively

updates the probabilities of choosing the size of the random groups to suit the problem better.

The proposed algorithm MOEA/D-RDG, as described above, was able to improve on the performance of MOEA/DVA, since no additional function evaluations are necessary to obtain the groups. Thus, the optimisation algorithm was able to spend a larger amount for the actual search [6]. This, together with the fact that many of the current state-of-the-art algorithms do not depend on interaction analyses any more, raises again the question of how important interaction-based groups actually are for the success of the optimisation. For instance, a study made by the author of this thesis in [26] also revealed that MOEA/DVA with random group shows in many cases better performance than the original, interaction-based MOEA/DVA.

3.4 Summary

This chapter summarised the related literature in the area of large-scale multi-objective optimisation. First, a brief overview showed that large-scale approaches have been studied in the single-objective field for many years, but have only recently begun to draw attention in the multi-objective community. Most related approaches are based on the concept of coevolution, although other techniques have been proposed. The concepts of the existing large-scale multi-objective algorithms were described, including some possible advantages and disadvantages. Afterwards, we described a selection of existing grouping methods from the literature, both in single- and multi-objective optimisation.

Chapter 4

Classification and Comparison of the State-of-the-Art

This chapter aims to summarise, compare and classify the existing approaches and grouping mechanisms in the large-scale multi-objective optimisation area. We point out common elements and properties as well as differences of these approaches. While an experimental comparison is carried out in Chapter 6, in this chapter the theoretical differences as well as possible advantages and disadvantages of different methods are explored. Furthermore, the experimental evaluations of the algorithms from the literature are analysed. We propose in Section 4.1 a classification scheme for large-scale multi-objective algorithms, and the existing algorithms are compared and categorised based on several proposed criteria, such as their dependency on groups, their required computational budget, their techniques for dimensionality reduction and others. In addition, in Section 4.2 a classification of grouping mechanisms is proposed and the related methods are categorised based on it. The components of algorithms which are interchangeable, like for instance grouping methods, are also called *building blocks* in the remainder of this work. Different candidates for each of such interchangeable blocks are either present in the related literature or are developed during the process of this dissertation and its preceding publications. The classification and comparison of large-scale optimisation algorithms form a new contribution in this thesis and has not been published before.

4.1 Comparison and Classification of Large-scale Algorithms

The goal of this section is to compare and categorise the related algorithms for large-scale multi-objective optimisation presented in Section 3.2. To do so, the similarities, common elements (building blocks) and differences based on different criteria are identified and presented in the following. We propose several criteria which are relevant to large-scale optimisation in multi-objective problems and classify the existing approaches accordingly.

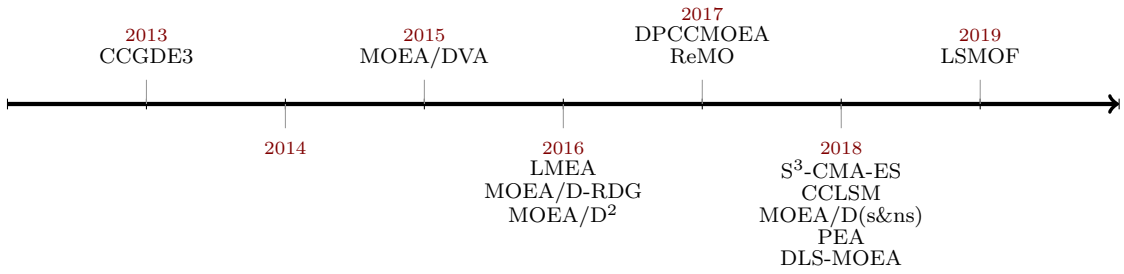


Figure 4.1: Publication history of large-scale multi-objective algorithms.

4.1.1 Similarities and Common Building Blocks

Reviewing the existing large-scale multi-objective optimisation literature reveals that most algorithms were in fact published within the last three years, with 11 out of 13 methods listed in Section 3.2 published in 2016 or later, and 6 of them published in 2018 or 2019. The publication history of the related methods is depicted in Fig. 4.1, and shows an increased attention of the research community towards this area in the last years. Looking closer at the structures of the state-of-the-art algorithms, we observe that on the one hand there are certain common elements in most of these algorithms, while on the other hand there is a diverse set of different techniques. We now first examine the state-of-the-art algorithms to identify similarities and common elements, so-called *building blocks*, which are present in the large-scale multi-objective algorithms.

The first common element of all the 13 methods is that all of them rely on existing "low-scale" algorithms to perform the actual optimisation, i.e. algorithms which work well with traditional multi- or single-objective problems and relatively low dimensionality. This is implemented in several different ways, but in all cases one or more of the sophisticated search strategies for single- and multi-objective areas are usually incorporated to optimise different parts of the whole problem. They are used to optimise either separate populations for certain areas of the search space (as for instance in S^3-CMA-ES) or used to optimise only certain groups of variables (as in the CC-based approaches like MOEA/DVA or CCLSM). In some cases, single-objective optimisation is used to optimise the problem based on an indicator-value (as in LSMOF or DLS-MOEA).

Since sophisticated algorithms in the multi-objective area have been developed over the last two decades, it is only natural to cover large-scale problems by harvesting the power of existing algorithms, i.e. by reducing the dimensionality of the large-scale problems, and then applying these algorithms. All of the mentioned methods, with the exception of DLS-MOEA, use some kind of dimensionality reduction technique in the decision space to enable related algorithms to deal with a lower-dimensional problem. In addition, some of the algorithms - including DLS-MOEA - apply a reduction of dimensionality also in the objective space.

These findings lead to one of the most striking observations: There is no common algorithm structure that covers all large-scale methods, and therefore no kind of *general framework* for successful large-scale algorithms seems to present itself. DLS-MOEA is one of the newest algorithms which, based on the experiments in the original article, performs very well on current large-scale problem instances. However, it is also the only method that does not reduce the dimensionality of the search space through any mechanism. On the other hand, some of the most successful algorithms like LSMOF and the WOF by the author of this thesis heavily rely on the reduction of dimensionality and reformulation of the original problem. Similarly, some algorithms show good performance using variable groups, while others perform equally well without ever using any grouping technique. Among all methods, the DLS-MOEA is the only example so far that shows that dimensionality reduction is not necessarily the definite answer to all large-scale optimisation problems, which is a surprising result given that all other algorithms in the last 6 years have relied on this principle.

Some further similarities are striking among the related methods. Among them, the algorithms CCGDE3, MOEA/D^2 and CCLSM are all very similar. In fact, CCGDE3 and MOEA/D^2 were proposed by the same authors in 2013 and 2016 respectively. They differ only in the aspect that the independent populations for the groups were replaced by only one population, so that the search for suitable partners from the other population for evaluation is not necessary any more. Following in a publication in 2018, the CCLSM is essentially the same algorithm as the MOEA/D^2, with the only difference that the random groups were replaced with an interaction-based grouping method.

Similar relationships exist between other algorithms as well. MOEA/D-RDG is equivalent to MOEA/DVA, only the interaction-based groups are replaced by the dynamic random-based method (DRG) described in Section 3.3.3. Further, there exist certain similarities between the three methods PEA, S^3-CMA-ES and DPCCMOEA. Based on the simplified flowcharts of all algorithms in the previous chapter, Table 4.1 shows an overview of which blocks are used in which methods, and how often each component is used in the literature in total. As mentioned before, the MOEA/D(s&ns) is excluded from this analysis, due to the uncertain quality of the work (see Section 3.2).

Based on Table 4.1, we see that the most common element in the literature is the "Contribution-based grouping", which occurs in 6 out of 12 methods. Similar popularity enjoy the building blocks "Interaction-based grouping" and "CC-based optimisation of convergence-variables", which occur 5 times each. 4 out of 12 methods use a form of "Convergence-detection". Further, the "Creation of independent populations" is used 3 times, which refers to actual independent populations for the original large-scale problem (i.e. not populations only containing certain variables as in the CC-based approaches). A minority at the current time are the indicator-based methods and transformation-based methods, whose building blocks only occur in ReMO, DLS-MOEA and LSMOF. It is further interesting that random groups are used 4 times, and that also the optimisation

	Building Blocks															
	Random Grouping	Interaction-based Grouping	Contribution-based Grouping	CC-based optimisation of large-scale problem	CC-based optimisation of convergence-variables	Optimise large-scale problem	Optimise diversity-variables	Indicator-based optimisation	Indicator-based local search	Optimise transformed problem	Optimisation of a single group of variables	Update global archive	Create independent populations	Convergence detection	Problem transformation
CCGDE3	✓			✓											
MOEA/DVA		✓	✓		✓	✓								✓	
LMEA		✓	✓		✓		✓								
MOEA/D-RDG	✓		✓		✓	✓								✓	
MOEA/D^2	✓			✓											
DPCCMOEA		✓	✓				✓				✓		✓		
ReMO										✓					✓
S^3-CMA-ES		✓	✓		✓		✓						✓	✓	
CCLSM		✓		✓											
PEA	✓		✓		✓							✓	✓	✓	
DLS-MOEA								✓	✓						
LSMOF						✓		✓		✓					✓
Σ	4	5	6	3	5	3	3	2	1	2	1	1	3	4	2

Table 4.1: Summary of building blocks in the related large-scale algorithms. The bottom row shows in how many methods a building block is used.

of the large-scale problem as a whole occurs in a quarter of all methods, without reducing the dimensionality.

The remainder of this section analyses these observations in further detail. The resulting classification of methods and which categories they belong to are summarised in Table 4.2, and details about experimental evaluations are shown in Table 4.3. The properties, similarities and differences are categorised in the next subsections based on the following criteria.

- Dimensionality Reduction in Decision Space
- Diversity Management
- Many-Objective Capabilities
- Parallelism
- Experimental Evaluation and Computational Budget

4.1.2 Dimensionality Reduction in Decision Space

Most of the large-scale algorithms use a dimensionality reduction mechanism to be able to optimise in a lower dimensional space with existing algorithms. To characterise the different approaches of reduction, we propose the following **three categories of dimensionality reduction** techniques. For convenience, the related algorithms and their respective categories of dimensionality reduction are summarised in Table 4.2.

The **first and most common category of dimensionality reduction** is based on **Cooperative Coevolution**. This strategy is used by 9 out of 12 related methods. However, not all methods actually apply coevolution in the way it was originally proposed in single-objective optimisation in 1994 [29]. While the CCGDE3 for instance applies CC in its original way by keeping independent subpopulations and chooses "partners" from each of the other subpopulations in each function evaluation, other methods like MOEA/DVA, and MOEA/D^2 do not keep these subpopulations independent. Instead, only one complete population is used for the whole problem and genetic operators are only applied to specific variables at a time. This eliminates the need to choose appropriate partners for each evaluation. For the CC-based dimensionality reduction, a segregation of variables into groups is usually necessary. Algorithms which apply this reduction technique are CCGDE3, MOEA/DVA, LMEA, MOEA/D-RDG, MOEA/D^2, DPCCMOEA, S^3-CMA-ES, CCLSM and PEA.

The **second category of dimensionality reduction** concerns the **transformation-based reduction approaches**. These methods also can use variable groups, but in contrast to CC-inspired algorithms, the reduction of the search space is done through a kind of problem transformation strategy. Algorithms in this category are ReMO and LSMOF. In case of ReMO, the problem is reformulated into a lower dimensional one by a transformation matrix which maps the vectors of n decision variables of the original search space into a lower-dimensional subspace. In a different way, the LSMOF method uses weight vectors to determine how the original solutions are altered through a set of weights.

The **third category** consists of algorithms which **do not use any dimensionality reduction** technique in the decision space. The only representative of this category is currently the DLS-MOEA algorithm, which uses indicator-based optimisation and local search mechanisms to achieve diversity, but does all of its optimisation in the original, high-dimensional decision space.

One further interesting property regards the usage of interaction-based variable groups. We saw that 9 out of the 12 existing approaches lie in the first category and use the CC-based method of dimensionality reductions. All of these 9 CC-based algorithms need to divide the variables into groups. However, not all of them actually use interaction-based information. Based on Table 4.1, the interaction analysis of variables is used only in 5 algorithms: MOEA/DVA, LMEA, DPCCMOEA, S^3-CMA-ES and CCLSM. The decision

Year		Algorithm	Dimensionality Reduction Category			Diversity Management Category				Many-objective	Parallel
			1	2	3	1-1	1-2	2	3		
2013	[3]	CCGDE3	✓					✓			
2015	[24]	MOEA/DVA	✓			✓		(✓)			
2016	[25]	LMEA	✓			✓				✓	
2016	[67]	MOEA/D-RDG	✓			✓					
2016	[79]	MOEA/D^2	✓					✓			
2017	[68]	DPCCMOEA	✓				✓				✓
2017	[78]	ReMO		✓				✓			
2018	[77]	S^3-CMA-ES	✓				✓			✓	◯
2018	[70]	CCLSM	✓					✓		✓	
2018	[71]	PEA	✓				✓			✓	✓
2018	[47]	DLS-MOEA			✓				✓		
2019	[69]	LSMOF		✓				✓	✓		

Table 4.2: Classification of existing large-scale methods based on identified categories. Many-objective capabilities are based on experiments in the literature (see Table 4.3). The ◯ indicates that S^3-CMA-ES can be parallelised without changes in the algorithm behaviour.

to not use interaction-based groups in the other 4 CC-based algorithms (CCGDE3, MOEA/D-RDG, MOEA/D^2 and PEA) might often be based on the disadvantage of a large computational budget to obtain the interaction-based groups, as seen in Section 3.3. It might be for this reason also that the usage of interaction-based methods seems to be disregarded by newer methods in favour of computational efficiency and specialised search strategies as in DLS-MOEA and LSMOF.

4.1.3 Diversity Management

Another way to categorise the large-scale algorithms is by their method of achieving diversity in the solution set. For this criteria, we propose again **three categories** based on the following analysis, where category 1 is divided further into two subcategories. Again, the algorithms and their corresponding categories of diversity management are listed in Table 4.2.

One way to achieve and maintain diversity, the **first category of diversity management**, is to **make use of the diversity-related variables** of a problem, as these mainly represent the trade-off relationship between objectives. As we see in Table 4.1 above, 6 out of 12 related algorithms make use of such variable contribution information. This strategy is used especially since the cost of obtaining this knowledge (i.e. the contribution-based grouping) is relatively low compared to other, interaction-based grouping methods. Among the algorithms which use the diversity-based variables, there are different approaches how to incorporate them into the optimisation. Algorithms which apply diversity-based groups and how they use these groups are listed in the following, and can be further divided into **two distinct groups**.

Methods in the **first group** (category 1-1) are MOEA/DVA, MOEA/D-RDG and LMEA. They use different diversity-variable values for each solution in the population.

- MOEA/DVA and MOEA/D-RDG initialise the diversity-based variables in the beginning of the search, and then leave them untouched while focussing the optimisation only on the convergence-variables. Only after a certain convergence is detected, the whole problem, including the convergence- and diversity-related variables, is optimised again until the termination criterion is reached.
- In contrast, LMEA optimises convergence- and diversity-variables only independently, but in turn over the course of the whole optimisation process. It is therefore more likely to deliver good intermediate results, since the optimisation of diversity does not just happen towards the end of the search as in MOEA/DVA.

Methods in the **second group** (category 1-2) are S^3-CMA-ES, PEA and DPCCMOEA. They use independent populations for the whole problem, where each population shares the same values for the diversity-based variables.

- S^3-CMA-ES achieves diversity through independent populations, each concentrating on specific parts of the objective space. Here, the diversity-variables are used to create multiple populations, where each population shares the same diversity-variables among its members. Only the convergence-variables are subject to optimisation. In this way, diversity is achieved because each population tries to achieve convergence for different areas of the search space and therefore objective space.
- PEA, which was proposed by the same authors as S^3-CMA-ES, follows the same principle, but with a focus on performing the optimisation of the populations in a parallel fashion (see below).
- DPCCMEA also follows this approach, with the difference that here the diversity-variables are not fixed, but also subject to optimisation in each of the independent populations. As in PEA, the focus and main contribution of DPCCMOEA lies more in the parallel implementation instead of the general methodology of large-scale optimisation.

Except using diversity-based variables, there are other approaches to tackle the issue of diversity. It is of interest for the development of future algorithms in this area that the usage of diversity-variables does not necessarily lead to superior diversity. Among the other 6 algorithms, which do not make use of such contribution information, there are 4 that **do not apply any specific method to obtain diversity**, which we identify as the **second category of diversity-management**. These methods mainly rely on the

hope that the diversity management of the applied metaheutristic, in most cases combined with a CC-based optimisation, is sufficient. The algorithms CCGDE3, MOEA/D^2, ReMO, and CCLSM are examples of algorithms which do not explicitly define a mechanism to increase or preserve diversity. And as seen above, with the exception of ReMO, all of them rely on coevolution. It is further noteworthy that the approach of MOEA/DVA, which originally belongs to the first category seen above, actually applies an optimisation of the complete large-scale problem towards the end of the search. This feature of MOEA/DVA is used with the specific intention to increase diversity, and we can therefore also sort MOEA/DVA partly into the second category.

The remaining algorithms, DLS-MEOA and LSMOF, both possess a form of diversity management which belongs to the **third category of diversity management**. Both algorithms utilise an **indicator-based approach**, where the original or transformed problem is optimised using the hypervolume indicator. This indicator is able to measure convergence and diversity of a solution set, and in contrast to many other performance indicators does not need a sample of the true Pareto-front to be applied. Since with this indicator all objective function values can be mapped to a one-dimensional number, single-objective search techniques are applied in both algorithms to optimise based on the hypervolume. In addition, the LSMOF method also follows a similar idea as in MOEA/DVA and optimises the whole, high-dimensional problem during the second half of the optimisation process. While DLS-MOEA uses the indicator-based search in combination with a local search mechanism to increase the diversity during the whole process, LSMOF focuses on convergence for the first 50% of the function evaluations, and achieves diversity only in the second stage of the algorithm. Therefore, the second half of the LSMOF process is representative of the second category as seen above.

4.1.4 Many-Objective Capabilities

One further observation that is related to the matter of diversity management concerns the reduction of dimensionality in the objective space. This kind of approach has begun to draw attention in the many-objective area, but has also been used recently in some of the large-scale approaches. The indicator-based approaches of DLS-MOEA and the LSMOF both reduce the objective space through the calculation and optimisation of the Hypervolume. It must be noted, however, that calculation of Hypervolume, at least when it is done for a whole set of solutions, is computationally expensive when the number of objectives increases, and as we see later this can be disadvantageous when the DLS-MOEA method is used. Using the Hypervolume indicator can therefore be problematic when applying the large-scale methods to many-objective optimisation in future studies.

Regarding the applicability of algorithms for many-objective problems, in fact there are 4 out of the 13 algorithms in the large-scale area which were originally intended to work for both large decision spaces and large objective spaces. As we describe in further detail

later, we also see this in Table 4.3, where LMEA, S^3-CMA-ES, CCLSM and PEA are the only algorithms which have been tested on more than 3 objective functions. Both of the indicator-based approaches were not tested for more than 2 and 3 objectives respectively. On the other hand, PEA, CCLSM and LMEA were tested with up to 10, the S^3-CMA-ES with up to 15-objective problems. This does, with the exception of the indicator-based methods, not imply that the remaining methods do not work well with many-objective cases. However, since there exist a variety of methodologies to tackle the large search spaces, it is likely that they show significant differences when applied to many-objective test problems. Some of the frameworks which are specifically designed to have interchangeable optimisers, like for instance LSMOF and ReMO, might be able to tackle many objectives directly through the use of many-objective optimisers like NSGA-III or RVEA inside their frameworks. Other algorithms might face more difficulties. Experimental and theoretical evaluation on the effects of building blocks on high-dimensional search as well as objective spaces might be a driving factor in the design of future algorithms. The algorithms of the current state-of-the-art which did report results for many-objective instances are marked in Table 4.2.

4.1.5 Parallelism

Another criterion which sets the algorithms apart is their parallel nature, i.e. their ability to run in parallel. Two of the related algorithms - namely PEA and DPCCMOEA - are developed specifically with the goal to speed up the evolutionary search process through parallel computation (as also indicated in Table 4.2). Therefore, their design is not only orientated on how to obtain the best solutions, but on how to achieve good performance while at the same time divide the search process into independent components. This is reflected in the choice of the used building blocks, and we saw earlier that their form of diversity management depends on independent populations and diversity-based variables (see Table 4.1). In these two methods, the experimental evaluation was also done in parallel and the speed-up in terms of execution time was used as a performance measure. Although only these were explicitly labelled as parallel algorithms, it is worth to take a look at how well the remaining methods can be used - or adapted - in a parallel way.

One main aspect that limits the efficiency of a parallel implementation is the need for communication between processes. In a traditional evolutionary algorithm, this is for instance the case when in each generation the information of the current population needs to be gathered to perform parental and environmental selection. Due to this fact, the PEA and DPCCMOEA both employ independent populations for the whole problem, so they can be optimised without the need of frequent communication between processes.

As most of the other large-scale approaches also employ a kind of dimensionality reduction that is often used with variable groups, it seems that most of them can easily be run in parallel. However, if we look for instance at the relatively simple CCGDE3 algorithm, we see that this might not necessarily be the case. CCGDE3 employs Cooperative

Coevolution in its original form, meaning that it holds independent populations for the variables in each group, and during the optimisation of one group, the variables of the respective other groups remain fixed. The way that CCGDE3 is written, however, optimises the populations iteratively, and the optimisation of the subsequent populations already uses the updated solution candidates of the previous population. Therefore, the optimisation of the populations is not independent, and optimising all of them with copies of the populations as they were when the main loop began might result in different results or slower convergence.

Unfortunately, the same is also true for most of the other methods. Due to the similarity to CCGDE3, the same argumentation applies to MOEA/D^2 and CCLSM. MOEA/DVA and LMEA also proceed in the same way, and by extension also the MOEA/D-RDG. In all these methods, which are CC-based, fitness evaluations might be done in parallel, but frequent communication is necessary in each generation if the same algorithm behaviour as in the original implementations is desired. However, if this condition is weakened, a parallel implementation of these approaches might be obtained by using fixed copies of all variables in the other groups in each generation, optimising each group with these copies of the variables, and then merging the resulting populations afterwards. In this way, these methods were to operate with independent population in a similar way as the above parallel algorithms. It is, however, required that an adequate method can select those solutions, and as each of the independent populations only provides updated values for the variables in one group, it is likely that the overall search process is slowed down, as only small parts of the whole solutions are updated in each generation.

The transformation-based methods ReMO and LSMOF also seem difficult to parallelise. In ReMO, the transformation only provides a smaller search space, but it seems not clear how this can help in a parallel implementation. The same is true for LSMOF, especially since the second half of the search process consists of a classical optimisation of the large-scale problem, so even a parallel implementation of the transformed optimisation would only speed up the first half of the search. In DLS-MOEA, one thing that might be done in parallel is the local search based on an archive and the hypervolume indicator. However, also in this case this only corresponds to a fraction of the overall search, and since this is done in each generation, processes would need to wait in idle time until all local searches are finished.

The one remaining candidate, which might offer a promising solution for a parallel version, is the S^3-CMA-ES. As said above, this algorithm is similar to DPCCMOEA and PEA, as it also uses independent populations. The only limitation in this case is the fact that S^3-CMA-ES contains a diversity-improvement step, which is done once all independent populations are considered converged. If this is implemented in a parallel way, this might result in idle time of processes until the last of the populations achieves the convergence goal. The speed-up that can be achieved is therefore limited by the properties of the problem, but in contrast to the above algorithms, it is possible to

parallelise the S^3-CMA-ES without altering its behaviour (this is marked with a ◯ in Table 4.2).

Lastly, it must be noted that the above considerations mostly apply on the actual optimisation procedure of the algorithms. Another question of interest might be how a parallel implementation of grouping methods could be obtained, especially for the high-budget interaction-based methods. This is also of concern for the S^3-CMA-ES and DPCCMOEA, as they both use interaction-based groups. An analysis in the original DPCCMOEA article addresses this problem explicitly by calculating that about 10% of the whole $10,000,000$ evaluations is spent on the interaction analysis and is therefore not parallelised.

4.1.6 Experimental Evaluation and Computational Budget

Now we take a look at the experimental evaluation that has been done in the literature so far with respect to large-scale multi-objective optimisation. With such an amount of different methods published within a short time, and on a topic that is still in development and in need of good methods, it is also natural that the evaluation criteria differ. Therefore, this thesis provides a comparison of the experiments done in the literature. A detailed overview of the experimental methodologies and results in each of the related articles is given in Appendix A. In the following, these findings are summarised and analysed.

In Table 4.3 we provide an overview of each of the current large-scale methods that exist at the current point in time. We list which other algorithms, large-scale and conventional, they were compared with, which benchmark functions were used with how many variables and how many objective functions, and lastly we provide the amount of function evaluations that were used in each of the experiments. The experiments done in the literature differ largely between works. As we see in Table 4.3, some algorithms were only compared to conventional "low-scale" algorithms and some have only used relatively "easy" benchmarks, like the ZDT functions. In some works, other large-scale methods were used for comparison as well, but the used numbers of variables and objective functions also varies greatly.

The algorithm which was used most often for comparison from the large-scale area might be the MOEA/DVA, which appears in 7 out of 11 works which were published after MOEA/DVA. LMEA was used 3 times and CCGDE3 2 times. In total, 5 out of 13 articles have also never compared their algorithms with other large-scale methods, although one of them is the CCGDE3, which was at time of its publication the only large-scale method. Most methods actually compare their large-scale methods with normal, established metaheuristic methods. This makes sense in the way that many of the large-scale methods use other algorithms inside them to optimise the formed subproblems as written above, so it is of interest how the performance compares to them. On the other hand, there seems to be a general lack of comparison between most of the

Year	Source	Proposed Method	Compared Algorithms large-scale	Compared Algorithms normal	Benchmarks	# Variables	# Objectives	# Function Evaluations
2013	[3]	CCGDE3	-	GDE3, NSGA-II	ZDT1-3, ZDT6	200 - 5000	2	up to 10,000,000
2015	[24]	MOEA/DVA	-	NSGA-III, SMS-EMOA, MOEA/D	UF1-10, WFG1-9, DTLZ1, DTLZ3	24 - 1000	2 - 3	up to 3,000,000
2016	[25]	LMEA	MOEA/DVA	MOEA/D, NSGA-III, KnEA	DTLZ1-7, WFG3, UF9, UF10, LSMOP1-9	100 - 5000	3 - 10	1,000,000 - 230,000,000
2016	[67]	MOEA/D-RDG	MOEA/DVA	MOEA/D	UF1-10, WFG1-9	800 - 1000	2 - 3	10,000,000
2016	[79]	MOEA/D^2	-	MOEA/D, GDE3	DTLZ1-7	200 - 1200	3	100,000
2017	[68]	DPCCMOEA	CCGDE3, MOEA/DVA	-	DTLZ1-7, WFG1-9	1000	3	10,000,000
2017	[78]	ReMO	-	NSGA-II, MOEA/D	modified versions of ZDT1-3	10,000	2	3000
2018	[77]	S^3-CMA-ES	MOEA/DVA, LMEA	NSGA-III, RVEA, BiGE	LSMOP1-9	500 - 1500	5 - 15	5,000,000 - 15,000,000
2018	[70]	CCLSM	-	NSGA-II, IBEA, NSGA-III	WFG2-3, UF5, LSMOP1, LSMOP5, LSMOP9	100 - 300	2 - 10	50,000
2018	[92]	MOEA/D(s&ns)	-	NSGA-II	ZDT1-3, LSMOP1, LSMOP5, LSMOP9	200 - 300	2 - 3	?
2018	[71]	PEA	LMEA, MOEA/DVA	NSGA-III, MaOEA-R&D, BiGE	LSMOP1-3, MaF1-7	307 - 1039	3 - 10	3,070,000 - 10,390,000
2018	[47]	DLS-MOEA	CCGDE3, ReMO, WOF-SMPSO, MOEA/DVA, LMEA	SMS-EMOA, MOEA/D, NSGA-II	ZDT4, DTLZ1, DTLZ3, DTLZ6, WFG1-9, UF1-7	1024 - 8192	2	10,000,000
2019	[69]	LSMOF	WOF-NSGA-II, WOF-SMPSO, MOEA/DVA	NSGA-II, MOEA/D-DE, SMS-EMOA, CMOPSO	DTLZ1-7, LSMOP1-9, WFG1-9	200 - 5000	2 - 3	50,000

Table 4.3: Experimental evaluations of related work in the area of large-scale multi-objective optimisation with their compared algorithms, used benchmarks, dimensionalities and computational budget.

state-of-the-art large-scale methods. A reason might of course be the lack of available codes online. Another reason might also be that in scientific articles, space is usually limited and a comprehensive comparison might not fit in each of these works. However, it is also possible that the high frequency of publications made it difficult to obtain the current state-of-the-art for some of these works. A lot of the published methods are, even at the time of writing this thesis, only a few months old, and many of the listed articles might have been written or under reviews simultaneously. It is therefore of great interest to the field to compare the performance of the newest members of the large-scale area with each other in the future, and some comparison with available state-of-the-art, especially some of the newest algorithms, is provided in this thesis in Chapter 6.

Looking at the used benchmarks and their dimensionality, we also observe great differences. In total, among all the studies, the DTLZ functions were used most often, in 7 out of 13 articles. Following are the LSMOP (6 times) and WFG (6 times), UF (5 times) and ZDT (4 times) benchmarks. However, not all problems of the respective families were used in all articles. Some only picked one or two of the problems of the respective family and did not report the performance for the other ones. In the ReMO article, only modified versions of the ZDT1-3 function were used instead of the original ones. In summary, none of the 13 studies compared the performance on all of the most relevant benchmark problems in the DTLZ, UF, WFG, and LSMOP benchmark suites (where the ZDT functions are excluded due to their relatively low complexity).

We would also like to point out that the number of objective functions and variables in Table 4.3 are not to be understood in the way that exhaustive combinations of these numbers were tested with all of the used benchmark functions. In many works, only certain benchmarks were used with certain numbers of variables and objectives (for further details refer to Appendix A). Often, especially with the LSMOP benchmarks, the problems with lower numbers of objectives were used also with lower numbers of variables. This leads to the situation where the large-scale instances were at the same time also many-objective instances. And while the results are in some cases impressive, this kind of evaluation makes it of course difficult to examine separately the performance in large search spaces and the performance in large objective spaces. This is for instance visible in the evaluation of the S^3-CMA-ES algorithm, which was outperformed on several problem instances by the NSGA-III [23] and RVEA [21] algorithms, which are dedicated many-objective algorithms and originally not designed for large search spaces. Such results can indicate that the actual challenge of the used problem instances was more related to the high number of objective functions rather than the high-dimensional decision space.

Taking a closer look at the numbers reveals that the term “large-scale” is also not universally defined in the literature. While some works, for instance CCLSM and MOEA/D(s&ns) propose large-scale methods and tested them with at most 300 decision variables, on the other end of the scale CCGDE3, LMEA and LSMOF used up to 5000

variables, DLS-MOEA used between 1024 and 8192 variables, and ReMO performed experiments with $10,000$ decision variables. If we turn this around and ask the question how the large-scale algorithms perform on rather low-dimensional problems, there is surprisingly less experimental evidence in the literature. Problem instances with fewer than 100 variables were only tested with the DLS-MOEA. One might argue that it is not actually necessary to develop algorithms which are universally superior for any number of variables, as there is no free lunch and in a real application the dimensions of the problems are known. Therefore, an educated choice can be made to employ either traditional or large-scale methods. However, from a scientific point of view, it is still of interest which parts, i.e. building blocks, of large-scale methods might lead to deterioration of performance in traditional benchmark sizes.

Regarding the number of objectives, it was mentioned above that only LMEA, S^3-CMA-ES, CCLSM and PEA were tested on more than 3 objective functions. Both of the indicator-based approaches were not tested for more than 2 (DLS-MOEA) and 3 (LSMOF) objectives respectively. On the other hand, PEA, CCLSM and LMEA were tested with up to 10, the S^3-CMA-ES with up to 15-objective problems. Since the current large-scale approaches provide a variety of different techniques, it is likely that they might show different behaviours when facing many-objective problems, and an experimental evaluation in this regard might bring valuable insights for future developments in this area.

Finally, a very interesting observation that sets many of the algorithms apart is the computational budget that is used in the respective experiments. Since a variety of algorithms exist which are able to solve current large-scale benchmarks through different methodologies, a question of interest is the performance of algorithms over time. The author of this thesis published a study in this regard in 2017 [6] which showed that LMEA and MOEA/DVA are unable to reach acceptable results before millions of function evaluations are used up, while the WOF algorithm (see Section 5.1) is able to deliver good approximations after just $100,000$ evaluations. This is due to the large overhead these two methods need for obtaining interaction-based groups. In the meantime since the study was conducted, however, a variety of new algorithms has been proposed, and many of them do not rely on interaction-based groups any more, as was discussed above.

Therefore, we now take a look at the computational budget these methods use in their original implementations by comparing the experimental evaluation in the respective articles. In the last column of Table 4.3 the number of function evaluations used in the articles' experiments is listed. Out of the 13 large-scale algorithms, only the ReMO, CCLSM, MOEA/D^2 and LSMOF use fewer than 1 million function evaluations. A number of 3 to 10 million is often used in most of the studies. In the extreme case, the experiments in LMEA used up to $230,000,000$ evaluations to test the algorithm's performance. Sometimes these high numbers may have been chosen due to a comparison with other algorithms which need a large overhead. In many cases, however, it is not clear

how the performance of the algorithms relates to the computational budget. Experiments regarding convergence behaviour (i.e. development of performance over time) were only rarely conducted in the related literature.

Furthermore, it is not clear whether these high numbers are feasible to realistic scenarios of real applications. As an example, the LMEA article reports an average runtime of around 2386 seconds for the 1000-variable instances, which is approximately 40 minutes for a single run. Although these numbers are faster than the reported times for other methods like MOEA/DVA, it corresponds to an average of 7124 function evaluations per second. Assuming that a real-world problem might need additional computational resources of only 1 second per evaluation (e.g. if physical simulations are part of the evaluation), such an algorithm would run for almost 200 days to approximate a 1000-variable problem. And although parallel computing might be able to reduce the time, this further emphasises the need for sophisticated search strategies which make use of the computational budget in an efficient and effective manner.

4.2 Comparison and Classification of Grouping Methods

In this section, we compare and categorise existing grouping mechanisms briefly. As mentioned in the previous chapter, the list of grouping mechanisms in this thesis is not exhaustive due to the large amount of variants in the single-objective literature. An extensive study of the properties and differences of variable grouping methods is an ongoing research topic. However, in the context of methodologies of large-scale methods, which is the focus of this thesis, it is of less importance how exactly the groups were created. It is in contrast of higher importance which influence these groups, especially the interaction-based groups, have on the success of the search process. This topic is addressed experimentally in Chapter 6. The relevant methods that are explained in Section 3.3 are compared and classified in this section.

For classification of grouping methods, we propose three different categories in this thesis. In Section 3.3 a number of methods to group variables together have already been identified. These grouping mechanisms can, as previously described, be divided into three main categories as follows.

1. *Simple methods* which do not use objective function information and are therefore applicable to single- as well as multi-objective problems.
2. *Interaction-based methods*, which take into account the change of objective functions and aim to identify interacting variables. These methods can be designed for either single- or multi-objective problems.
3. *Contribution-based methods*, which are only useful in multi-objective optimisation, as they divide based on contribution of variables to convergence or diversity of the solution set.

The resulting classification of the methods is shown in Table 4.4. In addition to the respective category, we list further criteria of the grouping mechanisms in the table. In the following, we briefly go into detail on these properties of the variable grouping mechanisms, and explore whether the method is applicable to multi-objective optimisation, how many parameters the user needs to set, whether group sizes are fixed or automatically allocated, and which computational budget the mechanisms consume.

The respective methods were already grouped by categories in the previous chapter, and each of the categories was introduced. As described, not all methods are applicable to multi-objective problems without further modification. It is, however, an advantage that all methods in the "simple" category can be applied to single-, multi- and many-objective optimisation without further modifications. Further, contribution-based groups are only applicable to multi-objective problems, as the notions of convergence and diversity might not have the same meaning or importance in the single-objective area. Regarding the interaction-based methods, we see that Differential Grouping and Differential Grouping 2 can not be applied to multi-objective problems, as they measure the differences in fitness values. However, the Interdependence Analysis that was proposed in MOEA/DVA and is used in a couple of the related algorithms, basically follows a similar idea, and is applicable to multiple objectives only because the interaction information in each single function is combined in a certain way before assigning groups. A generalisation of such a combination mechanism for interaction-information was proposed in [26].

Regarding the parameters that need to be set and the group sizes, we see that simple methods usually require the user to set only one parameter, which determines either how big each group is, or the resulting number of groups. Since these methods do not possess any kind of information about the problem, there is no intelligent way to determine group sizes automatically. In contrast, the interaction-based approaches and contribution-based methods usually set the sizes and numbers of groups automatically. While this is an advantage, it might also result in undesirable situations. For instance, in [26] it was shown that in case many interactions are present, large groups might be formed, which in the extreme case might lead to only one large group which contains all variables. While this situation might be optimal in terms of optimising interacting variables together, it renders the dimensionality reduction useless in the methods which depend on the CC-based approaches.

The Random-based Dynamic Grouping which is used in MOEA/D-RDG plays a special role among the grouping mechanisms. This method does not directly compute any interaction information, but rather uses it implicitly through learning which group sizes were successful in the past. Therefore, over time the algorithm might learn how many independent variable groups exist in the problem, although the groups are still created randomly, and there is no guarantee that the interacting variables end up together. Nonetheless, RDG turned out to work well, and it has the advantage of not using any function evaluations. A drawback would be that the user has to set the highest number of

parameters compared to the other methods. RDG requires to specify k group sizes, which the algorithm can choose from, and which efficiencies are learned. In total, however, it might still be a valid trade-off to set more parameters in the beginning in exchange for saving millions of function evaluations for the analysis in the other methods.

If we further compare the computational budget, we see that simple methods in general require no function evaluations at all, as the definition of simple methods exactly defines this property. We see further that the contribution information can usually, in both known methods, be obtained with a budget that rises linear in the number of variables, while all interaction-based methods suffer from quadratically rising computational costs. Note that the RDG method is classified here as an interaction-based method, although it has a computational budget of zero. RDG does not require additional function evaluations prior to the optimisation process, but on the other hand can not work without any evaluations at all, since the information about the success of the optimisation is needed to determine optimal group sizes. It can therefore be seen partly as a simple, partly as an interaction-based approach.

4.3 Summary

This chapter summarised and characterised the existing approaches in the area of large-scale multi-objective optimisation. First, the approaches from the literature are analysed based on their building blocks. Based on the simplified structures of the algorithms, the elements that frequently occur were identified in Table 4.1. The similarities and differences of the methods were then analysed based on different criteria. This lead to the classification of existing algorithms into multiple categories, based on their methods of dimensionality reductions, diversity management, possible parallel implementations and their capabilities to deal with many-objective problems. These findings were summarised in Table 4.2. After that, the algorithms were analysed based on their experimental performances in the literature and their required function evaluations (Table 4.3).

In general, we saw that the CC-based method to optimise large search spaces is the most prominent one among the existing algorithms. Other approaches like the transformation of the problem are used as well, which come with multiple advantages like a possible independence of interaction-based variable groups. Comparing the results of the experiments in the literature, it is also interesting to see that in the development of the area, which happened mainly in the last three years since the beginning of 2016, no clear superior method has emerged yet, and a variety of approaches of dimensionality reduction and diversity management seem to be successful. A major drawback of the existing work in this area is the lack of common experimental settings in terms of function evaluations. Most work focused on solving the benchmark problems as well as possible, but not many papers set these results into perspective with the required amount of computational budget. Especially the methods which rely on interaction-based groups

Year	Source	Method	Category	Multi-objective	# Parameters	Group sizes	Computational complexity
-	-	Random Grouping	Simple	✓	1	fixed	0
-	-	Linear Grouping	Simple	✓	1	fixed	0
2014	[84]	Differential Grouping	Interaction	✗	1	automatically	$O(\frac{n^2}{l})$ Worst-case: $n \cdot (n-1) + 2 \cdot n$
2015	[24]	Interdependence Analysis (MOEA/DVA)	Interaction	✓	1	automatically	$\frac{3n(n-1)*NIA}{2}$
2015	[24]	Control Variable Analysis (MOEA/DVA)	Contribution	✓	1	automatically	$n \cdot NCA$
2016	[67]	Dynamic Random (DRG)	Interaction / Simple	✓	k	dynamic	0
2016	[25]	Clustering-based Contribution Groups (LMEA)	Contribution	✓	2	automatically	$nSel \cdot nPer \cdot n$
2017	[87]	Differential Grouping 2	Interaction	✗	-	automatically	$\frac{n(n+1)}{2} + 1$

Table 4.4: Comparison of related grouping methods based on different categories.

might be at a disadvantage in this regard, as their required budgets rise quadratically with the dimensionality of the search space.

In Section 4.2, a classification and analysis was presented regarding the different categories of grouping mechanisms in the literature. The classification in simple methods, interaction-based and contribution-based methods is build on the information which the grouping methods aim to extract from the problem. Further criteria were used such as the ability to be applied to multiple objectives, the number of free parameters, the sizes of the obtained groups and the required computational budgets.

CHAPTER 5

Proposed Approaches for Large-scale Optimisation

In this chapter, the three developed approaches and contributions to large-scale multi-objective optimisation are described and analysed. These are different approaches that either improve the performance of existing algorithms on their own or in combination. The main algorithm developed in [4, 1] is the so-called *Weighted Optimisation Framework*. This approach belongs to the transformation-based methods, and makes use of dimensionality reduction, variable grouping methods and uses existing metaheuristics to optimise the transformed optimisation problems. This framework is explained in detail in the following Section 5.1. The second approach is the incorporation of variable groups inside existing mutation operators. This approach was published by the author in [5] and is introduced in further detail in Section 5.2. A third approach to tackle high-dimensional optimisation is described in Section 5.3, which uses linear combinations of existing solutions to reduce the dimensionality and to increase diversity at the same time. This approach was published by the author earlier in 2019 in [7].

The descriptions of the three methods are based on the respective original publications by the author. In addition, some improvements or adjustments have been made which differ from previous versions, and any differences in the implementations in this thesis to previously published versions of the algorithms are outlined in the respective sections. Section 5.4 gives a summary and the proposed approaches are classified and analysed based on the categories and criteria proposed in Chapter 4.

5.1 The Weighted Optimisation Framework

The Weighted Optimisation Framework (WOF) was proposed by the author of the thesis in [4, 1] and further extended in [6]. The aim of this framework is to reduce the dimensionality of optimisation problems, so that existing population-based algorithms are able to perform the optimisation in this new transformed search space in an efficient way.

The first version was proposed in 2016, which makes WOF the first transformation-based large-scale multi-objective algorithm developed.

Similar to coevolutionary methods, WOF requires a distribution of the decision variables into groups. In contrast to CC, however, the goal is not to optimise the variables in each group independently, but at the same time and by the same amount relative to their domain. The reduction of the problem's dimensionality is not achieved by keeping variable values in other groups fixed while focusing on one group. Instead, so-called *Transformation Functions* are used to optimise all variables in each group at the same time. In this way, the proposed approach provides a framework that performs dimensionality reduction and optimisation in smaller search spaces in a different way than existing coevolutionary methods. This proposed framework requires the use of a grouping mechanism, but is designed to be generic so that any grouping mechanism and metaheuristic optimiser from the literature can be used. It is noteworthy that the 2019 LSMOF method, which is described above, is strongly based on WOF, and uses similar strategies to perform the problem transformation. Respective differences between WOF and LSMOF are highlighted below.

In the following, we first explain the concept of problem transformation and the effect of the corresponding dimensionality reduction on the search space. The following subsections continue with the outline and details of the WOF algorithm. Different transformation functions as well as mechanisms to choose the necessary pivot solutions for the transformation are presented. After examining other parameters of the framework, a discussion is given in Section 5.1.8. The information and statements of this section are based on the contributions made by the author in [4, 1, 6].

5.1.1 Problem Transformation

The problem transformation, from a theoretical point of view, is essentially a method to create a new – possibly simpler – optimisation problem by applying a mathematical operation to the original problem.

Let Z be a multi-objective optimisation problem with n decision variables and m objectives as shown in Equation Eq. (2.1), where each variable x_i is real and has a domain with minimum value $x_{i,min}$ and a maximum value $x_{i,max}$. As described, the goal is to find the set of Pareto-optimal solutions $PS := \{\vec{x}^*\}$ (Pareto-set) and the corresponding objective values $PF := \{\vec{f}(\vec{x}^*)\}$ (Pareto-front) [1]. A transformation function to reformulate this problem is defined in Definition 5.1.

Definition 5.1 (Transformation Function) *A Transformation function ψ is a function*

$$\begin{aligned} \psi : [x_{i,min}, x_{i,max}] \times [w_{i,min}, w_{i,max}] &\rightarrow [x_{i,min}, x_{i,max}] \\ (x_i, w_i) &\mapsto \psi(x_i, w_i) \end{aligned}$$

The transformation function maps the domain of a single variable x_i together with a real value w_i back into its domain $[x_{i,min}, x_{i,max}]$. As an example, consider the simple transformation function $\psi(x_i, w_i) := x_i \cdot w_i$ with $x_i \in [0, 10]$ and $w_i \in [0, 1]$. For a given value of x_i and a weight value w_i we obtain a new value that lies in the domain of x_i. Note that a transformation function is neither required to be injective nor surjective, which allows transformation functions to limit the searchable space (for details see Sections 5.1.2 and 5.1.3). In the above example, the transformation is surjective, since every value in the domain of x_i can be reached by at least one combination of x_i and w_i. It is not injective, because if $x_i = 0$, all values of w_i map to 0. An example for a non-injective and non-surjective transformation function would be $\psi(x_i, w_i) := x_i \cdot 0.2 \cdot w_i$ with $x_i \in [0, 10]$ and $w_i \in [0, 1]$, since this function maps only to the interval $[0, 2]$.

Since in this work we deal with a vector $\vec{x}$ of decision variables, the transformation function in the following steps has to be applied to each variable of the optimisation problem, together with a corresponding weight value. For simplicity, we write $\psi(\vec{x}, \vec{w})$ as a notation indicating that the function is applied to each of the variable pairs (x_i, w_i) independently.

$$\psi(\vec{x}, \vec{w}) = \psi((x_1, ..., x_n), (w_1, ..., w_n)) := (\psi(x_1, w_1), ..., \psi(x_n, w_n)) \tag{5.1}$$

This concept is used to transform the original optimisation problem into a new one. To illustrate how this works, the following explanations use, without loss of generality, the above mentioned multiplication transformation function $\psi(x_i, w_i) := x_i \cdot w_i$. Using this transformation, the original fitness value $\vec{f}(\vec{x})$ can be written as $\vec{f}(\psi(\vec{w}, \vec{x}))$, with $\vec{w} = (1, ..., 1)$ a vector of ones, since

$$\vec{f}(\psi(\vec{w}_i, \vec{x})) = \vec{f}(w_1 x_1, ..., w_n x_n) = \vec{f}(x_1, ..., x_n) \tag{5.2}$$

To change the given solution vector $\vec{x}$ for the optimisation process, we can now change the values of $\vec{w}$. In this way, instead of optimising the vector $\vec{x}$, for any fixed values of $\vec{x}$, a vector $\vec{w}$ can be optimised to approximate an optimal solution [1].

It can be easily seen that the optimal solutions to the optimisation problem can still be found by changing the values of $\vec{w}$, provided an unbounded maximum for the domains of the w_i and all $x_i > 0$. In this case, for the optimal weights we obtain $\vec{f}(\psi(\vec{w}^*, \vec{x})) = \vec{f}(\vec{x}^*)$ for a fixed $\vec{x}$.

$$\begin{aligned} &\forall x_i : \exists w_i : w_i x_i = x_i^* \\ &\qquad\qquad \Rightarrow \\ &\forall \vec{x} \in \mathbb{R}^n : \exists \vec{w} : \psi(\vec{w}, \vec{x}) = \vec{x}^* \end{aligned} \tag{5.3}$$

By picking a fixed solution $\vec{x}'$, it is possible to reformulate the original optimisation problem Z with a transformation function to a new problem $Z_{\vec{x}'}$ which optimises the vector $\vec{w}$ using $\vec{x}'$ as an input parameter [1]:

$$\begin{aligned} Z_{\vec{x}'} : min \quad & \vec{f}_{\vec{x}'}(\vec{w}) = (f_{1,\vec{x}'}(\vec{w}), ..., f_{m,\vec{x}'}(\vec{w}))^T \\ s.t. \quad & \vec{w} \in \Phi \subseteq \mathbb{R}^n \\ & f_{o,\vec{x}'}(\vec{w}) = f_o(\psi(\vec{w}, \vec{x}')) \quad \forall o = 1, ..., m \\ & \psi(\vec{w}, \vec{x}') := (w_1 x'_1, ..., w_n x'_n) \end{aligned} \tag{5.4}$$

It must be noted that the multiplication is not necessarily a good choice for a transformation function. In practice, some x_i can be equal to zero and w_i has to have an upper bound. The more sophisticated transformation functions which are described in Section 5.1.5 are more suitable for application in the optimisation.

As a result of this process, a new problem $Z_{\vec{x}'}$ is created. This new optimisation problem still has the same number of decision variables as the original one, as $|w| = |x| = n$. However, the concept of transformation is used in the following to reduce the amount of variables to be optimised, i.e. to reduce the size of the vector $\vec{w}$.

5.1.2 Dimensionality Reduction

To reduce the dimensionality of the problem, the concept of variable groups is used as introduced in Section 2.4. With the help of variable groups, we are able to reduce the dimensionality of the vector $\vec{w}$ as follows. The original n variables of the problem Z are divided into a number of γ groups $(G_1, ..., G_\gamma)$ using a grouping mechanism Γ. These can be evenly sized or obtained by any other simple or intelligent mechanism. For simplicity, and without loss of generality, it is assumed in the following argumentation that evenly sized groups, each of size l (hence $\gamma \cdot l = n$) are used. This assumption can later be relaxed again.

Since the new, transformed problem optimises the variables in the vector $\vec{w}$, the task is now to reduce its dimensionality. Instead of assigning one w_i for each x_i in the transformation function, only one weight w_j is used for all variables within the same group G_j with l variables. The vector to be applied inside the transformation functions will then be

$$(\underbrace{w_1, w_1, w_1, ..., w_1}_{l \text{ times}}, ..., \underbrace{w_\gamma, w_\gamma, w_\gamma, ..., w_\gamma}_{l \text{ times}}) \tag{5.5}$$

Since there are only γ different values to be optimised, the vector $\vec{w}$ effectively reduces to $(w_1, ..., w_\gamma)$. Therefore, the above-mentioned example transformation function can then be reformulated as:

$$\psi(\vec{w}, \vec{x}) = (\underbrace{w_1 x_1, ..., w_1 x_l}_{\text{use } w_1}, ..., \underbrace{w_\gamma x_{n-l+1}, ..., w_\gamma x_n}_{\text{use } w_\gamma}) \tag{5.6}$$

As a result, the size of the vector $\vec{w}$ is reduced to $\gamma \ll n$ (same as the number of groups), and the newly formed optimisation problem $Z_{\vec{x}'}$ has then only γ decision variables. We can now pick an arbitrary but fixed solution $\vec{x}'$, and compute an approximation of the Pareto-set by optimising this new optimisation problem instead [1]. These fixed solutions that need to be picked before using the transformed problem are called "pivot solutions" or "candidate solutions" in the following.

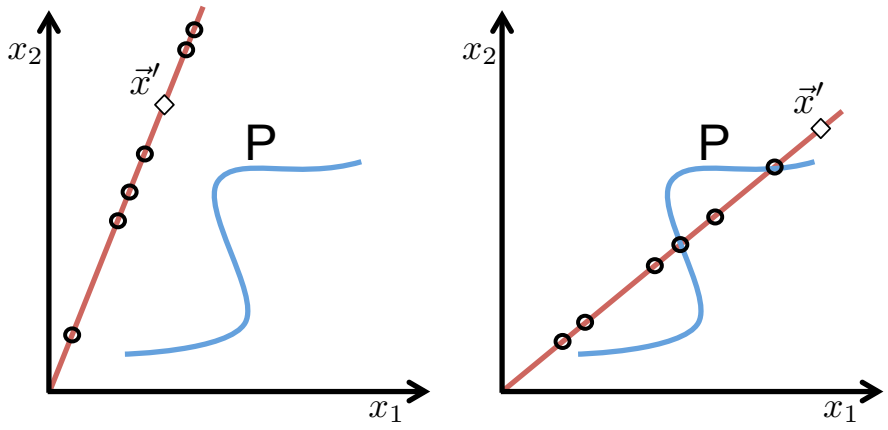
The drawback of this approach is that this reduction of the dimensionality limits the search to certain solutions that can be reached within the transformed problem. This happens due to the non-surjective transformation function, but also and more importantly due to the reachable subspace defined by the variable groups. By grouping the original variables together and applying the same w_j to a number of them, their values cannot be changed independently of each other any more, as they are all changed using the same w_j. This substantially limits the reachable solutions in the original search space. Geometrically, this dimensionality reduction can be seen as cutting out a γ-dimensional subspace out of the n-dimensional original search space. The searchable subspace of Ω, which can be reached by optimising $Z_{\vec{x}'}$, is defined mainly by three choices [1]:

1. The choice of the pivot solution $\vec{x}'$
2. The amount of groups and the decision of which variables are put together in the same group, i.e. the used mechanism Γ
3. The choice of the transformation function $\psi(\cdot)$

1. Choice of Pivot Solutions

An appropriate choice of $\vec{x}'$ is important. The chosen solution remains fixed for the time of optimising the weights and its variables are used for the creation of new solutions through applying the weight variables. Together with the two other points listed above, it defines the reachable subspace of Ω. In the following, a small example is used to illustrate this effect.

As an example, consider an optimisation problem with two decision variables x_1 and x_2. In Fig. 5.1 the decision space of this problem is shown together with a hypothetical



(a) "Suboptimal" choice of $\vec{x}'$ (b) "Better" choice of $\vec{x}'$

Figure 5.1: Examples for different choices of the pivot solution $\vec{x}'$ in the decision space. x_1 and x_2 belong to the same group, the resulting reachable subspace is shown as a red, straight line. The Pareto-set is shown as the line P in blue colour. The suboptimal pivot in Subfigure (a) does not lead to the discovery of optimal solutions, but can help to advance closer to them. Thus, it can still be useful for the overall search. Graphic taken from [1].

Pareto-set P. When we optimise the original problem Z, both of the variables can be altered separately, and therefore the whole decision space can be searched. As a result, it is possible to obtain a good approximation of P.

At this point, we use a variable grouping mechanism and put both of the variables, x_1 and x_2, into the same group. As described above, to reduce the dimensionality of the problem we apply only a single variable w_1 for all variables in this group inside the transformation function. Thus, we reduce the dimensionality of the problem from $n = 2$ to $\gamma = 1$. The newly formed optimisation problem $Z_{\vec{x}'}$ results in the optimisation of just one single variable w_1. This limits the reachable solutions to a 1-dimensional subspace [1]. If again multiplication is used as the transformation function as in the example above, the new search space of $Z_{\vec{x}'}$ is denoted as a red line from a chosen pivot solution $\vec{x}'$ to the origin in Fig. 5.1.

When we compare the scenes of Fig. 5.1a and Fig. 5.1b, we observe that the ability to reach the optimal solutions in this new search space depends on the choice of $\vec{x}'$. The same principle applies in higher dimensional search spaces. Every group of variables defines a subset of solutions that can be reached by the new optimisation process. It is to be expected, however, that even "suboptimal" choices of $\vec{x}'$ as in Fig. 5.1a can help the overall search process to advance closer to certain parts of the Pareto-set [1].

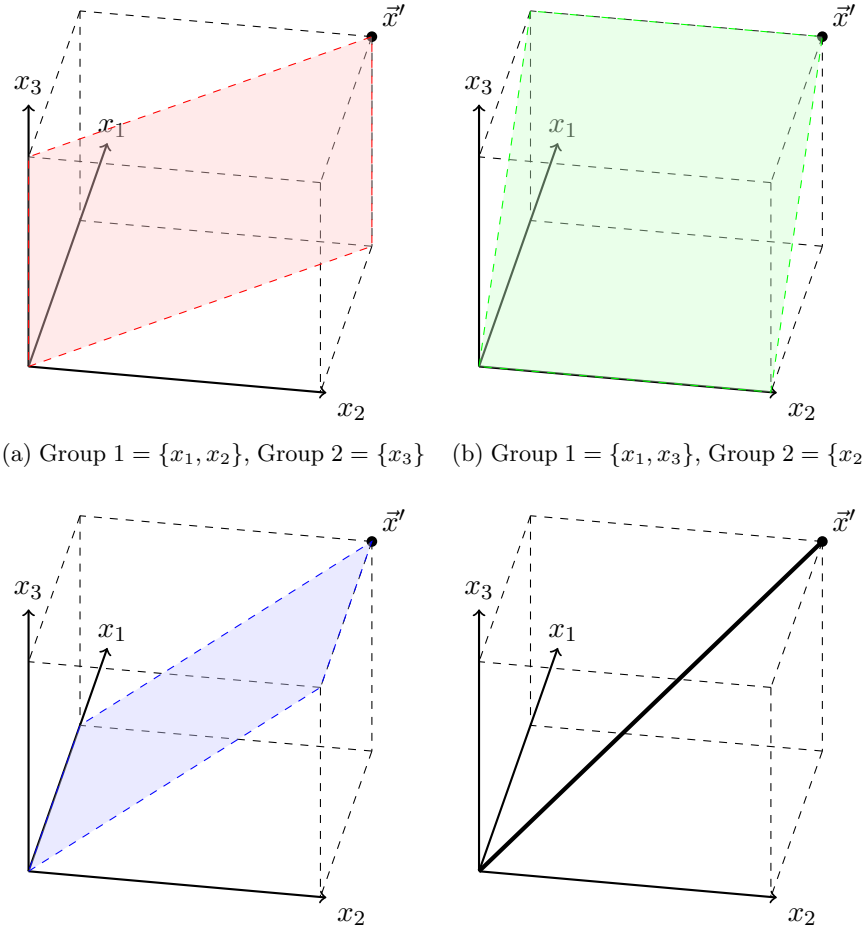
2. Choice of Grouping Mechanism

The grouping of the variables has, similar to the location of $\vec{x}'$, an influence on the subspace of Ω in which the search takes place [1]. In the example above, this can not directly be observed since there are only two decision variables. However, if we extend this example to a 3-dimensional search space, we find a situation as exemplarily shown in Fig. 5.2. A 3-dimensional search space is shown, where we use the solution $\vec{x}'$ as pivot to perform the transformation. We can see here that even if using the same solution $\vec{x}'$ in all subfigures, the searchable subspace of the original 3-dimensional space largely differs depending on which variables end up together in the same group. In Fig. 5.2a, x_1 and x_2 are grouped together while a second group consists only of x_3. The resulting search space is 2-dimensional, since there are two groups and therefore two variables w_1 and w_2. The resulting subspace that can be reached by optimising these two variables is denoted as a red shaded area in the figure. We can now observe that this subspace differs largely in Figs. 5.2b and 5.2c. These show the same situation with different groups, when x_1 and x_3 or x_2 and x_3 are put in a common group respectively. The last option is to put all variables in the same group and optimise only one weight for this group as seen in the previous 2-dimensional example. The corresponding situation in this 3-dimensional case is shown in Fig. 5.2d, where only a 1-dimensional subspace can be reached by optimising the one variable w_1, which changes all three original variables x_1, x_2 and x_3 at the same time.

3. Choice of Transformation Function

The third main influence for the reachable subspace of the newly formed problem is the transformation function. In addition, the transformation function also defines in which way the solutions' locations in the newly created search space change in response to a change in the variables w_j. In addition, it can affect the geometry of the lower-dimensional space, e.g. by using linear or non-linear transformation functions.

By defining a subspace in which new solutions can be produced based on the original values of $\vec{x}'$, the transformation process can also be regarded as a kind of "neighbourhood" function of the privot solution. However, in mutation operators or local search methods, the term "neighbourhood" is usually used to indicate a locality around the original solution. This is, however, not the case for the solutions created by the transformation process. Given that the transformation is usually used to produce more diverse solutions and explore larger areas of the original search space, such a locality is actually not desirable in this situation. Instead, it is assumed that a transformation can enable an algorithm to make large steps in the search space, depending on the values of the w_j. Different examples of transformation functions are given below.



(a) Group 1 = $\{x_1, x_2\}$, Group 2 = $\{x_3\}$ (b) Group 1 = $\{x_1, x_3\}$, Group 2 = $\{x_2\}$

(c) Group 1 = $\{x_1\}$, Group 2 = $\{x_2, x_3\}$ (d) Group 1 = $\{x_1, x_2, x_3\}$

Figure 5.2: Example of the influence of different variable groups on a 3-dimensional example with a linear transformation function.

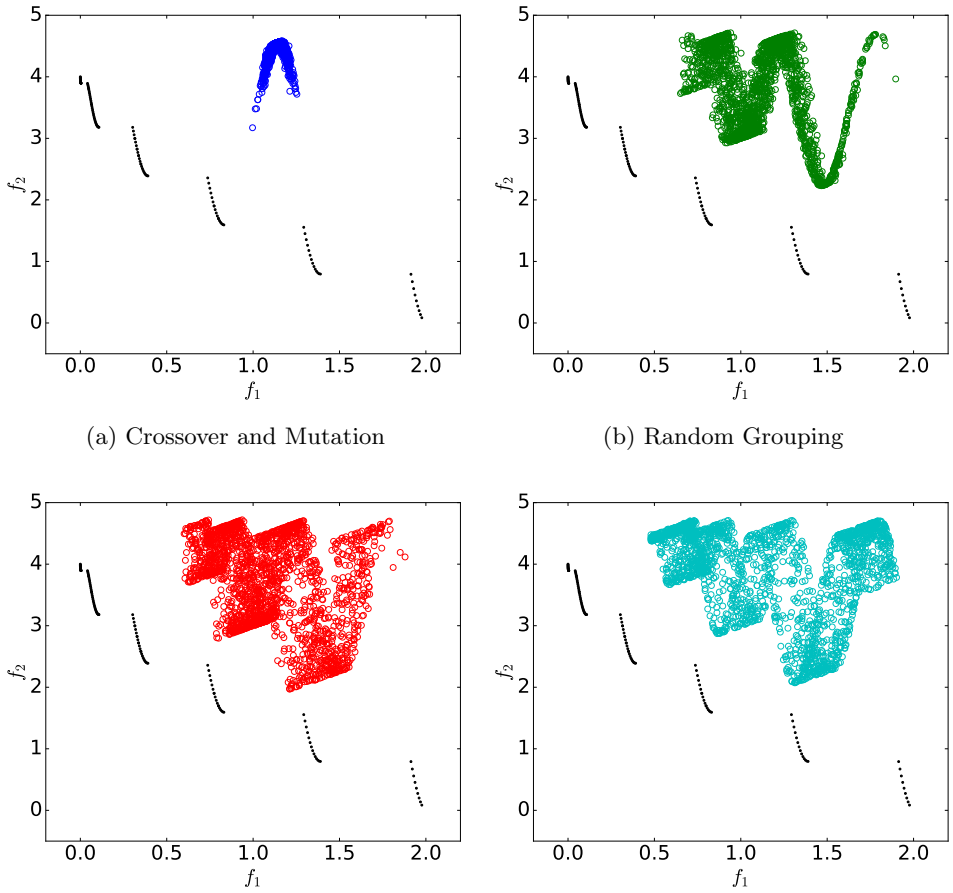
5.1.3 Influence of the Transformation on the Search Space

To further examine the effects and implications of the problem transformation, we perform some exemplary experiments. The following descriptions of these experiments are based on those in [1]. In Figs. 5.3 to 5.5, the effects of these steps are exemplarily shown on the three test problems WFG2 [28], UF2 [48] and ZDT1 [39] with $n = 1000$ decision variables each. In all three instances, we first created a random initial solution for the problem. From this solution, 2000 new solutions were created by different methods as explained in the following.

- In Figs. 5.3a, 5.4a and 5.5a a number of 2000 solutions are created from this initial solution by using a standard SBX crossover [100] with another randomly created solution and polynomial mutation [101, 102]. The mutation and crossover

parameters were standard values often used in the literature, i.e. polynomial mutation with a probability of $1/n$ (with n being the number of decision variables) and a distribution index of 20.0. The SBX crossover is used with a probability of 0.9 and a distribution index of 20.0.

- In the respective subfigures (b), (c) and (d), instead of using traditional genetic operators, 2000 random solutions are created from the same initial solution by creating random weight vectors $\vec{w}$ and by applying them in the proposed problem transformation with different grouping mechanisms. The initial solution is the same as in subfigures (a) and it is used as the pivot solution $\vec{x}'$ in these transformations. In Figs. 5.3b, 5.4b and 5.5b the variables were grouped using a random grouping mechanism (with $\gamma = 4$ groups), the remaining figures show the usage of ordered grouping (see Section 5.1.6) and Differential Grouping (see Section 3.3.3), where in the last variant DG is only applied to the first objective function.



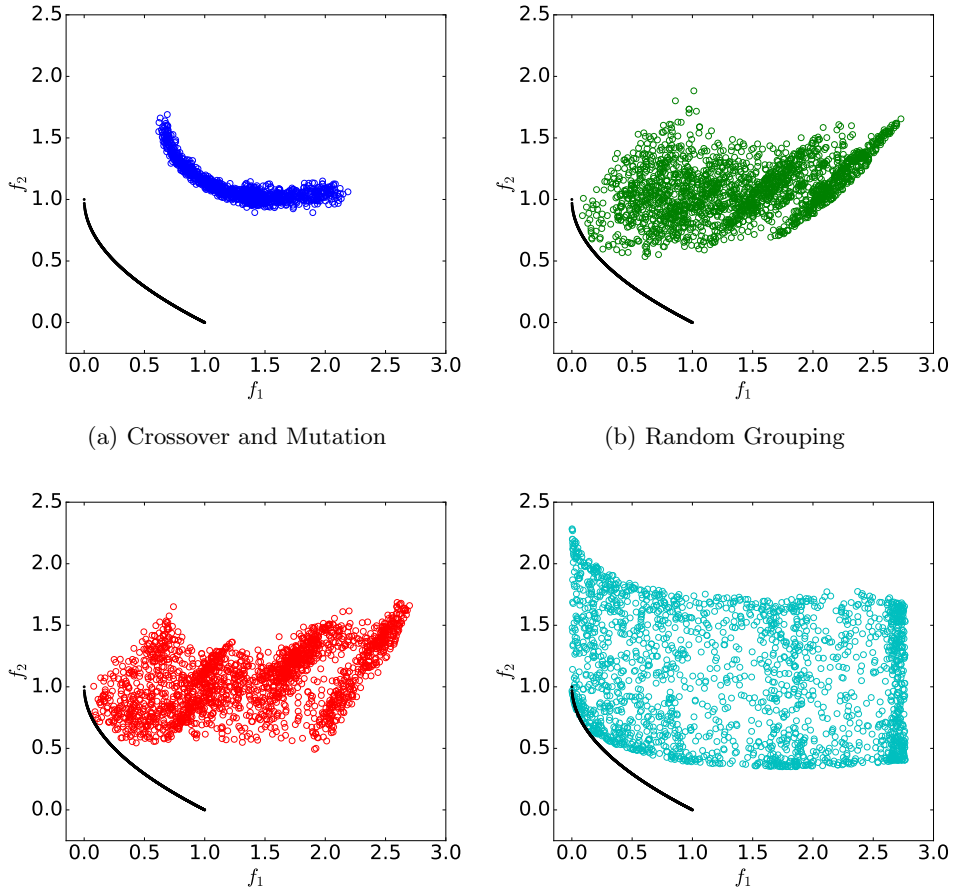
(a) Crossover and Mutation

(b) Random Grouping

(c) Ordered Grouping

(d) Differential Grouping

Figure 5.3: 2000 randomly created solutions for the WFG2 benchmark by using genetic operators and different grouping methods in the WOF transformation approach. The Pareto-front is shown as small black points. Graphic taken from [1].



(a) Crossover and Mutation

(b) Random Grouping

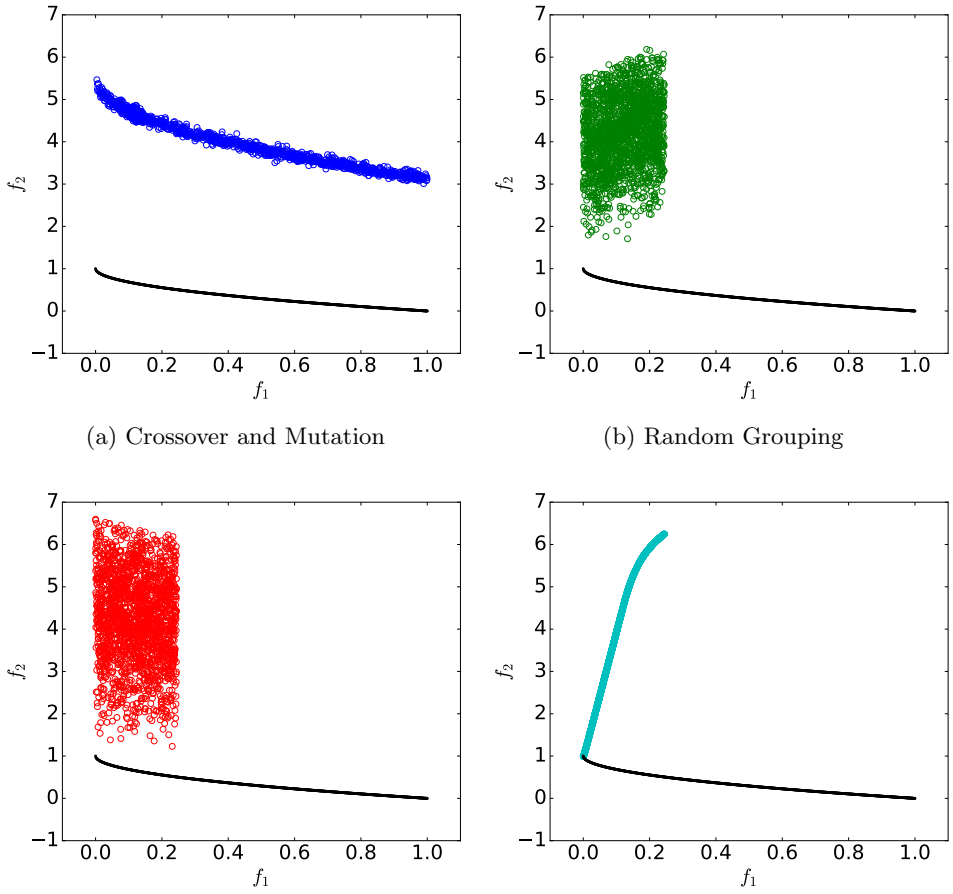
(c) Ordered Grouping

(d) Differential Grouping

Figure 5.4: 2000 randomly created solutions for the UF2 benchmark by using genetic operators and different grouping methods in the WOF transformation approach. The Pareto-front is shown as small black points. Graphic taken from [1].

We can observe large differences between the solutions obtained by the problem transformation process and the ones created by the genetic operators. In the case of WFG2, the different variants of the WOF transformation show solution sets that offer a far higher diversity and proximity to the Pareto-optimal solutions, compared to the solutions created by SBX crossover and polynomial mutation. This shows that the newly created search space contains very promising solutions for the overall search process. As a consequence, using an optimisation algorithm to obtain the best solutions reachable in this transformed space, we can expect an improved performance, in terms of convergence speed and diversity, towards promising solutions.

Looking at the UF2 problem in Fig. 5.4, we observe a similar situation. Even though there are slight differences between the different grouping mechanisms, all three of the transformed problems include solutions that lie very close to the true Pareto-front.



(a) Crossover and Mutation

(b) Random Grouping

(c) Ordered Grouping

(d) Differential Grouping

Figure 5.5: 2000 randomly created solutions for the ZDT1 benchmark by using genetic operators and different grouping methods in the WOF transformation approach. The Pareto-front is shown as small black points. Graphic taken from [1].

Furthermore, regarding the diversity of solutions, the transformed problems offer the possibility to reach a set that is at least as diverse as the one reached by classical operators.

Next, we look at the results of the ZDT1 problem in Fig. 5.5. In contrast to the positive influences of the transformation in the previous two examples, we now observe a loss of diversity. In Fig. 5.5a, the solutions obtained by the crossover and mutation operators are spread widely across the objective space, providing a very diverse set of solutions. This is achieved because these solutions are produced through a crossover with other, randomly created solutions. In contrast, the solutions created by the transformation process are concentrated in a certain part of the objective space (Figs. 5.5b to 5.5d). A reason for this effect is that the selected pivot solution $\vec{x}'$, which influences the reachable search space, happens to be on the "left side" of the shown objective space. The optimisation

of the transformed problem in this case can lead to fast convergence, especially in the case of Fig. 5.5d, but the diversity of the solutions may suffer. This effect is caused by the structure of the ZDT1 problem, since the diversity of the solution set is only defined by the first decision variable x_1, and the used transformation function in this case was the multiplication as described above. Therefore, since the lower and upper bounds of the w_j were set to 0.0 and 2.0 respectively, the transformed problem is not able to reach certain parts of the search space, as it can only alter the variables within a certain interval, based on the values in $\vec{x}'$. As mentioned above, this emphasises that other transformation functions than simple multiplication may be more suitable for the optimisation process of the transformed problem.

We can conclude from these three examples that, if appropriate choices for groups, transformation function and $\vec{x}'$ are made, the proposed transformation of the problem can result in an accelerated optimisation process in terms of both diversity and convergence. It is, however, important not to sacrifice the diversity in favour of a fast convergence, as in the case of Fig. 5.5d. In the following, these observations are used to propose the optimisation strategy of the WOF for large-scale problems.

5.1.4 The WOF Algorithm

The Weighted Optimisation Framework (WOF) is designed as a generic population-based meta-heuristic. Its main contribution is the transformation of the original problem, which creates new optimisation problems of lower dimensionality. These transformed problems are optimised using another optimisation algorithm within the framework. The choice of this second algorithm to optimise the transformed problems can, for instance, depend on the problem's properties, e.g. a many-objective metaheuristic may be used in a problem which contains more than 3 objective functions. This metaheuristic is used inside the framework for optimising two different problems Z and $Z_{\vec{x}'}$ in turns. It is further possible to use two different optimisation algorithms for the two problems. For instance, a classical evolutionary algorithm may be used for the transformed, low-dimensional problem, while the original, large-scale problem can be optimised with another large-scale method from the literature [1]. In the following, however, we concentrate on using only one optimiser for both problems.

A drawback of the transformed problem $Z_{\vec{x}'}$ is that it limits the search space that is accessible for the algorithm. On the other hand, it enables the algorithm to search a space with lower dimensionality more thoroughly, which can result in a faster convergence. The original problem Z in contrast can reach all possible solutions, but the ability to achieve a fast convergence and good exploration is limited in the high-dimensional space. To utilise the synergy of these two, WOF alternates two different phases of optimisation: A *regular* optimisation step and another *transformed* optimisation step, using multiple pivot solutions. The outline of this proposed method is shown in Fig. 5.6 and more detailed steps are given as pseudocode in Algorithm 3, Algorithm 4 and Algorithm 5 [1].

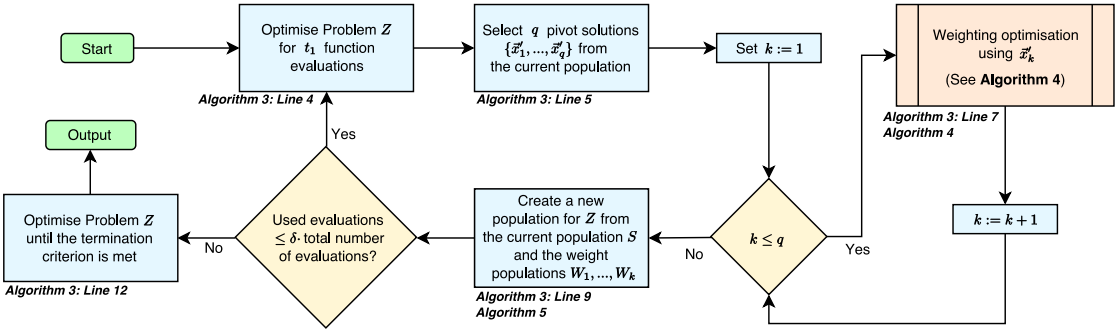


Figure 5.6: Outline of the Weighted Optimisation Framework. Graphic based on [1].

The inputs for the WOF algorithm are an optimisation problem Z, a population-based metaheuristic A, a grouping mechanism Γ (see Section 2.4) and a transformation function ψ (see Section 5.1.1). First, the population of the algorithm is initialised with a random population for the problem Z. In the main loop of the algorithm (Lines $3-10$ in Algorithm 3), the two different optimisation phases are carried out. First, the original problem Z is optimised using algorithm A for a predefined number of t_1 function evaluations (Line 4 in Algorithm 3).

Algorithm 3 WOF(Z,A,Γ,ψ) - Pseudocode based on [1].

Input: Problem Z, Optimisation Algorithm A, Grouping Mechanism Γ, Transformation Function ψ
Output: Solution population S

1: Initialisation
2: $S \leftarrow$ Random initial population for Z
3: **repeat**
4: $S \leftarrow A(Z, S, t_1)$ // Optimise Z with Algorithm A for t_1 evaluations, using S as a starting population.
5: $\{\vec{x}'_1, .., \vec{x}'_q\} \leftarrow$ Selection of q pivot solutions from S
6: **for** $k = 1$ **to** q **do**
7: $W_k \leftarrow$ WeightingOptimisation($\vec{x}'_k, Z, A, \Gamma, \psi$) // **Algorithm 4**
8: **end for**
9: $S \leftarrow$ updatePopulation($W_1, .., W_q, \vec{x}'_1, .., \vec{x}'_q, S$) // **Algorithm 5**
10: **until** $\delta \cdot total\#Evaluations$ used
11: **repeat**
12: $S \leftarrow A(Z, S, t_1)$ // Optimise Z with Algorithm A for t_1 evaluations, using S as a starting population.
13: **until** $total\#Evaluations$ used
14: **return** FirstNonDominatedFront(S)

Next, the weighting optimisation (Lines 6 to 8 in Algorithm 3) performs the transformation and optimisation of the weights q times. For this, q different pivot solutions $\vec{x}'_k$ ($k = 1, 2, ..., q$) are drawn from the current population (Line 5) based on a selection mechanism. For every one of them, the transformed optimisation is carried out as shown in Algorithm 4.

A suitable choice of the pivots helps in this step to preserve diversity in the population. Using the above described mechanism of problem transformation, multiple transformed problems $Z^k_{\vec{x}'}$ are created. At this point, we make use of a transformation function, as shown in Definition 5.1. In our implementation, the same function ψ is used for all transformed problems, although theoretically different functions could be used for each single problem as well.

Next, a population W_k of randomly created solutions (weights) for each of these $Z^k_{\vec{x}'}$ is optimised using the optimisation algorithm A. As a result of this step, we obtain a population W_k of weights, where the objective function values are optimised based on the values of the originally chosen solution $\vec{x}'_k$, i.e. the population W_k contains the best found solutions in the subspace defined by the transformation. This process of problem transformation and optimisation is carried out q times (once for each of the q pivot solutions), resulting in a set of q weight populations $\{W_1, ..., W_q\}$ [1].

The last step is to merge the original population and the newly obtained weight populations to perform an overall environmental selection (Line 9 in Algorithm 3). The weight populations $\{W_1, ..., W_q\}$ each contain vectors of weights which are optimised based on one of the solutions of the original population, i.e. the chosen $\vec{x}'_k$. If the population size of each of these W_k is c, and the original solution population S has a size of s individuals, by combining all weight vectors of all the populations with each solution of S, we can construct $q \cdot c \cdot s$ new solution candidates for the original problem. This results in a large number of additional needed function evaluations due to the creation of these new solutions.

To reduce this computational overhead different strategies can be used. The original version of WOF in [4, 1] used the strategy to only pick one solution from each W_k (using the largest Crowding Distance from the first non-dominated front[1]). This weight vector was used to create one new solution by applying it to each of the solutions in S, resulting in $q \cdot s$ new solutions. In contrast, a modified approach introduced in [6] used q selected weight vectors from each W_k and in addition combined each $\vec{x}'_k$ with all vectors in the respective W_k. This results in an overall of $q \cdot q \cdot s + q \cdot c$ new function evaluations for the environmental selection step, but showed a better exploitation of the information contained in the W_k. This second, modified version of the merging step is used in this thesis and is shown in detail in Algorithm 5, Lines 2 to 9.

The obtained new solutions in the sets S'_k for $k = 1, .., q$ are then combined with the population S (Line 10 in Algorithm 5). In a next step, duplicate solutions are removed from this union set to prevent negative effects on the population's diversity (Line 11). This elimination is based on the values in the objective space, i.e. if two solutions have the same objective function values, they are considered as duplicates and one of them is removed. Note that this elimination can also be done using the decision variable values. This increases the runtime of the algorithm, since the comparison has to be

Algorithm 4 WeightingOptimisation($\vec{x}'_k, Z, A, \Gamma, \psi$) - Pseudocode based on [1].

Input: Solution $\vec{x}'_k$, Problem Z, Optimisation Algorithm A, Grouping Mechanism Γ, Transformation Function ψ
Output: Population of weights W_k
1: Initialisation
2: Divide n variables into γ groups using Γ
3: $Z^k_{\vec{x}'} \leftarrow$ Build a transformed problem with γ decision variables (weights) from Z, $\vec{x}'_k$, Γ and ψ
4: $W_k \leftarrow$ Random population of weights for $Z^k_{\vec{x}'}$
5: $W_k \leftarrow A(Z^k_{\vec{x}'}, W_k, t_2)$ // Optimise $Z^k_{\vec{x}'}$ with Algorithm A for t_2 evaluations, using W_k as a starting population.
6: **return** W_k

done with the high-dimensional decision variable vectors. On the other hand, it can be beneficial for multi-modal problems where multiple areas of the search space map to the same objective function values. The concept of multi-modal multi-objective optimisation has recently drawn attention in the literature [103, 104, 105, 106, 107], and changing the elimination of duplicates in WOF is a possible modification for future multi-modal large-scale research.

Next, in case the newly formed solution set contains fewer than s solutions, additional solutions are created by genetic operators to fill the population (Lines 12 to 14 in Algorithm 5). As a last step, an environmental selection in the form of non-dominated sorting is carried out. This helps to eliminate worse solutions which may have been found in the weight optimisation, for instance due to a suboptimal selection of a pivot solution.

After the selection process, the main loop of the algorithm starts from the beginning with a normal optimisation step to alter the variable values independently of each other. The alternation of optimising the original and the transformed problems (Lines 3 - 10 in Algorithm 3) is repeated until a certain number of function evaluations is used up. Similar to the strategy used in MOEA/DVA, the performance of WOF is improved if a certain amount of computation is used for a final so-called "uniformity" optimisation. Since the weights of WOF usually lead to large "jumps" in the search space, and do not allow for independent alternation of variables, a normal optimisation with the used metaheuristic has shown to work better towards the terminal phase of the search. In this stage, the optimal values for the decision variables can be approximated better with traditional evolutionary operators, that do not include variable groups, and allow for independent changes in each variable [1]. To control at which point the optimisation of weights is stopped, WOF contains a parameter $\delta \in [0, 1]$, which defines which share of the total function evaluations are spent for the first phase (Line 11 in Algorithm 3).

Algorithm 5 UpdatePopulation($W_1, .., W_q, \vec{x}'_1, .., \vec{x}'_q, S$)

Input: Weight populations $W_1, .., W_q$, Pivot Solutions $\vec{x}'_1, .., \vec{x}'_q$, Solution population S
Output: Solution population S

```
s ← |S|
for k = 1 to q do
    S'_k ← ∅
    {w⃗_k^(1), .., w⃗_k^(q)} ← Selection of q individuals from W_k
    for r = 1 to q do
        S'_k ← S'_k ∪ {Apply w_k^(r) to each solution in population S}
    end for
    S'_k ← S'_k ∪ {Apply each individual in W_k to the solution x'_k}
end for
S ← S ∪ {S'_k}_{k=1,..,q}
Eliminate duplicate solutions from S
if |S| < s then
    Fill S by applying genetic operators to solutions from S until the population size s
    is reached
end if
S ← Perform non-dominated sorting on S
return S
```

5.1.5 Transformation Functions

In the following, four different transformation functions are defined and discussed that might be used in WOF. The first three of them ψ_1, ψ_2 and ψ_3 are included in the original publications of WOF [4, 1], while the fourth, parameter-free one was proposed in a later study [6] to improve the performance of the algorithm. As described, we assume the variables were divided into γ groups $G_1, ..., G_\gamma$ prior to the transformation, and $g(i) = j$ is the respective group x_i is assigned to [1]. w_j is the respective weight that is used to optimise the values of the variables in G_j.

ψ_1 - Product Transformation

The first transformation is based on a simple multiplication of the weight and the variable values, and is in the following described as in [1]. It is defined as follows:

$$\begin{aligned} &\psi_1(w_j, x'_i) = x_{i,new} := w_j \cdot x'_i \\ &w_j \in [0, 2] \end{aligned} \tag{5.7}$$

If the transformation results in values greater or smaller than the respective variable's boundaries, a repair mechanism is used to set the value to the exact boundary instead. This transformation function was used to motivate the idea above in Section 5.1.1. However, this function has some disadvantages. First, it can be expected that this transformation does not deal well with variable domains that involve both positive and negative values. Depending on whether the input value x'_i is positive or negative, a

multiplication with a positive weight can only obtain positive or negative new values respectively, and as a result make a certain part of the search space unreachable.

A second problem is that the progress that can be made by altering one weight variable w_j is determined by the absolute value of x_i, not its value relative to the search domain. Consider for instance the variable x_i with its domain of $x_i \in [0, 10]$. In the transformation step, we apply a variable w_j to it, which has a domain of $w_j \in [0, 2]$. Now based on the value of x_i in the chosen pivot solution, the search in the transfomation-induced subspace can be very limited. An absolute value of $x_i = 0.1$ enables the optimisation algorithm to explore the original search space in the interval $[0, 0.2]$, while a value of $x_i = 8$ results in a search that covers at least the whole domain of x_i [1]. As a result, the search ability of WOF can be very limited, and adaptions of the domain of the w_j may be required frequently. Since we do not know the optimal variable values for x_i, it is advisable that the search should not depend on absolute values. The ψ_1 transformation is therefore not recommended to use in practice.

ψ_2 - p-Value Transformation

The second function, called p-Value Transformation, was designed to overcome the disadvantages of the simple multiplication.

$$\begin{aligned} &\psi_2(w_j, x'_i) = x_{i,new} := x'_i + p \cdot (x'_{i,max} - x'_{i,min}) \cdot (w_j - 1.0) \\ &w_j \in [0, 2] \\ &p \in [0, 1] \end{aligned} \tag{5.8}$$

The value p in this function defines the range of possible change around the original value of the variable x_i, where $x_{i,min}$ and $x_{i,max}$ are the lower and upper bounds of the variable x_i. As before, should the result be lower or higher than the variables lower and upper bounds, the value is set to the respective boundary again. To ensure a symmetrical interval around the original value of x'_i, the value of w_j is adjusted depending on its variable domain. In this case, since the $w_j \in [0, 2]$, a value of 1.0 is subtracted, i.e. the values of w_j are mapped to the interval $[-1, 1]$. The value p determines how much change in the original search space is possible at most. For instance, setting $p = 0.3$, the value of x_i is altered by 30% of the width of the variable's domain, centered around the original value x'_i [1].

ψ_3 - Interval-Intersection Transformation

The third function is inspired by an approach suggested in [34] for single-objective optimisation.

$$\begin{aligned} &\psi_3(w_j, x'_i) = x_{i,new} := w_j \cdot x'_i \\ &w_j \in [\min_{h \in G(i)} \frac{x_{h,min}}{x_h}, \max_{h \in G(i)} \frac{x_{h,max}}{x_h}] \end{aligned} \tag{5.9}$$

where $x_{h,min}$ and $x_{h,max}$ are the lower and upper bounds of the variable x_h. The actual transformation formula of this function resembles the multiplication transportation ψ_1. However, the difference lies in the domains of the weight variables. The upper- and lower-bounds of the w_j are based on the other variables in the same group. We saw in the multiplication transformation that a domain of $[0, 2]$ may not be suitable, first because it limits the search too much if the absolute value of $\vec{x}'_i$ is too low, and second because it can produce many infeasible values if the absolute value is too high. The interval-intersection transformation solves this problem by determining for each variable x_i in a group $j = G(i)$ the maximal and a minimal possible weight value that can be used without exceeding the domain borders of x_i. The domain of the weight for this group is based on the highest and lowest possible values over all variables in the respective group. This interval $[w_{j,min}, w_{j,max}]$ can be seen as the intersection interval of all possible minimum and maximum values defined by each variable in the group [1].

The functions ψ_1 and ψ_2 require a method to repair values in case the domain of the variable is violated as a result of the transformation. An advantage of ψ_3 is that this kind of repair is no longer needed, as variable domains can never be violated. This can be helpful to prevent a lot of extremal values due to excessive repairs for certain applications. On the other hand, this approach might also be very limited in its capabilities since a near-boundary value for just one variable in a group can lead to almost no possible change in all other variables in that group. In [1], this transformation function was included to provide fair comparisons on benchmarks like the ZDT functions, since their optima include a large number of variables to obtain a value on their domain borders.

ψ_4 - Parameter-free Transformation

The fourth function was introduced by the author in [6]. The best-performing transformation function used in the original publication [1] was ψ_2, the *p-Value-Transformation*. Even though it achieved good results, its performance is dependent on a parameter p. Usually, a value of $p = 0.2$ is recommended in the literature (as shown by the sensitivity analysis in [1]). However, a disadvantage of the *p-Value* transformation is that only a part of the domain $[x_{i,min}, x_{i,max}]$ is covered by the transformed problem. Further, depending on the value of p and the absolute value of x'_i, infeasible values can be created that have to be repaired afterwards. As written in [6], it can therefore be of advantage to eliminate this parameter, so the algorithm is more adaptive when dealing with unknown problem properties and is able to search in the whole variable domain.

As a result, the parameter-free transformation function ψ_4 is defined as:

$$\psi_4(w_j, x'_i) = x_{i,new} := \begin{cases} x'_i + (w_j - 1.0) \cdot (x_{i,max} - x'_i) & \text{if } w_j > 1.0 \\ x_{i,min} + w_j \cdot (x'_i - x_{i,min}) & \text{if } w_j \leq 1.0 \end{cases}$$

$$w_j \in [0, 2]$$

where $x_{i,min}$ and $x_{i,max}$ are the respective lower and upper bounds of the variable x_i. Each new variable value $x_{i,new}$ is computed between $x_{i,min}$ and $x_{i,max}$ using the value of x'_i as a center point. The domains of the w_j, $j = 1, .., \gamma$ are located in the interval $[0, 2]$ as in ψ_2. All values in the range of $[0, 1]$ are mapped to the interval $[x_{i,min}, x'_i]$ in the original search space, while all values $(1, 2]$ are transformed to $(x'_i, x_{i,max}]$ [6]. In this way, ψ_4 is able to cover the whole domain of each variable in each group, therefore not limiting the search space artificially by a parameter. If $w_j = 0$ for a group j, all variables in that respective group are set to their lower bounds, while if $w_j = 2$, all variables are set to their upper bounds. A value of $w_j = 1$ results in no change of the variables at all.

5.1.6 Ordered Grouping

In addition to the simple grouping methods shown in Section 3.3.1, we propose a mechanism called "Ordered Grouping". This variant assigns variables to groups based on their absolute values, and achieved good performance in the original publication of the WOF method [1]. Since in the WOF, a pivot solution is picked from the population for the following optimisation process, the variable values of this solution are used to determine an ordering. The number of groups is specified beforehand, and the variables are assigned linearly to the groups after sorting them based on their values. As a result, the first group contains the variables which have the largest (or smallest, depending on the implementation) values in the picked solution, and so forth. It is worth noting that in problems where domains (upper and lower bounds) are not the same for all variables, a variant of this method where the values are normalised with their respective boundaries seems more suitable. In the remainder of this work, however, the classical ordering based on absolute values is used from the original WOF publication.

5.1.7 Choice of the Pivot Solutions $\vec{x}'$

The last design decision that is necessary for WOF is that of how to choose the $\vec{x}'_k$ solutions that are fixed in the weight optimisation steps. To increase the diversity in the solutions and prevent the algorithm from concentrating only on one part of the search space, this is done q times independently with different pivots. For this purpose, q different solutions need to be picked from the population in order to do the weight optimisation one time for each of them.

In order to choose these q solutions, different mechanisms can be used. Of course, that can be done completely at random. However, this has the disadvantage that there is no control over the amount of similarity between these solutions. To be more precise, it could happen that two solutions which are very similar to each other in the current population are picked, in which case it might be likely that the weight optimisation for each of these might not produce very different solutions in the end. As described in [1], experiments have shown that one of the weaknesses of WOF is the loss of diversity due to an increased convergence speed. An example for this loss of diversity in favour of

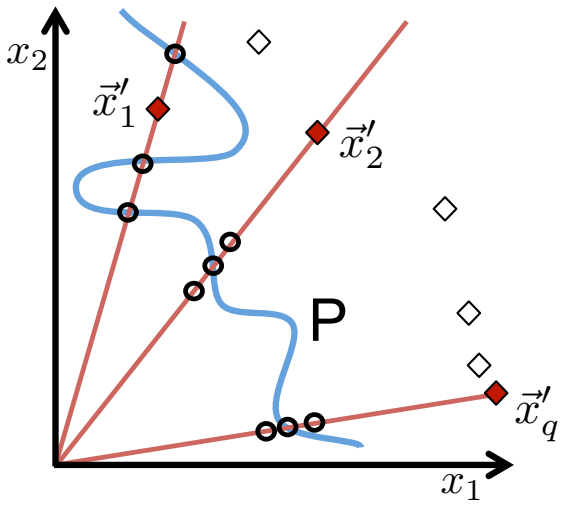


Figure 5.7: The optimisation of a transformed problem in WOF is performed with q different pivot solutions. The example shows the original population as diamonds and the solutions created from the transformed problems as circles. Graphic taken from [1].

convergence was shown earlier, see Fig. 5.5d. Therefore, it is of advantage to counter this development by an intelligent choice of the q solutions $\vec{x}'_k$, $k = 1, .., q$.

As also shown in [6], it can be of advantage to set q to a value of m (the number of objective functions) or larger. For the selection of the pivots it is further useful to include the diversity of the solutions, as shown in Fig. 5.7. Choosing solutions with a certain distance from each other in the objective space can lead to the exploration of different areas of the decision space with the weight optimisation. In the following, we present three different versions of choosing the pivot solutions $\vec{x}'_k$ and examine their possibilities briefly.

Choice of Pivots by Crowding Distance

The first method to select the q pivot solutions is by using the Crowding Distance metric from the literature [11]. From the current population of the original problem, the first non-dominated front is selected, and Crowding Distance is applied to select the solution in less crowded areas. This approach was used in [1]. By only using the first non-dominated front from the population, we ensure to use solutions which are (in the current generation) regarded as best-performing. Since the Crowding Distance assigns values of infinity to the extremal solutions (i.e. the solutions which have the highest or lowest values in any objective among the front), it is to be expected that these solutions are selected. The WOF algorithm would therefore do the optimisation with the solutions which lie on the far ends of the current non-dominated front. A disadvantage of this method is that it is rather unlikely to select a solution “in the middle” of the front, especially in the case of more than 2 objective functions (as the amount of “extremal” solutions rises as well).

Choice of Pivots by Tournament Selection

The second method that might come to mind is to use a usual selection method that is also applied before the genetic operators. As an example, a normal tournament selection as used in NSGA-II is possible, with the front number as the primary and the Crowding Distance as the secondary criteria. This ensures, similar to the method above, that solutions with in less crowded areas are preferred, but leaves room for the selection of solutions other than the extremal ones.

Choice of Pivots by Reference Lines

A method inspired by modern many-objective methods is to choose the $\vec{x}'_k$ by reference directions. This way of selection has first been used in WOF in [6] and showed, in combination with the newly introduced ψ_4 transformation function, good performance in terms of solution quality as well as convergence behaviour. The choice of solutions is done in the following way. The first m solutions are selected based on the distances to the axis of each of the m objectives. More clearly, the solution with the minimum angle between its function values (interpreted as a vector in the objective space) and the vector along the f_1 axis is selected as the first of the q solutions. This is repeated for each objective function $f_1, .., f_m$, resulting in m selected solutions. If q is larger than m, the next solution is chosen based on the smallest angle to the vector $(f_1, f_2, .., f_m) = (1, 1, .., 1)$, to include a solution that represents the "inner part" of the non-dominated front. After that, solutions are randomly added from the population until q solutions have been selected in total.

5.1.8 Discussion of the WOF Method

The WOF has the disadvantage that it needs to carefully balance between convergence and diversity. The mechanism of doing the weight optimisation multiple times achieves this balance. However, the recent study in [47], for instance, showed that WOF achieves superior performance mostly in those benchmarks which pose a challenge in terms of convergence towards the PF. In particular, it was argued that benchmarks like the UF problems from the CEC 2009 competition (UF benchmarks) pose stronger challenges to WOF as it is more difficult in these problems to achieve diversity along the PF. It must be noted, however, that this study used Crowding Distance to chose the pivot solutions. Another work by the author of the thesis done in [6] argued that performance, and in particular diversity, is generally increased when pivot solutions are chosen by reference lines as described above. The performance of WOF on the UF benchmarks using this technique is examined further in the evaluation in Chapter 6.

Another critical point worth noting is the relatively large amount of free parameters in WOF. While most other large-scale methods from the literature require some parameters to choose as well (e.g. number of groups), WOF, in its original version from [1], needs 6 parameters to be set by the user, and additional design decisions like the transformation

function and the grouping mechanism can be made. The sensitivity analysis in the same publication, however, showed some suggestions for the choice of parameter values, which can be used for good performance on most current benchmark problems. Using the parameter-free transformation function ψ_4 as described above, one additional parameter can be removed. In general, it is desirable to remove further parameters from the algorithm in future research by determining them automatically based on problem properties or search dynamics.

One aspect that gives WOF an advantage over other methods is the fact that it has been shown to work well with simple grouping mechanisms. It is therefore much better able to optimise problems with limited numbers of function evaluations, compared to methods like S^3-CMA-ES, LMEA or MOEA/DVA, since it does not depend on interaction-based or contribution-based groups. This aspect is also examined later in further detail in Chapter 6.

5.2 Groups in Mutation Operators

This section describes the second contribution of the author to multi-objective large-scale optimisation, which was proposed in [5]. It is based on the idea to include variable groups in the mutation operators of existing evolutionary algorithms. Since almost all population-based metaheuristics utilise a kind of mutation operator, changing the operator itself to include large-scale related concepts is an easy and fast way to enable a large variety of metaheuristics to deal with large-scale problems.

To create special mutation operators that are able to deal with large-scale problems, we include variable groups to determine which variables undergo mutation. For this purpose, the widely used Polynomial Mutation Operator [101] is used, although the concept can be extended to other mutation methods, for instance for non-real-valued representations, as well.

In the following, we introduce the concept of the well-known Polynomial Mutation Operator. Afterwards, we propose three new mutation operators. The first one is called the *Linked Polynomial Mutation*, which establishes a connection between the mutated variables in each individual in terms of the amount of change. Secondly, the concept of variable groups is incorporated into the Polynomial Mutation, called the *Grouped Polynomial Mutation*, by using groups to decide which variables are mutated. The third proposed operator, the *Grouped and Linked Polynomial Mutation Operator* utilises both of the above concepts. The following explanations are based on the initial publication by the author in [5].

Algorithm 6 Pseudocode of the Polynomial Mutation operator. Pseudocode based on [5]

Input: Solution $\vec{x}$, Probability p, Distribution Index η
Output: Mutated Solution $\vec{y}$

for $i = 1$ **to** the number of variables in $\vec{x}$ **do**
  $r \leftarrow$ random(0,1)
  if $r < p$ **then**
    $u \leftarrow$ random(0,1)
    if $u \leq 0.5$ **then**
      $\delta_1 = \frac{x_i - x_{i,min}}{x_{i,max} - x_{i,min}}$
      $\delta_q = (2u + (1-2u)(1-\delta_1)^{\eta+1})^{\frac{1}{\eta+1}} - 1$
    else
      $\delta_2 = \frac{x_{i,max} - x_i}{x_{i,max} - x_{i,min}}$
      $\delta_q = 1 - (2(1-u) + 2(u-0.5)(1-\delta_2)^{\eta+1})^{\frac{1}{\eta+1}}$
    end if
    $y_i = x_i + \delta_q(x_{i,max} - x_{i,min})$
    repair(y_i)
  else
    $y_i = x_i$
  end if
end for
return $\vec{y}$

5.2.1 Polynomial Mutation

The Polynomial Mutation operator was introduced in [101] and has since been a part of many metaheuristic optimisation algorithms, for instance in [11, 32, 108]. It is designed for real-valued variables, and utilises a polynomial distribution around the original value of a variable to sample a new, mutated value. Two versions of this operator were proposed in the literature, called *highly disruptive* and *non-highly disruptive* Polynomial Mutation. The difference between them is the distribution of the possible mutated values. The original, non-highly disruptive mutation has the disadvantage to become useless when the variable value gets closer to its domain border [109]. For this reason, the new highly disruptive version was proposed in 2008 [102]. In contrast to the original version, the complete domain of each variable was included in the distribution of the operator. A hybrid of both versions was also introduced in [109]. Due to its advantages, the highly disruptive version is used in this thesis. The functionality of this operator is depicted as pseudocode in Algorithm 6 and is described in further detail in the following [5].

Polynomial Mutation has two parameters: the mutation probability p and the distribution index η. For each variable x_i $(i = 1, ..., n)$ of a solution $\vec{x}$, the probability parameter p is used independently to decide whether this variable is subject to mutation. Thus, the expected number of mutated variables in a solution is $n \cdot p$. In the literature, a value of $p = \frac{1}{n}$ is often used [11, 108, 32]. As a result, only a single variable per solution is

mutated in the expected case. The second parameter, the distribution index η, influences the distribution from which the new, mutated value is chosen. The distribution index therefore influences how much change the mutation operator produces, and can be used to balance exploitation and exploration in the search process. A larger value of η corresponds to a higher probability that the drawn value from the distribution lies very close to the original x_i. In the literature, η is often set to 20.0 [11, 32].

The detailed steps of the operator are as follows. For each variable, it is decided whether a mutation is applied (Line 3 in Algorithm 6). After that, the operator draws another value u uniformly at random, which is used in the subsequent steps to determine the new value of x_i, i.e. the value that is chosen from the distribution defined by η. Depending on the value of u, a value from a distribution is drawn either on the left side or the right side of the original value of x_i (Lines 5 - 11 in Algorithm 6), and the update of the variable x_i is performed (Line 12). In case a drawn, new value for x_i exceeds the upper or lower limit of the variable, a repair mechanism is used in Line 13 which sets the value to the respective border [5].

In the following, certain changes are made to this operator, and the three proposed mutation operators for large-scale optimisation are described in detail. The newly proposed versions are called *Linked Polynomial*, *Grouped Polynomial* and *Grouped Linked Polynomial* mutation, where the last of these versions is the most effective operator based on the original publication, and is called the Grouped Linked Mutation Operator (GLMO) in the remainder of the thesis. In the following, we explain how the mutation can be equipped with a variable grouping mechanism, and how the amount of change for each variable can be influenced within these groups [5]. The experimental comparison between all the different versions inside various algorithms is carried out below in Chapter 6.

5.2.2 Linked Polynomial Mutation

The *Linked Polynomial Mutation* operator follows essentially the same workflow as the original Polynomial Mutation. The difference lies in the amount of change, which in this operator is not independent between the mutated variables. As we have seen above, the separability of a problem is usually of concern in large-scale optimisation. In a separable problem, all variables can be optimised independently of each other to obtain the optima of the problem. In contrast, in non-separable problems (where interactions between variables exist) the solution quality can be affected in a negative way if changes are made to one variable without a corresponding, suitable change to another, interacting variable.

To address this issue, we propose a version of the Polynomial Mutation operator where the amount of change in each variable is connected to the amount of change in the other mutated variables of the same solution. This "link" between the mutated variables is implemented by influencing the choice of the value u in the operator. In the pseudocode in Algorithm 6, the value for u (the amount of change) in Line 4 is now chosen prior

to the start of the loop in Line 1. In this way, all values for the mutated variables are determined by only one value for u, which is drawn one time and fix for all variables. Thus, the same amount of change (relative to the domain) is applied to all variables that are actually subject to mutation (according to the probability p) [5].

Using this concept can be beneficial for problems with interacting variables. If mutation is performed on multiple variables of a solution, the link between the variables will result in a similar change in both of the variables. This can lead to less disturbance in solution quality in case these variables are interacting with each other. However, this may not always be the case and is dependent on the specific types of interaction among the variables. Especially since the mutation probability is usually chosen so that the expected value of changed variables is 1 out of n variables, the effect of this mechanism alone, that is without any variable groups, might be rather limited. The effectiveness of this is examined later in the experiments in Chapter 6.

5.2.3 Grouped Polynomial Mutation

Next, the concept of variable groups is incorporated into the Polynomial Mutation operator. In this version, we assume that a grouping mechanism is used to separate the variables into multiple groups. Then, mutation is only applied to a group as a whole entity, i.e. to each variable in a group at the same time. This is expected to work well in large-scale problems when the groups are chosen in a way that interacting variables are put in the same group [5].

In the *Grouped Polynomial Mutation*, the decision of which variables are mutated is changed from a random probability per variable to a random choice of which group as a whole undergoes mutation. The underlying assumption is that through a suitable grouping mechanism, interacting variables, which should ideally be altered at the same time, belong to the same group. This is supposed to benefit the search process, especially in non-separable problems.

A difference to the original operator is that the grouped mutation does not require a probability parameter p. Which and how many of the n variables are subject to change is defined by the number of groups (γ), since one of the groups is chosen for the mutation randomly. The operator works as follows, as written in [5]. In a first step, a grouping mechanism is used on the solution $\vec{x}$ to separate the decision variables into γ groups, as described in Section 3.3. This can be done before the actual optimisation starts (in terms of interaction-based and contribution-based groups), or separately and anew each time the mutation operator is used (which might only be feasible with simple grouping methods). The second step is to select one of the γ groups randomly with equal probabilities for each group. Then, Polynomial Mutation is applied to every variable in the selected group (Lines 4-13 in Algorithm 6) [5]. More precisely, **separate** values for u are chosen and used in the mutation for each variable in the group. No additional probability value is

used. Which and how many of the variables are subject to mutation is entirely based on the random selection of one of the groups. All variables belonging to the remaining groups keep their original values. Assuming evenly sized groups, the amount of variables which are subject to change is $\frac{n}{\gamma}$, which is in the expected case significantly larger than the amount of mutated variables in the original and Linked Polynomial Mutation [5].

5.2.4 Grouped and Linked Polynomial Mutation

The *Grouped and Linked Polynomial Mutation Operator* (GLMO) incorporates both of the concepts from the previous operators. The detailed steps of this mutation operator are shown as pseudocode in Algorithm 7.

The necessary inputs for this operator are a grouping mechanism Γ and a distribution index η. In case the groups are already precomputed by a non-simple method as described above, the groups can also serve as an input directly. Otherwise, the whole mechanism Γ can be used as an input. In case a simple mechanism is used, it can even be beneficial to obtain different groups in each iteration or usage of the operator, as the specific simple groups can be tailored to the specific solution to be mutated (e.g. Ordered Grouping as seen in Section 5.1.6). Similar to the *Grouped Polynomial Mutation*, no additional mutation probability for the variables is needed. One of the γ groups is chosen at random and all variables in this group are mutated (Lines 1 and 2 in Algorithm 7). In addition, as in the *Linked Polynomial Mutation*, only one value for u is drawn from the distribution at the beginning of the operator (Line 3). This ensures that all variables in the chosen mutated group are changed equally, i.e. by the same amount relative to their domains. After choosing one of the groups, Polynomial Mutation is used on all variables in that group using the given value for u (Lines 4 to 14).

As explained, it can be advantageous to the search if interacting variables are changed at the same time, and we further assume that changing them by a similar amount can be useful. This operator combines both of these functionalities. In contrast to the linked version only, the variables that supposedly interact with each other may now belong to the same group. Further, since now a larger amount of variables is changed due to the mutation of a whole group, the link between the variables is expected to have an increased effect. The original study by the authors in [5] shows that this GLMO version performs better than each of the two proposed changes alone. It also shows that this mutation operator is able to greatly enhance the performance of existing methods on large-scale benchmark problems. The experimental section of this thesis evaluates the performance further by applying it to a larger set of different benchmark functions with different amounts of variables (see Chapter 6).

Algorithm 7 Pseudocode of the *Grouped and Linked Polynomial Mutation* operator. Pseudocode based on [5]

Input: Solution $\vec{x}$, Grouping Mechanism Γ, Distribution Index η
Output: Mutated Solution $\vec{y}$

$\{G_1, ..., G_\gamma\} \leftarrow$ Apply Γ to $\vec{x}$, producing γ groups
$j \leftarrow$ Pick a group index uniformly at random from $\{1, ..., \gamma\}$
$u \leftarrow$ random(0,1)
for all variables x_i with $i \in G_j$ **do**
  if $u \leq 0.5$ **then**
    $\delta_1 = \frac{x_i - x_{i,min}}{x_{i,max} - x_{i,min}}$
    $\delta_q = (2u + (1-2u)(1-\delta_1)^{\eta+1})^{\frac{1}{\eta+1}} - 1$
  else
    $\delta_2 = \frac{x_{i,max} - x_i}{x_{i,max} - x_{i,min}}$
    $\delta_q = 1 - (2(1-u) + 2(u-0.5)(1-\delta_2)^{\eta+1})^{\frac{1}{\eta+1}}$
  end if
  $y_i = x_i + \delta_q(x_{i,max} - x_{i,min})$
  repair(y_i)
end for
for all variables x_i with $i \notin G_j$ **do**
  $y_i = x_i$
end for
return $\vec{y}$

5.3 Dimensionality Reduction using Linear Combinations

The third proposed algorithm of this thesis is the concept of using linear combinations of solutions to reduce the dimensionality of the search space. This section describes the method based on the findings and explanations of the 2019 publication by the author in [7]. In particular, some of the mathematical descriptions of the concept in this section are strongly based on the descriptions in the author's related article.

The "Linear Combination-based Search Algorithm" (LCSA) is designed to increase the exploration abilities of existing algorithms while simultaneously performing a dimensionality reduction of the search space without using coevolution or variable groups. The concept is mainly based on the assumption that by using a linear combination of solutions in the n-dimensional search space, one can obtain other good or promising solutions. The concept of linear combinations has, for instance, also been used in [110] to preserve the feasibility in problems with linear constraints.

When metaheuristic, especially evolutionary, approaches are used, it is usually assumed that through a suitable encoding of the problem, recombinations and mutations of good solutions can be used to generate other promising solutions. Further, Pareto-optimal solutions, or approximations of such, may, to certain extent, be similar to each other. In some of the popular benchmark suites, for instance, all optimal solutions share the same

values for their convergence-related decision variables [44, 28]. These similar properties can be approximated by an optimisation algorithm, and the knowledge that certain values in certain variables are beneficial for the overall quality of solutions is implicitly coded in the population of the algorithm.

In this LCSA approach, the aim is to make use of this inherent information in the population. The method is based on the assumption that at each generation of the EA, the current population's members contain the information which (sub-)vector-space of the n-dimensional search space contains the (at that point in time) best or most promising solutions [7]. To utilise this concept and to search for solutions in this new subspace, a population of coefficient vectors is formed and used to create linear combinations of solutions. Optimising these coefficients with a metaheuristic can then help to increase the exploration and exploitation of the search space in areas that likely contain further improved solutions. Through such combinations of population members, the aim of the resulting EA is to improve the search process of multi- and many-objective optimisation algorithms in terms of solution quality.

The idea of extracting knowledge from the search process can be seen as related to the concept of "innovisation" from the literature ([111, 112]), specifically online innovisation. This concept aims to identify relevant information about the problem from the obtained solutions at runtime of the optimisation, and ideally feed it back to the optimisation process to further improve the search.

An advantage of this approach is the fact that population sizes are usually smaller than the number of decision variables (in large-scale optimisation). Therefore, even if all population members are used in the linear combinations, the proposed method is able to provide a reduction of dimensionality for large search spaces. In the following, the concept is introduced formally and the generic algorithm structure is presented. As the other proposed approaches in this thesis, the LCSA can be used with arbitrary population-based metaheuristics, and the evaluation of the approach on large-scale and other problems with various algorithms is performed in Chapter 6.

5.3.1 Concept of Linear Combinations of Solutions

Suppose we have a real-valued optimisation problem as shown in Eq. (2.1), containing n decision variables and m objectives. As written in [7], let the population of an algorithm be P and its size be $s := |P|$. At each given time of the optimisation process the population consists of s solution vectors each of dimensionality n: $P = \{\vec{x}^{(1)}, \vec{x}^{(2)}, ..., \vec{x}^{(s)}\}$. Each solution is a vector $\in \mathbb{R}^n$:

$$\vec{x}^{(i)} = (x_1^{(i)}\ x_2^{(i)}\ ...\ x_n^{(i)}) \tag{5.10}$$

The set P can be used to define a vector (sub)space, using the population members to span this space. The dimensionality of this space is given by the rank of the matrix of the spanning vectors [7]. Using the variables of the population members, we can create the matrix $\hat{X} \in \mathbb{R}^{s \times n}$, where each row contains one of the solutions in P.

$$\hat{X} = \begin{pmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(s)} & x_2^{(s)} & \cdots & x_n^{(s)} \end{pmatrix} \tag{5.11}$$

In an EA, new solutions for an optimisation problem are usually created by using recombination operators like, for instance, the arithmetic or simulated binary crossovers. On the other hand, multiple solutions of the population can also be combined linearly instead. The focus in the following lies on general linear combinations, although it is also possible to use smaller subsets of these, for instance convex or conical combinations. We define a linear combination of the solutions as follows.

$$\vec{x}' = \vec{y}\hat{X} = y_1\vec{x}^{(1)} + y_2\vec{x}^{(2)} + ... + y_s\vec{x}^{(s)} \tag{5.12}$$

where the vector $\vec{y}$ contains the combination coefficients:

$$\vec{y} = (y_1\ y_2\ ...\ y_s) \tag{5.13}$$

Using this concept, we can employ a metaheuristic algorithm to perform a search for better solutions from combinations of the existing ones. This new search space, which spanned by the s vectors in P, has a dimensionality equal to the rank of the matrix $\hat{X}$, i.e. $1 \leq rank(\hat{X}) \leq min\{n, s\}$.

Since this method combines all of the s existing solutions, it can also be seen as a variant of an s-parent crossover. However, in a multi-parent crossover new solutions are typically created through random combinations, while the environmental selection process of the EA is responsible for the decision whether this produced solution is an improvement. In contrast, our proposed approach uses an evolutionary process internally to find "optimal" or improving combinations, i.e. the parameters for the combination are not chosen randomly, but are subject to an optimisation process.

In the LCSA method, instead of optimising the original variables of the problem, the values of the vector $\vec{y}$ are optimised to search for promising linear combinations. Thus, assuming

that all s population members are used in the linear combinations, the dimensionality of the new optimisation problem is reduced to s decision variables as opposed to the n variables of the original problem. The effects this procedure has on the search can vary depending on the dimensionality of the problem.

In the following example, as described in [7], consider a 30-dimensional problem that is optimised using a population size of $s = 100$. In this case, searching for optimal linear combination coefficients results in a search in a 100-dimensional space. The created solutions by the combinations, however, still remain in the original, 30-dimensional space, and thus the formed vector space contains redundancy, since not all the vectors used for the combinations can be linearly independent. On the other hand, if the same mechanism is used in a high-dimensional problem, for instance with $n = 1000$ variables, we can observe a different effect. The $s = 100$ population members can at most define a 100-dimensional subspace of the original search space. Thus, the optimisation algorithm searches in an – at most – 100-dimensional subspace of the 1000-dimensional original search space. In this case, the optimisation in the space of linear combinations serves as a dimensionality reduction technique. If this technique is used in the beginning of the search, the whole population still consists of mostly randomly created solutions, where it is not guaranteed that good solutions actually lie in the defined subspace. However, after the optimisation already progressed for a certain time, we can assume that the population started to converge towards promising areas of the search space. Thus, the spanned vector space, i.e. a combination of the current variable values, may contain additional promising solutions which can help to approximate the Pareto-optimal areas.

This way of dimensionality reduction makes the LCSA suitable for large-scale problems, and through the exploration of subspaces, we can also expect a good performance in multi- and many-objective optimisation in terms of diversity of solutions. A possible drawback is, of course, that the lower dimensional space might not contain the actual Pareto-optimal solutions, and therefore optimising only the linear combinations might make it impossible for the EA to find these optimal regions of the original search space. To counter this risk, in the algorithm structure which is described in the following subsection, the optimisation of the original problem and the optimisation of linear-combinations take turns to harvest the best of both search spaces.

5.3.2 Algorithm Structure of the LCSA

The proposed concept of the LCSA can be used inside arbitrary metaheuristic optimisation algorithms. In the following, the algorithm structure is described as written in the author's contribution in [7]. We define a population Q of $\vec{y}$-vectors, where each vector in the population defines one linear combination of the members of P as described above. Thus, any optimisation algorithm can be used on this newly formed population to search for promising linear combinations of the underlying original solutions. This optimisation of the population Q, i.e. the search in a promising subspace of Ω defined by P, is

Algorithm 8 Linear Combination-based Search Algorithm LCSA

Input: Problem Z, Optimisation algorithm A
Output: Solution population P

```
1: P ← Random initial population for Z
2: while termination criterion not met do
3:    for i = 1 to iter_1 do
4:       P ← Perform one iteration of A on population P
5:    end for
6:    X̂ ← Decision variable values of the current population P
7:    Q ← Random initial population of linear-combination-vectors
8:    for i = 1 to iter_2 do
9:       Q ← Perform one iteration of A on population Q
10:      P ← EnvironmentalSelection(P ∪ Q)
11:   end for
12: end while
13: return P
```

employed inside other, existing metaheuristic algorithms as an additional search step. More precisely, the search mechanism of the original (arbitrary) metaheuristic is used in turns with the proposed linear combination-based search [7].

The following mathematical description is identical to the one given in the earlier publication by the author in [7]. Let $\hat{X}$ be the matrix of the decision variable values of all solutions in P as seen above, where each row in $\hat{X}$ corresponds to one solution in P. As a result, $\hat{X}$ is an $s \times n$ matrix, where s is the number of solutions in P. In the same way, let $\hat{Y}$ be the matrix of the decision variable values (i.e. coefficients of linear combinations) of the solutions in Q. The population size of Q is t, therefore $\hat{Y} \in \mathbb{R}^{t \times s}$. The original objective function evaluation can be applied to the new population by simply multiplying $\hat{X}$ with $\hat{Y}$ and computing $\vec{f}(\hat{Y}\hat{X})$, i.e. applying $\vec{f}$ to each row in $\hat{Y}\hat{X}$. For practical reasons and to limit the search space of the new problem, we also define lower and upper bounds for the variables y_i, i.e. $y_i \in [y_{i,min}, y_{i,max}]$, $i = 1, .., s'$.

The resulting LCSA optimisation approach works as follows, and is shown in Algorithm 8. In the main loop of the algorithm, the population P of the original problem is optimised with a multi-objective algorithm A for a specified number $iter_1$ of iterations (Lines 3 to 5 in Algorithm 8). Then, we build the matrix $\hat{X}$ out of the decision variables' values of the current population P. To start the linear combination-based search phase, a random population Q of linear-combination-vectors is created (Line 7). The algorithm then optimises Q for a certain amount of iterations $iter_2$ using the same optimisation algorithm A (Lines 8 to 11). During this step, all evaluated solutions are also used to update the original population P, using the environmental selection method of A. As a result, we obtain an updated population P for the next iteration of the main loop.

5.3.3 Discussion and Modifications of the LCSA

The LCSA switches between two optimisation phases, which optimise either the original problem or the coefficients of the linear combinations. The resulting algorithm is able to explore the original search space frequently during its runtime, while at the same giving increased attention to the exploitation of promising subspaces. If the upper and lower bounds of the coefficients ($y_i \in [y_{i,min}, y_{i,max}]$) are limited to be between zero and one, the search can only exploit the convex combinations of the existing solutions, i.e. the "inner" area of the simplex spanned by the (non-dominated) population members. The experiments in the previous publication [7] and the experiments conducted in this thesis, however, make use of larger domains for these coefficients, which allows extrapolation and therefore further enhances the exploration of the algorithm.

Previous results in [7] showed that this method was able to increase the performance in 60 problem instances from different benchmark families, both in many-objectives and in large-scale problems. The LCSA was able to increase the solution quality for two standard algorithms NSGA-II and GDE3, and even improved the performance of two many-objective algorithms: NSGA-III and RVEA.

Even though these results showed the potential of the linear combinations, the original version had the drawback that it used the NSGA-II optimiser exclusively for the optimisation of the coefficients $\vec{y}$. This use of NSGA-II can be a disadvantage when optimising a many-objective problem. To further harvest the advantages of existing methods, the version proposed in this dissertation thesis uses the employed metaheuristic in both, the original search space and the reduced space of the combination coefficients. This means, if SMPSO is used with this method, SMPSO is used to search in both spaces, in contrast to the version in [7]. In this way, using a many-objective algorithm within the LCSA framework is expected to deal well with many-objective large-scale problems as well.

One further modification to concentrate on promising solutions can be to use only the non-dominated solutions in the population for the linear combinations instead of the whole population. From a theoretical point of view, this can help to achieve a faster convergence, since the algorithm concentrates more on the "best" subspace spanned by the first front. In case the first front is significantly smaller than the population size, this measure also reduces the amount of decision variables of the linear coefficient problem, as fewer coefficients are necessary. On the other hand, focussing only on the non-dominated solutions can also be disadvantageous to the overall search. For instance, it can lead to a reduced exploration of the search space, especially in the early stages of the optimisation. Moreover, it is known that Pareto-dominance is not a well-performing concept in many-objective problems, and limiting the linear combinations to those non-dominated solutions can affect the diversity in the objective space negatively. For these reasons, the version in this thesis uses the whole population for the linear combinations.

5.4 Discussion and Classification of Proposed Methods

In this section, the three developed large-scale optimisation approaches are compared to each other and classified based on the different categories and characteristics proposed in Chapter 4.

Comparing the three proposed methods in terms of overall complexity, we can see that the WOF, in comparison with the other two, seems relatively complex, as it requires multiple different building blocks such as the selection of the pivot solutions, the transformation functions and the grouping mechanisms. The different GLMO methods are, in contrast, probably some of the simplest ways to incorporate variable groups into traditional algorithms and enable them to deal with large-scale problems. GLMO only requires the change of the mutation operator, and since many algorithms might use the same mutation operators, this can be applied to a large variety of algorithms without further change. It is, in this way, also relatively easy to dynamically activate or deactivate the large-scale capabilities inside an algorithm by simply choosing between modified or original mutation operators.

In the following, we take a look at the different building blocks from the literature as seen in Chapter 4, and examine which of these blocks are present in the proposed methods. The results of this analysis are summarised in Table 5.1. We observe that most of the building blocks identified in the literature are absent in the proposed methods. The WOF and GLMO approaches are marked partly for the "random grouping" building block as a representative of simple grouping methods. Both of them can be (and partly were) used in the literature with random groups before, but both were used with other grouping methods like the ordered grouping as well (Section 5.1.6). Mainly, both algorithms do not use any interaction-based or contribution-based groups. Other than that, we can see that both WOF and LCSA use the blocks to create and optimise a transformed, lower-dimensional problem. Looking back to the situation in Table 4.1, only two methods, ReMO and LSMOF, make use of this kind of technique in the literature, and it should be noted that LSMOF does so while being based on WOF. One common building block in all of the three proposed methods is that they all, at some point in their process, optimise the original large-scale problem. In case of the GLMO, in fact, only the original problem is optimised, but with a modified operator, while all optimisation takes place on the original problem.

Dimensionality Reduction

In the following, we categorise the proposed methods based on the categories in Chapter 4 as shown in Table 5.2. Both GLMO and LCSA require different building blocks for their dimensionality reduction techniques. While the former needs a variable grouping method inside the mutation operator, the latter relies on reducing the dimensionality through a search in a spanned subspace as described in Section 5.3. WOF uses pivot

	Building Blocks															
	Random Grouping	Interaction-based Grouping	Contribution-based Grouping	CC-based optimisation of large-scale problem	CC-based optimisation of convergence-variables	Optimise large-scale problem	Optimise diversity-variables	Indicator-based optimisation	Indicator-based local search	Optimise transformed problem	Optimisation of a single group of variables	Update global archive	Create independent populations	Convergence detection	Problem transformation
WOF	(✓)					✓				✓					✓
GLMO	(✓)					✓									
LCSA						✓				✓					✓

Table 5.1: Building blocks from the related works which are present in the proposed large-scale algorithms.

solutions and variable groups to transform the problem. Since in both WOF and LCSA, the new, lower dimensional problem is created by the introduction of new variables which span a new search space inside the large-scale problem, both of them can be regarded as transformation-based approaches in the classification of dimensionality reduction (category 2). GLMO, on the other hand, can be seen as part of the coevolution-based group of algorithms, since it changes variables in the mutation operator only in certain groups while leaving the other groups fixed. On the other hand, the crossover or PSO movements (depending on the used metaheuristic) are always performed on the original large-scale problem. Although other approaches like MOEA/DVA also apply optimisation on the original large-scale problem, the related methods usually divide the CC-based optimisation and the large-scale optimisation in temporal order (meaning the algorithm applies one of these approaches at a time, after each other). This means that whenever CC-based optimisation is used, the whole evolutionary process using crossover and mutation is applied only to a single variable group. In contrast, this division is not done in GLMO, and only the original problem is optimised. As a result, GLMO falls in the "no reduction at all" category of dimensionality reduction (category 3). If we look again at Table 4.2, we see that the only other method in the literature so far which is not using dimensionality reduction is the DLS-MOEA. Since now GLMO also falls into that category, with the proposed methods included still only 2 out of 15 methods make use of this strategy.

Diversity Management

Next, we take a look at the ways of diversity management of the three methods. As described above and in the original publications, the WOF method, applied to a certain pivot solution, usually can leads to a fast convergence speed towards some optimal

Year		Algorithm	Dimensionality Reduction Category 1	2	3	Diversity Management Category 1-1	1-2	2	3	4	Many-objective	Parallel
2016	[4] [1]	WOF		✓						✓	✓	(✓)
2016	[5]	GLMO			✓			✓			✓	○
2019	[7]	LCSA		✓				✓			✓	

Table 5.2: Classification of the proposed large-scale methods. WOF is the only method in the new category 4 of diversity management which is not existent in the related literature.

solutions. The drawback of converging too fast to only one region of the Pareto-Front is the risk of loosing diversity. Therefore, WOF is designed to achieve diversity by using multiple pivot solutions $\vec{x}'_k$ in each of the weighting optimisation steps, to obtain convergence to different parts of the Pareto-front and therefore achieve and maintain diversity. The mechanism that allows WOF to balance between diversity and convergence is the selection of the pivot solutions, and to do the transformation of the problem multiple times with these solutions before merging the populations. Unfortunately, this method of achieving diversity is not used in any of the related works and it is different from the proposed main categories and subcategories introduced in Chapter 4. WOF is not part of the categories 1-1 or 1-2 since it does not use diversity-related variables. It does not use indicator-based optimisation as in the third category. However, since category 2 consists of those methods which do not explicitly have a mechanism for managing diversity, this category does not fit to WOF either. We therefore introduce a **fourth main category of diversity management**, which we define as **achieving diversity through pivot solutions**. WOF is currently the only representative of this way of obtianing diversity. The forth category is added into Table 5.2.

Noteworthy is that the new LSMOF algorithm from the literature [69] is strongly based on WOF in the way that it uses the transformation function with weight vectors to create the new subproblems. It does, however, as described above, not make use of any mechanism to select specific solutions as candidates for this transformation. In fact, since the transformation step is only done once in the beginning of LSMOF, it uses every solution in the initial population of the algorithm to do a transformation. And while this is probably not harmful to the diversity, the LSMOF clearly does not take any steps of using specific solutions for increasing diversity either, and especially not in the remaining parts of the algorithm. Instead, since all these transformed problems are optimised at the same time, the Hypervolume indicator is used as described in Section 3.2.

The GLMO approach does not possess a specific method of diversity management in the basic implementation, since it only specifies using (simple) variable groups. GLMO basically takes care of diversity by always optimising the whole, high-dimensional problem with its crossover operator. Therefore, like the other algorithms in category 2 of diversity management, it relies on the selection mechanism of the used metaheuristic to achieve diversity. If GLMO is applied to a NSGA-II or NSGA-III algorithm, it is to be expected

that the final performance in terms of diversity is entirely depended on the diversity capabilities of those algorithms. However, this classification relies on the assumption that the used grouping mechanism in the mutation operator is either a simple method or an interaction-based method as described in Section 4.2. It is, on the other hand, very easy to use also contribution-based groups or even a combination of contribution-based and other methods to obtain the groups for the mutation. For the sake of computational budget, these groups might have to be precomputed in the beginning of the algorithm, as they should not be redone in every iteration like the simple methods can. If contribution-based groups are applied in the GLMO method, it might raise the computational budget needed for the approach, but it might be beneficial for the overall performance and the diversity of the problem. The GLMO can therefore easily be extended in this way to be a member of category 1-1 of diversity management. Nonetheless, in its original version, and as used in this thesis, it only uses simple groups, and by that does not manage diversity. In Table 5.2 the approach is listed in category 2 as a result.

Regarding the LCSA, it achieves diversity basically through the assumption that the inter- and extrapolation of solutions in the transformed problem can produce diverse solution candidates, and that the selection mechanism of the used metaheuristic is able to keep these in the population. In addition, LCSA optimises the original problem in turn with the transformed one, which makes it also similar to other methods like WOF, MOEA/DVA or LSMOF in the way that there are phases of optimising the original problem. LCSA therefore also belongs to diversity-management category 2, which does not include specialised ways of obtaining or retaining diversity (see Table 5.2). However, the linear search mechanism is actually designed and shown in the original publication to benefit diversity [7]. This is because it enables the algorithm to produce solutions through the extrapolation in a linear hyperplane, which can lead to a good exploration and produce more diverse solutions within a certain converged area in the search space. Even though LCSA belongs technically to category 2, because it relies on the used metaheuristic, it is designed with the goal of increasing diversity in the population. Nevertheless, it does not manage this increased diversity with its own mechanism.

Dealing with Many-objective Problems

Following the analysis of the related work in Table 4.2, we now give a brief analysis of the many-objective capabilities of the proposed methods as well. In their original publications, only LCSA was actually designed for many-objective problems, while WOF ans GLMO were mostly used with 2- and 3-objective problems. However, all of them possess certain many-objective capabilities if the employed metaheuristic is able to deal with many-objective problems.

For GLMO, the diversity management is entirely up to the used metaheuristic, and the modified mutation operator can be used directly inside a many-objective algorithm like NSGA-III. In the case of WOF, the same applies, but another aspect of WOF can

affect its efficiency for many-objective problems as well. This regards the selection and especially the amount of chosen pivot solutions $\vec{x}'_k$. These solutions are used to control the diversity of the algorithm, and it is recommended to use a number of pivot solutions that is at least the number of objective functions. Therefore, if reference directions are used for the pivot-selection, as was proposed in [6] and described in Section 5.1.7, at least one transformed problem is formed for the extremal solutions in each objective function, i.e. the solution which has the best value in one objective. Using this recommendation of the parameter q, we can assume that WOF has a kind of built-in scalability for increasing numbers of objectives. On the other hand, it is not to be expected that WOF would achieve a satisfactory performance for many-objective problems when using NSGA-II as the optimiser, even if the number of pivot solutions is increased, due to the Pareto-dominance-based selection mechanism of NSGA-II. Therefore, the many-objective capabilities of WOF depend on different aspects, but the approach can be adjusted to be applicable to many-objective problems using the correct selection mechanisms and metaheuristics.

Regarding the LCSA, since the experiments in the original publication actually showed good performance for many-objective instances, it can be assumed that it is a promising way to increase the performance of existing many-objective algorithms. It was shown that the linear search through extrapolation is able to significantly increase the solution quality of NSGA-III as well as RVEA on 4- and 5-objective benchmark problems. This shows that not only does the approach work well for many-objective optimisation, it also further increased the performance of already well-performing algorithms on these problems. Since the goals of the LCSA is not just dimensionality reduction, but also the identification of relevant subspaces to increase exploration, among the three proposed algorithms, the LCSA is the one whose design is intended to work for many-objective optimisation.

To further explore the capabilities of the proposed methods, in the experimental evaluation of this thesis (see Chapter 6) all of the three approaches are also tested on many-objective problems, and show good performance when they are used with many-objective algorithms. For this reason, Table 5.2 lists all three approaches as suitable for many-objective optimisation.

Parallel Implementations of the Proposed Methods

Next, we focus on the possible parallel implementations of the proposed methods. It is clear that the GLMO is actually not easy to parallelise on its own, or at least not easier than to parallelise any existing metaheuristic that uses the mutation. Since the algorithm structure does not deviate from the usual flow of the underlying metaheuristic, it is therefore dependent on the underlying algorithm if GLMO works well for parallel computation. This is denoted with a ◯ in Table 5.2.

To parallelise WOF, the same is true as for GLMO, which means if a parallel algorithm is used, parts of the optimisation can be done in parallel. However, this is only partly effective, since WOF consists of the different steps of optimisation, namely the normal optimisation of the original problem and the optimisation of the q transformed problems. The optimisation of the q different independent problems can be done in parallel with any metaheuristic, which makes WOF easy to parallelise in these phases. An issue is, on the other hand, that these parallel processes only exist for a certain time and there is frequent need for communication between cores to distribute the created problems and gather their results. The merging of the populations and the subsequent large-scale optimisation phase needs to be done in a central instance again, before new q problems are created. In addition, WOF stops this alternation at a certain point in time to focus entirely on the original problem. WOF is therefore, in parts, parallelisable, but is by design not meant to work very efficiently on multiple cores at the same time.

The LCSA is probably the hardest to parallelise among the three proposed algorithms, since it has alternating phases similar to WOF that would require communication and data transfer, except that it does not use multiple transformed problems, but just a single one. Therefore, even if LCSA is used with a parallel EA, there is increased need for central coordination.

5.5 Summary

In this chapter, the proposed approaches to solve large-scale optimisation problems were explained and analysed. The three methods WOF, GLMO and LCSA were described together with their components in detail and possible advantages and shortcomings were discussed. The proposed algorithms were further classified based on the identified categories from the previous chapter. The comparison of the methods showed various differences in terms of dimensionality reduction and diversification, and revealed that the proposed techniques are substantially different in many aspects from most of the large-scale work in the literature.

Chapter 6

Evaluation

In this chapter, the proposed methods are experimentally evaluated, alongside some of the most prominent and recent related algorithms from the literature. This is done by performing a variety of different experiments with various versions of the metaheuristics, varying parameter settings and computational budgets. The goal of this chapter is to contribute an in-depth analysis which compares the state-of-the-art using the same benchmarks, parameters and computational budgets. 184 different problem instances are used for this purpose, ranging from low-dimensional to large-scale and from 2-objective to many-objective instances with different properties. In total, the experiments for this thesis required multiple weeks of calculation time in parallel on multiple machines with between 20 and 60 cores each, and is the most extensive experimental evaluation of large-scale methods up to date (to the best of our knowledge). The details are as follows.

After introducing the general parameter settings for the evaluation, the main contributions of this thesis to large-scale optimisation, namely the WOF, GLMO and LCSA are evaluated in Sections 6.2 to 6.4. Each of the proposed methods is examined in detail on its own by applying them with different configurations to a variety of benchmark functions and by comparing them with their respective low-scale optimisers. Section 6.5 compares the performance of the three algorithms with each other.

In a next step, some of the respective best performing configurations from each of these methods are used to compare their performance to state-of-the-art large-scale algorithms. This part is a central focus of the evaluation chapter, as it sets the proposed methods into relation with the current state of large-scale optimisation algorithms (Section 6.6). These experiments are performed on different computational budgets to compare with methods that are based on interaction-based groups.

In the last step, we investigate the influence of interaction-based groups on the success of certain large-scale methods (Section 6.7). To do so, the used interaction-based groups of these methods are replaced by random groups and their performance is compared

again to that of other large-scale techniques. The last section of this chapter provides a summary and discussion of the experiments.

6.1 General Experiment Settings

In this section, the general settings of the experiments are described. These settings apply to most of the following experiments in this chapter. Wherever the settings of a specific experiment differ from the ones described here, this is indicated respectively. In the following, we first describe general experiment settings, and then go into detail into the configuration of related algorithms, implementation details and configurations of the proposed methods and the used benchmarks and their settings in the subsequent subsections.

The experiments were done using the PlatEmo framework, version 2.0.4 [113]. All experiments are performed for 31 independent runs for each configuration of an algorithm on the specific problem instance. The results are examined for statistical significance using the two-sided pairwise Mann-Whitney-U Test (also called the Wilcoxon Ranksum Test) [114, 115], with the null hypothesis that the distributions of the two samples which are compared have equal medians. A threshold value of 0.01 is used, i.e. statistically significant differences between the performance of two algorithms is assumed for a p-value smaller than 0.01.

The two performance indicators for the experiments are the IGD metric [65] and the Hypervolume (HV) indicator [75, 76], as described in Section 2.7. The IGD metric is commonly used in the literature for multi- and many-objective optimisation, and is in general able to measure convergence and diversity of the obtained solution sets simultaneously. It serves as the main evaluation metric for measuring final solution quality as well as convergence behaviour over time, and was also used by many of the existing large-scale algorithms [67, 68, 69, 24, 25, 70, 71]. The HV indicator is used as the secondary metric. Due to its large computational effort and the extent of the experiments (which took multiple weeks of parallel computation), HV values are only computed on the final solution sets of each algorithm, and only the IGD is used to track the behaviour of the algorithms during the runtime. Furthermore, two of the related algorithms which are used in the experiments (DLS-MOEA and LSMOF) use the Hypervolume internally in their indicator-based optimisation. It can therefore mislead the evaluation if the same metric is used for evaluation that is internally optimised by some of the algorithms.

The IGD values are computed using a sample of the respective Pareto-front of the benchmarks. The samples are provided by the used PlatEmo framework. The sample sizes are based on the number of objectives and are set to $10,000$ points for the 2-objective instances and to 9870, 9880 and 8855 points for the 3-, 4- and 5-objective problems. The used reference points used for the Hypervolume calculations are obtained using the respective nadir points of samples of the benchmarks, multiplied with a factor of 2.0 in

each dimension. These samples to obtain the nadir points are of size 2000, 1953, 1771 and 1820 for the 2-, 3-, 4- and 5-objective instances respectively. Note that the sizes of the reference sets produced by the framework can be smaller than these numbers, for instance in the case of disconnected or degenerated Pareto-fronts.

All algorithms use a computational budget of $100,000$ function evaluations in all experiments if not stated otherwise. A budget of $10,000,000$ evaluations is used instead in some of the experiments to compare with the state-of-the-art. The population sizes are set to 100, 91, 84 and 85 for the 2-, 3-, 4- and 5-objective instances respectively. These numbers are chosen for all algorithms, so that the population sizes account for classical and many-objective algorithms, which often require evenly distributed reference directions.

6.1.1 Configuration of State-of-the-Art Algorithms

Implementations of the related works from Section 3.2 have been obtained through contacting the corresponding authors of the algorithms. In addition, some of the large-scale algorithms have been provided by the used PlatEmo framework, and have been altered by the author of the thesis to resemble the original publications as closely as possible. All of our own proposed contributions have also been implemented in the PlatEmo framework, which provides implementations of the used benchmark functions as well. Based on the available sources, the authors were able to obtain implementations of the CCGDE3, LMEA, MOEA/DVA, S^3-CMA-ES, DLS-MOEA, ReMO and LSMOF algorithms from the literature. These methods in total represent all of the different categories of dimensionality-reduction, diversity management, and many-objective capabilities as proposed in Chapter 4. The CCGDE3 method is excluded from the experimental evaluation, since it was shown in various publications that its performance is not comparable to that of other modern mechanisms, and at the same time its methods of dimensionality reduction and diversity management are represented by the MOEA/DVA and LMEA techniques as well.

Some of related methods are implemented in multiple different versions as follows. ReMO is implemented using NSGA-II as well as MOEA/D as internal optimisers, while the LSMOF implementation is used with the NSGA-II and SMPSO optimisers, as these versions were also used in the original publication [69]. Further, to examine the effect of interaction-based groups on the performance of MOEA/DVA, LMEA and S^3-CMA-ES, we implemented each of the algorithms as a random-group-based version. In these algorithms, the interaction-based groups are replaced with a random grouping. In this way, we can also apply and compare these search techniques on a smaller computational budgets, since the original versions are not applicable without a minimum of multiple million evaluations. As a result, a total of 11 different related algorithms are used in the experiments, denoted as LMEA, MOEA/DVA, S^3-CMA-ES, randomLMEA, randomMOEA/DVA, randomS^3-CMA-ES, DLS-MOEA, Re-NSGA-II, Re-MOEA/D, LS-NSGA-II and LS-SMPSO.

Whenever possible, the parameters used in the respective original publications are used in our experiments to follow the recommendations that the original authors of the methods made. Further, we use the same parameters in all algorithms that depend on the same mechanisms, e.g. when random grouping is used, all algorithms use the same number of groups, when interaction-based grouping is used, the same parameters for the grouping is used in LMEA and MOE/DVA. The further parameter settings of the related methods are listed in the following. If deviations from these parameters exist in a specific experiment, this is stated in the respective section.

- LMEA is configured using the parameter values $nSel = 2$, $nPer = 4$ and $nCor = 6$.
- MOEA/DVA in used with $NCA = 50$ and $NIA = 6$.
- S^3-CMA-ES uses $nPer = 50$ and further retains the same parameters as in the source code provided by the authors and in the PlatEmo framework. The number of subpopulations is set to 5, the size of the subpopulations is 10. The group size for separable variables in the interaction analysis is set to 35.
- The randomised versions of LMEA, MOEA/DVA and S^3-CMA-ES use $\gamma = 4$ groups of even size, where the last group contains additional variables in case n is not evenly divisible by 4.
- DLS-MOEA was obtained from the original authors and is configured as in the original publication, with the exception of reducing the number of generations per phase. In the original publication, each of the two phases was used for $20,000$ generations. However, this roughly corresponds to $800,000$ function evaluations, which exceeds the number of maximum evaluations in most of our experiments. Therefore, the number of generations for each phase is set to 200. The crossover probability is set to 0.9 as in the original work.
- Re-NSGA-II and Re-MOEA/D were obtained as source codes from the original authors, and the parameters are taken from their implementation. The size of the transformed problems is set to $v = 50$. In Re-NSGA-II, crossover probability is 0.7 and mutation probability is set to 0.4. The mutation rate and step size are set to 0.02 and 0.2, and the lower and upper bounds of the variables of the transformed problems are set to -1 and 1 respectively. Re-MOEA/D uses a neighbourhood size of 15. Since in the original publication ReMO was only applied to ZDT problems, where the variable domains are between 0.0 and 1.0, and this is not necessarily the case in the more complex benchmarks used in our experiments, we adapt an additional mapping of the variables into the domain of the current problem's domains after the transformation step in each function evaluation.
- The LS-NSGA-II and LS-SMPSO codes are also obtained through contacting with the authors of the method. The number of generations for the weight optimisation is set to 10, the population sizes of the transformed problems is set to 30.

- Wherever an algorithm uses polynomial mutation [101, 102], it is configured with a distribution index of 20.0 and a probability of $1/n$. An exception are the cases where the structure of an algorithm explicitly changes these values, as in DLS-MOEA or LMEA. SBX crossover [100], where applicable, is used with a distribution index of 20.0 and a crossover probability of 1.0.

6.1.2 Configuration of the Proposed Methods

Regarding the proposed approaches from Chapter 5, each of the methods is implemented with multiple different optimisation algorithms and in different configurations as follows.

Five different variants of the WOF are used in this chapter, using different internal optimisers. First, NSGA-II and SMPSO are used, as they have been used in earlier publications and in the related work. In addition, MOEA/D and NSGA-III are used inside WOF for the first time, which have not been present in the literature so far. The fifth version of WOF makes use of a random choice of the internal optimiser. This new version of WOF is called *WOF-Randomised*, and is aimed to make use of the potentials of each of the separate optimisers, which minimises the risk of using a non-suitable optimiser for a specific problem instance. WOF-Randomised works as follows. In the first phase of the optimisation process (determined by the parameter δ), the algorithms to optimise the original problem and the transformed problems are drawn randomly from the pool of NSGA-II, SMPSO, NSGA-III and MOEA/D each time. In the second phase of the search, NSGA-III is used in order to obtain and preserve diversity. For all five WOF versions, the following parameters are used: $\delta = 0.5$, $\gamma = 4$ with the ordered grouping method, $t_1 = 1000$, $t_2 = 500$, $q = m + 1$, the transformation function used is the parameter-free transformation (ψ_4), and the pivot solutions are drawn based on reference lines.

The proposed extensions to the mutation operators, namely the grouped mutation, the linked mutation, and the grouped linked mutations, are implemented into the NSGA-II, SMPSO and NSGA-III algorithms. For each of these three optimisation algorithms, besides the original version, four additional versions were created. One where the mutation is replaced by a linked mutation, another where the mutation is replaced with a grouped mutation, and one where the grouped and linked mutation is used. In addition, as in the original work, we compare one version of the optimisers which uses normal polynomial mutation, but with a high mutation probability, which equals 0.25. This is done to make a fair comparison, since a number of $\gamma = 4$ groups is used in the modified operators, and the expected amount of changed variables is comparable to a high mutation rate in the normal polynomial mutation. The ordered grouping method was used to create the groups. In all of these operator versions, a distribution index of 20.0 is used.

The linear search mechanism LCSA was implemented into NSGA-II, SMPSO and NSGA-III as well. Here, the original problem is optimised for 100 generations at a time before one linear search optimisation step is carried out. The modified linear search uses the

same optimiser and environmental selection strategies, the same population size, and is done for 30 generations at a time. The whole population is used to create the new problem instead of only the non-dominated solutions, which means the new linear search problem contains the same amount of decision variables as the population size of the original problem. The lower and upper bounds of the coefficient variables are set to $[y_{i,min}, y_{i,max}] = [-10.0, 10.0]$ for all variables.

6.1.3 Benchmark Problem Specification

Regarding the used benchmark problems and the corresponding numbers of variables and objectives, we test the algorithms on a total of 184 different problem instances to cover a wide range of different combinations. Among them are low-scale and large-scale problems, multi- and many-objective instances, and different benchmark families. The details are listed below.

1. We use the four most prominent benchmark families, which also are mostly applied in the related algorithm's literature. These are the DTLZ1-7 [40], WFG1-9 [28], UF1-10 [48], and LSMOP1-9 [44] benchmarks, which amount to a total of 35 distinct test problems.

2. The DTLZ1-7 problems are used with $n = 40$ and $n = 1000$ variables, and each of these in combination with $m = 2$, 3, 4 and 5 objective functions, resulting in 56 different instances of the DTLZ benchmark suite.

3. The WFG1-9 problems are used with $n = 40$ and $n = 1000$ variables, and each of these in combination with $m = 2$ and $m = 3$ objective functions. The number of diversity-related variables, which can be freely scaled in the WFG suite, is set to $n/4$ in all instances. For the WFG2 and WFG3 functions, the 2-objective instances are specified with 41 and 1001 variables respectively, as the problem structure requires these configurations. In total, we obtain 36 instances of the WFG problems.

4. The UF1-10 problems from the CEC2009 competition on multi-objective optimisation are used, where the UF1-7 problems are 2-objective, while UF8-10 contain 3 objective functions. All of them are used with $n = 40$ and $n = 1000$ variables, which amounts to 20 instances of the UF problems.

5. The LSMOP benchmark suite is used, where each of the 9 functions is used with $m = 2$, 3, 4 and 5 objective functions. The number of variables is in the original work of the LSMOP linked to the number of objectives, and it is used in this way in many of the recent publications. Therefore, we follow this approach and use each of the functions with $n = m \cdot 100$ variables for the different numbers of m. In addition, we use all the of the instances with $n = 1000$ variables respectively, to obtain more results for large search spaces even with low numbers of objectives. In

total, this amounts to 72 different instances of the LSMOP benchmark suite. The parameter n_k in the problems is set to its standard value of 5.

The total of 184 different problem instances enables the evaluation to examine the performance of the algorithms broadly from low-scale, traditional problem sizes, with 40 variables and 2 objectives, up until large-scale instances as used in related works, with up to 5 objectives and 1000 variables. The combinations of these settings further allow to examine whether algorithms perform well on large search spaces independently of the number of objectives. The problem setup is identical for all experiments, except where stated otherwise for certain reasons. This is the case where only 2- and 3-objective instances are used due to the computational overhead of Hypervolume calculations in DLS-MOEA, and where experiments with up to $10,000,000$ function evaluations would render the computation time intractable if applied to all of the 184 instances.

6.1.4 Presentation of Results

Due to the vast amount of experiments in this evaluation, it is necessary to accumulate the results to make them presentable in the course of this chapter. The detailed results of all experiments with regard to the final solution quality (i.e. final IGD values) for all problem instances and all algorithms in the different experiments are given in Appendix B. Tables B.1 to B.47 show the median obtained IGD values after the total amount of function evaluations for the specific experiments, together with the interquartile range (IQR). The respective best algorithm in each row (i.e. for each problem instance) is shown with a grey background in the cell, while statistical significance is computed to this respective best method and denoted with an asterisk in the other method's columns where applicable.

In this chapter, the results are summarised in the following form. The 184 problem instances are divided into four categories and presented as exemplarily shown in Table 6.1. In these tables, the respective algorithms which took part in the experiment are shown in the rows and columns, and each cell shows the pairwise comparison between two methods. The numbers in the cells indicate on how many problem instances the algorithm in the row performed significantly better than the algorithm in the column. The numbers are given as percentages from the total number of benchmarks, and we refer to these numbers as *winning rates* or *winning scores* in the remainder. For instance, looking at Table 6.2, the cell in the second row (WOF-SMPSO) and the third column (NSGAII) indicates how often WOF-SMPSO performed significantly better than NSGA-II. In the same way, the numbers in the third row and second column indicate how often NSGA-II outperformed WOF-SMPSO.

The four coloured numbers in each cell correspond to the different problem categories to enable a detailed analysis. For each comparison, the black number in the upper left shows the comparison on all problems of the experiment (in case of Table 6.2 all 184

Table 6.1: Example of the presentation of winning rates for different problem categories. The four numbers indicate the amounts of wins of *Algorithm A* in the row (based on statistical significance) against *Algorithm B* in the column for the respective four categories.

		Algorithm B	
184	56		
92	64		
Algorithm A		60.08	23.78
		25.00	9.56

instances). The red numbers in the upper right show the winning scores for only the low-scale problems which contain 40 or 41 decision variables, regardless of the number of objectives. The blue numbers in the lower left represent the performance on all large-scale problems, which contain 1000 or 1001 variables and arbitrary numbers of objective functions. Finally, the green numbers on the lower right show the winning scores on the many-objective instances, which contain 4 or 5 objective functions (with any number of variables). To obtain a better understanding of the numbers, the respective amounts of instances in each of the four categories are shown in each table in the upper left, next to the algorithms' names. Using this kind of analysis, we can deduct from Table 6.2, for instance, that WOF-SMPSO outperformed NSGA-II on low-scale instances with 40 variables in 24 out of 56 problems (42.85%), while NSGA-II performed superior in 51.78% of cases (29 out of 56). The remaining $56 - 24 - 29 = 3$ cases resulted in a draw between these two algorithms. Tables 6.2 to 6.17 in this chapter show the results based on the detailed IGD values in Appendix B. The same winning rate tables based on HV results are shown in Appendix C.

As we saw in Section 4.1, the amount of required function evaluations differs largely between algorithms in the literature. Some algorithms often need millions of evaluations to achieve a good solution quality, hence the actual usefulness of them for a decision maker is difficult to judge by only the final solution sets. Therefore, it is of interest to explore each algorithm's behaviour and solution quality over time, the convergence behaviour. For this purpose, we show selected problem instances as convergence plots, where the IGD values of the compared algorithms are shown against the used function evaluations. In this kind of plots, for instance in Fig. 6.2, from the 31 independent runs, for each algorithm the respective run which achieved the median IGD value at the end of the optimisation process is shown. IGD values are displayed on the vertical axes and displayed on a logarithmic scale.

Table 6.2: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations of the SMPSO and NSGA-II algorithms and their WOF-versions as well as the randomised WOF algorithm. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	SMPSO	WOF-SMPSO	NSGA-II	WOF-NSGA-II	WOF-Randomised
SMPSO	—	0.54 | 0.00 0.00 | 0.00	43.47 | 12.50 65.21 | 50.00	19.56 | 14.28 22.82 | 35.93	2.71 | 3.57 2.17 | 1.56
WOF-SMPSO	90.76 | 78.57 95.65 | 85.93	—	80.43 | 42.85 96.73 | 89.06	60.86 | 37.50 77.17 | 53.12	32.06 | 17.85 43.47 | 9.37
NSGA-II	40.21 | 62.50 25.00 | 29.68	16.84 | 51.78 2.17 | 10.93	—	17.39 | 41.07 4.34 | 14.06	3.80 | 12.50 0.00 | 0.00
WOF-NSGA-II	73.36 | 67.85 76.08 | 59.37	24.45 | 41.07 13.04 | 31.25	66.84 | 17.85 91.30 | 71.87	—	10.86 | 10.71 9.78 | 7.81
WOF-Randomised	90.21 | 78.57 94.56 | 90.62	45.65 | 62.50 35.86 | 59.37	87.50 | 60.71 100.00 | 96.87	73.91 | 60.71 81.52 | 84.37	—

6.2 Evaluation of the Weighted Optimisation Framework

In this section, we compare the performance of different versions of the WOF algorithm framework with the respective original algorithms, and examine how the proposed method is suitable to increase the performance of algorithms on large-sale optimisation. In total, we compare 9 different algorithms, which are the four original methods NSGA-II, SMPSO, MOEA/D and NSGA-II, the four respective WOF versions of these methods, and the above-described randomised version of WOF. All 184 problem instances are used in this experiment. For the analysis in the winning-score tables, these problems include 56 low-scale instances, 92 large-scale instances, and 64 many-objective instances. The winning scores of this experiment are shown in Table 6.2 and Table 6.3, as well as in Tables B.1 to B.8 in Appendix B for all problems.

Regarding WOF-SMPSO and SMPSO, we see in Table 6.2 that the WOF version outperfoms SMPSO in 90.76% of all problems (167 out of 184 instances), where the remaining instances consist of 16 draws and only 1 out of 184 times (0.54%) the original SMPSO was able to perform better than WOF-SMPSO. We further observe that the good performance occurs in all kinds of problem categories, where WOF wins in 44 out 56 instances (78.57%) in the low-scale problems, 88 out of 92 times (95.65%) in the large-scale and 55 out of 64 times (85.93%) in the many-objective area. If we further take a look at the detailed results in Appendix B, we see that the superior performance

Table 6.3: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations of the NSGA-III and MOEA/D algorithms and their WOF-versions as well as the randomised WOF algorithm. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	NSGA-III	WOF-NSGA-III	MOEA/D	WOF-MOEA/D	WOF-Randomised
NSGA-III	—	21.73 | 55.35 5.43 | 23.43	45.65 | 58.92 47.82 | 20.31	26.08 | 71.42 4.34 | 14.06	12.50 | 37.50 1.08 | 10.93
WOF-NSGA-III	62.50 | 12.50 88.04 | 62.50	—	66.84 | 50.00 84.78 | 54.68	45.65 | 67.85 33.69 | 31.25	4.89 | 0.00 5.43 | 7.81
MOEA/D	45.10 | 21.42 46.73 | 68.75	22.82 | 37.50 11.95 | 37.50	—	25.00 | 50.00 8.69 | 25.00	13.04 | 23.21 6.52 | 18.75
WOF-MOEA/D	67.39 | 17.85 90.21 | 78.12	40.21 | 25.00 50.00 | 51.56	61.41 | 25.00 86.95 | 57.81	—	18.47 | 12.50 15.21 | 35.93
WOF-Randomised	72.28 | 19.64 95.65 | 73.43	61.95 | 44.64 72.82 | 57.81	73.36 | 51.78 91.30 | 60.93	57.60 | 66.07 57.60 | 32.81	—

is spread across all problem families. We can conclude from this that WOF is able to improve the performance of SMPSO for almost all problems independent of their properties or dimensionality in decision or objective space.

The NSGA-II variants show in general a similar performance, although the superiority over the original NSGA-II is a little lower than in the SMPSO case. WOF-NSGA-II outperforms NSGA-II in 123 of 184 instances (66.84%), while NSGA-II wins in 32 instances (17.39%). Interestingly, we can see that the performance in terms of low-scale problems differs from the one on large-scale and many-objective problems. This is visible in Table 6.2 (third row, fourth column), where NSGA-II can only outperfom its WOF version in 4 out of 92 large-scale problems (4.34%), and 9 out of 64 many-objective instances (14.06%). Therefore, we can conclude that WOF-NSGA-II shows a superior performance on the large-scale and many-objective problems, but does not improve the results of NSGA-II on classical problems with small numbers of variables on the same scale as with the SMPSO.

Next, we take a look at the results for NSGA-III in Table 6.3, which is a dedicated many-objective optimiser. Also in this case we see that the WOF is able to enhance the performance strongly. While NSGA-III is still able to claim around 55% of the wins for low-scale problems, WOF-NSGA-III clearly outperforms NSGA-III on most of the large-scale and many-objective instances (88.04% and 62.5% respectively), with a

Table 6.4: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations of NSGA-II and its grouped and linked versions. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	NSGA-II	GroupedNSGA-II	LinkedNSGA-II	GroupLinkNSGA-II	HighProbNSGA-II
NSGA-II	—	50.54 | 87.50 39.13 | 46.87	1.63 | 0.00 3.26 | 0.00	18.47 | 50.00 5.43 | 20.31	58.15 | 87.50 51.08 | 54.68
GroupedNSGA-II	30.43 | 3.57 42.39 | 21.87	—	29.89 | 3.57 42.39 | 20.31	1.08 | 1.78 0.00 | 0.00	53.80 | 46.42 60.86 | 43.75
LinkedNSGA-II	5.43 | 8.92 3.26 | 1.56	51.08 | 87.50 38.04 | 45.31	—	21.19 | 58.92 5.43 | 20.31	59.23 | 87.50 52.17 | 54.68
GroupLinkNSGA-II	69.02 | 17.85 91.30 | 73.43	89.13 | 80.35 94.56 | 89.06	69.02 | 14.28 92.39 | 73.43	—	94.02 | 87.50 97.82 | 95.31
HighProbNSGA-II	25.00 | 1.78 38.04 | 17.18	5.97 | 1.78 7.60 | 1.56	25.00 | 1.78 39.13 | 15.62	1.08 | 3.57 0.00 | 0.00	—

Table 6.5: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations of NSGA-III and its grouped and linked versions. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	NSGA-III	GroupedNSGA-III	LinkedNSGA-III	GroupLinkNSGA-III	HighProbNSGA-III
NSGA-III	—	61.41 | 83.92 52.17 | 67.18	0.00 | 0.00 0.00 | 0.00	21.73 | 55.35 8.69 | 21.87	66.84 | 85.71 58.69 | 71.87
GroupedNSGA-III	28.26 | 7.14 40.21 | 18.75	—	27.17 | 7.14 39.13 | 18.75	1.63 | 5.35 0.00 | 0.00	56.52 | 37.50 71.73 | 50.00
LinkedNSGA-III	8.15 | 16.07 4.34 | 3.12	62.50 | 85.71 54.34 | 71.87	—	21.73 | 55.35 8.69 | 21.87	69.02 | 89.28 61.95 | 75.00
GroupLinkNSGA-III	71.19 | 23.21 90.21 | 73.43	95.65 | 85.71 100.00 | 100.00	69.56 | 17.85 90.21 | 71.87	—	96.73 | 89.28 100.00 | 100.00
HighProbNSGA-III	20.10 | 5.35 31.52 | 14.06	9.78 | 16.07 5.43 | 9.37	20.65 | 5.35 32.60 | 14.06	1.63 | 5.35 0.00 | 0.00	—

Table 6.6: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations of SMPSO and its grouped and linked versions. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	SMPSO	GroupedSMPSO	LinkedSMPSO	GroupLinkSMPSO	HighProbSMPSO
SMPSO	—	4.89 | 1.78 8.69 | 3.12	1.63 | 1.78 1.08 | 1.56	2.17 | 5.35 1.08 | 1.56	7.60 | 7.14 7.60 | 3.12
GroupedSMPSO	35.86 | 21.42 47.82 | 3.12	—	34.23 | 17.85 46.73 | 3.12	2.71 | 8.92 0.00 | 0.00	25.00 | 7.14 40.21 | 3.12
LinkedSMPSO	2.17 | 1.78 2.17 | 0.00	5.43 | 5.35 7.60 | 3.12	—	3.26 | 10.71 0.00 | 0.00	8.69 | 8.92 9.78 | 4.68
GroupLinkSMPSO	63.58 | 37.50 76.08 | 39.06	61.95 | 33.92 73.91 | 43.75	62.50 | 33.92 75.00 | 42.18	—	61.95 | 33.92 72.82 | 40.62
HighProbSMPSO	28.80 | 17.85 35.86 | 0.00	1.08 | 1.78 0.00 | 0.00	27.17 | 10.71 36.95 | 0.00	2.17 | 7.14 0.00 | 0.00	—

Table 6.7: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations. The original NSGA-II, SMPSO and NSGA-III algorithms are shown along their LCSA-enhanced versions. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	SMPSO	xSMPSO	NSGA-II	xNSGA-II	NSGA-III	xNSGA-III
SMPSO	—	7.06 | 17.85 2.17 | 3.12	44.56 | 14.28 66.30 | 53.12	24.45 | 12.50 32.60 | 34.37	27.17 | 7.14 45.65 | 12.50	14.13 | 3.57 22.82 | 7.81
xSMPSO	65.76 | 23.21 79.34 | 73.43	—	71.73 | 21.42 92.39 | 85.93	47.82 | 19.64 64.13 | 53.12	61.95 | 17.85 80.43 | 60.93	36.41 | 17.85 50.00 | 18.75
NSGA-II	41.30 | 64.28 26.08 | 32.81	25.00 | 67.85 7.60 | 12.50	—	27.71 | 33.92 27.17 | 28.12	14.67 | 14.28 13.04 | 4.68	14.13 | 25.00 9.78 | 1.56
xNSGA-II	68.47 | 71.42 65.21 | 64.06	30.97 | 66.07 14.13 | 25.00	52.17 | 10.71 68.47 | 62.50	—	53.26 | 17.85 68.47 | 57.81	17.93 | 19.64 16.30 | 1.56
NSGA-III	65.76 | 78.57 50.00 | 81.25	32.06 | 73.21 15.21 | 31.25	66.30 | 57.14 70.65 | 90.62	35.86 | 51.78 30.43 | 37.50	—	32.06 | 48.21 27.17 | 28.12
xNSGA-III	78.26 | 78.57 73.91 | 85.93	46.73 | 71.42 33.69 | 54.68	77.17 | 51.78 86.95 | 95.31	53.80 | 48.21 56.52 | 82.81	52.71 | 8.92 72.82 | 60.93	—

Table 6.8: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations. WOF and LSMOF are compared using the NSGA-II and SMPSO algorithms. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-SMPSO	LS-SMPSO	WOF-NSGA-II	LS-NSGA-II	WOF-Randomised
WOF-SMPSO	—	61.41 | 66.07 60.86 | 34.37	60.86 | 37.50 77.17 | 53.12	51.63 | 33.92 60.86 | 37.50	32.06 | 17.85 43.47 | 9.37
LS-SMPSO	9.23 | 1.78 13.04 | 21.87	—	40.21 | 26.78 48.91 | 42.18	34.78 | 19.64 41.30 | 35.93	15.21 | 14.28 18.47 | 10.93
WOF-NSGA-II	24.45 | 41.07 13.04 | 31.25	49.45 | 58.92 45.65 | 45.31	—	37.50 | 33.92 41.30 | 32.81	10.86 | 10.71 9.78 | 7.81
LS-NSGA-II	26.08 | 42.85 19.56 | 29.68	42.93 | 62.50 39.13 | 39.06	29.89 | 30.35 34.78 | 23.43	—	17.39 | 28.57 15.21 | 9.37
WOF-Randomised	45.65 | 62.50 35.86 | 59.37	67.93 | 71.42 66.30 | 68.75	73.91 | 60.71 81.52 | 84.37	68.47 | 57.14 70.65 | 75.00	—

large number of non-significant differences (draws) between the two in the low-scale and many-objective cases as well. Overall, NSGA-III can only outperfrom its WOF-enhanced version in 21.73% of all problems, and in large-scale instances only in 5.43% (5 out of 92 instances). This shows that the WOF method is suitable also for many-objective optimisation if an appropriate method like NSGA-III is used. Regarding MOEA/D in comparison with WOF-MOEA/D, the picture looks very similar to the one in NSGA-III, with similar winning scores in all four categories. An interesting observation is here that WOF internally uses the concept of Pareto-dominance to merge its created population from the different subproblems. Nonetheless, it is able to improve the performance also in algorithms like MOEA/D, which on their own do not rely on this concept.

These results, which reflect the final algorithm performance with respect to obtained IGD values, show that WOF is able to significantly outperform existing methods in large-scale optimisation. It is further able to significantly increase the performance of existing algorithms in many cases on traditional, low-scale problems. The application to 4 different optimisers shows that the improvement is strongest when applied to the SMPSO algorithm, but in general suggest that the WOF method can be successfully used with any arbitrary metaheuristic.

To bring out the best of the four versions of WOF, the randomised version as described above is compared to the respective other versions and the original algorithms. The

Table 6.9: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations. WOF is compared with the random-group-based MOEA/DVA, LMEA and S^3-CMA-ES. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	randomLMEA	randomMOEA/DVA	randomS3-CMA-ES
WOF-SMPSO	—	60.86 | 37.50 77.17 | 53.12	32.06 | 17.85 43.47 | 9.37	72.28 | 33.92 90.21 | 60.93	86.41 | 66.07 95.65 | 78.12	89.67 | 85.71 94.56 | 76.56
WOF-NSGA-II	24.45 | 41.07 13.04 | 31.25	—	10.86 | 10.71 9.78 | 7.81	66.84 | 30.35 83.69 | 53.12	73.36 | 64.28 77.17 | 53.12	89.13 | 87.50 93.47 | 76.56
WOF-Randomised	45.65 | 62.50 35.86 | 59.37	73.91 | 60.71 81.52 | 84.37	—	78.26 | 44.64 95.65 | 67.18	92.39 | 83.92 95.65 | 90.62	91.84 | 89.28 95.65 | 82.81
randomLMEA	25.54 | 60.71 8.69 | 34.37	29.89 | 62.50 14.13 | 42.18	16.84 | 41.07 3.26 | 28.12	—	71.73 | 78.57 72.82 | 67.18	75.54 | 94.64 80.43 | 67.18
randomMOEA/DVA	10.32 | 25.00 4.34 | 18.75	21.73 | 25.00 19.56 | 39.06	3.80 | 7.14 2.17 | 3.12	20.10 | 16.07 22.82 | 23.43	—	59.23 | 80.35 56.52 | 48.43
randomS3-CMA-ES	7.60 | 7.14 5.43 | 20.31	9.23 | 8.92 6.52 | 20.31	5.97 | 5.35 3.26 | 14.06	18.47 | 3.57 15.21 | 23.43	31.52 | 10.71 34.78 | 39.06	—

Table 6.10: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations. WOF is compared with ReMO using NSGA-II and SMPSO. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-NSGA-II	ReNSGA-II	WOF-MOEA/D	ReMOEA/D	WOF-Randomised
WOF-NSGA-II	—	91.30 | 94.64 90.21 | 84.37	35.86 | 46.42 29.34 | 23.43	88.58 | 94.64 85.86 | 71.87	10.86 | 10.71 9.78 | 7.81
ReNSGA-II	5.97 | 3.57 7.60 | 12.50	—	0.54 | 1.78 0.00 | 0.00	69.56 | 85.71 66.30 | 35.93	0.00 | 0.00 0.00 | 0.00
WOF-MOEA/D	48.36 | 28.57 58.69 | 65.62	94.02 | 85.71 97.82 | 95.31	—	99.45 | 100.00 100.00 | 98.43	18.47 | 12.50 15.21 | 35.93
ReMOEA/D	7.60 | 3.57 9.78 | 21.87	23.91 | 8.92 28.26 | 53.12	0.00 | 0.00 0.00 | 0.00	—	1.08 | 0.00 2.17 | 3.12
WOF-Randomised	73.91 | 60.71 81.52 | 84.37	98.91 | 96.42 100.00 | 100.00	57.60 | 66.07 57.60 | 32.81	98.36 | 100.00 96.73 | 95.31	—

Table 6.11: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations. WOF is compared with DLS-MOEA. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

102 | 42 60 | 0	WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	DLS-MOEA
WOF-SMPSO	—	63.72 | 33.33 85.00	43.13 | 23.80 56.66	74.50 | 52.38 90.00
WOF-NSGA-II	21.56 | 40.47 8.33	—	11.76 | 14.28 10.00	61.76 | 35.71 80.00
WOF-Randomised	41.17 | 61.90 26.66	68.62 | 52.38 80.00	—	70.58 | 42.85 90.00
DLS-MOEA	19.60 | 35.71 8.33	18.62 | 28.57 11.66	11.76 | 19.04 6.66	—

Table 6.12: Winning rates using the IGD indicator for different problem categories using 10,000,000 evaluations. WOF is compared with MOEA/DVA and LMEA. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

42 | 0 42 | 0	WOF-SMPSO	WOF-Randomised	LMEA	MOEA/DVA
WOF-SMPSO	—	38.09 | 38.09 |	66.66 | 66.66 |	47.61 | 47.61 |
WOF-Randomised	50.00 | 50.00 |	—	76.19 | 76.19 |	47.61 | 47.61 |
LMEA	26.19 | 26.19 |	19.04 | 19.04 |	—	16.66 | 16.66 |
MOEA/DVA	47.61 | 47.61 |	47.61 | 47.61 |	76.19 | 76.19 |	—

Table 6.13: Winning rates using the IGD indicator for different problem categories using $10,000,000$ evaluations. WOF is compared with MOEA/DVA, LMEA and S^3-CMA-ES. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

28 | 0 28 | 0	WOF-SMPSO	WOF-Randomised	LMEA	MOEA/DVA	S3-CMA-ES
WOF-SMPSO	—	42.85 42.85	82.14 82.14	64.28 64.28	82.14 82.14
WOF-Randomised	46.42 46.42	—	82.14 82.14	64.28 64.28	75.00 75.00
LMEA	17.85 17.85	14.28 14.28	—	21.42 21.42	53.57 53.57
MOEA/DVA	28.57 28.57	32.14 32.14	67.85 67.85	—	60.71 60.71
S3-CMA-ES	17.85 17.85	17.85 17.85	42.85 42.85	32.14 32.14	—

numbers in Tables 6.2 and 6.3 show that the randomised version of WOF is able to improve the performance even further. In different problems the 4 different algorithms which are used internally might perform differently. By using all of them with an expected share of $\frac{1}{4}$, there is a high chance that the respective best method will be able to produce some solutions which enable the rest of the algorithm to advance towards better solutions as well. Since the second half of the randomised WOF uses only the NSGA-III optimiser, a good diversity is also expected for the final solution sets. The winning rates of the 4 original algorithms against the randomised WOF are in general very low, with 2.71% for SMPSO and 3.8%, 12.5% and 13.04% for NSGA-II, NSGA-III and MOEA/D respectively. Note that these numbers even include the low-dimensional problems as well. If applied to large-scale problems only, randomised WOF outperforms all algorithms by far, winning for instance 100.0% of all large-scale problems against the original NSGA-II, and over 90% of instances against SMPSO, NSGA-III and MEOEA/D. It also outperfroms the other WOF versions in many cases, winning 81.52% of problems against the WOF-NSGA-II and 72.82% against WOF-NSGA-III. With around 35%, the WOF-SMPSO is the only algorithm against which the randomised WOF looses more often that it wins. All this shows that a combination of different algorithms inside one framework can be a promising direction for future large-scale algorithms.

Table 6.14: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations. WOF is compared with modified versions of MOEA/DVA, LMEA and S^3-CMA-ES. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

28 | 0 28 | 0	WOF-SMPSO	WOF-Randomised	groupInfLMEA	groupInfMOEA/DVA	groupInfS3-CMA-ES
WOF-SMPSO	—	46.42 46.42	96.42 96.42	96.42 96.42	96.42 96.42
WOF-Randomised	28.57 28.57	—	100.00 100.00	96.42 96.42	100.00 100.00
groupInfLMEA	3.57 3.57	0.00 0.00	—	67.85 67.85	42.85 42.85
groupInfMOEA/DVA	0.00 0.00	0.00 0.00	14.28 14.28	—	21.42 21.42
groupInfS3-CMA-ES	3.57 3.57	0.00 0.00	42.85 42.85	67.85 67.85	—

Table 6.15: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations. The proposed WOF, GLMO and LCSA are compared using NSGA-II. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

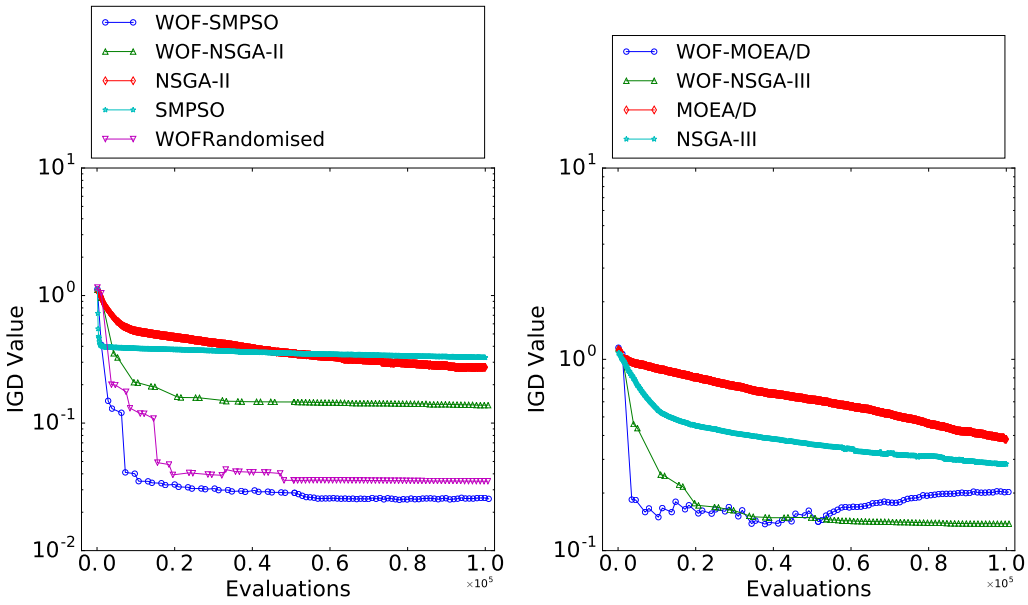
184 | 56 92 | 64	WOF-NSGA-II	xNSGA-II	GroupLinkNSGA-II
WOF-NSGA-II	—	47.82 | 17.85 66.30 | 42.18	39.13 | 48.21 38.04 | 37.50
xNSGA-II	16.30 | 21.42 14.13 | 20.31	—	33.69 | 50.00 23.91 | 46.87
GroupLinkNSGA-II	41.30 | 21.42 53.26 | 35.93	51.08 | 21.42 67.39 | 42.18	—

Table 6.16: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations. The proposed WOF, GLMO and LCSA are compared using NSGA-III. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-NSGA-III	xNSGA-III	GroupLinkNSGA-III
WOF-NSGA-III	—	45.10 | 17.85 61.95 | 28.12	37.50 | 41.07 38.04 | 37.50
xNSGA-III	33.15 | 46.42 27.17 | 51.56	—	36.41 | 48.21 29.34 | 48.43
GroupLinkNSGA-III	44.56 | 23.21 55.43 | 48.43	51.08 | 21.42 64.13 | 40.62	—

Table 6.17: Winning rates using the IGD indicator for different problem categories using 100,000 evaluations. The proposed WOF, GLMO and LCSA are compared using SMPSO. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-SMPSO	xSMPSO	GroupLinkSMPSO
WOF-SMPSO	—	60.32 | 67.85 61.95 | 28.12	65.76 | 58.92 66.30 | 73.43
xSMPSO	9.78 | 7.14 9.78 | 20.31	—	32.06 | 21.42 30.43 | 64.06
GroupLinkSMPSO	13.58 | 3.57 19.56 | 6.25	44.56 | 32.14 54.34 | 10.93	—

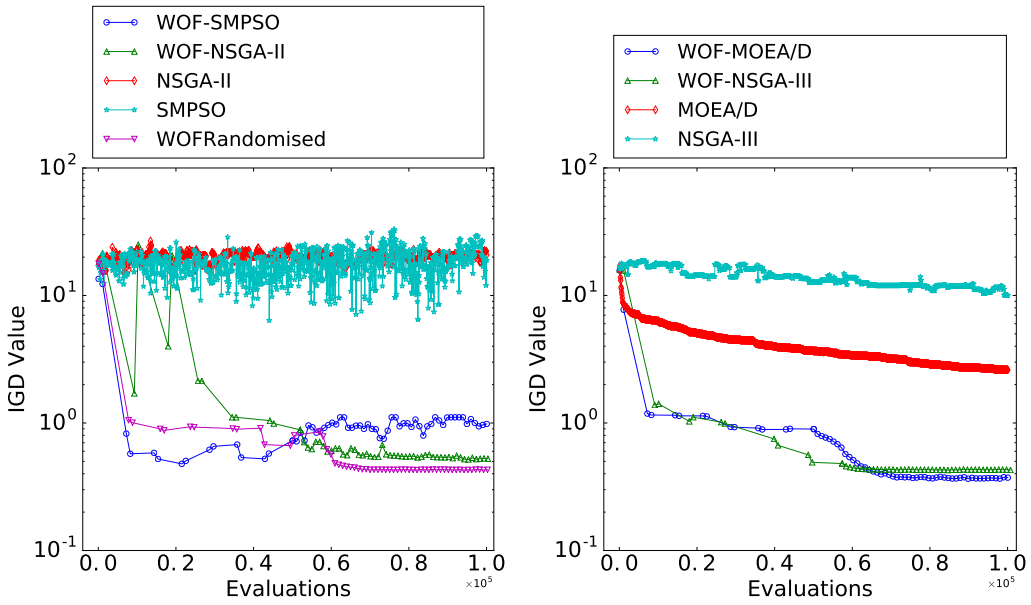


(a) SMPSO and NSGA-II (b) MOEA/D and NSGA-III

Figure 6.1: Convergence behaviour of the original and respective WOF algorithms on the 3-objective UF3 problem with 1000 variables.

Next, we take a brief look at the convergence behaviour to examine if the good performance is not only reflected in the final solution sets after the complete 100,000 function evaluations, but also during the search process. In Figs. 6.1 and 6.2 we show examples of the performance of the algorithms over the amount of function evaluations for the UF3 problem with 2 objectives and 1000 variables as well as LSMOP5 with 5 objectives and 1000 variables. We observe in these figures not only that all four WOF versions perform better than the respective original algorithm, we also see a rapid convergence towards low IGD values in the very beginning of the search process. This finding is consistent with the observations in [1, 6], where WOF shows a very fast convergence in all problem instances as well. The observed behaviour reflects the ability of WOF to converge to promising areas of the search space easily with the transformed, lower dimensional problem. This is especially useful in the beginning, while in later phases of the search the spread along the PF is more important and the transformed problems can not lead to large jumps in solution quality any more.

In general, this behaviour of the WOF algorithm is beneficial for the overall search, and desired behaviour that can lead to advantages when applied to practical applications. WOF is able to provide a reasonable solution quality after a very small amount of function evaluations, which might be helpful when resources are limited. From that point, it gradually improves, and the selection of multipe pivot solutions helps to keep diversity despite the fast convergence, to prevent the population from collapsing to a small area on the PF.



(a) SMPSO and NSGA-II

(b) MOEA/D and NSGA-III

Figure 6.2: Convergence behaviour of the original and respective WOF algorithms on the 5-objective LSMOP5 problem with 1000 variables.

6.3 Evaluation of the Grouped and Linked Mutation Operator

In this section, we evaluate the performance of the proposed GLMO and its associated methods as described above in Section 5.2. We compare the different version of the grouped and linked mutation operators inside of three well-known metaheuristic algorithms with their respective original version. For each of the three used algorithms (NSGA-II, NSGA-III and SMPSO) we implemented 5 versions which are compared with each other in Tables 6.4 to 6.6. The original version is compared with three versions that use only grouped mutation, only linked mutation or the combination in the grouped and linked mutation operator (GLMO). All algorithms use the polynomial mutation operator. The fifth algorithm uses the standard polynomial mutation but with a high mutation probability. This serves as a baseline to test whether the positive effect on the solution quality actually comes from the linkage or the groups, or if this is merely an effect from an increased amount of changed variables per iteration. Therefore, the mutation probability is set to produce the same expected amount of change in a solution as the grouped mutations. Since 4 evenly sized groups are used in all experiments, the mutation probability in this algorithm version is set to 0.25. All other settings of algorithm parameters and experiment specifications are identical to the descriptions above in Section 6.1.

As the focus lies on the effect of the proposed changes, we are not primarily interested in which of the underlying algorithms (NSGA-II, NSGA-III, SMPSO) performed best

or worst. As we want to show the effects of the proposed changes to the mutation operators, in the following we take a look at each of the three algorithms separately. We first concentrate on the final solution quality before taking a look at the convergence behaviour.

Regarding the NSGA-II and its derived versions, the results are shown in Table 6.4. The first observation we can draw from these winning-rates is that the linked algorithm does not perform better than the original NSGA-II in most instances, and only outperforms it in a little over 5% of the problems. This behaviour makes sense since both of these versions use a mutation probability of $1/n$, and therefore the expected amount of mutated genes in each individual is 1 out of the n variables. As a result, linking the amount of change between all mutated variables might in most cases not have any effect, since it only applies when more than 1 variables is changed in the first place.

The next observation regards the grouped NSGA-II version, especially in comparison with the high mutation probability version. Here we can see in Table 6.4 that the GroupedNSGA-II performs better than the original NSGA-II in about 30% of the cases, and better than the HighProbabilityNSGA-II in $53,8\%$ of all problems. Since NSGA-II also outperforms the grouped version in around 50% of problems, a clear superior performance between these two can not be observed when all problem instances are considered. However, when only looking at the large-scale problems, these results differ, and the GroupedNSGA-II and NSGA-II win against each other in 42% and 39% respectively, the remaining instances resulting in draws. If we compare with the version using a high mutation probability, we see that the grouped version performs significantly better on 60.86% of the large-scale problems, while on the other hand, the high probability NSGA-II can only outperfom the grouped NSGA-II in 7.6% of large-scale problems. This is especially interesting since the expected amount of change to each individual remains the same in both operators (1/4 of the n variables). However, there seems to be an influence in the choice of which of the variables are mutated. Since we used the ordered grouping mechanism in these experiments, the grouped version usually mutates solutions with relatively similar values (relative to their domains), which might in the current benchmark problems lead to groups that seem more beneficial. Without the detailed knowledge on why these ordered groups are favourable for each specific benchmark, we can nevertheless conclude that the choice of which variables are being mutated has in influence besides the mere increase of mutation compared to the normal, low mutation rate.

The most important observation concerns the version that uses both the groups and the links between the variables. In this GLMO operator, even though both effects on their own did not lead to a clearly superior performance over the original NSGA-II, we can observe that the "GroupLinkNSGA-II" in Table 6.4 can obtain significantly better results than the original NSGA-II in 91.30% of the 92 different large-scale instances and $73,43\%$ of the many-objective problems. This shows that the solution quality can be significantly

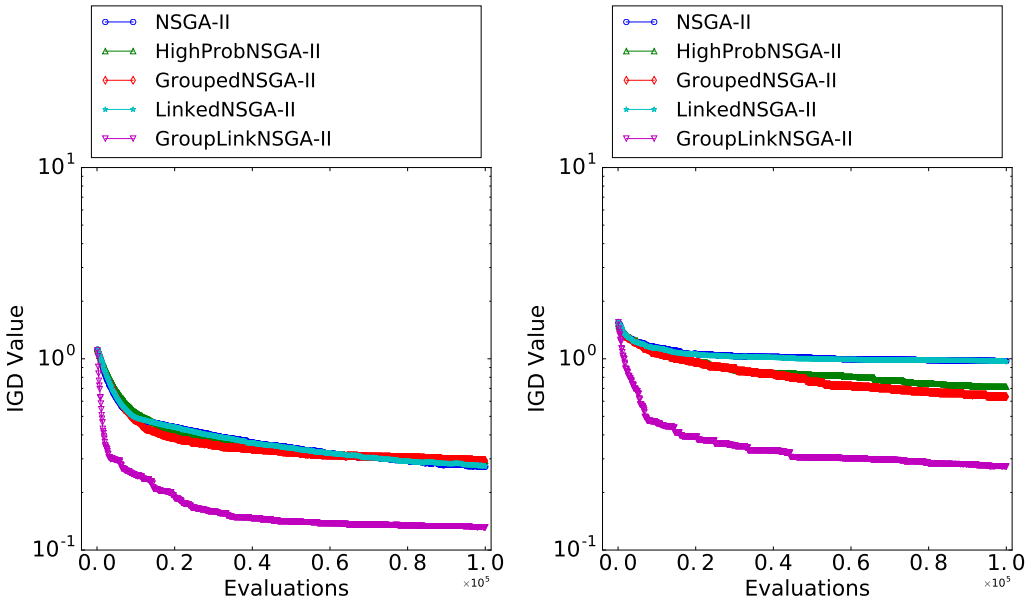
improved using the proposed GLMO mutation operator, and that variable groups and especially the link between the amount of mutation for the variables can enhance the effectiveness of traditional algorithms in the large-scale area.

When we look at the results of the NSGA-III and SMPSO in Tables 6.5 and 6.6, we observe a similar picture. The GroupLinkNSGA-III algorithm obtains better results than NSGA-III in over 90% of the large-scale problems, and over 73% of the many-objective problems. Although NSGA-III is already a dedicated many-objective algorithm, and we can naturally expect it to work well for the 64 many-objective instances, we still observe a large increase in performance in those many-objective instances when we apply the modified mutation operator, which confirms the observation that these operators might not just be useful for large-scale, but also for many-objective problems.

The same operator applied to SMPSO yields similar result. However, the winning rates compared to the original SMPSO are with 76% and 39% for large-scale and many-objective instances respectively a little lower than for the other optimisers. On the other hand, the original SMPSO can only win against the GroupLinkSMPSO version in less than 2% of the cases, suggesting that in more than 20% of the cases there is no statistical difference between the two algorithms.

Another interesting aspect is the convergence behaviour of the proposed methods. In Figs. 6.3 to 6.5 we show exemplarily the development of the IGD values for the 2-objective WFG5 and UF3 problems with 1000 variables for the algorithm versions using NSGA-II, SMPSO and NSGA-III respectively. In all three figures it is clearly observable that the GLMO which uses both groups and links between variables performs superiorly, not just with a statistically signifiant difference, but also by a large margin. In contrast, even though in general there exist such significant differences between the other algorithm versions as well, we see that in the median performance, the results of the other three versions are always very close to their respective original algorithm. What is also visible is that all algorithms have the phase of highest progression in the beginning of the search, while after the first 10% of the evaluations the IGD values decrease only slowly. The big difference between the GLMO and the other versions is that this progression in the beginning is much more significant, which may indicate a much better exploration of the high-dimensional search space in the beginning of the search compared to the other algorithms.

In conclusion, we can see that the modified mutation operators, although they require very little change to the operator, and no changes at all to the used metaheuristic, can significantly increase the performance of existing algorithms for large-scale and also for many-objective optimisation. Especially the combination of variable groups and variable linkage can lead to increased performance in over 90% of the 92 large-scale problems and over 70% in many-objective instances when the NSGA-II or NSGA-III algorithms are



(a) UF3 (b) WFG5

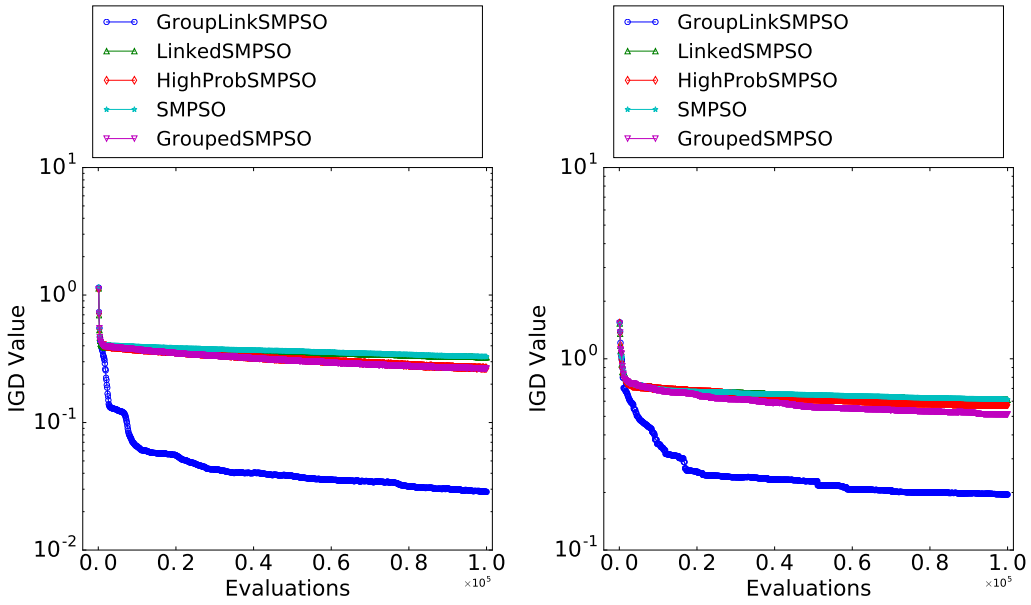
Figure 6.3: Convergence behaviour of the different NSGA-II versions using the grouped and linked mutation operators on the 2-objective UF3 and WFG5 problems with 1000 variables.

used. The convergence analysis further confirms a superior performance of the GLMO not only in solution quality, but also in the speed of convergence towards the Pareto-front.

6.4 Evaluation of the Linear Combination-based Search Algorithm

In this section, we take a look at the performance of the Linear Combination-based Search Algorithm (LCSA). Similar to the previous sections, we apply the LCSA method to three different optimisation algorithms: NSGA-II, NSGA-III and SMPSO. The experiment and algorithm settings are the same as described in Section 6.1. All 184 problem instances are used in this experiment. In total, 6 different algorithms are compared with each other, where the three versions using the LCSA are denoted with an "x" in front of their names in the following tables and figures, i.e. "xNSGA-II", "xNSGA-III" and "xSMPSO".

In general, the results in Table 6.7 indicate a superior performance of the LCSA-versions over their respective original algorithms for all three metaheuristics. The numbers show that for SMPSO, NSGA-II and NSGA-II respectively, the LCSA version performs better in 79%, 68% and 72% of the large-scale instances and in about 73%, 62% and 61% of instances from the many-objective area. We can also observe that this superior performance is not observed in such a magnitude when we look at the low-scale (i.e. 40-variable) problems in the upper right of each table cell.



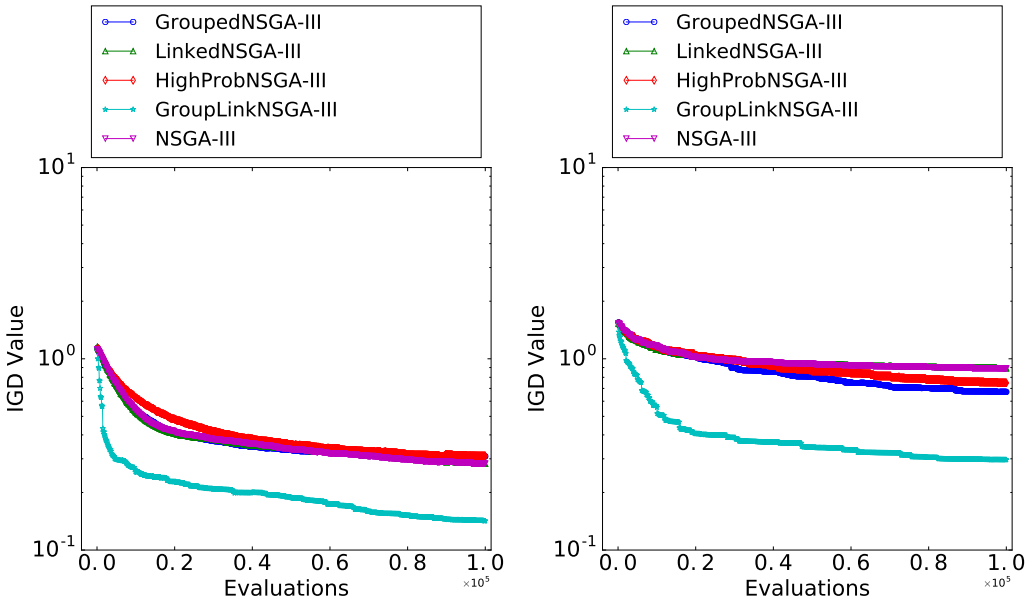
(a) UF3 (b) WFG5

Figure 6.4: Convergence behaviour of the different SMPSO versions using the grouped and linked mutation operators on the 2-objective UF3 and WFG5 problems with 1000 variables.

The best performance among all the 6 algorithm versions is often achieved by xSMPSO, NSGA-III or xNSGA-III (see Tables B.21 to B.24 in Appendix B). Especially in the large-scale problems, with lower numbers of objective functions, the xSMPSO works best, while with increased numbers of objective functions the xNSGA-III performs best among all versions. Best performances of the original NSGA-III algorithm are usually only acquired in the low-dimensional problem instances. These results show that also the third proposed large-scale approach is able to greatly enhance the performance of existing methods on large-scale optimisation and to a similar extent than the GLMO method.

To take look at the convergence behaviour, in Fig. 6.6, the IGD development for four different problem instances are shown. Fig. 6.6a shows the performance for the 2-objective DTLZ4 problem with 1000 variables. This problem is an example where the LCSA method actually leads to a reduced solution quality at the end of the optimisation for all three algorithms. The performance of NSGA-II, NSGA-III as well as both of the LCSA-enhanced versions of these shows very little difference in the early stages of the optimisation, until approximately $20,000$ function evaluations were used.

The linear combinations are not able to identify better solutions in this DTLZ4 problem, which is clearly visible in the repeated "straight" phases of IGD development that are interrupted by the normal optimisation, during which the IGD values shrink. This could be a result of the properties of this specific problem. According to an analysis done in



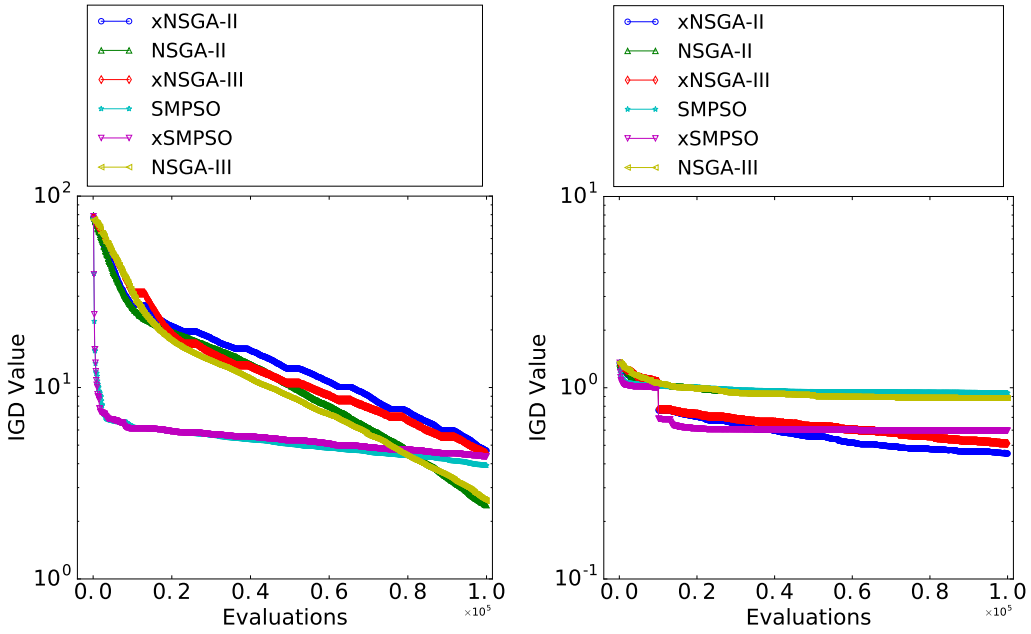
(a) UF3 (b) WFG5

Figure 6.5: Convergence behaviour of the different NSGA-III versions using the grouped and linked mutation operators on the 2-objective UF3 and WFG5 problems with 1000 variables.

[42], the DTLZ4 problem has a concave PF and is separable and unimodal. This means that it can in fact be solvable with little difficulties by traditional methods, as it does not possess local optima and can be solved without taking variable interactions into account. However, exactly for this reason, the LCSA method might have difficulties with it, as the phases of linear-combining solutions alter all variables at the same time, which may not hurt the solution quality, but on the other hand, neither advance the IGD values as much as a normal optimisation could with the same computational budget. For this reason, the original version of the algorithms might perform better on this type of problem.

In contrast, this claim is not supported by the results shown in Fig. 6.6b, where the convergence for the WFG7 problem are shown. WFG7 has the same properties as DTLZ4, with a concave PF, separability and unimodality. Nonetheless, we observe that xNSGA-II and xNSGA-III perform better on this problem than their respective original versions. This indicates that the low effectiveness of LCSA on the DTLZ4 problem might result from other properties, or the specific problem structure of the DTLZ benchmark family.

Figs. 6.6c and 6.6d show the IGD developments for the 3-objective WFG6 and 5-objective LSMOP9 benchmarks. In these cases we see that the LCSA performs in all three algorithms superiorly to the original versions on the large-scale instances with higher numbers of objective functions, which is in line with the high winning rates on the many-objective instances described above.



(a) 2-objective DTLZ4

(b) 2-objective WFG7

(c) 3-objective WFG6

(d) 5-objective LSMOP9

Figure 6.6: Convergence behaviour of the LCSA-based algorithms (denoted with an "x") and the respective original versions. All problems have 1000 decision variables.

6.5 Comparison of the Proposed Methods

After evaluating each of the three proposed large-scale methods with their respective original algorithm version, we will now take a look at how these methods compare to each other. We have seen in the previous three sections, that WOF, GLMO and LCSA are all able to significantly improve the performance of existing methods on large-scale and partly many-objective problems. All of them can be used with arbitrary metaheuristics, and the positive effects have been shown for different PSO and EA methods, which indicates that they are flexible and can be used with algorithms for specific applications in the future as well.

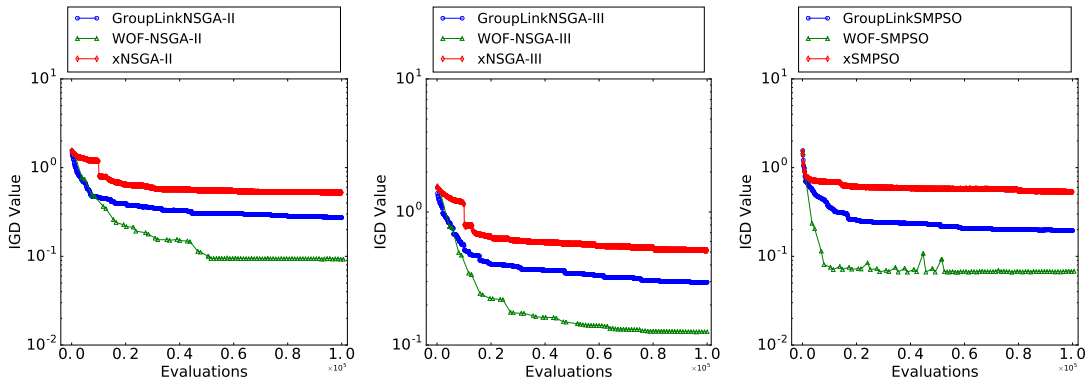
To compare them now with each other and evaluate possible differences between them, we compare the different methods using the SMPSO, NSGA-II and NSGA-III versions respectively with each other. In Table 6.15 we show the winning rates when comparing WOF-NSGA-II, xNSGA-II and GroupLinkNSGA-II with each other. The same comparison is shown when all methods use SMPSO or NSGA-III respectively in Tables 6.16 and 6.17.

First we look at the NSGA-II performances of the three proposed techniques. The winning rates are, in general, mixed, with WOF winning against LCSA in over 66% and against GLMO in over 38% of large-scale instances. It is, however, outperformed by GLMO in over 53% of the 92 large-scale problems. The LCSA seems, at least in its NSGA-II version, the weakest among the three, with WOF and GLMO on par in the different categories, and in general a high number of non-significant differences between the algorithms.

Looking at the NSGA-III versions in Table 6.16, we see a similar picture, which is not surprising since both algorithms use common elements. When comparing these numbers, it seems that the GLMO is actually the slightly better choice compared to WOF in the large-scale instances and the many-objective instances, but both algorithms show significantly better performance over each other in over 37% of instances in all categories of problems. We can see that the GLMO seems to have slightly higher winning rates in the large-scale instances, while the WOF is more useful in the 40-variable instances, indicating a better robustness to changing numbers of variables.

This impression for both of the NSGA-based algorithms is not the same when the SMPSO is used as the optimiser. These results are shown in Table 6.17. In these cases, WOF-SMPSO outperforms both of the other proposed techniques in about 66% and 62% respectively on the large-scale instances, and the GLMO technique in over 73% of the many-objective problems. Interestingly, using SMPSO, WOF and LCSA perform very similarly on the many-objective problems, with over 50% of problems resulting in no statistical difference at all. Especially in the small-scale problems, WOF outperforms the other two methods by far, which is consistent with the results from NSGA-III.

Taking a look at the convergence speed of the three large-scale methods, we show in Fig. 6.7 the 2-objective WFG5 problems with 1000 variables for the three techniques using each of the three optimisers in the subfigures. For this specific problem, the WOF method is the best with all 3 algorithms, and further shows the fastest convergence among the three proposed large-scale techniques.



(a) NSGA-II variants (b) NSGA-III variants (c) SMPSO variants

Figure 6.7: Convergence behaviour of the three proposed large-scale methods using the NSGA-II, NSGA-III and SMPSO algorithms on the WFG5 benchmark with 2 objective functions and 1000 decision variables.

In conclusion, the results indicate that WOF is the best large-scale strategy among the three proposed ones when the SMPSO is used as the optimisation strategy. Using NSGA-based variants, GLMO and WOF perform on par, with GLMO having slightly higher winning rates on the large-scale instances. Based on the results of this comparison, we identify the WOF as the most efficient of the three proposed techniques. Due to this fact and the large amount of data and limited space in this work, in the following experiments different versions of WOF are being used in comparison with the related works from the literature.

6.6 Comparison with Related Large-scale Approaches

After the evaluation of each of the proposed methods on their own, this section compares the performance of the proposed methods with the state-of-the-art in large scale optimisation.

We showed in the previous three sections that the WOF, GLMO and LCSA are all able to greatly improve the performance of different multi- and many-objective metaheuristics. As described in the beginning of the chapter, we also reached out to all the authors of the related state-of-the-art in large-scale optimisation, which were described in Chapter 3. The obtained algorithms and their configuration were described above in Section 6.1.

In this section, we perform the experiments in two different parts in the following two Sections 6.6.1 and 6.6.2. First, we compare only those algorithms which can be compared on a relatively low computational budget in Section 6.6.1, i.e. algorithms which do not require expensive grouping methods. These experiments use a computational budget of $100,000$ function evaluations as in the experiments in the previous three sections. Due to the fact that some of the algorithms (MOEA/DVA, LMEA and S^3-CMA-ES) rely on interaction-based groups, these require a large computational budget to run. These larger experiments, which use $10,000,000$ function evaluations, are the subject of Section 6.6.2. However, in order to evaluate whether their respective search strategies can work efficiently even without the interaction-based groups, we also use these three algorithms with random grouping strategies in the low-budget experiments.

The algorithms compared in this sections are as follows.

- Three of the previously introduced and compared WOF versions: WOF-NSGA-II, WOF-SMPSO and WOF-Randomised
- Two different versions of the LSMOF algorithm: LS-NSGA-II and LS-SMPSO
- Three versions of MOEA/DVA, LMEA and S^3-CMA-ES, which use contribution-based groups but have their interaction-based grouping methods replaced with random groups. These are called randomMOEA/DVA, randomLMEA and randomS^3-CMA-ES.
- The DLS-MOEA algorithm from the literature.
- Two version of the ReMO algorithm which use NSGA-II and MOEA/D respectively as the optimiser, called ReNSGA-II and ReMOEA/D.
- The original versions of MOEA/DVA, LMEA and S^3-CMA-ES.

For most of the experiments, especially the ones using $100,000$ function evaluations (low-budget experiments), all 184 benchmark functions are used as described in Section 6.1. The exception from this is only the comparison with DLS-MOEA, which is further explained below.

Due to various reasons, the amount of used benchmark functions in the $10,000,000$ evaluation experiments (large-budget experiments) also differ in certain cases from the common setting. To be precise, certain benchmark functions were excluded from some of the experiments due to their extremely large, infeasible computational overhead. Some additional insights on these decisions are elaborated below and in the respective subsections. The details on the used benchmarks for the experiments are as follows.

- The low-budget comparison between the WOF methods and the LSMOF versions, the ReMO versions and the random-grouping-based versions of MOEA/DVA, LMEA

and S^3-CMA-ES are done using all of the described 184 problem instances in the same way as the previous subsections.

- The low-budget comparison between the WOF versions and DLS-MOEA are done using 102 problem instances. The LSMOP problems are used with $n = 1000$ variables, and the instances with 200 to 500 variables are not used. Further, the many-objective problems are excluded from these experiments. DLS-MOEA, as was theoretically analysed in previous chapters (see Chapters 3 and 4), needs to compute the Hypervolume of a solution set multiple times in each iteration of the algorithm. Using this method on many-objective problems would result in expected calculation times of multiple months and is not feasible within the time frame of this thesis, nor would such a long calculation time be of practical use for real-world applications, even if the results should be satisfactory. The DTLZ1-7, WFG1-9 and UF1-10 problems are used in these experiments in the same way as in the previous sections with the combinations of $n = 40$ and $n = 1000$ variables and $m = 2$ and $m = 3$ objectives, resulting in 102 problem instances overall, 42 of them low-scale and 60 of them large-scale.

- The large-budget comparison between WOF-SMPSO, WOF-Randomised, LMEA and MOEA/DVA were done only on large-scale problem instances, as they are the main focus of the evaluation. The low-scale (40-variable) problems, as well as the many-objective problems, were not used in these cases. The WFG problems were not used in these experiments due to exceptionally long computation time of the function evaluations, especially in the problems WFG6 and WFG9. The results of these experiments are listed and analysed below in Table 6.12, and are shown in the appendix of the thesis in further detail. In total, 42 different large-scale problem instances were used which include the LSMOP1-9, DTLZ1-7 and UF1-10 problems, all of them used with $n = 1000$ decision variables and with $m = 2$ and $m = 3$ objective functions.

- Finally, the large-budget comparison with the S^3-CMA-ES algorithm is performed only on the DTLZ1-7 as well as the LSMOP2 and LSMOP4-9 benchmarks, all of them with 1000 decision variables and with 2 and 3 objectives each. The reason to exclude the low-scale problems and the WFG problems lies again in the computational budget as described above. However, the exclusion of the UF problems and the LSMOP1 and LSMOP3 is attributed to a major weakness of the S^3-CMA-ES method when dealing with these problems, which is explained in detail below in the analysis. As a result, S^3-CMA-ES is compared on 28 large-scale problem instances.

In the following, the results of these experiments are described and analysed.

6.6.1 Results and Analysis: Small Budget

This section deals with the analysis of the experiments using 100,000 function evaluations. The results of these comparisons are shown in Tables 6.8 to 6.11, and in further detail in Appendices B and C.

Comparison with LSMOF

First, we pay attention to the comparison between WOF and LSMOF. This pair of algorithms is of special interest as the LSMOF is strongly based on the WOF and uses similar concepts, that were first introduced in the original publication of WOF. Both algorithms belong to the transformation-based category in terms of dimensionality reduction and make use of transformation functions as introduced in Sections 5.1.1 and 5.1.5. As described in Section 3.2, LSMOF does not use variable groups and does not alternate between different optimisation steps in contrast to WOF. In Table 6.8 we see the resulting winning rates of the different versions of both methods. To enable a fair comparison, the same optimisation methods were implemented in both frameworks, i.e. SMPSO and NSGA-II. In the results, a striking observation is that WOF seems to benefit more from a combination with the SMPSO algorithm, while the LSMOF can obtain higher winning rates when both frameworks utilise the NSGA-II algorithm. For further comparison, the randomised version of WOF, which chooses the metaheuristic randomly, is also applied in these experiments. The findings are outlined in further detail in the following.

WOF-SMPSO performs significantly better than LS-SMPSO in 61.41% of all problem instances, while LS-SMPSO is only able to outperform WOF-SMPSO in 9.23% of all 184 problems. If we look closer into the different categories of problems, we see that this difference is even larger on the low-scale problems, where WOF-SMPSO wins in 66.07% while LS-SMPSO can only win in 1.78% of the cases (which corresponds to exactly one instance out of the 56 low-scale problems). In the many-objective category, the performances lie closer together, with WOF winning in around 34% and LSMOF winning in around 21% of the cases. Especially in the many-objective cases, these numbers together only sum up to a little over 50%, suggesting that in almost half of the many-objective cases both of these frameworks perform evenly, i.e. without significant differences in their final solution quality.

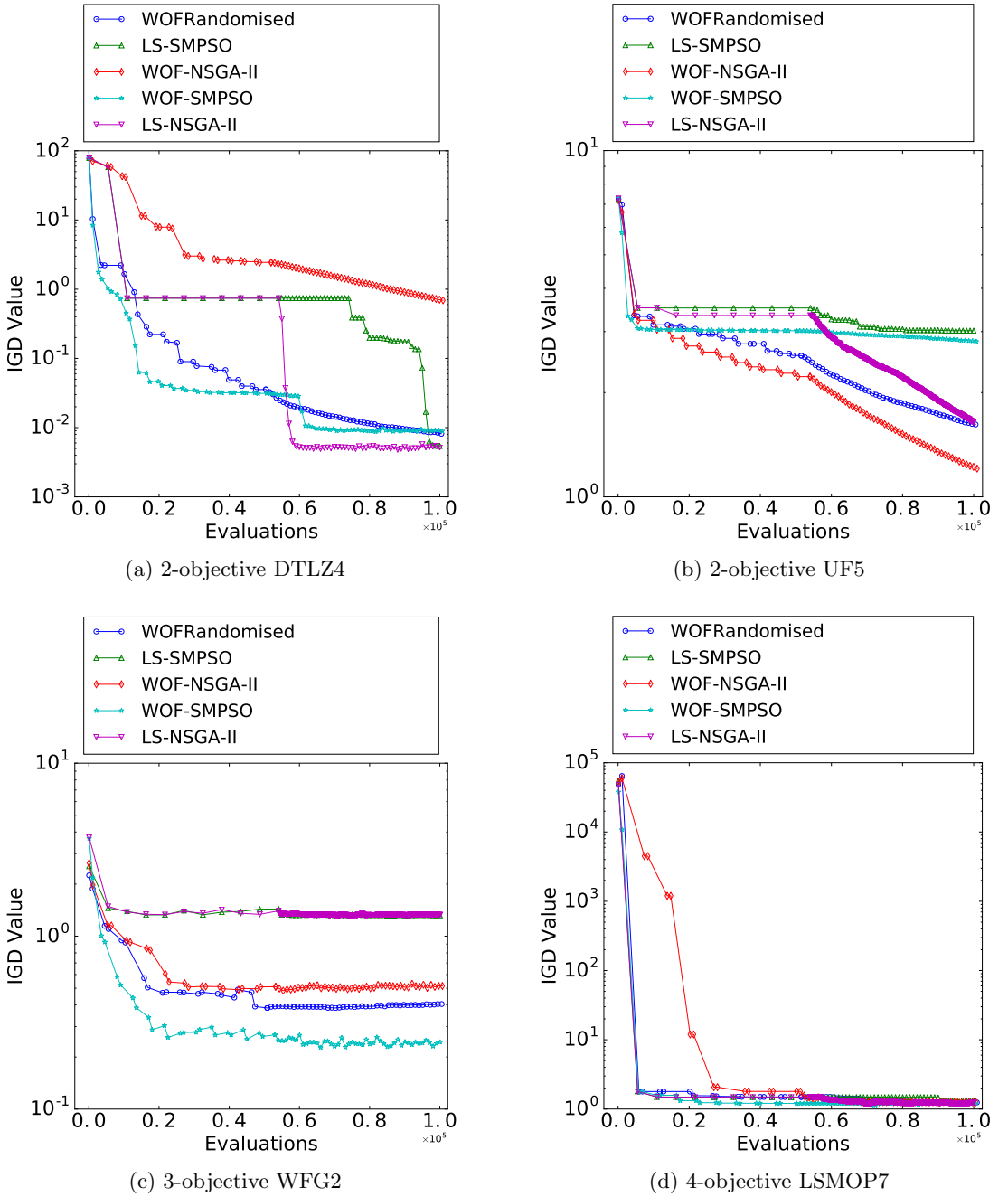
These results already indicate that WOF seems to be more robust in terms of scalability of the search space, as it can almost every time outperform LSMOF on low-scale problems. This is further supported by the most important category of benchmarks, which are the 92 large-scale instances. Out of these, WOF-SMPSO outperforms LS-SMPSO significantly in 60.86% of instances (56 out of 92), while LS-SMPSO can only show superior performance in 13.04% (12 out of 92) problems. The remaining 24 instances result in a draw, which is not surprising considering their related underlying transformation mechanics. Based on the numbers we can conclude for the SMPSO that even though LSMOF is the newer

algorithm, it is not able to show convincing performance compared with WOF, and is outperformed by WOF on most of the problems in both low-scale and large-scale area.

Next, we look at the comparison between the NSGA-II variants in both of the frameworks. Surprisingly, the results here look less clear, and the performance of both methods lies closer together. While the results in the many-objective category are almost identical to the SMPSO results, overall WOF outperforms LSMOF in 37.5% of instances while LSMOF wins in 29.89% of cases. This still leaves a little over 20% of 184 problems where both methods perform equally according to our statistical test. In the low-scale problems, both methods also show a similar strength over each other, with winning rates of little over 30%. Although these numbers indicate that none of the algorithms is really superior on the low-scale instances, it also suggests that, depending on the problem, one algorithm might perform better than the other in almost 60% of the cases. This highlights again the need to choose appropriate metaheuristics in real applications, and that the right choice of the metaheuristic can depend on the problem's characteristics. On the most important large-scale problem instances, WOF-NSGA-II shows slightly higher winning rates with 41.3% over 34.78% of instances. This indicates that WOF is, for large-scale problems, still the better choice when NSGA-II is preferred as the optimisation strategy, although, as said earlier, the superiority of WOF is much stronger when SMPSO is applied as the optimiser.

This leads to another important observation, which regards the overall best performance. Even though the winning rates can help to see how two algorithms compare directly with each other, the tables in Appendix B show that the overall best performance of all five algorithms is most often achieved by one of the WOF versions, usually by the WOF-SMPSO or the randomised WOF. This is also visible in the last row of Table 6.8, which shows very high winning rates of the randomised WOF in comparison with the LSMOF as also with the other WOF versions. However, this comparison is of course only partly fair, since this version makes use of MOEA/D and NSGA-III as well. Based on these findings, it is suggested that while LSMOF might be a valuable alternative in NSGA-II-based large-scale optimisation, the performance of WOF overall, when more successful optimisers like the SMPSO are used, is clearly superior to those of LSMOF.

It is noted that these results only partly match the reported findings from the original LSMOF article. The parameters used in the present thesis are the same as suggested in the original WOF publication [1], which were obtained by a sensitivity analysis of all of WOF's parameters. The experiments in the LSMOF study [69] have used different parameter settings for WOF, and mostly used NSGA-II in their experiments, while only limited experiments were reported for the SMPSO versions. This can be a reason for the mostly superior performance of the LSMOF method in [69], which was not observed in the same way in our experiments.



(a) 2-objective DTLZ4

(b) 2-objective UF5

(c) 3-objective WFG2

(d) 4-objective LSMOP7

Figure 6.8: Convergence behaviour of the LSMOF and WOF algorithms. All problems have 1000 decision variables.

Next, we take a look at some of the convergence behaviours of both methods. Four selected benchmark instances are shown in Fig. 6.8 from each of the four benchmark families with 2, 3 and 4 objectives. These four subfigures reveal interesting behaviour of the algorithms. First of all, it is clearly visible from in both methods that they change their behaviour after around half of the total function evaluations. This is the point where both techniques switch their behaviour from using transformation-based approaches to using the normal metaheuristic optimisation of the large-scale problem. Especially in Figs. 6.8a and 6.8b this change is clearly visible in the development of IGD values. The effect can be attributed to a sudden increase in diversity of the solution set, since the transformation-based steps are meant to converge quickly, and the second phase is meant to spread the good solution candidates further along the PF.

In Fig. 6.8a, the 2-objective DTLZ4 problem is shown, where the LSMOF algorithms obtain in the end of the optimisation a better solution quality than the WOF-based algorithms. In contrast, during the optimisation the performances are mixed. WOF-NSGA-II is almost from the beginning the worst performing of all 5 methods, and especially performs worse than the corresponding LS-NSGA-II. Overall, LS-NSGA-II and LS-SMPSO perform in the same way until the change occurs after the first half of the available evaluations. The good performance of these LSMOF methods is only achieved during this second phase. In case of LS-NSGA-II this happens shortly after the first half of evaluations, but in the LS-SMPSO, the final solution quality that outperforms the WOF versions is only achieved gradually and mostly during the last few generations of the search process. The WOF-SMPSO, on the other hand, achieves a relatively good IGD value from the start of the search and gradually improves on this even during the first half of the optimisation. During the first 50% of evaluations, the WOF-SMPSO (and also the randomised WOF) achieve the best IGD values, and keep improving after the change towards the normal optimisation. These results indicate that for a small amount of function evaluations, the WOF methods might be the better alternative, even the problem instances that LSMOF wins after the total amount of evaluations. On the other hand, if only $50,000$ evaluations had been used, the change of the algorithms' behaviour might have occurred earlier as well in the LSMOF methods. Nonetheless, it might be beneficial to use the WOF method over the LSMOF, since its search process can be stopped at earlier points in time and it produces acceptable solution quality throughout the whole process. This behaviour of fast convergence towards the beginning of the search might be desirable for many real-world applications. It also indicates that the alternation between normal and transformed optimisation is beneficial in the WOF algorithm. LSMOF optimises the transformed problems only in the first half of the search, which might, as in this case, result in a situation where the algorithm wastes many function evaluations without improvement, before the normal optimisation occurs. WOF alternates these phases and has an advantage of a more stable progress over time.

A similar behaviour is visible in Fig. 6.8b. First, we can observe that both PSO-based algorithms perform better than the NSGA-II-based ones, and that the WOF versions

perform better than the LSMOF versions in both cases. We also observe the sudden increase in performance after 50% of the used evaluations, which fits to the observations in [47], where the UF problems are identified as mainly diversity-oriented, meaning that the main challenge in these problems is diversity, not convergence. The decrease of the IGD values in this case fits to this observation, as the algorithms start to optimise diversity mostly after in the second half of the search.

In Fig. 6.8c we show the WFG2 problem, which has a disconnected PF. As a result, if an algorithm covers parts of the PF in earlier stages of the search, it might be difficult to find the other parts of it later on. Keeping this in mind, the convergence behaviour of the LSMOF methods in Fig. 6.8c make sense, because the LSMOF is designed to only optimise convergence in the first half. If by chance only certain solutions are found during this stage which belong to only a part of the PF, it might explain the overall bad performance in the WFG2 problem, and the inability to improve the IGD values throughout the rest of the search. In contrast, WOF has a diversity-balancing mechanism through its pivot solutions throughout the whole search, which enables it in this case to achieve better IGD values and to solve the WFG2 problem better. Finally, Fig. 6.8d shows an example from the many-objective area, where all algorithms perform more or less equally, and, except for the WOF-NSGA-II, also show the same convergence behaviour.

In summary, the comparison between WOF and LSMOF shows that both algorithms, although based on similar concepts, have various differences in their performances. If SMPSO is used as the internal optimiser, WOF is overall the superior method and outperforms LSMOF in most problem instances. If NSGA-II is used, the performance is more similar and both algorithms outperform each other on multiple occasions, and perform in general on par to each other. In the convergence analysis, WOF often achieves faster convergence towards good solutions in earlier stages of the optimisation. In addition, WOF seems more robust to different numbers of variables, as it performs better on low-dimensional problems when the SMPSO is used.

Comparison with ReMO

In this part we show the results and analysis when WOF is compared to the transformation-based ReMO method from the literature. The results in terms of winning rates are shown in Table 6.10 and in further detail in Appendices B and C. Like in the original publication in [78], we use ReMO with the two algorithms NSGA-II and MOEA/D, and the same two algorithms are used in the WOF framework to compare the performance. In Table 6.10, the WOF and ReMO versions of NSGA-II and MOEA/D are compared, along with the randomised version of WOF as in the previous experiments.

Overall, the results show that ReMO is not competitive to WOF, no matter which optimisation method is used. Similar results have been reported also in [47], where Re-NSGA-II was not able to perform on par with the DLS-MOEA. The results on the

NSGA-II versions show that WOF-NSGA-II significantly outperforms Re-NSGA-II in over 90% of all instances, in around 94% of low-scale and 90% of large-scale problems, and around 84% of many-objective problems. Re-NSGA-II can only perform better than WOF-NSGA-II in 7 out of 92 large-scale problems showing its inferiority compared to the Weighted Optimisation Framework. The results of the MOEA/D algorithms show an even clearer picture, with WOF-MOEA/D winning over Re-MOEA/D in 99.45% of all problems, among these 100.0% of all low-scale and 100.0% of all large-scale problems. Only in one out of 184 problem instances both algorithms perform equally, and in fact Re-MOEA/D never wins against WOF-MOEA/D even a single time, as seen in Table 6.10, row 4, column 3. ReMO is, based on these findings, the only algorithm in the study which is completely outperformed by another.

The random embedding approach of ReMO relies heavily on the fact that the random matrix, which is sampled in the beginning of the algorithm, results in a suitable transformation of the problem. In ReMO, this means a suitable embedding of the problem into the low-dimensional search space through the matrix is essential. However, this is only possible if there exists a useful embedding in the first place. In the original publication, ReMO was proposed and used purely to solve large-scale problems with low effective dimensions. As such, ReMO does not claim to solve any large-scale problems, but only those which include a large part of variables that do not or only weakly contribute to the objective function values. This assumption makes ReMO almost incapable of solving any problems which can not be embedded in a low-dimensional space without significant loss of the reachable search space. The used benchmarks in the large-scale literature and in this thesis require all variables to be part of the optimisation process, with most of them equally contributing to the objective function values. It is therefore not surprising that ReMO is outperformed by far, almost entirely, in the present study. Based on the results in this section, we conclude that ReMO can hardly be seen as a general large-scale optimisation alternative for future studies. However, it is the author's opinion that the random embedding approach might offer multiple valuable ways for extension and development of new transformation-based algorithms in future work.

Comparison with Randomised Algorithms

Now we take a closer look at how the random-group-based versions of MOEA/DVA, LMEA and S^3-CMA-ES perform against different versions of the Weighted Optimisation Framework. In Table 6.9 we list the winning rates of all six algorithms in comparison with each other, and the detailed IGD results are shown in Appendix B in Tables B.27, B.30, B.33 and B.36. The winning rates using the Hypervolume indicator are shown in Appendix C.

Since the three modified algorithms from the literature usually rely on interaction-based groups, it is expected that their performance is to some extent inferior to the results that are usually reported in the literature. All three methods, however, still use their original

contribution-based grouping mechanisms, and removing the interaction-based groups enables them to run of a small computational budget of $100,000$ function evaluations. In this way, we can examine in these experiments how well these three methods perform with their search strategies when only a limited computational budget or predefined variable groups are available.

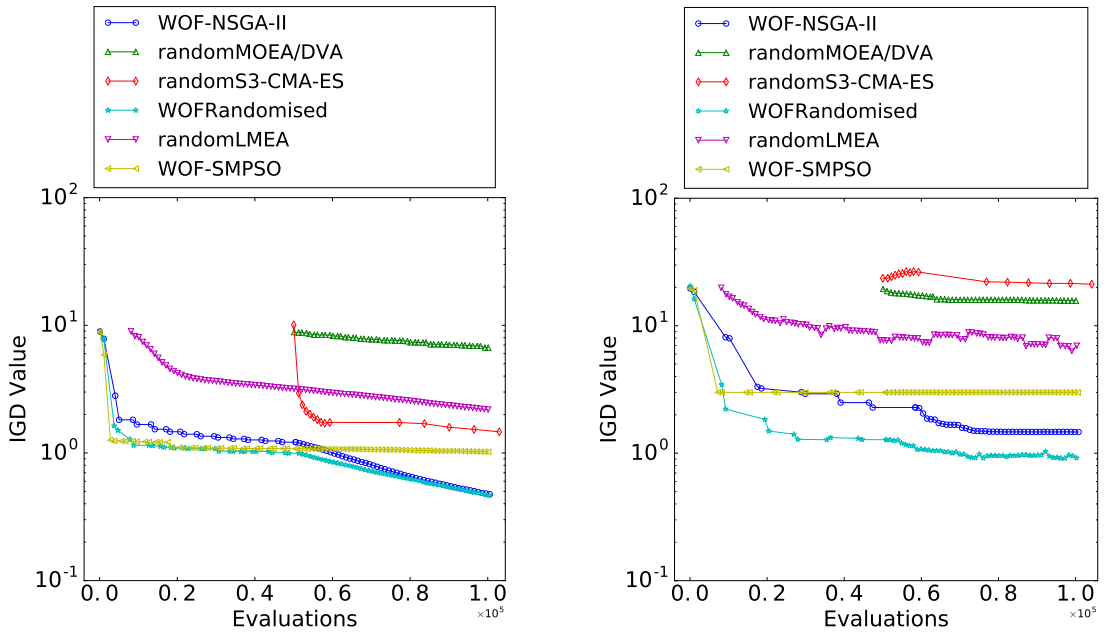
Comparing the three methods with each other first (the last three columns and last three rows of Table 6.9), it is visible that the search mechanism of LMEA works best with these random groups, and can outperform the other two on the entire set of benchmarks in around 71% (MOEA/DVA) and 75% (S^3-CMA-ES) respectively. On the large-scale instances, the picture remains the same, and the LMEA further obtains the highest winning rates (over 67%) on the many-objective instances and the low-scale problems. These results indicate that LMEA possesses a search mechanism that is not only more robust against non-optimal variable groups, but can also perform a suitable optimisation with its search mechanism alone. In contrast, the other two algorithms rely more on either the large amount of available evaluations or the quality of variable groups, as they do not perform well with random groups.

The more interesting observations stem from comparing the three WOF versions with the random-group versions of these related methods. Using the same computational budget, we observe that all of the three WOF versions outperform the LMEA, MOEA/DVA and S^3-CMA-ES algorithms by far. The randomised WOF version performs significantly better than all of the three related ones in over 95% of instances among the large-scale problems. Similar high numbers are also obtained in the many-objective problems and the overall benchmark set. Even though we saw earlier that the random-group LMEA obtained better results that the corresponding MOEA/DVA and S^3-CMA-ES methods, we can see that it is outperformed significantly in over 83% of large-scale problems by WOF-NSGA-II and over 90% by WOF-SMPSO and the randomised version of WOF. In general, the winning rates of all of the related algorithms remain very low compared to the WOF versions, often in the area of less than 10% overall as well as on the large-scale problems.

Looking at the convergence behaviour, further details can be identified. In Fig. 6.9, the convergence behaviour is shown for the 1000-variable UF6 problem with 2 objectives and the 1000-variable DTLZ7 problem with 5 objectives. The first observation is that the MOEA/DVA and S^3-CMA-ES only start their optimisation after they already used up half of the total evaluations. This is due to the contribution-based grouping methods, which consume $n \cdot NCA$ function evaluations (see Table 4.4). For this reason, both of these methods suffer from a reduced amount of resources for the actual optimisation, even though the interaction-based groups have been replaced. The same applies to the LMEA, which starts its optimisation after $8,000$ evaluations due to its clustering-based mechanism. Unfortunately, we can not observe that these found contribution-based groups have a big influence on the development of the IGD values. The desired effect

seems visible only for the S^3-CMA-ES in the UF6 problem, where the IGD is increased rapidly in the beginning of its optimisation. The other algorithms seem not to be able to reach similar convergence rates as the transformation-based WOF versions, which indicates that the CC-based approaches of the three related methods might be at a disadvantage.

In summary, these results show that the transformation-based search strategy seems to have a definite advantage in the case when only random or other simple variable groups are available. It also suggests that most of the good performance of MOEA/DVA, LMEA and S^3-CMA-ES comes not from their actual superior exploration mechanism in the search space, but either from the quality of the variable groups or the sheer amount of available evaluations that are usually used in the literature. To test these findings further, we show later in Section 6.6.2 the experiments using a large computational budget of $10,000,000$ function evaluations. In this way it is analysed whether the performance of WOF can still keep up in these scenarios where the three related algorithms can make use of their interaction-based variable groups.



(a) 2-objective UF6

(b) 5-objective DTLZ7

Figure 6.9: Convergence behaviour of the random-group-based algorithms from the literature and different WOF versions. All problems have 1000 decision variables.

Comparison with DLS-MOEA

The last analysis in the low-budget experiments concern the recent DLS-MOEA algorithm. It was only published recently in the end of the year 2018 and relies mostly on a local search using an archive and the Hypervolume indicator. In Table 6.11 we list the winning

rates of the DLS-MOEA in comparison with the three tested WOF versions. In addition, the appendix shows the detailed performance results in terms of IGD in Appendix B and the Hypervolume in Appendix C.

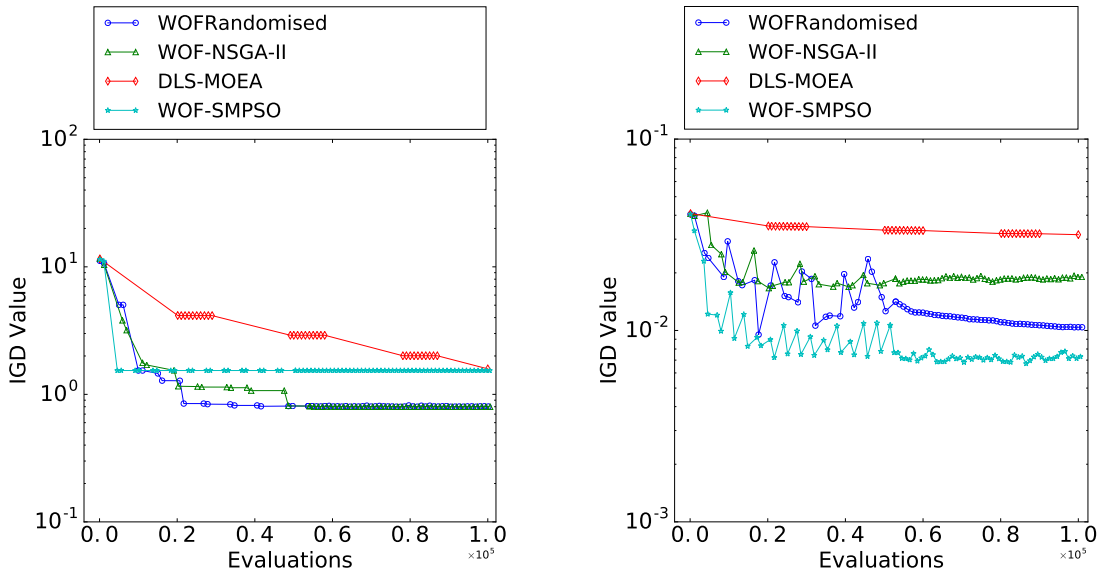
The results performed on 102 benchmark instances show in general a favourable performance of WOF compared to DLS-MOEA. DLS-MOEA performs significantly worse than WOF-SMPSO, WOF-NSGA-II and WOF-Randomised in 74.5%, 61.76% and 70.58% of problems instances respectively. In only the large-scale benchmarks, the WOF versions perform significantly better than DLS-MOEA in 90%, 80% and 90% respectively. If we take a look in the appendix at Tables B.40 to B.43, we see that among all 60 large-scale instances, DLS-MOEA only achieves the best performance among all 4 algorithms 2 times in the 3-objective LSMOP2 and LSMOP4 problems. Both of these problems are non-separable and multimodal according to [42]. Further, the results on other problems with the same characteristics like LSMOP6-8 show inferior performance of DLS-MOEA. It is therefore difficult to examine why these three problems favour the DLS-MOEA search strategy based on the properties of the problems. However, these results show that WOF methods in general outperform DLS-MOEA on 90% of cases on different benchmark families and with different numbers of variables.

As was described in the beginning of this section, the many-objective instances were not used in these experiments because the DLS-MOEA relies heavily on the calculation of Hypervolume in the objective space. This can be considered as a weakness of this approach, since it uses a performance indicator that, on the one hand, can measure diversity and convergence of solutions without the need of a reference set, but on the other hand is computationally expensive to calculate, especially when the number of objectives increases. Therefore, the computational budget needed by DLS-MOEA in terms of computation time is dependent on the number of objectives, which gives it a disadvantage for many-objective optimisation.

The reported results of DLS-MOEA in comparison with WOF in the original work [47] differ from the results obtained in our experiments. In the present thesis, a larger variety of benchmarks with different numbers of variables and objectives was used. However, if we focus our analysis on only those benchmarks used in [47], we also observe differences. More precisely, the original study claimed that DLS-MOEA works especially well on problems with certain properties, i.e. which require a great amount of care to obtain diversity, while convergence can be achieved relatively easily. The problems that were identified to lie in this category were the UF benchmark problems. Table B.41 shows the result of WOF methods and DLS-MOEA on the 40-variable and 1000-variable UF instances in our experiments. We can see that the best performance is acquired in the 40-variable problems by DLS-MOEA in UF2, 4, 5, 9 and 10, while in the other instances DLS-MOEA ties with one of the WOF versions or performs significantly worse. However, the situation which is of interest in this thesis are the large-scale 1000-variable tests in the second half of Table B.41. It is visible that DLS-MOEA performs significantly

worse than one of the WOF versions in all 10 benchmarks except UF7, where it ties with WOF-NSGA-II. This means that DLS-MOEA never actually achieves a best performance in any of the large-scale UF problems. This is a contrast to the result reported in [47], and can be explained with the different version of the WOF algorithm which was used in that study. In [47] the WOF-SMPSO was used in the version from the article published in [1], which chose the pivot solutions by Crowding Distance and employed the p-Value transformation function. The WOF version used in the present work, on the other hand, chooses the pivots by reference directions and uses the new parameter-free transformation function which was introduced in [6] and explained earlier in this thesis in Section 5.1.5. Furthermore, the experiments in [47] used a computational budget of $10,000,000$ evaluations.

In Fig. 6.10b we show the convergence behaviour for the 2-objective LSMOP2 and the 3-objective DTLZ7 problems. An interesting observation is that the IGD values of DLS-MOEA are not changing during the local search phase of the search. DLS-MOEA is alternating between two different stages of optimisation, and we can observe that one that uses the proposed dual local search has no real effect on the development of the IGD in these two problems. The advancement in IGD happens only during those phases where the Hypervolume-based optimisation takes place. This is not the case in all benchmarks, but it is an interesting observation which might be due to the multi-modality of those selected problems.



(a) 3-objective DTLZ7

(b) 2-objective LSMOP2

Figure 6.10: Convergence behaviour of the DLS-MOEA from the literature and different WOF versions. All problems have 1000 decision variables.

6.6.2 Results and Analysis: Large Budget

In this section the results of the large-budget experiments are analysed. These experiments used $10,000,000$ function evaluations and are conducted with WOF-SMPSO, WOF-Randomised, LMEA, MOEA/DVA and S^3-CMA-ES. The respective settings of the algorithms are the same as in Section 6.1.1. The results are summarised in Tables 6.12 and 6.13 and details are found in Appendices B and C. As described above, the amount of used benchmarks varies in these experiments due to the large computation time when applying all algorithms to a large set of different benchmarks. Therefore, only large-scale instances were used in these experiments, and further details regarding the exclusion of other benchmarks for algorithmic reasons in case of S^3-CMA-ES are given below.

Comparison with MOEA/DVA and LMEA

At first, we pay attention to the results in Table 6.12, where we see the winning rates of WOF-SMPSO and WOF-Randomised (the two best performing WOF versions in the previous experiments) compared to the original LMEA and MOEA/DVA on 42 different large-scale instances from the 2- and 3-objective LSMOP, UF and DTLZ families. In these experiments we obtain mixed results and, more importantly, different results than above when the same algorithms were used with random groups and small computational budgets.

Comparing MOEA/DVA and LMEA reveals that LMEA can only win in around 16% of problem instances while it is outperformed by MOEA/DVA in over 76% of cases. This picture opposes that of Table 6.9 and indicates that MOEA/DVA uses its large computational budget more efficiently than LMEA. More precisely, MOEA/DVA has a mechanisms of updating the population during the interaction-based grouping. If a solution produced throughout the interaction analysis can dominate a population member, the population is updated accordingly. Therefore, a large share of function evaluations (see Table 4.4) is used to actually contribute to the search process in contrast to LMEA, where the interaction-based analysis is carried out independently of the subsequent optimisation steps. This implementation detail may explain why MOEA/DVA can outperform LMEA in the experiments that include the interaction-based grouping mechanisms. This theory is explored further below in Section 6.7, where exactly this property is deactivated to examine the influence of variable groups on the search in more detail.

When taking the WOF algorithms into account, we observe that LMEA is not able to compare with the rest of the algorithms in their unity. Out of all 42 large-scale problems, LMEA performs significantly worse than one of the other three method on 40 benchmark instances, ties with the other algorithms on the 2-objective DTLZ7 and performs best only one time, on the 3-objective DTLZ7. The performance of MOEA/DVA compared to WOF reveals more mixed results, with WOF-SMPSO winning against MOEA/DVA in 20 instances, and also loosing in 20 instances. The same holds for the randomised WOF version. A closer look into the tables in the appendix (Tables B.37 to B.39) reveals that

results on the LSMOP are generally mixed, while MOEA/DVA seems to be the better option in almost all UF problems and WOF seems to be the better choice in almost all DTLZ problems.

Comparison with S^3-CMA-ES

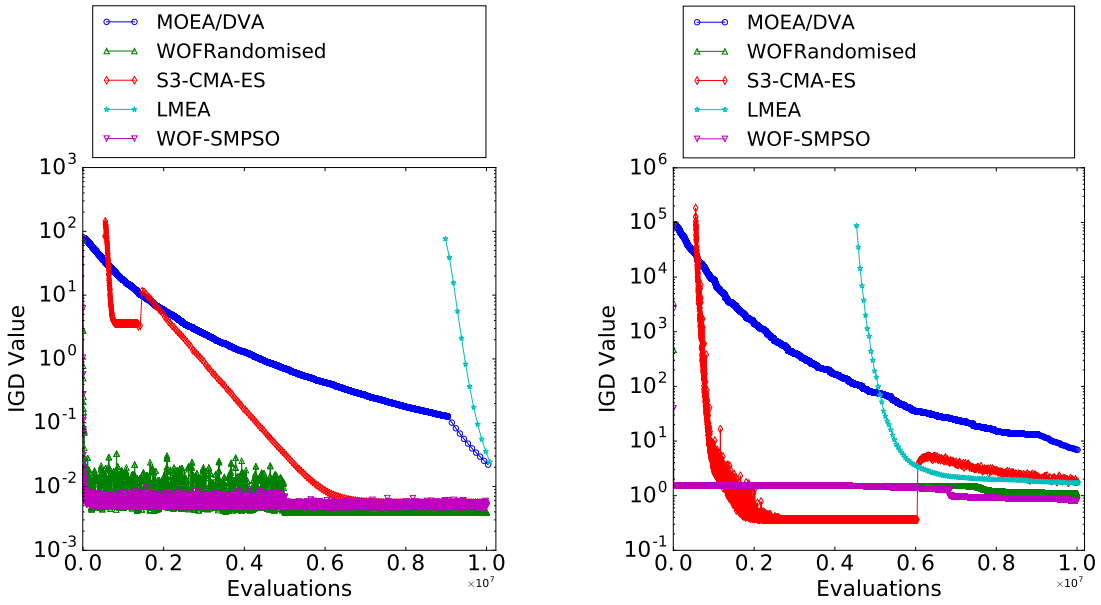
The last comparison with the state-of-the-art on a large computational budget involves the S^3-CMA-ES algorithm. This method is based on the concept of covariance matrix adaptation, and also utilises interaction-based variable groups. The results based on 28 large-scale problems are shown in Table 6.13 and Tables B.44, B.45 and C.12 in the appendices of this thesis.

The results, in comparison with MOEA/DVA, LMEA and two WOF versions, do reveal a rather poor performance of S^3-CMA-ES, which is significantly worse than the state-of-the-art in 82% and 75% of cases compared to the WOF-SMPSO and WOF-Randomised respectively. It is further outperformed by MOEA/DVA and LMEA in 60% and 53% of benchmarks respectively. While these numbers suggest that S^3-CMA-ES is at least comparable to LMEA in its performance (winning 42% against LMEA), it shows a worse performance especially in comparison with WOF. And while S^3-CMA-ES can win in 9 out of 28 instances (32%) against MOEA/DVA, it must be noted that these experiments did not include the UF benchmarks (reasons are explained below). It is therefore expected that if more benchmarks like UF were used, the percentage of wins of MOEA/DVA compared to S^3-CMA-ES would lie even higher, since we have seen in the above experiment that MOEA/DVA performs mostly superior to other algorithms on the UF benchmarks.

Next, we focus on a major weakness of the S^3-CMA-ES algorithm. While this method is built upon the (otherwise successful) concept of covariance matrix adaptation, a big problem actually lies in the size of the covariance matrices in the algorithm. S^3-CMA-ES uses interaction-based groups, and optimises each group separately using its own covariance matrix adaption instance to optimise only the variables in a specific group. Therefore, each of the matrices' dimensions are determined by the sizes of the variable groups. Since the dimensions of the matrices rise quadratically with the number of variables, a quadratic amount of memory is needed to store them. The issue, however, lies in the computation time, since inside of the algorithm matrix multiplications and inversions are calculated frequently. For an optimisation problem containing equal-sized groups, which do not contain too many variables each, this results in several moderate-sized matrices. However, when problems are non-separable, the interaction analysis produces (in the worst case) one large group of variables which includes the entirety of variables. In our experiments, this has been the case for all of the UF problems except UF3 and the LSMOP1 and LSMOP3 problems. In these cases, the algorithm operates with one large matrix of size 1000×1000 and as such uses a large computation time for multiplication and especially inversion of these. In our experiments, while S^3-CMA-ES

solved other problems with 1000 variables like LSMOP8 or DTLZ1 in around 14 minutes on a single core, for some of the UF functions a single run of the algorithm took more than 22 hours of calculation time. In case of the LSMOP benchmarks, not a single run had finished after more than 46 hours of computation time. Since this kind of runtime would make it impossible to test this algorithm for over 30 independent runs on these problems, the UF and LSMOP1 and LSMOP3 problems were therefore omitted in this experiment.

In conclusion, S^3-CMA-ES certainly has interesting properties, since it is easily adapted into a parallel algorithm (see Table 4.2). On the other hand, it is outperformed in most instances by WOF or MOEA/DVA, and it heavily depends on suitable variable groups of small to moderate sizes. If this algorithm was applied to real-world problems with non-separable variables, its long computation time, apart from the usual required time for the $10,000,000$ function evaluations, might render S^3-CMA-ES unsuitable in practise.



(a) 2-objective DTLZ2

(b) 2-objective LSMOP7

Figure 6.11: Convergence behaviour of LMEA, MOEA/DVA, S^3-CMA-ES and different WOF versions. All problems have 1000 decision variables.

Finally, we look into the convergence behaviour of LMEA, MOEA/DVA and S^3-CMA-ES in Fig. 6.11. Multiple interesting features are visible in these two plots of the 2-objective DTLZ2 and LSMOP7 problems. First of all, due to the contribution-based and interaction-based grouping phases, the MOEA/DVA, LMEA and S^3-CMA-ES start their optimisation procedures later than the two WOF versions. This is visible immediately in the IGD values of LMEA and S^3-CMA-ES since they do not produce IGD values before their groups are finished. Here we also see immediately the difference in the grouping mechanisms. S^3-CMA-ES uses a version of DG2, and therefore finishes the grouping phases after

around 500,000 evaluations. In contrast, LMEA and MOEA/DVA need multiple millions of evaluations in both benchmarks before they obtain the interaction-based groups. In MOEA/DVA, however, this is not immediately visible, since it saves the solutions created during the interaction analysis and constantly updates its initial population during the process. Especially in Fig. 6.11a it is visible that the IGD values gradually improve until the optimisation starts after around 9 million evaluations, and the IGD improves rapidly afterwards.

Another interesting observation is the sudden increase in IGD of the S^3-CMA-ES. This algorithm has two different stages, which are the independent optimisation of the populations until convergence and the diversity optimisation. It is visible that the IGD converges in the beginning and once no more improvement is possible, the algorithm detects that all populations have converged, starts the optimisation of the diversity-related variables, and creates new independent populations for the next iteration. However, this mechanism seems to increase the IGD values in both benchmark functions, and leads to an overall worse performance. This insight can be valuable to reconsider the way the diversity optimisation is carried out in S^3-CMA-ES, or how the preservation of good solutions in an archive is implemented in future versions of the S^3-CMA-ES.

6.7 Influence and Efficiency of Grouping Mechanisms

In the previous sections we have seen that some algorithms are based on costly interaction-based grouping methods, and are only able to show their full potential when used with large computational budgets (as seen in the random-grouped versions in Table 6.9). Therefore, the reason for the good performance is not clear, i.e. whether the good performance stems from the actual search mechanism of these algorithms or whether the good quality of the interaction-based groups is responsible for the successful optimisation result.

For instance, MOEA/DVA saves the created solutions during the interaction-based grouping to update the population. Therefore, it uses the large amount of function evaluations already partly to search better solutions. The resulting updated population is, when compared to the initial random population, very likely to provide a better starting point for the actual search. On the one hand, after comparing MOEA/DVA with another algorithm for a total of multiple million evaluations, the final performance can show which algorithm makes better use of the total budget. On the other hand, it is very difficult to decide whether the good performance of MOEA/DVA stems form (a) the special optimisation procedure after the grouping phases, (b) the updated start population or (c) the quality of the obtained groups. There is a chance that the good performance in the end is only the result of the algorithm's search process after the grouping phase, and would have lead to the same good results if random groups had been used. However, this can not be examined by just replacing different grouping mechanisms

with another, since the varying computational budget then allows the algorithm to use more evaluations for the optimisation.

Therefore, in this section we examine the performance of the three interaction-based methods from the previous experiments in further detail. The goal of this experiment is to find out which influence interaction-based groups have on the actual performance of the three most prominent algorithms using this concept. This analysis can especially be of interest considering that "good" variable groups in reality might be obtained through the inclusion of expert knowledge, and therefore it is of interest to know whether MOEA/DVA, LMEA and S^3-CMA-ES are competitive algorithms to use in these scenarios, even when only a limited computational budget is available for the actual optimisation phase of the problem.

Experimental Outline

To test for the influence of the interaction-based groups and to assess whether the search mechanisms alone are competitive, we use MOEA/DVA, LMEA and S^3-CMA-ES and implement special versions of them as follows. The normal version of each algorithm is modified so that the used function evaluations are not counted during the group-finding phase. Therefore, each of them performs their contribution-based and interaction-based grouping mechanisms, and starts the following optimisation phase as if no evaluations had been used so far.

In addition, no population update is done in the case of MOEA/DVA, so that after finding the interaction-based groups, all algorithms start with a random initial population. For comparison, the WOF-SMPSO and WOF-Randomised algorithms are used. In the previous sections, these two algorithms have already been compared with the random-group-based versions (see Section 6.6.1 and Table 6.9) and the original versions of LMEA, MOEA/DVA and S^3-CMA-ES on a large computational budget (see Section 6.6.2 and Table 6.12).

The algorithms perform the optimisation in these experiments with a budget of $100,000$ function evaluations. This means that the WOF versions use the standard parameters as described above with simple variable groups, while the three related methods use interaction-based groups, but all algorithms use the same budget for search process after the groups are formed.

We point out that this kind of experiment can usually not be seen as a fair comparison, as the versions with the interaction-based groups actually use millions of additional evaluations to obtain knowledge of the problem. It would therefore not be surprising to see that the interaction-based algorithms can outperform the WOF versions in this case. However, this experiment can reveal information on whether the high computational effort that is associated with interaction-based groups is actually justified and should be pursued in future large-scale algorithms. We can also obtain insight on whether the

actual search procedure of the three algorithms causes the good performance, and see how well these methods can perform on a fairly low computational budget.

The remaining settings and parameters of this experiment are as described in Section 6.1 and Section 6.6.2 respectively. Since the S^3-CMA-ES is involved in the experiments, the same 28 large-scale benchmark problems as in the previous section are used. The results are analysed in detail in the following and especially compared with the performances in the previous sections regarding the other version of the three related interaction-based methods.

Results and Analysis

The resulting winning rates of the five algorithms are shown in Table 6.14. Since the algorithms are altered to assess the influence of variable groups, the versions of the three related algorithms in Table 6.14 are called "groupInfMOEA/DVA", "groupInfLMEA" and "groupInfS3-CMA-ES".

Similar to the previous experiments, the WOF methods obtain high winning rates compared with the other three algorithms. WOF-SMPSO performs significantly better than LMEA, MOEA/DVA and S^3-CMA-ES in 27 out of 28 instances, and shows no statistical difference on the one remaining problem when compared with MOEA/DVA. Similarly, WOF-Randomised wins in 100% of all cases against LMEA and S^3-CMA-ES and in all but one instances against MOEA/DVA. None of the three related algorithms is able to win a single time against the randomised WOF version.

These winning rates of the WOF algorithms are higher than the ones obtained in the above experiment in Table 6.12. In those experiments all algorithms used $10,000,000$ evaluations, and the three related methods use up a major part of it for their interaction analysis. Nonetheless, they still used more than $1,000,000$ evaluations afterwards for the actual optimisation, which enabled them to obtain competitive IGD values for some benchmark functions. Compared to the present experiment, the difference lies only in the fact that once the contribution-based and interaction-based groups are obtained, the algorithms can only use $100,000$ evaluations for the optimisation. The numbers in Table 6.14 indicate that this amount is not sufficient to make use of the obtained variable groups. We can conclude that even when optimal groups (or groups of suitable quality) have been found, or are given from external sources, the search mechanisms of the LMEA, MOEA/DVA and S^3-CMA-ES can only obtain good results when used with a sufficiently large computational budget. If only limited budget is available, methods like WOF make better use of these computational resources.

Next, we take a look at the differences to the random group-based versions of the algorithms. In Table 6.9 we saw that LMEA in most cases performs superiorly to MOEA/DVA and S^3-CMA-ES on the large-scale problems when random groups are used, winning 72% and 80% respectively. Slightly different results are observed in Table 6.14,

although the winning rates of the groupInfLMEA over the other algorithms are still the highest with approximately 67% and 42% respectively. However, groupInfS^3-CMA-ES also wins against groupInfLMEA in 42% of the instances, which makes both algorithms equally strong. Surprisingly, groupInfS^3-CMA-ES also wins against the respective MOEA/DVA version in over 67% of the cases, which makes MOEA/DVA the weakest of the three algorithms in this experiment.

In summary, we see that the MOEA/DVA performs better in comparison with the other methods when a higher amount of function evaluations is available, which suggests that it reaches good performance mainly through its optimisation procedure rather than through the quality of groups alone. Overall, the results in this section reveal that the WOF algorithms can outperform the related methods on almost all the problem instances, even though the three methods from the literature used millions of additional function evaluations for their interaction analyses. Therefore, we can deduce that the high computational effort to obtain "good" variable groups does not contribute to the overall solution quality in most cases.

6.8 Summary and Discussion

In this section the results of the different experiments are summarised and discussed with respect to the different algorithm categories and properties.

Each of the three proposed approaches was evaluated on its own using a set of 184 different benchmark instances with different dimensionalities and properties. The results showed that each of the three methods is able to significantly increase the performance of existing traditional algorithms. This holds for the large-scale problem instances, but also in many cases for problems with lower numbers of variables and many-objective instances. In terms of complexity, the GLMO is the simplest approach, while the WOF requires the most change in algorithm behaviour. The performance of the methods, when compared to each other, reveals that the WOF method is generally the strongest among the three when the SMPSO is used as the internal optimiser, although the numbers suggest that the GLMO performs slightly better when NSGA-based algorithms are used.

In comparison with the state-of-the-art, the results show that the WOF-based algorithms perform superiorly in the majority of experiments. Using moderate computational budgets of $100,000$ function evaluations, WOF performs superiorly to ReMO in almost all problem instances, and achieves winning rates over 90% when compared with the DLS-MOEA. In comparison with the LSMOF, WOF performs superiorly using the SMPSO. Both methods are competitive using NSGA-II or NSGA-III, although the overall convergence speed of WOF is higher.

The three methods MOEA/DVA, LMEA and S^3-CMA-ES from the literature were tested in multiple different versions to test their abilities to work with random groups,

interaction-based groups and different computational budgets. As a result, we conclude that these methods do not perform well with random groups and with small computational budgets. On the other hand, their search mechanisms seem to benefit from higher budgets for the optimisation phases more than from the correct interaction-based segregation of variables. In summary, these three algorithms usually require a large computational budget to be effective, which can render them inferior when applied to costly real-world evaluation functions. In addition, their performance was in many cases inferior to WOF even after millions of additional evaluations.

Among these three algorithms, using the large computational budget, MOEA/DVA performs in general best among them, even though the interaction-based grouping method of S^3-CMA-ES requires fewer function evaluations and thus leaves more budget for optimisation. A critical weakness of S^3-CMA-ES was discovered in the course of the experiments which regards the computational effort of the algorithm for highly interacting problems, i.e. problems for which the grouping methods produce few very large groups. In these cases S^3-CMA-ES is unlikely to produce results in feasible time for real applications with expensive function evaluations.

Hypervolume Results

In Appendix C the winning scores of all experiments with regard to the Hypervolume indicator are shown. The used reference points used for the Hypervolume calculations are obtained using the respective nadir points of the benchmarks, multiplied with a factor of 2.0 in each dimension to account for Hypervolume contributions of extremal solutions of the obtained fronts. Since not all algorithms are able to obtain solutions close enough to the true Pareto-fronts in several problem instances, the Hypervolume values amount to zero in many problem instances. For this reason, the results using the IGD indicator are generally more reliable to judge the differences in algorithms' performances.

The results using the Hypervolume metric show that in most experiments the winning rates are lower than the corresponding IGD winning rates, resulting from an increased amount of ties between algorithms due to the zero-values. This effect influences all algorithms in the same way, and it is visible that despite the lower overall rates, the relations between algorithms stay the same. Therefore, the results overall match the ones from the IGD indicator, and confirm the strengths and weaknesses of the individual methods compared to each other in all experiments.

Analysis based on Algorithms' Categories

Finally, we perform a brief analysis which aims to identify different levels of performance not based on individual algorithms, but on the basis of the classifications from Chapter 4.

First, we take a look at the different categories of dimensionality reduction. Among the algorithms used in our evaluation, LMEA, MOEA/DVA and S^3-CMA-ES are CC-based

methods (category 1). The methods ReMO, LSMOF, WOF and LCSA are transformation-based methods (category 2), and DLS-MOEA and GLMO belong to category 3 and do not use dimensionality reduction. Between the representatives of these three categories, no clear lines can be drawn based on their performances, i.e. no class of dimensionality reduction seems superior to another.

Especially among the transformation-based approaches, LSMOF, WOF and LCSA perform very well, while we saw that ReMO is outperformed in almost all the benchmarks. We further saw that both WOF and LSMOF usually have a short phase of rapid convergence in the beginning of the search, though differences exist between both methods. We can deduce that on the one side, transformation-based approaches can be very powerful in terms of solution quality and convergence speed, but on the other side it is of importance that a suitable transformation is done, and that not any random transformation, as in the case of ReMO, is effective.

Looking at the CC-based approaches, we also obtain mixed results, with partly competitive, partly inferior performance compared to the transformation-based WOF. The CC-based approaches show differences in their performance, with MOEA/DVA being in general more competitive to WOF than the other algorithms on large-scale problems. The methods DLS-MOEA and GLMO, which do not reduce the dimensionality of the problem, show good performance in general, but while DLS-MOEA is mostly inferior to the WOF algorithm, GLMO is very competitive in its results. This is a surprising finding and shows that dimensionality reduction is not necessarily the best answer to large-scale optimisation.

Taking a look at the different categories of diversity management, LMEA, MOEA/DVA and S^3-CMA-ES belong to categories 1-1 and 1-2 respectively, which all make use of diversity-related variables through contribution-based grouping mechanisms. ReMO, GLMO and LCSA do not implement a special mechanism of diversity management, while DLS-MOEA and partly LSMOF use indicator-based approaches. Each of the three main categories has advantages and contains at least one method which performs reasonably well in the experiments. It is therefore not possible to attribute the performance to a specific type of diversity management.

In summary, our experiments do not reveal a clear superior strategy of dimensionality reduction or diversity management. While the LSMOF and WOF methods are two of the most promising algorithms, the GLMO and MOEA/DVA from the other two reduction categories show competitive performance as well. Moreover, ReMO was one of the weakest methods in the evaluations. We can conclude that, if used properly, all three categories of dimensionality reduction can be of advantage and should be taken into account for the future development of large-scale algorithms. However, the dependency of CC-based approaches on suitable variable groups and the reduction of their computational budget might be one of the biggest drawbacks of this category.

CHAPTER 7

Conclusions and Future Work

In this chapter we summarise the findings from this thesis and provide conclusions regarding the research objectives from Chapter 1. Finally, several promising topics for future research in the large-scale area are outlined.

The present thesis deals with the metaheuristic optimisation of large-scale multi-objective problems. The basic concepts from this research area were introduced in Chapter 2, including principles of multi-objective problems, evolutionary algorithms, variable groups based on interaction or contribution and cooperative coevolution. The following Chapter 3 presented the state-of-the-art from the literature in this research area, and outlined the concepts and properties of all 13 large-scale multi-objective algorithms published until the end of March 2019. Furthermore, an overview of popular grouping mechanisms was given.

Chapters 4 and 5 presented new contributions of this thesis. First, the existing large-scale algorithms and grouping mechanisms were analysed based on their components (building-blocks), and their properties. Based on the findings, classification schemes were proposed. The algorithms were analysed in detail based on their diversification, their dimensionality reduction and their capabilities for many-objective optimisation and possibilities for parallel implementations. In addition, the methods were analysed based on their required computational budgets, and previous experiments from the literature were analysed with respect to the used benchmark functions and dimensionality of the tested problems. Existing grouping methods from the literature were classified based on their computational budget and their segregation criteria into simple methods, contribution-based methods and interaction-based methods. The advantages and disadvantages, especially with respect to the necessary function evaluations to obtain the variable groups, were outlined.

In Chapter 5 the proposed approaches to solve large-scale optimisation problems were explained and analysed. The three methods, called GLMO, WOF and LCSA, were described together with their components in detail and their possible advantages and

shortcomings were discussed. The proposed algorithms were further classified based on the identified categories from the previous chapter.

Lastly, the experimental evaluation gave insights in the performance of the proposed methods and 6 of the related algorithms from the literature as well as different variations of them. Various experiments were performed with up to 184 different benchmark instances with different properties and dimensionalities. Since many of the algorithms are further configurable with arbitrary metaheuristics, four different metaheuristics (NSGA-II, NSGA-III, MOEA/D and SMPSO) were employed inside the large-scale methods. The results revealed a competitive and in many cases superior performance of the proposed methods compared to the state-of-the-art. The WOF was identified as the strongest of the three methods, and was able to significantly outperform most other algorithms with low computational budgets. In comparison with the state-of-the-art using interaction-based grouping methods, and using a large computational budget, the MOEA/DVA showed best performance among the related methods and was competitive to WOF in several cases. In addition to the final solution quality in terms of IGD and hypervolume indicators, the results were further compared based on convergence speed, revealing fast initial convergence of transformation-based algorithms. Generally, no clearly superior strategy could be identified among the classes of dimensionality reduction or diversity management. However, several weaknesses of single algorithms, related to the computational budget, the used matrix operations or the hypervolume indicator, were identified, which may make them unsuitable for real applications. It is the author's belief that a transformation of the problem is especially useful in early stages of optimisation, but CC-based methods can become more useful for parallel implementations and problems with many interactions between the variables.

Research Objectives

In conclusion of this thesis, we recapitulate on the research objectives stated in the introduction of this thesis and summarise the results for each of them.

Objective 1: Analysis and Classification of State-of-the-Art The first objective was to examine what the challenges of large search spaces are and which approaches have been developed in recent years for multi-objective large-scale optimisation. This objective was achieved first through a literature review, leading to a description of the existing large-scale algorithms and their components, called building-blocks. The methods and their building blocks were analysed based on various criteria and afterwards classified into categories. The further analysis of these categories and the experimental evaluations from the literature revealed that many common elements exist among large-scale algorithms, but also that experimental evaluations and computational resources often differ among publications. Several theoretical weaknesses and advantages of the existing methods were

also discussed. The classification scheme of the methods may be helpful in the future development of algorithms and to better understand the existing techniques.

Objective 2: Examination of Grouping Mechanisms The second objective concerned grouping mechanisms with the aim to examine how existing grouping methods for variables work and how important different groups are for the results of the optimisation with different algorithms. To achieve this goal, a selection of methods from the literature to obtain variable groups were described and analysed. As a result, a classification scheme for grouping methods was proposed. The methods were divided into simple methods, interaction-based and contribution-based methods, and the importance of interaction-based groups for the results of the optimisation, especially in CC-based algorithms, was evaluated experimentally. CC-based methods work better with interaction-based groups compared to random groups, but in exchange require a large computational budget to be carried out.

Objective 3: Proposal of new Algorithms The third objective of this thesis was to propose new methods to solve large-scale multi-objective problems and to improve the search abilities of current algorithms. Three distinct search methods were proposed in this thesis which make use of problem transformation and variable groups to achieve fast convergence and diversity when solving such large-scale problems. The three methods were formally described in detail and analysed theoretically. A classification based on the previously introduced categories was done for the proposed methods. The experimental evaluation showed in most cases superior behaviour both in terms of solution quality and convergence speed compared to the state-of-the-art.

Objective 4: Experimental Evaluation The fourth objective dealt with the experimental evaluation of the proposed methods in comparison with the state-of-the-art. Several experiments were conducted, comparing the proposed algorithms with classical methods and with each other. Several methods from the literature and the proposed WOF were then compared with each other using up to 184 different benchmark instances with varying dimensionality of search space and objective space. Many of the used algorithms were compared for the first time directly with each other and with a common parameter set. Further, different computational budgets were used and some related algorithms were used in altered versions to test for specific influences of grouping mechanisms. The findings show good performance of all three proposed methods, and that they are superior to many and competitive to some state-of-the-art algorithms. Aside from algorithms' performances, practical issues have been identified which include the dependency on indicators for high-dimensional objective spaces or the weakness of CMA-based approaches for large variable groups. As a result of the evaluation, several insights were gained which can be helpful for the future development of large-scale algorithms.

Future Work

Based on the findings of this thesis, we observe a development in the large-scale multi-objective area from the beginning in the year 2013 to modern techniques like the LSMOF or DLS-MOEA. The techniques used in this area have developed from simple CC-based algorithms which proved to be effective in single-objective optimisation to more advanced transformation-based methods working with lower computational budgets. It is also visible that the topic has received higher attention recently, with most articles published within the last 2 years. The future of the large-scale area still holds many challenges to face which can be the subjects of future work. Some possible issues that might be promising are listed in the following.

- One of the greatest challenges of large-scale optimisation is the issue of finding relevant groups for the decision variables with reasonable computational budgets. It is crucial for many of the current algorithms to obtain knowledge about variable interaction and contribution, and our experiments have shown that these can not be replaced by random groups in CC-based methods without loss of solution quality. While in real applications these might be obtained with the assistance of expert knowledge, it might be promising to develop methods which can obtain more of these variable properties during runtime or through surrogates to save function evaluation while at the same time improve the performance of current algorithms.

- Another topic of interest can be the further development of transformation-based approaches. We saw that a transformation of the problem can greatly enhance the search process, but should not be used without other optimisation techniques like optimising the original problem or using indicator-based methods. The difference in performance we observed between for instance LCSA and ReMO further shows that the type of transformation is important. Future work in this area can incorporate more information obtained from the search process like statistical measures of the current population into the transformation. Transformation functions can further be used more flexibly or only on parts of the variables, e.g. in combination with only certain variable groups.

- More attention can go towards suitable exploration techniques in high-dimensional spaces. Simple techniques like the GLMO reveal that changes in existing operators lead to more exploration and more change in the population's variables, which leads to better performance of the algorithms. It can therefore be promising to develop further specialised operators for selection, crossover or velocity updates in large-scale PSO techniques.

- In some algorithms a criterion is needed to determine when to stop using the transformation or CC-based optimisation and switch to the next phase of the algorithm, for instance to achieve diversity. This is done in WOF and LSMOF

through a fixed parameter. MOEA/DVA or S^3-CMA-ES aim to detect convergence during runtime. However, it is an open topic how to actually best detect convergence and how often this should be done during the optimisation. Suitable mechanisms can be developed in the future which enable large-scale methods to better balance their computational budget between different techniques of optimisation.

- An important factor is further the test of the existing methods on real-world applications. It is not yet explored how well the proposed methods or techniques from the literature perform when confronted with real optimisation problems. This relates to a general challenge in the field of multi-objective optimisation which usually relies on benchmark functions. The assumption that benchmarks represent the properties of real problems adequately is a topic of ongoing research in metaheuristic optimisation in general.

- Finally, it is the author's belief that great enhancement of large-scale multi- and many-objective algorithms can be achieved through hybrid approaches. We saw that transformation function can achieve fast convergence speeds, but may be more effective in the beginning of the search. We also observed that CC-based approaches or changed operators can be beneficial, and that PSO techniques and EA-based methods can show different performances. The randomised version of WOF achieved in general better performance than using just one metaheuristic as the internal optimiser. Future work might go towards detection of problem properties and applying suitable mechanisms automatically. Based on the measured convergence, diversity and overall performance of the search, different methods of dimensionality reduction might be applied either to the whole problem or to parts of it.

Using these and other future research directions, the development of large-scale algorithms can accelerate in the upcoming years. More research can produce algorithms which focus not only on large search spaces but also on many objective functions simultaneously. Hybrid techniques and dynamically changing components and operators can lead the way to future algorithms that outperform the current methods, including the ones proposed in this thesis. In this way, more elaborate algorithms can incorporate single-, multi- and many-objective optimisation with arbitrary numbers of variables. The author hopes that the insights and proposed methods from this thesis will be helpful for the further development of such methods on large-scale multi-objective optimisation. Overall, the author believes that the research area of metaheuristic optimisation will produce more advanced and wholesome techniques in the future which can aid to solve increasingly complex problems of modern society and technology.

Bibliography

[1] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "A framework for large-scale multi-objective optimization based on problem transformation," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 260–275, April 2018.

[2] M. Ueno, S. Usui, H. Tanaka, and A. Watanabe, "Technological overview of the next generation shinkansen high-speed train series N700," Central Japan Railway Company, Tokyo, Japan, 2008.

[3] L. M. Antonio and C. A. Coello Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," in *IEEE Congress on Evolutionary Computation (CEC)*, 2013, pp. 2758–2765.

[4] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "Weighted optimization framework for large-scale multi-objective optimization," in *ACM Genetic and Evolutionary Computation Conference (GECCO) Companion*. ACM, 2016, pp. 83–84.

[5] ——, "Mutation operators based on variable grouping for multi-objective large-scale optimization," in *IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec 2016, pp. 1–8.

[6] H. Zille and S. Mostaghim, "Comparison study of large-scale optimisation techniques on the LSMOP benchmark functions," in *IEEE Symposium Series on Computational Intelligence (SSCI)*, November 2017, pp. 1–8.

[7] ——, "Linear search mechanism for multi- and many-objective optimisation," in *Evolutionary Multi-Criterion Optimization*, K. Deb, E. Goodman, C. A. Coello Coello, K. Klamroth, K. Miettinen, S. Mostaghim, and P. Reed, Eds. Cham: Springer International Publishing, 2019, pp. 399–410.

[8] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, 2009.

[9] R. Datta and K. Deb, *Evolutionary constrained optimization*. Springer, 2014.

[10] H. Varian, *Intermediate Microeconomics: A Modern Approach: Ninth International Student Edition*. W.W. Norton, 2014.

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

[12] R. Kruse, C. Borgelt, C. Braune, S. Mostaghim, M. Steinbrecher, F. Klawonn, and C. Moewes, *Computational Intelligence: A Methodological Introduction*, ser. Texts in Computer Science. Springer London, 2016.

[13] M. Dorigo and T. Stützle, *Ant Colony Optimization*. Scituate, MA, USA: Bradford Company, 2004.

[14] A. Lewis, S. Mostaghim, and M. Randall, *Biologically-Inspired Optimisation Methods: Parallel Algorithms, Systems and Applications*, ser. Studies in Computational Intelligence. Springer Berlin Heidelberg, 2009.

[15] S. Mostaghim, J. Branke, and H. Schmeck, "Multi-objective particle swarm optimization on computer grids," in *ACM Genetic and Evolutionary Computation Conference (GECCO)*. New York, NY, USA: ACM, 2007, pp. 869–875.

[16] S. Mostaghim and J. Teich, "Strategies for finding good local guides in multi-objective particle swarm optimization (mopso)," in *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No.03EX706)*, April 2003, pp. 26–33.

[17] C. Darwin, *On The Origin of Species by Means of Natural Selection, or Preservation of Favoured Races in the Struggle for Life*. London: John Murray, 1859.

[18] H. Ishibuchi, M. Yamane, N. Akedo, and Y. Nojima, "Many-objective and many-variable test problems for visual examination of multiobjective search," in *IEEE Congress on Evolutionary Computation (CEC)*, 2013, pp. 1491–1498.

[19] R. C. Purshouse and P. J. Fleming, "Evolutionary many-objective optimisation: An exploratory analysis," in *IEEE Congress on Evolutionary Computation (CEC)*, vol. 3. IEEE, 2003, pp. 2066–2073.

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

[21] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 773–791, Oct 2016.

[22] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 5, pp. 694–716, Oct 2015.

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

[24] X. Ma, F. Liu, Y. Qi, X. Wang, L. Li, L. Jiao, M. Yin, and M. Gong, "A multiobjective evolutionary algorithm based on decision variable analyses for multiob-

jective optimization problems with large-scale variables," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 2, pp. 275–298, 2016.

[25] X. Zhang, Y. Tian, Y. Jin, and R. Cheng, "A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 97–112, Feb 2018.

[26] F. Sander, H. Zille, and S. Mostaghim, "Transfer strategies from single- to multi-objective grouping mechanisms," in *ACM Genetic and Evolutionary Computation Conference (GECCO)*. New York, NY, USA: ACM, 2018, pp. 729–736.

[27] F. Sander, "Variable grouping mechanisms and transfer strategies in multi-objective optimisation," Master's thesis, Otto von Guericke University Magdeburg, 2018.

[28] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.

[29] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *International Conference on Parallel Problem Solving from Nature*. Springer, 1994, pp. 249–257.

[30] K. Y. Yip, P. Patel, P. M. Kim, D. M. Engelman, D. McDermott, and M. Gerstein, "An integrated system for studying residue coevolution in proteins," *Bioinformatics*, vol. 24, no. 2, pp. 290–292, 2008.

[31] M. A. Potter and K. A. De Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evolutionary Computation*, vol. 8, no. 1, pp. 1–29, 2000.

[32] A. W. Iorio and X. Li, "A cooperative coevolutionary multiobjective algorithm using non-dominated sorting," in *ACM Genetic and Evolutionary Computation Conference (GECCO)*. Springer, 2004, pp. 537–548.

[33] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Information Sciences*, vol. 178, no. 15, pp. 2985–2999, 2008.

[34] X. Li and X. Yao, "Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2009, pp. 1546–1553.

[35] Z. Yang, J. Zhang, K. Tang, X. Yao, and A. C. Sanderson, "An adaptive coevolutionary differential evolution algorithm for large-scale optimization," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2009, pp. 102–109.

[36] W. Chen, T. Weise, Z. Yang, and K. Tang, "Large-scale global optimization using cooperative coevolution with variable interaction learning," in *Parallel Problem Solving from Nature, PPSN XI*, ser. Lecture Notes in Computer Science, R. Schaefer, C. Cotta, J. Kołodziej, and G. Rudolph, Eds. Springer Berlin Heidelberg, 2010, vol. 6239, pp. 300–309.

[37] X. Li and X. Yao, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 2, pp. 210–224, 2012.

[38] H. Zille, "Cooperative coevolution for large-scale multiobjective distance minimization problems," Master's thesis, Karlsruhe Institute of Technology (KIT), 2014.

[39] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[40] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," in *IEEE Congress on Evolutionary Computation (CEC)*, vol. 1, 2002, pp. 825–830.

[41] H. Zille, A. Kottenhahn, and S. Mostaghim, "Dynamic distance minimization problems for dynamic multi-objective optimization," in *IEEE Congress on Evolutionary Computation (CEC)*, June 2017, pp. 952–959.

[42] S. Zapotecas-Martínez, C. A. Coello Coello, H. E. Aguirre, and K. Tanaka, "A review of features and limitations of existing scalable multiobjective test suites," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 1, pp. 130–142, Feb 2019.

[43] K. Li, M. N. Omidvar, K. Deb, and X. Yao, "Variable interaction in multi-objective optimization problems," in *Parallel Problem Solving from Nature – PPSN XIV*, J. Handl, E. Hart, P. R. Lewis, M. López-Ibáñez, G. Ochoa, and B. Paechter, Eds. Cham: Springer International Publishing, 2016, pp. 399–409.

[44] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "Test problems for large-scale multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 47, no. 12, pp. 4108–4121, Dec 2017.

[45] R. Cheng, M. Li, Y. Tian, X. Zhang, S. Yang, Y. Jin, and X. Yao, "Benchmark functions for the CEC'2017 competition on many-objective optimization," School of Computer Science, University of Birmingham, Tech. Rep. CSR-17-01, 2017.

[46] ——, "A benchmark test suite for evolutionary many-objective optimization," *Complex & Intelligent Systems*, vol. 3, no. 1, pp. 67–81, Mar 2017.

[47] W. Hong, K. Tang, A. Zhou, H. Ishibuchi, and X. Yao, "A scalable indicator-based evolutionary algorithm for large-scale multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 525–537, June 2019.

[48] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari, "Multiobjective optimization test instances for the CEC 2009 special session and competition," School of Computer Science and Electronic Engineering, University of Essex, Tech. Rep. CES-487, 2008.

[49] M. Köppen and K. Yoshida, "Many-objective particle swarm optimization by gradual leader selection," in *Adaptive and Natural Computing Algorithms*. Springer, 2007, pp. 323–331.

[50] ——, "Substitute distance assignments in nsga-ii for handling many-objective optimization problems," in *Evolutionary Multi-Criterion Optimization*. Springer, 2007, pp. 727–741.

[51] O. Schütze, A. Lara, and C. A. Coello Coello, "On the influence of the number of objectives on the hardness of a multiobjective optimization problem," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 4, pp. 444–455, 2011.

[52] H. Ishibuchi, Y. Hitotsuyanagi, N. Tsukamoto, and Y. Nojima, "Many-objective test problems to visually examine the behavior of multiobjective evolution in a decision space," in *Parallel Problem Solving from Nature, PPSN XI*. Springer, 2010, pp. 91–100.

[53] H. Masuda, Y. Nojima, and H. Ishibuchi, "Visual examination of the behavior of emo algorithms for many-objective optimization with many decision variables," in *IEEE Congress on Evolutionary Computation*, 2014, pp. 2633–2640.

[54] S. Mostaghim, C. Steup, and H. Zille, "Multi-objective distance minimization problems – applications in technical systems," *at-Automatisierungstechnik*, vol. 66, no. 11, pp. 964–974, 2018.

[55] H. Ishibuchi, N. Akedo, and Y. Nojima, "A many-objective test problem for visually examining diversity maintenance behavior in a decision space," in *ACM Genetic and Evolutionary Computation Conference (GECCO)*. ACM, 2011, pp. 649–656.

[56] J. E. Fieldsend, T. Chugh, R. Allmendinger, and K. Miettinen, "A feature rich distance-based many-objective visualisable test problem generator," in *ACM Genetic and Evolutionary Computation Conference (GECCO)*. New York, NY, USA: ACM, 2019, pp. 541–549.

[57] Y. Nojima, T. Fukase, Y. Liu, N. Masuyama, and H. Ishibuchi, "Constrained multi-objective distance minimization problems," in *ACM Genetic and Evolutionary Computation Conference (GECCO)*. New York, NY, USA: ACM, 2019, pp. 586–594.

[58] M. Heibig, H. Zille, M. Javadi, and S. Mostaghim, "Performance of dynamic algorithms on the dynamic distance minimization problem," in *ACM Genetic and Evolutionary Computation Conference (GECCO) Companion*. New York, NY, USA: ACM, 2019, pp. 205–206.

[59] H. Zille and S. Mostaghim, "Properties of scalable distance minimization problems using the manhattan metric," in *IEEE Congress on Evolutionary Computation (CEC)*, May 2015, pp. 2875–2882.

[60] ——, "Using ϵ-dominance for hidden and degenerated pareto-fronts," in *IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec 2015, pp. 845–852.

[61] S. K. Goh, K. C. Tan, A. Al-Mamun, and H. A. Abbass, "Evolutionary big optimization (BigOpt) of signals," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2015, pp. 3332–3339.

[62] Y. Zhang, J. Liu, M. Zhou, and Z. Jiang, "A multi-objective memetic algorithm based on decomposition for big optimization problems," *Memetic Computing*, vol. 8, no. 1, pp. 45–61, 2016.

[63] S. Elsayed and R. Sarker, "An adaptive configuration of differential evolution algorithms for big data," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2015, pp. 695–702.

[64] D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithm research: A history and analysis," Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright- Patterson AFB, Ohio, Tech. Rep. TR-98-03, 1998.

[65] C. A. Coello Coello and M. Reyes Sierra, "A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm," in *MICAI 2004: Advances in Artificial Intelligence*, R. Monroy, G. Arroyo-Figueroa, L. E. Sucar, and H. Sossa, Eds. Berlin, Heidelberg: Springer, 2004, pp. 688–697.

[66] L. C. T. Bezerra, M. López-Ibáñez, and T. Stützle, "An empirical assessment of the properties of inverted generational distance on multi- and many-objective optimization," in *Evolutionary Multi-Criterion Optimization*, H. Trautmann, G. Rudolph, K. Klamroth, O. Schütze, M. Wiecek, Y. Jin, and C. Grimme, Eds. Cham: Springer International Publishing, 2017, pp. 31–45.

[67] A. Song, Q. Yang, W. N. Chen, and J. Zhang, "A random-based dynamic grouping strategy for large scale multi-objective optimization," in *IEEE Congress on Evolutionary Computation (CEC)*, July 2016, pp. 468–475.

[68] B. Cao, J. Zhao, Z. Lv, and X. Liu, "A distributed parallel cooperative coevolutionary multiobjective evolutionary algorithm for large-scale optimization," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 2030–2038, Aug 2017.

[69] C. He, L. Li, Y. Tian, X. Zhang, R. Cheng, Y. Jin, and X. Yao, "Accelerating large-scale multi-objective optimization via problem reformulation," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 6, pp. 949–961, Dec 2019.

[70] M. Li and J. Wei, "A cooperative co-evolutionary algorithm for large-scale multi-objective optimization problems," in *ACM Genetic and Evolutionary Computation Conference (GECCO)*. New York, NY, USA: ACM, 2018, pp. 1716–1721.

[71] H. Chen, X. Zhu, W. Pedrycz, S. Yin, G. Wu, and H. Yan, "Pea: Parallel evolutionary algorithm by separating convergence and diversity for large-scale multi-objective optimization," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*, July 2018, pp. 223–232.

[72] H. Ishibuchi, H. Masuda, Y. Tanigaki, and Y. Nojima, "Modified distance calculation in generational distance and inverted generational distance," in *Evolutionary Multi-Criterion Optimization*, A. Gaspar-Cunha, C. Henggeler Antunes, and C. C. Coello, Eds. Cham: Springer International Publishing, 2015, pp. 110–125.

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

[74] H. Ishibuchi, H. Masuda, and Y. Nojima, "A study on performance evaluation ability of a modified inverted generational distance indicator," in *ACM Genetic and Evolutionary Computation Conference (GECCO)*. New York, NY, USA: ACM, 2015, pp. 695–702.

[75] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 29–38, 2006.

[76] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[77] H. Chen, R. Cheng, J. Wen, H. Li, and J. Weng, "Solving large-scale many-objective optimization problems by covariance matrix adaptation evolution strategy with scalable small subpopulations," *Information Sciences*, vol. 509, pp. 457 – 469, 2020.

[78] H. Qian and Y. Yu, "Solving high-dimensional multi-objective optimization problems with low effective dimensions," in *Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, USA, 2017, pp. 875–881.

[79] L. Miguel Antonio and C. A. Coello Coello, "Decomposition-based approach for solving large scale multi-objective problems," in *Parallel Problem Solving from Nature – PPSN XIV*, J. Handl, E. Hart, P. R. Lewis, M. López-Ibáñez, G. Ochoa, and B. Paechter, Eds. Cham: Springer International Publishing, 2016, pp. 525–534.

[80] S. Mahdavi, M. E. Shiri, and S. Rahnamayan, "Metaheuristics in large-scale global continues optimization: A survey," *Information Sciences*, vol. 295, pp. 407–428, 2015.

[81] "Efficient implementation of an active set algorithm for large-scale portfolio selection," *Computers & Operations Research*, vol. 35, no. 12, pp. 3945 – 3961, 2008, part Special Issue: Telecommunications Network Engineering.

[82] W. Chen and K. Tang, "Impact of problem decomposition on cooperative coevolution," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2013, pp. 733–740.

[83] Z. Yang, K. Tang, and X. Yao, "Differential evolution for high-dimensional function optimization," in *IEEE Congress on Evolutionary Computation (CEC)*, 2007, pp. 3523–3530.

[84] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 378–393, 2014.

[85] M. N. Omidvar, Y. Mei, and X. Li, "Effective decomposition of large-scale separable continuous functions for cooperative co-evolutionary algorithms," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2014, pp. 1305–1312.

[86] Y. Sun, M. Kirley, and S. K. Halgamuge, "Extended differential grouping for large scale global optimization with direct and indirect variable interactions," in *ACM Genetic and Evolutionary Computation Conference (GECCO)*. ACM Press, 2015, pp. 313–320.

[87] M. N. Omidvar, M. Yang, Y. Mei, X. Li, and X. Yao, "DG2: A faster and more accurate differential grouping for large-scale black-box optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 6, pp. 929–942, Dec 2017.

[88] R. Cheng and Y. Jin, "A Competitive Swarm Optimizer for Large Scale Optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 2, pp. 191–204, 2015.

[89] Y. Mei, X. Li, and X. Yao, "Cooperative coevolution with route distance grouping for large-scale capacitated arc routing problems," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 435–449, June 2014.

[90] H. Zille, S. Nunokawa, S. Mostaghim, M. Miki, and T. Hiroyasu, "Comparison study of moeas with high dimensional objective and decision spaces," in *Japanese Society for Evolutionary Computation Symposium*, Kirishima, Japan, Dec 2013.

[91] S. Kukkonen and J. Lampinen, "GDE3: The third evolution step of generalized differential evolution," in *IEEE Congress on Evolutionary Computation (CEC)*, vol. 1. IEEE, 2005, pp. 443–450.

[92] S. Basu, A. Mondal, and A. Basu, "A cooperative co-evolutionary approach for multi-objective optimization," in *Recent Trends in Signal and Image Processing*, S. Bhattacharyya, A. Mukherjee, H. Bhaumik, S. Das, and K. Yoshida, Eds. Singapore: Springer Singapore, 2019, pp. 57–65.

[93] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, moea/d and nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284–302, April 2009.

[94] M. Emmerich, N. Beume, and B. Naujoks, "An emo algorithm using the hypervolume measure as selection criterion," in *Evolutionary Multi-Criterion Optimization*, C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 62–76.

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

[96] P. Yang, K. Tang, and X. Yao, "Turning high-dimensional optimization into computationally expensive optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 143–156, Feb 2018.

[97] K. Klamroth, S. Mostaghim, B. Naujoks, S. Poles, R. Purshouse, G. Rudolph, S. Ruzika, S. Sayın, M. M. Wiecek, and X. Yao, "Multiobjective optimization for interwoven systems," *Journal of Multi-Criteria Decision Analysis*, vol. 24, no. 1-2, pp. 71–81, 2017.

[98] Z. Yang, K. Tang, and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2008, pp. 1663–1670.

[99] C. A. Coello Coello and M. R. Sierra, "A coevolutionary multi-objective evolutionary algorithm," in *IEEE Congress on Evolutionary Computation (CEC)*, vol. 1. IEEE, 2003, pp. 482–489.

[100] K. Deb and A. Kumar, "Real-coded genetic algorithms with simulated binary crossover: Studies on multimodal and multiobjective problems," *Complex Systems*, vol. 9, pp. 431–454, 1995.

[101] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Computer Science and Informatics*, vol. 26, no. 1, pp. 30–45, 1996.

[102] K. Deb and S. Tiwari, "Omni-optimizer: A generic evolutionary algorithm for single and multi-objective optimization," *European Journal of Operational Research*, vol. 185, no. 3, pp. 1062–1087, 2008.

[103] M. Javadi, H. Zille, and S. Mostaghim, "Modified crowding distance and mutation for multimodal multi-objective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. New York, NY, USA: ACM, 2019, pp. 211–212.

[104] ——, "The effects of crowding distance and mutation in multimodal and multi-objective optimization problems," accepted for publication at the EUROGEN conference, Guimarães, Portugal. September 2019.

[105] J. Liang, W. Xu, C. Yue, K. Yu, H. Song, O. D. Crisalle, and B. Qu, "Multimodal multiobjective optimization with differential evolution," *Swarm and Evolutionary Computation*, vol. 44, pp. 1028–1059, 2019.

[106] C. Yue, B. Qu, and J. Liang, "A multiobjective particle swarm optimizer using ring topology for solving multimodal multiobjective problems," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 5, pp. 805–817, 2018.

[107] J. Liang, Q. Guo, C. Yue, B. Qu, and K. Yu, "A self-organizing multi-objective particle swarm optimization algorithm for multimodal multi-objective problems," in *International Conference on Swarm Intelligence*. Shanghai, China: Springer, 2018, pp. 550–560.

[108] A. J. Nebro, J. J. Durillo, P. J. Garcia Nieto, C. A. Coello Coello, F. Luna, and E. Alba, "SMPSO: A new PSO-based metaheuristic for multi-objective optimization," in *Symposium on Computational Intelligence in Multi-criteria Decision-making*. IEEE, 2009, pp. 66–73.

[109] M. Hamdan, "On the disruption-level of polynomial mutation for evolutionary multi-objective optimisation algorithms," *Computing and Informatics*, vol. 29, no. 5, pp. 783–800, 2010.

[110] S. Mostaghim, W. Halter, and A. Wille, "Linear multi-objective particle swarm optimization," in *Stigmergic Optimization*. Berlin, Heidelberg: Springer, 2006, pp. 209–238.

[111] K. Deb and A. Srinivasan, "Innovization: Innovating design principles through optimization," in *ACM Genetic and Evolutionary Computation Conference (GECCO)*. ACM, 2006, pp. 1629–1636.

[112] A. Gaur and K. Deb, "Effect of size and order of variables in rules for multi-objective repair-based innovization procedure," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2017, pp. 2177–2184.

[113] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "Platemo: A MATLAB platform for evolutionary multi-objective optimization," *CoRR*, vol. abs/1701.00879, 2017.

[114] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, 1947.

[115] M. Hollander, D. Wolfe, and E. Chicken, *Nonparametric Statistical Methods*, ser. Wiley Series in Probability and Statistics. Wiley, 2013.

APPENDIX A

Experimental Evaluations in the Literature

In this appendix, we describe the details of the experiments in the related literature. For each of the 13 algorithms listed in Section 3.2, the methodology of the experiments and the results based on their respective original publications are described in the following. A detailed summary of these related experiments and the analysis of the algorithms' performances is given in Section 4.1.6 and Table 4.3.

CCGDE3

The experiments in the CCGDE3 work were performed with up to $10,000,000$ function evaluations, using the ZDT1-3 and ZDT6 [39] problems with up to 5000 decision variables. The algorithms were stopped after a suitable approximation of the true Pareto-front was reached, i.e. when the population reached a value of 95% of the Hypervolume indicator with respect to the Hypervolume of a sample of the true Pareto-front. The needed evaluations were compared between the different algorithms (CCGDE3, GDE3 and NSGA-II). In result, CCGDE3 was able to obtain 95% approximations in all four tested problems even with up to $5,000$ variables, while the classical algorithms failed to do so within the budget of $10,000,000$ evaluations in the 4000 and 5000 variable instances of all problems. The computational budget to do so, however, was still quite high. To solve the 1000-variable instances, CCGDE3 needed between around $155,000$ and $1,280,000$ evaluations, depending on the used problem. For the 5000 variable instances, between $1,180,000$ and $5,760,000$ evaluations were needed.

Although the performance was originally not tested for more complicated benchmark functions, later experiments by the author of this thesis in [1] compared its performance among others with the WOF method (Section 5.1). The same ZDT experiments with up to 5000 variables were repeated as well as tests with up to 1000-dimensional problems of the WFG [28], DTLZ [40] and CEC2009 competition (UF) [48] benchmark families. The results showed that CCGDE3 was significantly outperformed by the WOF method

and in most instances even by classical (i.e. non-large-scale) optimisation methods. This may be due to the inability to change the groups during runtime, making this algorithm potentially vulnerable to problems with high interactions between variables. Another drawback was the high computational budget used in the CCGDE3 study. Using up to $5,760,000$ function evaluations to solve ZDT problems may be unsuitable for real applications, and is far more than other modern methods require for the same task.

MOEA/D^2

For the experiments in this MOEA/D^2 article, the DTLZ1-7 problems were used with 200 to 1200 variables and three objective functions. The variables were divided so that each group always contained 2 variables, i.e. they used 100 groups for the 200-variable problems, 200 groups for the 400-variable problems, and so forth. The obtained Hypervolume values were compared with those of the MOEA/D and GDE3 algorithms after a total of $100,000$ function evaluations. The results showed a significant improvement in performance compared to both of the standard (and low-scale) algorithms, although it is noteworthy that no other large-scale methods, like for instance CCGDE3, were used in the experiments for comparison.

CCLSM

The CCLSM experiments were done using $50,000$ evaluations on selected problem instances, i.e. WFG2, WFG3, UF5, LSMOP1, 5 and 9. The number of objectives varied between 2 and 10, and the number of variables between 100 and 300. However, it must be noted that no exhaustive combinations of these numbers were reported, but merely an arbitrary selection with no clear reason on why exactly these and only these results were reported. WFG3 was only tested with 7 and 10 objectives, WFG2 with 5 objectives, LSMOP1 with 3 and 5 objectives, while LSMOP5 was tested with 4 and 5 objectives. Furthermore, only averages were shown, and no spread measure of performance was reported. The number of independent runs was not revealed in the paper, and no test for statistical significance was done. In addition, the method was not able to deliver an acceptable approximation of the Pareto-front in a 2-objective LSMOP9 problem with only 100 decision variables, and a comparison with other large-scale methods was not done.

MOEA/D(s&ns)

The MOEA/D(s&ns) was compared with NSGA-II on ZDT1-3 and LSMOP1, 5 and 9 problems with 2 and 3 objectives and 200 and 300 variables respectively. In the results, however, the different test problems were compared with the result that the obtained IGD values were lower on the ZDT problems than on the LSMOP problems. The used number of independent runs was not reported, nor was the number of function evaluations, and no statistical tests were conducted.

MOEA/DVA

In the MOEA/DVA article, experiments were performed with 200 decision variables using the 2-objective ZDT4 and UF1-6 and the 3-objective DTLZ1, DTLZ3 and UF10 benchmarks. The maximum number of function evaluations was set between $1,200,000$ to $3,000,000$, depending on the test problem. MOEA/DVA outperformed other popular algorithms in these tests, although it must be noted that none of them was specifically designed for large-scale problems. Further, MOEA/DVA was used with 1000-variable problems in this and other subsequent studies.

MOEA/D-RDG

The dynamic grouping strategy MOEA/D-RDG was compared in [67], using a computational budget of $10,000,000$ function evaluations, with the original MOEA/DVA and MOEA/D. They used the UF1-7 problems with 2 objectives and the UF8-10 and WFG1-9 problems with 3 objective functions and between 800 and 1000 variables. The results showed that the proposed RDG version can improve the performance of the original MOEA/DVA in most of the problem instances.

LMEA

For the evaluation of LMEA in [25], large-scale instances of the LSMOP benchmarks were used with 500 variables and 5 objectives. The maximum function evaluations were set to $6,800,000$. LMEA performed superior to the other algorithms in the study on the LSMOP1-3, 5, 8 and 9 problems. On the other hand, it was outperformed by MOEA/DVA on the LSMOP4 and LSMOP7 and by the non-large-scale method NSGA-III on the LSMOP6 benchmark. In further experiments, LMEA was tested on selected problems from the DTLZ, WFG and UF families with up to 1000 decision variables and 10 objectives. A computational budget of $17,000,000$ function evaluations was used in these experiments. It was outperformed by MOEA/DVA on some instances DTLZ1 and 3 and all instances of the UF9 and 10 problems. However, LMEA performed best on all instances of DTLZ5 and 6, WFG3 and some instances of DTLZ1, 2, 3 and 7, showing that it can work well on large-scale problems which include many variables and many objectives at the same time.

Finally, LMEA was used on instances of up to 5000 variables with the DTLZ1-7, WFG3 and UF9 and UF10 problems. However, the performance of these high-dimensional instances was not compared to any other methods, and used up to $230,000,000$ function evaluations. The goal of these experiments in the original article was to show that the performance of LMEA does not deteriorate when increasing the number of variables. This claim, however, needs further investigation, since this behaviour can be a result of a constant complexity of the used benchmarks, despite scaling the variable numbers. Furthermore, as the number of used evaluations was scaled up dramatically, it is not surprising to observe respective similar solution quality in the end of the search process.

A budget of $230,000,000$ evaluations to solve a 5000-variable problem might be seen as the biggest disadvantage of the LMEA method. Similar to the interaction-based decomposition in MOEA/DVA, the number of required evaluations for obtaining the groups rises quadratically with the decision variables. For the 1000-variable instances, the authors of the LMEA article report an average runtime of around 2386 seconds, which is approximately 40 minutes for a single run. Although these numbers are faster than the reported times for the MOEA/DVA, it corresponds to an average of 7124 function evaluations per second. It remains unclear whether these numbers are in any way related to real-world applications, since complex simulations may consume much more time for each single evaluation.

DPCCMOEA

The experiments in the DPCCMOEA publication compared this method with CCGDE3 and MOEA/DVA on 3-objective instances with 100 variables of the DTLZ1-7 and WFG1-9 problems. The number of function evaluations was set to $10,000,000$. As expected, the proposed distributed method was able to achieve a large speed-up in computation time compared to its sequential competitors. In addition, the DPCCMOEA was also able to achieve better solution quality on several problem instances, although it must be noted that a pairwise test for statistical significance was not reported and the resulting IGD and Hypervolume values were in fact very close together between MOEA/DVA and DPCCMOEA in most instances. Only problems with 100 variables were tested, which makes the computational budget of 10 million function evaluations seem generous, especially in the light of other recent large-scale algorithms and their capabilities, which solve problems with many more variables using the same budgets.

S^3-CMA-ES

For testing the S^3-CMA-ES, the experiments in the original article were performed using the LSMOP1-9 problems with 5, 8, 10 and 15 objective functions, containing around 500, 800, 1000 and 1500 variables respectively. The used function evaluations were determined based on the amount of variables, i.e. $5,000,000$, $8,000,000$, $10,000,000$ and $15,000,000$ evaluations respectively for each of the problem instances. S^3-CMA-ES performed well on most problem instances, but was in some cases outperformed by MOEA/DVA. Moreover, it was also outperformed on several problem instances by the NSGA-III and RVEA algorithms, which are dedicated many-objective algorithms and originally not designed for large search spaces. This might indicate that the actual challenge of the used problem instances was more related to the high number of objective functions rather than the high-dimensional decision space. The results, however, show that the decomposition into subpopulations and variable groups in S^3-CMA-ES, although a large computational budget was needed, is able to deal well with LSMOP's high-dimensional objective spaces as well as search spaces.

PEA

In the results of the PEA article, using a budget roughly between $3,000,000$ and $10,000,000$ function evaluations, the PEA was compared with LMEA and MOEA/DVA, along with some other, non-large-scale methods. The LSMOP1-3 and MaF1-7 problems were used with between 307 and 1039 variables and 3 to 10 objective functions. While the PEA algorithm was able to decrease the computation time due to its parallel nature, it also showed significant improvements in terms of IGD values compared to LMEA and MOEA/DVA.

ReMO

The experimental evaluation of ReMO was performed with modified versions of the ZDT1-3 functions. To test for the ability of the algorithms to deal with low effective dimensions, the ZDT functions were used with 30 variables, but embedded in a $10,000$-dimensional decision space. This means that 9970 variables were artificially added to the problems, which had to be optimised by the used metaheuristics, but had no or almost no influence on the actual objective function values. Using their random embedding approach with $v = 50$, the ReMO algorithms were able to search in only a 50-dimensional space to solve these problems. The ReMO approach was not tested on more complicated benchmarks or problems without the property of having low effective dimensions. The results showed superior performance compared to the respective original NSGA-II and MOEA/D algrithms. Notable is that this study used a very low function evaluation budget of just 3000 evaluations. It is, however, not surprising that the NSGA-II and MOEA/D algorithms were not able to obtain good solutions in a $10,000$-dimensional problem after such a short time. The results of the ReMO versions improved on these performances, although the approach was not tested on more complicated benchmarks or problems without the property of having low effective dimensions. Experiments performed in the later DLS-MOEA work [47] showed that Re-NSGA-II could not compare in other high-dimensional benchmarks against DLS-MOEA, WOF-SMPSO or even CCGDE3.

DLS-MOEA

The DLS-MOEA aims to produce more diverse solution sets in problems where diversity is difficult to obtain. The experimental evaluation in the article was therefore focused mainly on problems with such properties. The experimental evaluation was conducted using several state-of-the-art large-scale algorithms, among them the Re-NSGA-II, CCGDE3, LMEA, MOEA/DVA and the WOF-SMPSO (see Section 5.1). The focus lay on large search spaces and all experiments were done with only 2 objective functions, and up to $10,000,000$ function evaluations were used for problem instances with 1024 to 8192 decision variables. Overall, DLS-MOEA achieved good performances on a variety of large-scale instances. However, the average best performance for the convergence-related and the diversity-type I related problems was achieved by the WOF-SMPSO method,

while the DLS-MOEA was able to perform best on average on the diversity-type II, i.e. the UF1-7, problems.

LSMOF

The experiments of the LSMOF article compared the algorithm with several low-dimensional methods and showed that the framework is able to increase the performance for different instances of the LSMOP, DTLZ and WFG benchmark suites. Only a relatively small budget of $50,000$ function evaluations was used, and the number of objectives were chosen as 2 and 3, with 200 to 1000 variables. In addition, LSMOF was compared with MOEA/DVA and WOF (see Section 5.1), with decision-variables rising up to 5000. The results showed that the LSMOF which employs NSGA-II as the internal optimiser (LS-NSGA-II) was able to perform on par with WOF-NSGA-II and MOEA/DVA on most problem instances and outperform them in several cases, while in other instances MOEA/DVA or WOF-NSGA-II showed significantly better performance. When employing SMPSO as the optimiser, WOF-SMPSO and the LS-SMPSO performed on par on most instances of the WFG problems and mixed results on the DTLZ function. The LSMOP problems were only tested with the NSGA-II-versions of both algorithms.

APPENDIX B

Detailed IGD Results

In this Appendix we list the detailed results regarding the IGD indicator of all algorithms in the different experiments from Chapter 6. Shown are the medians and interquartile ranges over the set of 31 independent runs of the respective algorithm on each benchmark. The respective best performance for each benchmark instance (each row) is shown in bold front and marked with a grey background. Statistical significance is assumed for $p < 0.01$ and indicated by an asterisk for each algorithm compared to the respective best in each row.

Table B.1: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100,000 function evaluations.

		SMPSO	WOF-SMPSO	NSGA-II	WOF-NSGA-II	WOF-Randomised
n = 200						
LSMOP1	2	8.61E-1 * (1.6E-1)	**5.34E-2** (3.1E-2)	3.25E-1 * (4.0E-2)	5.78E-1 * (8.4E-2)	1.80E-1 * (1.0E-1)
LSMOP2	2	8.95E-2 * (2.7E-3)	**1.31E-2** (8.7E-4)	9.68E-2 * (4.6E-3)	4.78E-2 * (9.0E-3)	2.08E-2 * (5.6E-3)
LSMOP3	2	2.42E1 * (1.6E0)	1.49E0 * (1.0E-2)	1.30E1 * (2.2E0)	**1.06E0** (2.4E-1)	1.45E0 * (5.1E-1)
LSMOP4	2	1.14E-1 * (2.2E-2)	**5.75E-2** (9.2E-3)	1.45E-1 * (4.0E-3)	1.16E-1 * (6.5E-3)	8.34E-2 * (1.6E-2)
LSMOP5	2	1.05E0 * (3.5E-1)	3.52E-1 (2.9E-1)	3.80E-1 * (5.9E-2)	**3.43E-1** (2.6E-2)	3.94E-1 (8.6E-2)
LSMOP6	2	5.06E-1 * (6.0E-2)	**1.08E-1** (1.5E-2)	8.86E-1 * (4.4E-2)	6.57E-1 * (4.7E-2)	3.93E-1 * (2.0E-1)
LSMOP7	2	6.75E1 * (2.2E1)	1.47E0 * (4.7E-3)	3.70E0 * (1.8E0)	1.47E0 * (5.0E-3)	**1.47E0** (1.0E-2)
LSMOP8	2	7.80E-1 * (2.2E-1)	**1.18E-1** (1.0E-1)	3.81E-1 * (9.7E-2)	7.42E-1 * (3.9E-1)	2.53E-1 * (1.7E-1)
LSMOP9	2	**4.22E-1** (1.0E-2)	8.10E-1 * (2.5E-1)	1.40E0 * (1.4E-1)	8.10E-1 * (1.5E-9)	8.10E-1 * (—)
n = 300						
LSMOP1	3	2.23E0 * (4.7E-1)	**1.78E-1** (4.8E-2)	2.18E0 * (4.7E-1)	3.09E-1 * (5.8E-2)	3.03E-1 * (1.7E-1)
LSMOP2	3	9.84E-2 * (6.1E-3)	**7.44E-2** (6.3E-3)	1.05E-1 * (3.8E-3)	1.15E-1 * (4.5E-3)	8.34E-2 * (5.2E-3)
LSMOP3	3	1.50E1 * (5.4E0)	8.60E-1 * (3.7E-4)	7.77E0 * (1.8E0)	**8.26E-1** (8.8E-2)	8.60E-1 * (1.5E-4)
LSMOP4	3	2.61E-1 * (9.8E-3)	1.98E-1 * (8.8E-3)	2.65E-1 * (1.0E-2)	2.76E-1 * (1.0E-2)	**1.84E-1** (1.2E-2)
LSMOP5	3	6.05E0 * (7.6E-1)	4.50E-1 (1.6E-1)	6.02E0 * (1.1E0)	5.21E-1 * (2.2E-2)	**4.26E-1** (7.7E-2)
LSMOP6	3	2.02E3 * (1.7E3)	**7.52E-1** (2.1E-1)	5.16E0 * (1.3E2)	1.24E0 * (4.2E-3)	1.24E0 * (1.7E-2)
LSMOP7	3	1.52E0 * (2.5E-2)	8.98E-1 * (3.4E-3)	1.57E0 * (6.7E-2)	9.15E-1 * (1.1E-2)	**8.70E-1** (1.7E-2)
LSMOP8	3	9.80E-1 * (1.2E-2)	**1.05E-1** (2.1E-2)	4.59E-1 * (6.3E-2)	3.61E-1 * (1.8E-2)	2.25E-1 * (1.5E-1)
LSMOP9	3	1.57E1 * (3.1E0)	1.53E0 * (4.0E-1)	2.95E0 * (8.9E-1)	**1.14E0** (4.6E-4)	1.15E0 * (2.1E-1)
n = 400						
LSMOP1	4	4.92E0 * (2.2E0)	4.98E-1 (1.0E-1)	5.98E0 * (1.1E0)	6.54E-1 * (3.8E-2)	**4.89E-1** (1.7E-1)
LSMOP2	4	1.70E-1 * (9.4E-3)	1.50E-1 (7.6E-3)	1.81E-1 * (1.7E-2)	1.85E-1 * (7.9E-3)	**1.48E-1** (4.7E-3)
LSMOP3	4	1.97E1 * (2.8E0)	1.78E0 * (2.7E-3)	1.36E1 * (2.6E0)	1.77E0 * (5.4E-2)	**1.52E0** (4.6E-1)
LSMOP4	4	2.53E-1 * (1.8E-2)	2.24E-1 * (1.4E-2)	2.66E-1 * (1.8E-2)	2.76E-1 * (1.1E-2)	**2.21E-1** (9.3E-3)
LSMOP5	4	1.85E1 * (5.9E0)	5.44E-1 (3.3E-1)	1.83E1 * (2.2E0)	4.66E-1 * (7.0E-3)	**4.54E-1** (1.7E-2)
LSMOP6	4	1.27E0 * (1.0E-11)	8.86E-1 (1.0E-1)	1.27E0 * (7.4E-3)	8.89E-1 * (7.0E-3)	**8.69E-1** (8.2E-3)
LSMOP7	4	5.23E4 * (5.2E4)	1.23E0 (4.5E-2)	1.91E4 * (5.6E3)	1.23E0 (1.1E-2)	**1.22E0** (2.6E-1)
LSMOP8	4	1.15E1 * (4.9E0)	4.89E-1 * (1.2E-1)	9.23E0 * (1.2E0)	4.66E-1 * (4.1E-3)	**4.49E-1** (5.1E-2)
LSMOP9	4	1.34E1 * (2.9E0)	2.24E0 * (—)	5.80E0 * (1.0E0)	**1.46E0** (1.5E-3)	1.77E0 * (4.6E-1)
n = 500						
LSMOP1	5	5.55E0 * (2.4E0)	8.35E-1 * (7.0E-2)	8.89E0 * (1.2E0)	9.11E-1 * (1.6E-2)	**7.80E-1** (2.4E-1)
LSMOP2	5	2.07E-1 * (1.3E-2)	1.91E-1 * (5.8E-3)	2.17E-1 * (1.5E-2)	2.19E-1 * (1.5E-2)	**1.73E-1** (2.3E-3)
LSMOP3	5	2.01E1 * (3.4E0)	**9.58E-1** (—)	1.92E1 * (2.4E0)	9.59E-1 * (5.5E-3)	9.58E-1 (—)
LSMOP4	5	3.43E-1 * (3.1E-2)	2.95E-1 * (1.3E-2)	3.72E-1 * (1.8E-2)	3.44E-1 * (1.9E-2)	**2.85E-1** (5.2E-3)
LSMOP5	5	2.12E1 * (6.0E0)	9.52E-1 * (3.1E-1)	2.02E1 * (3.5E0)	5.47E-1 * (6.7E-2)	**4.28E-1** (2.8E-2)
LSMOP6	5	8.51E4 * (2.2E4)	1.80E0 * (2.6E-1)	5.17E4 * (1.8E4)	1.36E0 * (1.1E-1)	**1.12E0** (1.4E-1)
LSMOP7	5	3.35E0 * (1.8E-1)	1.65E0 * (2.9E-1)	3.47E0 * (1.0E-1)	1.28E0 * (1.5E-1)	**1.03E0** (2.0E-1)
LSMOP8	5	1.21E0 * (1.4E-2)	7.66E-1 * (1.7E-1)	1.21E0 * (2.2E-2)	4.13E-1 * (2.7E-2)	**3.42E-1** (1.4E-2)
LSMOP9	5	5.40E1 * (7.1E0)	3.00E0 * (2.5E-4)	2.92E1 * (7.9E0)	**1.82E0** (5.4E-3)	1.90E0 * (6.1E-1)
n = 1000						
LSMOP1	2	1.76E0 * (1.3E-1)	**7.17E-2** (6.2E-3)	3.55E0 * (3.3E-1)	6.46E-1 * (9.1E-2)	1.74E-1 * (1.1E-1)
LSMOP2	2	2.56E-2 * (8.5E-4)	**7.29E-3** (3.9E-4)	3.62E-2 * (6.7E-4)	1.90E-2 * (5.2E-4)	1.03E-2 * (3.0E-3)
LSMOP3	2	2.81E1 * (5.3E-1)	**1.56E0** (1.9E-3)	2.12E1 * (1.1E0)	1.57E0 * (2.5E-3)	1.57E0 * (3.7E-3)
LSMOP4	2	5.34E-2 * (5.6E-4)	**2.04E-2** (5.9E-4)	6.07E-2 * (9.4E-4)	4.13E-2 * (4.3E-3)	2.37E-2 * (2.5E-3)
LSMOP5	2	3.89E0 * (2.4E-1)	7.42E-1 (2.9E-1)	1.01E1 * (7.3E-1)	7.42E-1 (1.6E-1)	**7.41E-1** (4.7E-2)
LSMOP6	2	7.58E-1 * (1.9E-3)	**1.73E-1** (3.4E-3)	7.74E-1 * (8.4E-4)	6.71E-1 * (1.3E-3)	3.49E-1 * (1.8E-1)
LSMOP7	2	2.04E3 * (3.6E2)	1.51E0 (6.0E-4)	3.91E3 * (4.5E3)	1.51E0 * (2.4E-3)	**1.51E0** (2.8E-3)
LSMOP8	2	2.95E0 * (3.5E-1)	**2.12E-1** (5.9E-1)	5.00E0 * (6.3E-1)	7.42E-1 * (4.0E-2)	2.14E-1 (1.8E-1)
LSMOP9	2	2.80E0 * (7.0E-1)	**4.67E-1** (9.5E-3)	1.39E0 * (1.7E-1)	8.08E-1 * (1.6E-3)	4.90E-1 * (7.5E-1)
LSMOP1	3	2.55E0 * (5.8E-1)	**2.00E-1** (4.1E-2)	6.23E0 * (8.2E-1)	5.93E-1 * (3.9E-2)	3.24E-1 * (1.4E-1)
LSMOP2	3	6.45E-2 * (1.6E-3)	6.12E-2 * (5.2E-3)	7.04E-2 * (6.5E-3)	7.58E-2 * (8.1E-3)	**5.16E-2** (5.7E-4)
LSMOP3	3	1.65E1 * (4.0E0)	**8.60E-1** (—)	1.89E1 * (6.9E0)	8.60E-1 * (4.4E-4)	8.60E-1 (2.6E-8)
LSMOP4	3	1.19E-1 * (4.5E-3)	**8.91E-2** (4.9E-3)	1.31E-1 * (4.7E-3)	1.43E-1 * (5.2E-3)	9.31E-2 * (7.0E-3)
LSMOP5	3	6.07E0 * (5.8E-1)	**4.29E-1** (2.3E-1)	1.59E1 * (1.1E0)	5.41E-1 * (5.1E-4)	4.51E-1 (7.3E-2)
LSMOP6	3	3.13E3 * (1.7E3)	**9.11E-1** (5.8E-1)	1.09E4 * (2.9E3)	1.31E0 * (1.8E-3)	1.31E0 * (9.8E-2)
LSMOP7	3	1.08E0 * (1.8E-3)	8.49E-1 (1.0E-1)	1.10E0 * (4.8E-3)	8.56E-1 * (4.3E-3)	**8.47E-1** (3.3E-3)
LSMOP8	3	9.57E-1 * (7.6E-2)	**9.22E-2** (1.5E-2)	9.58E-1 * (8.5E-3)	3.43E-1 * (5.7E-2)	1.86E-1 * (7.8E-2)
LSMOP9	3	2.65E1 * (2.6E0)	**1.11E0** (5.7E-1)	1.39E1 * (2.7E0)	1.14E0 * (3.4E-4)	1.14E0 * (1.7E-2)
LSMOP1	4	5.71E0 * (2.1E0)	**4.91E-1** (1.3E-1)	8.43E0 * (1.0E0)	8.33E-1 * (3.1E-2)	5.98E-1 * (1.7E-1)
LSMOP2	4	1.33E-1 * (7.1E-3)	1.26E-1 * (5.7E-3)	1.44E-1 * (5.5E-3)	1.46E-1 * (9.9E-3)	**1.16E-1** (1.2E-3)
LSMOP3	4	1.98E1 * (2.5E0)	**1.81E0** (3.3E-3)	2.04E1 * (1.6E0)	1.81E0 * (2.4E-3)	1.81E0 (2.4E-2)
LSMOP4	4	1.70E-1 * (8.9E-3)	1.55E-1 * (6.9E-3)	1.84E-1 * (7.6E-3)	1.87E-1 * (9.4E-3)	**1.48E-1** (4.2E-3)
LSMOP5	4	2.02E1 * (4.9E0)	4.94E-1 (2.5E-1)	2.11E1 * (1.3E0)	4.65E-1 * (6.2E-3)	**4.57E-1** (1.3E-2)
LSMOP6	4	1.12E0 * (2.3E-12)	9.09E-1 (1.1E-1)	1.12E0 * (7.9E-4)	9.05E-1 * (4.9E-3)	**8.97E-1** (7.3E-3)
LSMOP7	4	5.48E4 * (3.9E4)	**1.20E0** (2.5E-2)	3.98E4 * (8.9E3)	1.25E0 * (1.5E-2)	1.24E0 * (1.1E-1)
LSMOP8	4	1.05E1 * (3.7E0)	4.89E-1 * (1.3E-1)	1.26E1 * (8.8E-1)	4.65E-1 * (4.0E-3)	**4.54E-1** (2.4E-2)
LSMOP9	4	1.48E1 * (1.7E0)	2.24E0 (1.4E0)	2.27E1 * (3.3E0)	**1.46E0** (7.1E-4)	1.74E0 * (4.5E-1)
LSMOP1	5	7.02E0 * (2.0E0)	8.46E-1 (1.0E-1)	9.20E0 * (1.2E0)	9.17E-1 * (1.4E-2)	**8.33E-1** (1.7E-1)
LSMOP2	5	1.82E-1 * (1.1E-2)	1.82E-1 * (1.1E-2)	1.92E-1 * (8.4E-3)	1.91E-1 * (9.2E-3)	**1.52E-1** (1.3E-3)
LSMOP3	5	2.17E1 * (2.4E0)	**9.58E-1** (—)	2.34E1 * (1.4E0)	9.62E-1 * (6.7E-2)	9.58E-1 (—)
LSMOP4	5	2.56E-1 * (1.4E-2)	2.33E-1 * (1.3E-2)	2.87E-1 * (1.9E-2)	2.71E-1 * (1.5E-2)	**2.22E-1** (5.6E-3)
LSMOP5	5	1.89E1 * (7.4E0)	9.83E-1 * (2.2E-1)	2.08E1 * (3.2E0)	5.23E-1 * (4.6E-2)	**4.29E-1** (6.7E-2)
LSMOP6	5	7.59E4 * (4.5E4)	1.83E0 * (2.8E-1)	5.30E4 * (1.5E4)	1.48E0 * (2.2E-1)	**1.26E0** (4.0E-1)
LSMOP7	5	2.05E0 * (3.6E-2)	1.35E0 * (1.8E-1)	2.09E0 * (4.4E-2)	1.16E0 * (1.4E-1)	**1.02E0** (1.4E-1)
LSMOP8	5	1.15E0 * (2.3E-3)	6.59E-1 * (1.5E-1)	1.15E0 * (7.5E-3)	4.08E-1 * (2.5E-2)	**3.32E-1** (1.1E-2)
LSMOP9	5	6.17E1 * (2.8E0)	3.00E0 (2.2E0)	7.38E1 * (1.1E1)	**1.81E0** (7.4E-3)	1.86E0 * (3.4E-1)

Table B.2: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100, 000 function evaluations.

		NSGA-III	WOF-NSGA-III	MOEA/D	WOF-MOEA/D	WOF-Randomised
n = 200						
LSMOP1	2	5.35E-1 * (1.5E-1)	5.31E-1 * (1.3E-1)	2.20E-1 * (6.2E-2)	2.13E-1 (3.1E-2)	**1.80E-1** (1.0E-1)
LSMOP2	2	9.24E-2 * (3.3E-3)	4.59E-2 * (7.5E-3)	1.21E-1 * (4.4E-3)	6.83E-2 * (7.1E-3)	**2.08E-2** (5.6E-3)
LSMOP3	2	1.25E1 * (3.5E0)	9.65E-1 * (4.3E-1)	**7.07E-1** (8.3E-2)	7.08E-1 (3.3E-2)	1.45E0 * (5.1E-1)
LSMOP4	2	1.40E-1 * (3.4E-3)	1.16E-1 * (7.6E-3)	1.62E-1 * (1.1E-2)	1.46E-1 * (7.7E-3)	**8.34E-2** (1.6E-2)
LSMOP5	2	5.01E-1 * (1.6E-1)	3.45E-1 * (3.5E-1)	7.42E-1 * (4.0E-1)	**2.01E-1** (1.1E-2)	3.94E-1 * (8.6E-2)
LSMOP6	2	8.90E-1 * (1.0E-2)	6.61E-1 * (5.0E-2)	5.79E-1 * (4.2E-2)	7.47E-1 * (1.9E-3)	**3.93E-1** (2.0E-1)
LSMOP7	2	1.12E1 * (1.7E1)	1.47E0 * (9.9E-3)	3.55E0 * (1.6E0)	**1.46E0** (2.6E-3)	1.47E0 * (1.0E-2)
LSMOP8	2	4.82E-1 * (1.7E-1)	7.42E-1 * (3.9E-1)	2.12E-1 * (4.1E-1)	**1.14E-1** (8.9E-3)	2.53E-1 * (1.7E-1)
LSMOP9	2	1.50E0 * (1.0E-1)	8.10E-1 * (6.6E-8)	4.32E-1 * (3.1E-2)	**1.74E-1** (4.1E-2)	8.10E-1 * (—)
n = 300						
LSMOP1	3	9.03E-1 * (1.9E-1)	4.04E-1 * (1.0E-1)	4.81E-1 * (2.4E-1)	4.36E-1 * (8.2E-2)	**3.03E-1** (1.7E-1)
LSMOP2	3	**8.31E-2** (1.4E-3)	8.85E-2 * (3.1E-3)	8.89E-2 * (1.0E-3)	9.23E-2 * (1.8E-3)	8.34E-2 (5.2E-3)
LSMOP3	3	3.72E0 * (9.2E-1)	**8.60E-1** (4.1E-2)	9.29E-1 (5.6E-1)	8.60E-1 (8.0E-6)	8.60E-1 (1.5E-4)
LSMOP4	3	1.97E-1 * (4.2E-3)	2.18E-1 * (1.2E-2)	2.49E-1 * (6.3E-3)	2.60E-1 * (2.2E-2)	**1.84E-1** (1.2E-2)
LSMOP5	3	2.84E0 * (9.9E-1)	5.32E-1 * (3.0E-2)	6.83E-1 * (3.2E-1)	**2.51E-1** (2.3E-1)	4.26E-1 * (7.7E-2)
LSMOP6	3	5.14E0 * (4.6E0)	**1.24E0** (3.9E-3)	2.00E0 * (1.3E0)	1.58E0 * (2.6E-1)	1.24E0 (1.7E-2)
LSMOP7	3	1.54E0 * (1.9E-2)	8.97E-1 (1.7E-2)	**8.57E-1** (2.1E-1)	9.46E-1 * (1.1E-4)	8.70E-1 (1.7E-2)
LSMOP8	3	3.69E-1 * (5.0E-2)	3.62E-1 * (9.6E-4)	5.93E-1 * (7.7E-3)	2.28E-1 (3.2E-2)	**2.25E-1** (1.5E-1)
LSMOP9	3	2.92E0 * (9.0E-2)	1.14E0 * (2.9E-3)	**4.86E-1** (5.6E-2)	1.15E0 * (1.1E-2)	1.15E0 * (2.1E-1)
n = 400						
LSMOP1	4	3.22E0 * (4.3E-1)	6.06E-1 * (7.0E-2)	2.43E0 * (1.0E0)	6.07E-1 * (7.4E-2)	**4.89E-1** (1.7E-1)
LSMOP2	4	1.51E-1 * (1.1E-3)	1.53E-1 * (3.2E-3)	1.56E-1 * (1.2E-3)	1.52E-1 * (2.3E-3)	**1.48E-1** (4.7E-3)
LSMOP3	4	1.69E1 * (4.1E0)	1.67E0 * (2.1E-1)	1.66E0 * (9.3E-1)	**9.43E-1** (2.2E-2)	1.52E0 * (4.6E-1)
LSMOP4	4	**2.01E-1** (3.8E-3)	2.31E-1 * (6.6E-3)	2.19E-1 * (5.8E-3)	2.58E-1 * (7.3E-2)	2.21E-1 * (9.3E-3)
LSMOP5	4	5.99E0 * (1.1E0)	4.57E-1 * (1.2E-3)	7.14E-1 * (1.9E-1)	1.04E0 * (2.2E-5)	**4.54E-1** (1.7E-2)
LSMOP6	4	1.22E0 * (9.0E-3)	8.75E-1 * (3.4E-3)	9.16E-1 * (5.6E-2)	1.05E0 * (1.2E-3)	**8.69E-1** (8.2E-3)
LSMOP7	4	4.48E2 * (2.9E2)	**1.21E0** (2.9E-3)	2.99E0 * (1.0E0)	1.75E0 * (2.5E-3)	1.22E0 * (2.6E-1)
LSMOP8	4	2.33E0 * (6.7E-1)	4.57E-1 * (1.4E-3)	6.13E-1 * (4.4E-2)	1.04E0 * (5.9E-1)	**4.49E-1** (5.1E-2)
LSMOP9	4	7.07E0 * (7.4E-1)	1.48E0 * (2.3E-2)	**6.85E-1** (9.0E-2)	1.47E0 * (3.2E-2)	1.77E0 * (4.6E-1)
n = 500						
LSMOP1	5	2.98E0 * (6.9E-1)	8.64E-1 * (3.2E-2)	1.18E0 * (1.2E0)	**5.37E-1** (2.5E-1)	7.80E-1 * (2.4E-1)
LSMOP2	5	1.75E-1 * (4.2E-4)	1.74E-1 * (2.0E-3)	1.74E-1 * (5.8E-4)	**1.67E-1** (8.3E-3)	1.73E-1 * (2.3E-3)
LSMOP3	5	1.14E1 * (2.6E0)	9.60E-1 * (1.0E-1)	1.00E0 * (5.6E-1)	**9.58E-1** (2.1E-8)	9.58E-1 * (—)
LSMOP4	5	2.91E-1 * (3.6E-3)	2.93E-1 * (6.9E-3)	2.87E-1 * (8.1E-3)	**2.52E-1** (8.1E-3)	2.85E-1 * (5.2E-3)
LSMOP5	5	7.53E0 * (9.7E-1)	4.20E-1 (2.9E-2)	7.88E-1 * (3.5E-1)	**3.58E-1** (6.3E-1)	4.28E-1 (2.8E-2)
LSMOP6	5	2.94E1 * (6.4E1)	1.12E0 (2.4E-2)	1.32E0 * (4.7E-1)	1.40E0 * (8.6E-2)	**1.12E0** (1.4E-1)
LSMOP7	5	2.66E0 * (1.5E-1)	1.06E0 (8.2E-2)	1.10E0 (2.6E-1)	1.11E0 (7.6E-4)	**1.03E0** (2.0E-1)
LSMOP8	5	1.15E0 * (1.2E-2)	3.42E-1 * (1.1E-2)	7.60E-1 * (1.1E-2)	**3.27E-1** (2.2E-2)	3.42E-1 * (1.4E-2)
LSMOP9	5	1.42E1 * (3.4E0)	1.84E0 * (3.9E-2)	**1.12E0** (7.9E-1)	1.87E0 * (1.4E-2)	1.90E0 * (6.1E-1)
n = 1000						
LSMOP1	2	2.94E0 * (2.6E-1)	6.26E-1 * (9.0E-2)	2.06E0 * (4.7E-1)	2.71E-1 * (1.5E-2)	**1.74E-1** (1.1E-1)
LSMOP2	2	3.42E-2 * (4.1E-4)	1.36E-2 * (1.7E-3)	3.67E-2 * (4.5E-4)	1.91E-2 * (1.0E-3)	**1.03E-2** (3.0E-3)
LSMOP3	2	2.10E1 * (1.7E0)	1.57E0 * (5.7E-3)	1.26E1 * (5.6E0)	**1.47E0** (2.1E-1)	1.57E0 * (3.7E-3)
LSMOP4	2	4.84E-2 * (1.1E-3)	3.24E-2 * (2.5E-3)	6.21E-2 * (3.2E-3)	5.86E-2 * (1.6E-3)	**2.37E-2** (2.5E-3)
LSMOP5	2	9.01E0 * (6.9E-1)	7.40E-1 (4.8E-2)	4.07E0 * (7.0E-1)	**7.34E-1** (6.2E-3)	7.41E-1 (4.7E-2)
LSMOP6	2	7.73E-1 * (4.8E-4)	6.71E-1 * (1.5E-3)	4.46E-1 * (3.2E-1)	7.47E-1 * (1.2E-4)	**3.49E-1** (1.8E-1)
LSMOP7	2	6.62E3 * (2.4E3)	1.51E0 * (2.2E-3)	2.30E2 * (4.5E2)	1.51E0 (6.2E-4)	**1.51E0** (2.8E-3)
LSMOP8	2	3.77E0 * (3.6E-1)	7.42E-1 * (3.2E-5)	2.99E0 * (4.1E-1)	4.78E-1 * (5.9E-2)	**2.14E-1** (1.8E-1)
LSMOP9	2	1.33E0 * (1.6E-1)	8.08E-1 * (1.1E-3)	2.51E0 * (9.2E-1)	**8.94E-2** (5.7E-1)	4.90E-1 (7.5E-1)
LSMOP1	3	2.91E0 * (3.4E-1)	5.94E-1 * (3.0E-2)	3.87E0 * (8.9E-1)	4.32E-1 * (7.3E-2)	**3.24E-1** (1.4E-1)
LSMOP2	3	5.18E-2 (1.3E-4)	5.22E-2 * (4.6E-4)	5.20E-2 * (1.1E-4)	5.46E-2 * (1.2E-3)	**5.16E-2** (5.7E-4)
LSMOP3	3	1.06E1 * (9.1E-1)	8.60E-1 * (8.7E-4)	6.96E0 * (2.1E0)	**8.60E-1** (8.4E-6)	8.60E-1 * (2.6E-8)
LSMOP4	3	9.89E-2 * (1.6E-3)	1.05E-1 * (2.9E-3)	1.08E-1 * (1.7E-3)	1.19E-1 * (3.8E-3)	**9.31E-2** (7.0E-3)
LSMOP5	3	7.55E0 * (5.7E-1)	5.41E-1 * (4.2E-4)	5.11E0 * (7.5E-1)	**3.98E-1** (1.7E-1)	4.51E-1 (7.3E-2)
LSMOP6	3	2.25E3 * (9.7E2)	**1.31E0** (3.1E-3)	5.16E0 * (1.3E0)	1.67E0 * (2.0E-3)	1.31E0 (9.8E-2)
LSMOP7	3	1.09E0 * (2.7E-3)	8.54E-1 * (3.8E-3)	**7.60E-1** (1.8E-2)	9.47E-1 * (6.0E-4)	8.47E-1 * (3.3E-3)
LSMOP8	3	7.68E-1 * (4.5E-2)	3.59E-1 * (7.5E-2)	5.52E-1 * (5.0E-4)	2.12E-1 (3.4E-2)	**1.86E-1** (7.8E-2)
LSMOP9	3	1.26E1 * (1.9E0)	1.14E0 * (1.1E-2)	**9.90E-1** (2.2E-1)	1.16E0 * (2.7E-2)	1.14E0 * (1.7E-2)
LSMOP1	4	5.67E0 * (7.4E-1)	7.50E-1 * (4.3E-2)	4.51E0 * (9.5E-1)	6.37E-1 (4.7E-2)	**5.98E-1** (1.7E-1)
LSMOP2	4	1.18E-1 * (2.4E-4)	1.17E-1 * (1.3E-3)	1.18E-1 * (2.1E-4)	1.18E-1 * (2.7E-3)	**1.16E-1** (1.2E-3)
LSMOP3	4	2.02E1 * (2.8E0)	1.81E0 * (2.4E-3)	8.82E0 * (5.9E0)	**1.07E0** (4.6E-2)	1.81E0 * (2.4E-2)
LSMOP4	4	**1.45E-1** (9.4E-4)	1.53E-1 * (1.8E-3)	1.52E-1 * (1.9E-3)	1.61E-1 * (6.7E-3)	1.48E-1 * (4.2E-3)
LSMOP5	4	1.06E1 * (1.2E0)	4.57E-1 (9.7E-4)	5.20E0 * (1.3E0)	1.04E0 * (2.2E-4)	**4.57E-1** (1.3E-2)
LSMOP6	4	1.11E0 * (1.1E-3)	8.97E-1 * (1.7E-3)	**7.72E-1** (5.9E-3)	1.05E0 * (1.2E-3)	8.97E-1 * (7.3E-3)
LSMOP7	4	4.61E3 * (1.9E3)	**1.23E0** (2.1E-3)	7.22E0 * (3.1E0)	1.78E0 * (7.6E-1)	1.24E0 * (1.1E-1)
LSMOP8	4	4.33E0 * (6.4E-1)	4.57E-1 * (1.5E-3)	7.21E-1 * (8.5E-2)	4.57E-1 (6.3E-1)	**4.54E-1** (2.4E-2)
LSMOP9	4	2.28E1 * (3.8E0)	1.47E0 * (2.7E-2)	**7.01E-1** (5.4E-2)	1.46E0 * (1.0E-2)	1.74E0 * (4.5E-1)
LSMOP1	5	4.78E0 * (8.7E-1)	8.97E-1 * (2.3E-2)	3.63E0 * (1.4E0)	**6.44E-1** (1.8E-1)	8.33E-1 * (1.7E-1)
LSMOP2	5	1.53E-1 * (1.6E-4)	1.53E-1 * (1.2E-3)	1.53E-1 * (2.7E-4)	1.55E-1 (1.7E-2)	**1.52E-1** (1.3E-3)
LSMOP3	5	1.29E1 * (2.7E0)	9.64E-1 * (8.5E-2)	3.21E0 * (3.7E0)	**9.58E-1** (8.1E-9)	9.58E-1 * (—)
LSMOP4	5	2.24E-1 * (1.7E-3)	2.28E-1 * (1.7E-3)	2.20E-1 * (4.2E-3)	**2.09E-1** (7.0E-3)	2.22E-1 * (5.6E-3)
LSMOP5	5	1.00E1 * (1.0E0)	4.27E-1 * (3.6E-2)	2.60E0 * (1.3E0)	**3.72E-1** (1.3E-1)	4.29E-1 (6.7E-2)
LSMOP6	5	3.46E2 * (3.6E2)	**1.17E0** (2.3E-2)	3.33E0 * (2.2E0)	1.95E0 * (2.5E-1)	1.26E0 * (4.0E-1)
LSMOP7	5	1.84E0 * (3.3E-2)	1.02E0 (1.1E-1)	1.42E0 * (2.4E-1)	1.11E0 (6.6E-4)	**1.02E0** (1.4E-1)
LSMOP8	5	1.15E0 * (2.3E-3)	3.38E-1 (1.0E-2)	7.85E-1 * (1.8E-2)	**3.29E-1** (2.3E-2)	3.32E-1 (1.1E-2)
LSMOP9	5	4.61E1 * (5.9E0)	1.84E0 * (4.1E-2)	**1.19E0** (7.0E-1)	1.88E0 * (2.5E-2)	1.86E0 * (3.4E-1)

Table B.3: Performance comparison using the IGD indicator on the UF benchmarks using 100,000 function evaluations.

		SMPSO	WOF-SMPSO	NSGA-II	WOF-NSGA-II	WOF-Randomised
n = 40						
UF1	2	1.19E-1 * (3.6E-2)	9.14E-2 (2.2E-2)	1.10E-1 * (4.6E-2)	1.01E-1 * (2.6E-2)	**9.04E-2** (2.2E-2)
UF2	2	6.23E-2 * (6.0E-3)	5.41E-2 * (5.7E-3)	4.32E-2 (1.7E-2)	**4.11E-2** (7.4E-3)	4.21E-2 (8.2E-3)
UF3	2	2.49E-1 * (1.2E-1)	**1.33E-1** (3.6E-2)	2.12E-1 * (6.4E-2)	1.69E-1 * (2.0E-2)	1.68E-1 * (4.0E-2)
UF4	2	9.83E-2 * (1.8E-2)	5.19E-2 * (8.4E-3)	4.91E-2 * (2.0E-3)	**4.72E-2** (1.1E-3)	4.88E-2 (5.6E-3)
UF5	2	2.40E0 * (1.5E0)	6.24E-1 * (1.6E-1)	**2.80E-1** (9.2E-2)	3.86E-1 * (2.0E-1)	3.48E-1 (2.0E-1)
UF6	2	3.84E-1 * (1.0E-1)	3.22E-1 * (1.7E-1)	1.27E-1 (1.5E-1)	**1.14E-1** (1.8E-1)	1.27E-1 (2.4E-1)
UF7	2	5.65E-2 * (2.3E-1)	**4.68E-2** (9.4E-3)	6.64E-2 * (2.4E-1)	5.56E-2 * (1.4E-2)	5.08E-2 (1.6E-2)
UF8	3	4.22E-1 * (1.1E-1)	2.52E-1 (5.0E-2)	3.10E-1 * (4.4E-2)	**2.24E-1** (1.2E-1)	5.28E-1 * (5.4E-2)
UF9	3	5.82E-1 * (1.2E-1)	4.81E-1 * (8.9E-2)	3.41E-1 (9.8E-2)	**3.06E-1** (1.4E-1)	3.13E-1 (2.3E-1)
UF10	3	2.65E0 * (1.1E0)	1.28E0 * (2.9E-1)	4.76E-1 (1.1E-1)	**4.30E-1** (1.4E-1)	5.20E-1 * (3.9E-2)
n = 1000						
UF1	2	1.38E0 * (1.7E-2)	2.72E-1 * (5.9E-3)	2.85E-1 * (9.0E-2)	**1.57E-1** (3.7E-2)	1.96E-1 * (1.6E-2)
UF2	2	1.68E-1 * (7.5E-3)	**8.50E-2** (4.1E-4)	2.32E-1 * (1.1E-2)	1.13E-1 * (1.3E-2)	8.69E-2 * (3.2E-3)
UF3	2	3.26E-1 * (7.9E-3)	**2.54E-2** (3.8E-3)	2.74E-1 * (8.3E-3)	1.38E-1 * (1.0E-2)	3.50E-2 * (1.0E-2)
UF4	2	1.36E-1 * (6.5E-4)	**5.49E-2** (9.0E-3)	1.61E-1 * (6.5E-3)	7.40E-2 * (5.0E-3)	7.08E-2 * (1.8E-2)
UF5	2	5.56E0 * (8.7E-2)	2.81E0 * (1.1E-1)	2.05E0 * (4.5E-1)	**1.20E0** (1.9E-1)	1.61E0 * (5.3E-1)
UF6	2	5.54E0 * (2.9E-1)	1.02E0 * (5.9E-2)	8.93E-1 * (2.0E-1)	4.74E-1 (2.6E-1)	**4.64E-1** (3.1E-1)
UF7	2	1.42E0 * (2.9E-2)	2.77E-1 * (7.4E-3)	3.21E-1 * (5.9E-2)	**1.50E-1** (2.6E-2)	1.67E-1 * (2.0E-2)
UF8	3	5.53E-1 * (4.4E-2)	**3.56E-1** (4.4E-3)	9.55E-1 * (1.3E-1)	4.67E-1 * (2.9E-2)	5.42E-1 * (2.1E-3)
UF9	3	8.02E-1 * (2.9E-2)	**5.70E-1** (7.2E-3)	7.72E-1 * (5.5E-2)	6.52E-1 * (8.4E-2)	6.20E-1 * (9.6E-2)
UF10	3	4.88E0 * (4.9E-1)	2.08E0 * (7.6E-1)	3.61E0 * (1.5E0)	1.90E0 * (4.1E-1)	**8.97E-1** (2.9E-1)

Table B.4: Performance comparison using the IGD indicator on the UF benchmarks using 100,000 function evaluations.

		NSGA-III	WOF-NSGA-III	MOEA/D	WOF-MOEA/D	WOF-Randomised
n = 40						
UF1	2	1.09E-1 * (1.9E-2)	9.71E-2 * (7.6E-3)	2.29E-1 * (2.0E-1)	**6.98E-2** (1.2E-2)	9.04E-2 * (2.2E-2)
UF2	2	**4.15E-2** (1.5E-2)	4.50E-2 (8.9E-3)	1.82E-1 * (1.2E-1)	4.77E-2 * (1.1E-2)	4.21E-2 (8.2E-3)
UF3	2	2.15E-1 * (7.2E-2)	2.04E-1 * (1.9E-2)	3.20E-1 * (2.5E-2)	2.24E-1 * (4.3E-2)	**1.68E-1** (4.0E-2)
UF4	2	5.05E-2 * (2.2E-3)	**4.69E-2** (1.0E-3)	8.31E-2 * (6.4E-3)	8.27E-2 * (5.4E-3)	4.88E-2 (5.6E-3)
UF5	2	**2.70E-1** (7.1E-2)	2.92E-1 (1.4E-1)	4.69E-1 * (1.1E-1)	5.16E-1 * (1.7E-1)	3.48E-1 * (2.0E-1)
UF6	2	**1.09E-1** (3.7E-2)	1.18E-1 (1.0E-1)	4.09E-1 * (1.3E-1)	1.88E-1 * (2.8E-1)	1.27E-1 (2.4E-1)
UF7	2	5.25E-2 (2.1E-1)	5.45E-2 (1.4E-2)	4.17E-1 * (2.1E-1)	3.43E-1 (3.0E-1)	**5.08E-2** (1.6E-2)
UF8	3	5.31E-1 (1.9E-2)	5.28E-1 (3.8E-1)	2.72E-1 (5.9E-1)	2.91E-1 (1.6E-2)	5.28E-1 (5.4E-2)
UF9	3	4.05E-1 * (1.9E-1)	**1.69E-1** (1.9E-1)	3.68E-1 * (4.6E-2)	3.47E-1 * (1.1E-1)	3.13E-1 (2.3E-1)
UF10	3	**4.29E-1** (1.4E-1)	5.07E-1 (2.1E-1)	6.99E-1 * (1.4E-1)	7.12E-1 * (1.3E-1)	5.20E-1 * (3.9E-2)
n = 1000						
UF1	2	2.80E-1 * (4.8E-2)	**1.79E-1** (3.7E-2)	8.99E-1 * (2.9E-1)	2.67E-1 * (1.4E-2)	1.96E-1 * (1.6E-2)
UF2	2	2.23E-1 * (1.1E-2)	1.16E-1 * (1.6E-2)	4.54E-1 * (6.2E-2)	9.84E-2 * (4.6E-3)	**8.69E-2** (3.2E-3)
UF3	2	2.83E-1 * (1.1E-2)	1.38E-1 * (1.5E-2)	3.82E-1 * (2.1E-2)	2.02E-1 * (1.8E-2)	**3.50E-2** (1.0E-2)
UF4	2	1.61E-1 * (2.9E-3)	7.31E-2 (3.7E-3)	2.00E-1 * (5.3E-3)	1.04E-1 * (6.7E-3)	**7.08E-2** (1.8E-2)
UF5	2	1.84E0 * (5.0E-1)	**1.35E0** (2.6E-1)	3.12E0 * (2.0E-1)	1.41E0 (1.6E-1)	1.61E0 (5.3E-1)
UF6	2	8.73E-1 * (1.7E-1)	5.83E-1 (3.4E-1)	2.42E0 * (3.7E-1)	7.74E-1 * (1.1E-1)	**4.64E-1** (3.1E-1)
UF7	2	3.13E-1 * (7.6E-2)	**1.50E-1** (2.8E-2)	9.04E-1 * (2.3E-1)	4.71E-1 * (2.4E-1)	1.67E-1 * (2.0E-2)
UF8	3	8.62E-1 (6.0E-2)	5.58E-1 (6.7E-2)	1.66E0 * (3.0E-1)	**3.31E-1** (6.4E-1)	5.42E-1 (2.1E-3)
UF9	3	8.34E-1 * (2.9E-2)	6.76E-1 (1.5E-1)	7.77E-1 * (3.0E-2)	7.14E-1 * (1.0E-1)	**6.20E-1** (9.6E-2)
UF10	3	3.01E0 * (3.4E-1)	2.16E0 * (1.0E0)	2.57E0 * (5.8E-1)	**8.59E-1** (9.8E-2)	8.97E-1 (2.9E-1)

Table B.5: Performance comparison using the IGD indicator on the WFG benchmarks using 100, 000 function evaluations.

		SMPSO	WOF-SMPSO	NSGA-II	WOF-NSGA-II	WOF-Randomised
n = 40						
WFG1	2	1.22E0 * (2.9E-2)	1.12E0 * (7.3E-2)	**4.98E-2** (2.5E-2)	1.71E-1 * (6.1E-2)	3.49E-1 * (9.5E-2)
WFG2	2	2.49E-1 * (1.8E-2)	2.80E-2 * (7.5E-3)	6.65E-1 * (8.0E-3)	2.87E-2 (1.1E-2)	**2.44E-2** (6.1E-3)
WFG3	2	6.15E-2 * (2.3E-2)	3.00E-2 * (6.4E-3)	2.75E-2 * (8.7E-3)	2.98E-2 * (8.7E-3)	**2.16E-2** (3.7E-3)
WFG4	2	1.01E-1 * (8.9E-3)	7.68E-2 * (3.4E-2)	2.08E-2 * (2.1E-3)	2.24E-2 * (3.4E-3)	**1.85E-2** (2.4E-3)
WFG5	2	9.09E-2 * (3.2E-2)	**6.79E-2** (4.6E-3)	7.16E-2 * (9.3E-4)	7.22E-2 * (7.1E-4)	6.94E-2 (1.2E-4)
WFG6	2	4.75E-2 * (2.4E-3)	**2.28E-2** (9.8E-3)	6.05E-2 * (1.5E-2)	8.32E-2 * (1.2E-3)	4.13E-2 * (4.9E-2)
WFG7	2	2.39E-2 * (1.0E-2)	1.88E-2 * (1.6E-3)	1.71E-2 * (9.5E-4)	1.83E-2 * (1.2E-3)	**1.38E-2** (5.7E-4)
WFG8	2	3.29E-1 * (4.5E-2)	**1.07E-1** (8.7E-2)	2.39E-1 * (1.1E-2)	2.15E-1 * (1.6E-2)	2.09E-1 * (5.9E-2)
WFG9	2	9.62E-2 * (2.0E-2)	**4.27E-2** (1.3E-2)	9.34E-2 * (1.5E-3)	9.15E-2 * (4.6E-2)	8.28E-2 (5.4E-2)
WFG1	3	1.53E0 * (1.4E-2)	1.51E0 * (2.1E-2)	**6.63E-1** (9.0E-2)	9.77E-1 * (7.2E-2)	1.18E0 * (1.0E-1)
WFG2	3	3.10E-1 * (1.3E-1)	2.36E-1 * (1.7E-2)	5.13E-1 * (1.1E-2)	2.24E-1 * (9.0E-3)	**1.66E-1** (4.8E-3)
WFG3	3	2.57E-1 * (9.0E-2)	1.43E-1 * (2.6E-2)	1.30E-1 (3.2E-2)	1.35E-1 (2.7E-2)	**1.20E-1** (3.0E-2)
WFG4	3	4.12E-1 * (3.7E-2)	3.46E-1 * (3.2E-2)	2.89E-1 * (1.3E-2)	2.91E-1 * (1.2E-2)	**2.35E-1** (4.5E-3)
WFG5	3	4.81E-1 * (1.3E-1)	3.04E-1 * (2.5E-2)	2.93E-1 * (1.4E-2)	2.92E-1 * (1.4E-2)	**2.36E-1** (2.5E-3)
WFG6	3	3.44E-1 * (5.1E-2)	3.48E-1 * (6.2E-2)	3.01E-1 * (1.4E-2)	3.13E-1 * (1.9E-2)	**2.41E-1** (4.1E-3)
WFG7	3	4.61E-1 * (1.1E-1)	2.99E-1 * (1.9E-2)	2.84E-1 * (2.1E-2)	3.13E-1 * (1.1E-1)	**2.28E-1** (4.0E-2)
WFG8	3	7.03E-1 * (7.4E-2)	5.60E-1 * (3.4E-2)	4.56E-1 * (2.4E-2)	5.02E-1 * (3.6E-2)	**3.53E-1** (6.4E-2)
WFG9	3	4.00E-1 * (5.0E-2)	3.38E-1 * (3.1E-2)	3.26E-1 * (2.1E-2)	3.18E-1 * (1.0E-2)	**2.46E-1** (6.3E-3)
n = 1000						
WFG1	2	1.31E0 * (9.7E-3)	**1.20E0** (5.1E-2)	1.71E0 * (3.0E-2)	1.27E0 * (1.2E-2)	1.22E0 * (2.7E-2)
WFG2	2	8.99E-1 * (5.4E-1)	**7.25E-2** (2.1E-2)	8.95E-1 * (2.0E-2)	2.35E-1 * (7.5E-2)	1.40E-1 * (2.9E-2)
WFG3	2	8.97E-1 * (4.5E-2)	**8.94E-2** (1.3E-2)	8.39E-1 * (4.6E-2)	2.12E-1 * (2.9E-2)	1.25E-1 * (3.1E-2)
WFG4	2	4.90E-1 * (1.4E-2)	**1.11E-1** (8.2E-3)	9.78E-1 * (6.6E-2)	1.77E-1 * (2.3E-2)	1.27E-1 * (1.8E-2)
WFG5	2	6.03E-1 * (1.1E-2)	**6.78E-2** (4.8E-3)	9.74E-1 * (5.6E-2)	9.28E-2 * (6.6E-2)	6.97E-2 (3.3E-3)
WFG6	2	2.86E-1 * (2.7E-2)	1.69E-2 * (1.4E-3)	1.03E0 * (7.5E-2)	2.34E-2 * (5.4E-3)	**1.35E-2** (3.6E-3)
WFG7	2	9.40E-1 * (3.2E-2)	**7.70E-2** (1.1E-2)	9.24E-1 * (2.7E-2)	2.41E-1 * (2.7E-2)	1.39E-1 * (2.9E-2)
WFG8	2	1.29E0 * (7.8E-2)	**1.03E-1** (1.0E-1)	1.09E0 * (4.1E-2)	3.59E-1 * (4.3E-2)	2.81E-1 * (7.8E-2)
WFG9	2	4.54E-1 * (7.0E-2)	**3.83E-2** (7.2E-3)	9.57E-1 * (7.4E-2)	1.04E-1 * (5.2E-2)	5.05E-2 * (2.9E-2)
WFG1	3	1.75E0 * (4.5E-2)	1.51E0 * (1.9E-2)	1.82E0 * (5.5E-2)	1.55E0 * (3.6E-2)	**1.50E0** (1.8E-2)
WFG2	3	1.76E0 * (2.6E-2)	**2.43E-1** (1.6E-2)	1.78E0 * (1.5E-2)	5.15E-1 * (1.1E-1)	4.04E-1 * (1.0E-1)
WFG3	3	8.94E-1 * (1.0E-1)	**1.13E-1** (3.0E-2)	8.64E-1 * (6.4E-2)	3.97E-1 * (1.5E-1)	3.98E-1 * (1.4E-1)
WFG4	3	1.37E0 * (9.3E-2)	**3.70E-1** (2.4E-2)	1.86E0 * (7.4E-2)	6.48E-1 * (1.2E-1)	4.22E-1 * (9.3E-2)
WFG5	3	1.57E0 * (3.8E-2)	4.28E-1 (1.4E-1)	1.66E0 * (6.9E-2)	6.22E-1 * (1.0E-1)	**4.01E-1** (1.1E-1)
WFG6	3	1.18E0 * (5.4E-2)	3.78E-1 * (5.9E-2)	1.59E0 * (5.1E-2)	4.03E-1 * (8.8E-2)	**2.46E-1** (1.8E-2)
WFG7	3	1.60E0 * (5.7E-2)	**3.58E-1** (4.1E-2)	1.78E0 * (5.0E-2)	7.19E-1 * (3.7E-2)	5.29E-1 * (5.5E-2)
WFG8	3	1.78E0 * (9.0E-2)	**6.19E-1** (7.5E-2)	1.83E0 * (5.1E-2)	1.07E0 * (1.2E-1)	7.72E-1 * (1.3E-1)
WFG9	3	1.21E0 * (1.0E-1)	**3.68E-1** (3.6E-2)	1.56E0 * (9.9E-2)	7.51E-1 * (1.2E-1)	4.31E-1 * (1.1E-1)

Table B.6: Performance comparison using the IGD indicator on the WFG benchmarks using 100,000 function evaluations.

		NSGA-III	WOF-NSGA-III	MOEA/D	WOF-MOEA/D	WOF-Randomised
n = 40						
WFG1	2	**1.29E-1** (3.2E-2)	3.34E-1 * (7.1E-2)	6.14E-1 * (7.5E-2)	1.01E0 * (6.5E-2)	3.49E-1 * (9.5E-2)
WFG2	2	6.64E-1 * (5.7E-3)	3.13E-2 * (1.2E-2)	7.82E-1 * (1.2E-2)	9.68E-2 * (9.2E-3)	**2.44E-2** (6.1E-3)
WFG3	2	2.97E-2 * (1.2E-2)	2.86E-2 * (5.6E-3)	9.62E-2 * (6.3E-2)	4.81E-2 * (4.8E-3)	**2.16E-2** (3.7E-3)
WFG4	2	**1.65E-2** (2.0E-3)	1.96E-2 * (2.3E-3)	3.59E-2 * (6.4E-3)	7.19E-2 * (8.5E-3)	1.85E-2 * (2.4E-3)
WFG5	2	7.04E-2 * (5.6E-4)	7.03E-2 * (9.0E-4)	7.80E-2 * (6.6E-3)	7.28E-2 * (2.4E-3)	**6.94E-2** (1.2E-4)
WFG6	2	5.92E-2 (1.3E-2)	8.10E-2 * (2.5E-4)	8.24E-2 * (2.0E-2)	8.54E-2 * (5.4E-3)	**4.13E-2** (4.9E-2)
WFG7	2	1.59E-2 * (1.2E-2)	1.48E-2 * (1.2E-3)	3.03E-2 * (2.0E-2)	4.67E-2 * (1.5E-2)	**1.38E-2** (5.7E-4)
WFG8	2	2.56E-1 * (2.2E-2)	2.19E-1 * (1.6E-2)	2.45E-1 * (3.3E-2)	**1.67E-1** (1.8E-2)	2.09E-1 * (5.9E-2)
WFG9	2	9.23E-2 * (2.1E-3)	7.69E-2 (3.4E-2)	**7.56E-2** (2.4E-2)	9.31E-2 * (2.9E-3)	8.28E-2 (5.4E-2)
WFG1	3	8.38E-1 (1.2E-1)	1.20E0 * (8.8E-2)	**7.82E-1** (1.0E-1)	1.38E0 * (7.5E-2)	1.18E0 * (1.0E-1)
WFG2	3	4.90E-1 * (6.7E-3)	1.69E-1 * (3.7E-3)	6.25E-1 * (8.4E-1)	3.02E-1 * (3.0E-2)	**1.66E-1** (4.8E-3)
WFG3	3	**8.27E-2** (2.1E-2)	1.32E-1 * (3.4E-2)	4.12E-1 * (2.9E-1)	1.74E-1 * (3.0E-2)	1.20E-1 * (3.0E-2)
WFG4	3	**2.31E-1** (2.7E-3)	2.38E-1 * (5.2E-3)	2.56E-1 * (5.5E-3)	2.97E-1 * (2.5E-2)	2.35E-1 * (4.5E-3)
WFG5	3	2.37E-1 * (2.5E-3)	2.36E-1 (4.5E-3)	2.50E-1 * (3.7E-3)	2.91E-1 * (1.0E-2)	**2.36E-1** (2.5E-3)
WFG6	3	**2.36E-1** (6.7E-3)	2.40E-1 * (3.6E-3)	2.66E-1 * (2.0E-2)	3.28E-1 * (1.9E-2)	2.41E-1 * (4.1E-3)
WFG7	3	2.30E-1 (1.0E-2)	2.36E-1 (2.2E-2)	2.73E-1 * (4.4E-2)	2.95E-1 * (3.8E-2)	**2.28E-1** (4.0E-2)
WFG8	3	3.53E-1 (1.2E-2)	3.76E-1 (2.0E-2)	3.60E-1 (2.7E-2)	4.93E-1 * (8.8E-2)	**3.53E-1** (6.4E-2)
WFG9	3	**2.45E-1** (5.0E-3)	2.48E-1 * (5.4E-3)	2.86E-1 * (2.2E-2)	3.15E-1 * (2.1E-2)	2.46E-1 (6.3E-3)
n = 1000						
WFG1	2	1.65E0 * (3.4E-2)	1.26E0 * (9.9E-3)	1.89E0 * (7.2E-2)	1.26E0 * (8.7E-3)	**1.22E0** (2.7E-2)
WFG2	2	8.88E-1 * (2.9E-2)	2.45E-1 * (7.6E-2)	1.47E0 * (4.4E-1)	2.17E-1 * (3.6E-2)	**1.40E-1** (2.9E-2)
WFG3	2	7.87E-1 * (3.6E-2)	2.10E-1 * (3.5E-2)	1.05E0 * (3.6E-2)	1.71E-1 * (5.1E-2)	**1.25E-1** (3.1E-2)
WFG4	2	9.13E-1 * (5.2E-2)	1.66E-1 * (2.7E-2)	1.05E0 * (7.5E-2)	1.64E-1 * (3.1E-2)	**1.27E-1** (1.8E-2)
WFG5	2	8.92E-1 * (5.8E-2)	1.25E-1 * (6.3E-2)	1.18E0 * (6.3E-2)	1.01E-1 * (1.6E-2)	**6.97E-2** (3.3E-3)
WFG6	2	9.14E-1 * (5.9E-2)	1.62E-2 * (3.7E-3)	2.47E0 * (1.9E-2)	7.35E-2 * (3.5E-2)	**1.35E-2** (3.6E-3)
WFG7	2	8.87E-1 * (3.9E-2)	2.38E-1 * (3.3E-2)	1.01E0 * (5.9E-2)	2.23E-1 * (9.9E-2)	**1.39E-1** (2.9E-2)
WFG8	2	1.05E0 * (3.8E-2)	3.45E-1 * (3.6E-2)	1.31E0 * (5.0E-2)	3.77E-1 * (5.6E-2)	**2.81E-1** (7.8E-2)
WFG9	2	9.14E-1 * (6.0E-2)	1.18E-1 * (7.1E-2)	2.19E0 * (4.4E-2)	8.67E-2 * (3.6E-2)	**5.05E-2** (2.9E-2)
WFG1	3	1.85E0 * (6.1E-2)	1.55E0 * (4.6E-2)	1.89E0 * (4.6E-2)	1.62E0 * (5.5E-2)	**1.50E0** (1.8E-2)
WFG2	3	1.76E0 * (1.8E-2)	4.88E-1 * (9.2E-2)	1.84E0 * (2.4E-2)	7.63E-1 * (1.0E-1)	**4.04E-1** (1.0E-1)
WFG3	3	1.17E0 * (1.8E-1)	5.57E-1 * (1.2E-1)	1.33E0 * (8.1E-2)	**3.39E-1** (9.8E-2)	3.98E-1 (1.4E-1)
WFG4	3	2.84E0 * (9.5E-2)	4.60E-1 * (5.2E-2)	1.36E0 * (1.1E-1)	5.25E-1 * (7.4E-2)	**4.22E-1** (9.3E-2)
WFG5	3	1.64E0 * (9.5E-2)	6.73E-1 * (7.1E-1)	1.29E0 * (6.5E-2)	4.63E-1 * (4.0E-1)	**4.01E-1** (1.1E-1)
WFG6	3	1.45E0 * (4.8E-2)	**2.44E-1** (2.3E-2)	1.82E0 * (2.2E-1)	3.85E-1 * (6.4E-2)	2.46E-1 (1.8E-2)
WFG7	3	2.55E0 * (7.9E-1)	5.98E-1 * (6.4E-2)	1.30E0 * (1.1E-1)	6.61E-1 * (8.3E-2)	**5.29E-1** (5.5E-2)
WFG8	3	1.90E0 * (8.7E-2)	8.50E-1 * (1.1E-1)	1.49E0 * (1.1E-1)	1.35E0 * (2.1E-1)	**7.72E-1** (1.3E-1)
WFG9	3	1.39E0 * (9.2E-2)	5.29E-1 * (1.6E-1)	1.51E0 * (1.9E-1)	5.88E-1 * (7.6E-2)	**4.31E-1** (1.1E-1)

Table B.7: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100, 000 function evaluations.

		SMPSO	WOF-SMPSO	NSGA-II	WOF-NSGA-II	WOF-Randomised
n = 40						
DTLZ1	2	6.00E1 * (8.0E1)	**2.25E-3** (1.1E-4)	1.11E0 * (7.7E-1)	6.22E0 * (4.4E0)	3.33E0 * (1.1E1)
DTLZ2	2	5.10E-3 * (2.3E-4)	5.11E-3 * (2.9E-4)	5.08E-3 * (2.8E-4)	5.05E-3 * (2.4E-4)	**3.96E-3** (3.4E-7)
DTLZ3	2	1.75E2 * (2.0E2)	**4.98E-3** (6.0E-4)	2.20E0 * (2.2E0)	2.00E1 * (7.5E0)	1.03E1 * (1.9E1)
DTLZ4	2	5.49E-3 * (7.3E-1)	5.31E-3 * (7.3E-1)	5.09E-3 * (2.0E-4)	5.19E-3 * (4.4E-4)	**3.96E-3** (2.8E-6)
DTLZ5	2	5.21E-3 * (3.2E-4)	5.07E-3 * (2.0E-4)	5.15E-3 * (2.1E-4)	5.03E-3 * (2.0E-4)	**3.96E-3** (1.9E-7)
DTLZ6	2	5.16E-3 * (2.1E-4)	5.15E-3 * (3.3E-4)	7.36E-3 * (5.6E-3)	5.62E-3 * (3.7E-4)	**3.96E-3** (5.3E-8)
DTLZ7	2	5.46E-3 (4.3E-1)	4.42E-1 * (4.3E-1)	**5.30E-3** (3.0E-4)	5.44E-3 (4.3E-1)	4.42E-1 (4.3E-1)
DTLZ1	3	1.57E1 * (1.9E1)	**7.27E-2** (3.8E-1)	8.93E0 * (3.5E0)	2.77E1 * (1.2E1)	1.11E1 * (2.4E1)
DTLZ2	3	9.55E-2 * (8.2E-3)	8.41E-2 * (6.4E-3)	7.32E-2 * (2.6E-3)	7.32E-2 * (3.4E-3)	**5.44E-2** (6.1E-6)
DTLZ3	3	3.26E1 * (5.6E1)	**1.12E-1** (2.7E-1)	1.65E1 * (6.0E0)	6.04E1 * (4.2E1)	2.41E1 * (3.3E1)
DTLZ4	3	3.70E-1 * (2.8E-1)	2.46E-1 * (1.8E-1)	7.24E-2 * (4.7E-3)	7.10E-2 * (3.2E-3)	**5.44E-2** (8.3E-6)
DTLZ5	3	6.59E-3 * (6.2E-4)	**5.80E-3** (2.6E-4)	6.45E-3 * (4.3E-4)	6.48E-3 * (5.1E-4)	1.37E-2 * (2.9E-3)
DTLZ6	3	5.98E-3 (1.1E0)	**5.98E-3** (5.0E-4)	6.75E-3 * (6.4E-3)	6.46E-3 * (7.0E-4)	2.29E-2 * (3.5E-3)
DTLZ7	3	1.05E-1 * (1.2E-2)	1.04E-1 * (2.5E-2)	**8.19E-2** (4.6E-3)	8.54E-2 * (7.1E-1)	8.01E-1 * (7.2E-1)
DTLZ1	4	1.88E1 (2.7E1)	2.06E1 * (1.9E1)	3.13E1 * (9.1E0)	1.60E2 * (4.9E1)	**1.02E1** (1.4E1)
DTLZ2	4	7.69E-1 * (2.1E-1)	6.12E-1 * (1.2E-1)	1.67E-1 * (7.0E-3)	1.64E-1 * (8.1E-3)	**1.40E-1** (2.0E-5)
DTLZ3	4	5.93E1 * (7.4E1)	**2.29E1** (4.5E1)	5.76E1 * (2.4E1)	2.99E2 * (1.4E2)	3.25E1 (7.3E1)
DTLZ4	4	4.05E-1 * (8.3E-2)	4.10E-1 * (6.6E-2)	1.64E-1 * (6.6E-3)	1.63E-1 * (5.7E-3)	**1.40E-1** (7.5E-5)
DTLZ5	4	2.77E-1 * (8.9E-2)	2.71E-1 * (1.8E-1)	1.23E-1 * (2.5E-2)	1.52E-1 * (4.1E-2)	**8.13E-2** (3.5E-2)
DTLZ6	4	1.56E1 * (4.1E0)	**1.50E-1** (2.3E-1)	1.47E1 * (1.7E0)	1.02E1 * (2.2E0)	1.83E-1 (6.1E-2)
DTLZ7	4	3.49E-1 * (6.6E-2)	3.26E-1 * (2.6E-2)	**2.36E-1** (1.6E-2)	2.41E-1 (8.9E-1)	3.83E-1 (9.1E-1)
DTLZ1	5	1.42E2 * (2.4E2)	**1.95E1** (3.1E1)	5.42E1 * (1.9E1)	2.02E2 * (5.6E1)	2.10E1 (2.6E1)
DTLZ2	5	1.07E0 * (3.2E-1)	8.86E-1 * (1.7E-1)	2.62E-1 * (9.4E-3)	2.71E-1 * (1.3E-2)	**2.12E-1** (1.4E-4)
DTLZ3	5	2.56E2 * (2.0E2)	5.18E1 (4.9E1)	1.95E2 * (6.3E1)	6.24E2 * (1.6E2)	**4.41E1** (8.4E1)
DTLZ4	5	7.86E-1 * (1.5E-1)	7.12E-1 * (1.0E-1)	2.60E-1 * (1.1E-2)	2.73E-1 * (1.3E-2)	**2.12E-1** (1.7E-4)
DTLZ5	5	5.18E-1 * (3.2E-1)	4.61E-1 * (2.7E-1)	1.09E0 * (4.7E-1)	1.47E0 * (6.6E-1)	**1.45E-1** (4.0E-2)
DTLZ6	5	1.72E1 * (4.9E0)	3.29E0 * (5.3E0)	2.35E1 * (1.7E0)	1.96E1 * (3.8E0)	**4.52E-1** (1.9E-1)
DTLZ7	5	6.94E-1 * (1.9E-1)	5.25E-1 * (2.5E-2)	4.04E-1 (2.0E-2)	4.01E-1 (1.9E-2)	**3.97E-1** (5.8E-2)
n = 1000						
DTLZ1	2	1.86E3 * (1.6E3)	**2.32E-3** (5.1E-1)	4.37E3 * (1.6E2)	5.15E3 * (1.7E3)	7.98E2 * (9.7E2)
DTLZ2	2	3.15E0 * (5.1E-1)	5.89E-3 * (8.1E-4)	1.88E0 * (2.0E-1)	7.18E-1 * (6.3E-1)	**4.56E-3** (1.0E-3)
DTLZ3	2	5.32E3 * (4.4E3)	**1.14E-2** (2.1E0)	1.15E4 * (4.1E2)	1.38E4 * (4.5E3)	1.51E3 * (2.7E3)
DTLZ4	2	3.66E0 * (1.3E0)	8.93E-3 (4.3E-3)	2.38E0 * (4.6E-1)	6.91E-1 * (4.0E-1)	**8.13E-3** (4.8E-3)
DTLZ5	2	3.14E0 * (4.3E-1)	5.87E-3 (5.8E-4)	1.95E0 * (2.6E-1)	7.07E-1 * (5.2E-1)	**4.77E-3** (2.3E-3)
DTLZ6	2	4.28E2 * (2.3E1)	5.18E-3 * (3.7E-4)	5.83E2 * (1.2E1)	7.65E1 * (4.7E1)	**3.96E-3** (6.2E-8)
DTLZ7	2	4.84E0 * (4.0E-1)	4.42E-1 * (2.5E-4)	1.69E0 * (1.5E-1)	4.42E-1 * (1.1E-4)	**4.42E-1** (4.3E-1)
DTLZ1	3	7.30E2 * (5.5E2)	**2.40E0** (4.1E1)	7.46E3 * (3.2E2)	5.40E3 * (1.8E3)	1.02E3 * (1.1E3)
DTLZ2	3	4.92E0 * (1.2E0)	1.62E-1 * (9.0E-2)	8.01E0 * (3.9E-1)	1.85E0 * (1.3E0)	**5.89E-2** (4.1E-3)
DTLZ3	3	1.20E3 * (1.9E3)	**5.62E-1** (1.0E1)	1.79E4 * (6.4E2)	1.74E4 * (4.4E3)	3.45E3 * (2.9E3)
DTLZ4	3	1.95E0 * (1.5E0)	2.78E-1 * (2.2E-1)	9.66E0 * (1.3E0)	1.82E0 * (1.5E0)	**6.55E-2** (1.0E-2)
DTLZ5	3	5.74E0 * (1.2E0)	5.14E-2 * (5.1E-2)	9.90E0 * (9.4E-1)	2.25E0 * (2.1E0)	**2.50E-2** (1.2E-2)
DTLZ6	3	4.86E2 * (2.2E1)	**5.89E-3** (2.5E-4)	7.52E2 * (7.0E0)	2.33E2 * (9.5E1)	2.43E-2 * (3.3E-3)
DTLZ7	3	8.05E0 * (4.7E-1)	1.53E0 * (1.1E0)	2.75E0 * (1.8E-1)	**7.99E-1** (7.1E-1)	8.03E-1 * (4.2E-1)
DTLZ1	4	7.55E2 (1.4E3)	**5.24E2** (8.9E2)	1.09E4 * (7.0E2)	6.32E3 * (3.2E3)	7.55E2 (1.0E3)
DTLZ2	4	1.84E1 * (5.0E0)	1.06E1 * (4.1E0)	2.27E1 * (2.0E0)	5.22E0 * (2.3E0)	**2.26E-1** (2.1E-1)
DTLZ3	4	1.64E3 (1.0E3)	**1.02E3** (1.7E3)	3.63E4 * (1.3E3)	2.08E4 * (6.4E3)	4.10E3 * (3.9E3)
DTLZ4	4	6.05E0 * (2.7E0)	3.78E0 * (1.6E0)	2.51E1 * (2.1E0)	8.91E0 * (5.0E0)	**7.17E-1** (9.0E-1)
DTLZ5	4	3.14E1 * (1.8E1)	2.25E1 * (2.0E1)	5.75E1 * (2.2E0)	4.84E1 * (1.7E1)	**2.34E0** (4.5E0)
DTLZ6	4	4.95E2 * (2.2E1)	4.30E0 * (2.2E1)	8.70E2 * (4.2E0)	6.26E2 * (8.2E1)	**3.06E-1** (3.1E-1)
DTLZ7	4	1.16E1 * (6.3E-1)	2.24E0 * (1.9E0)	5.13E0 * (2.0E-1)	**1.11E0** (4.9E-1)	1.13E0 * (4.9E-1)
DTLZ1	5	5.00E3 * (6.5E3)	**9.53E2** (6.4E2)	1.24E4 * (6.5E2)	8.15E3 * (3.2E3)	1.96E3 * (1.3E3)
DTLZ2	5	2.70E1 * (1.5E1)	1.93E1 * (5.7E0)	4.50E1 * (2.6E0)	3.44E1 * (7.2E0)	**6.81E0** (7.0E0)
DTLZ3	5	9.01E3 * (5.0E3)	**1.66E3** (2.5E3)	6.28E4 * (2.6E3)	2.46E4 * (1.1E4)	6.62E3 * (4.4E3)
DTLZ4	5	1.47E1 * (4.4E0)	**8.66E0** (3.6E0)	4.69E1 * (4.4E0)	3.44E1 * (8.4E0)	1.18E1 (5.6E0)
DTLZ5	5	5.26E1 * (2.7E1)	3.12E1 * (1.7E1)	7.25E1 * (2.5E0)	6.70E1 * (1.8E1)	**1.61E1** (2.0E1)
DTLZ6	5	5.02E2 * (2.5E1)	**9.17E1** (1.4E2)	8.80E2 * (3.3E0)	6.48E2 * (8.3E1)	2.28E2 * (1.3E2)
DTLZ7	5	1.52E1 * (5.7E-1)	3.00E0 * (—)	9.30E0 * (6.4E-1)	1.46E0 (1.0E0)	**9.18E-1** (1.0E0)

Table B.8: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100,000 function evaluations.

		NSGA-III	WOF-NSGA-III	MOEA/D	WOF-MOEA/D	WOF-Randomised
n = 40						
DTLZ1	2	2.21E0 (2.1E0)	1.32E1 * (9.3E0)	**1.52E0** (1.7E0)	4.40E0 * (2.2E0)	3.33E0 * (1.1E1)
DTLZ2	2	3.96E-3 * (5.3E-8)	3.96E-3 * (1.1E-6)	**3.96E-3** (6.3E-8)	4.42E-3 * (1.9E-3)	3.96E-3 * (3.4E-7)
DTLZ3	2	4.52E0 (5.2E0)	3.86E1 * (3.1E1)	**4.36E0** (3.9E0)	1.61E1 * (2.0E1)	1.03E1 (1.9E1)
DTLZ4	2	3.96E-3 * (6.2E-7)	3.96E-3 * (1.2E-6)	**3.96E-3** (7.3E-1)	5.87E-3 * (2.4E-3)	3.96E-3 * (2.8E-6)
DTLZ5	2	3.96E-3 * (9.2E-8)	3.96E-3 * (1.5E-6)	**3.96E-3** (6.5E-8)	4.46E-3 * (2.0E-3)	3.96E-3 * (1.9E-7)
DTLZ6	2	3.96E-3 * (4.8E-8)	3.96E-3 * (6.3E-8)	**3.96E-3** (1.6E-8)	6.52E-3 * (7.4E-4)	3.96E-3 * (5.3E-8)
DTLZ7	2	**5.06E-3** (7.8E-5)	5.14E-3 * (4.3E-1)	7.53E-3 * (4.3E-1)	4.46E-1 * (4.0E-4)	4.42E-1 * (4.3E-1)
DTLZ1	3	**5.55E0** (3.0E0)	2.07E1 * (1.0E1)	7.52E0 (3.7E0)	1.08E1 * (1.1E1)	1.11E1 (2.4E1)
DTLZ2	3	5.44E-2 * (1.1E-6)	5.44E-2 * (1.0E-5)	**5.44E-2** (3.7E-7)	5.50E-2 * (2.9E-3)	5.44E-2 * (6.1E-6)
DTLZ3	3	**1.17E1** (7.0E0)	5.18E1 * (3.7E1)	3.95E1 * (1.9E1)	2.52E1 * (3.1E1)	2.41E1 (3.3E1)
DTLZ4	3	**5.44E-2** (7.3E-3)	5.44E-2 * (2.5E-5)	5.41E-1 (8.9E-1)	5.47E-2 * (2.1E-3)	5.44E-2 (8.3E-6)
DTLZ5	3	1.38E-2 (2.7E-3)	**1.31E-2** (1.9E-3)	3.38E-2 * (4.9E-5)	3.16E-2 * (2.0E-3)	1.37E-2 (2.9E-3)
DTLZ6	3	**2.05E-2** (3.0E-3)	2.15E-2 (4.6E-3)	3.38E-2 * (8.4E-5)	3.52E-2 * (4.2E-3)	2.29E-2 * (3.5E-3)
DTLZ7	3	**7.61E-2** (6.0E-3)	7.98E-1 * (7.2E-1)	1.41E-1 * (5.6E-4)	8.02E-1 * (6.5E-1)	8.01E-1 * (7.2E-1)
DTLZ1	4	1.10E1 * (6.9E0)	7.17E1 * (3.2E1)	1.12E1 * (5.3E0)	**5.74E0** (6.6E0)	1.02E1 (1.4E1)
DTLZ2	4	1.40E-1 * (5.1E-6)	1.40E-1 * (2.0E-5)	**1.40E-1** (2.5E-6)	1.48E-1 * (3.9E-3)	1.40E-1 * (2.0E-5)
DTLZ3	4	2.97E1 (1.0E1)	1.59E2 * (1.1E2)	5.05E1 * (2.1E1)	**2.32E1** (4.8E1)	3.25E1 (7.3E1)
DTLZ4	4	**1.40E-1** (3.1E-1)	1.40E-1 (5.3E-5)	7.51E-1 * (5.8E-1)	4.58E-1 * (6.0E-1)	1.40E-1 (7.5E-5)
DTLZ5	4	6.27E-2 * (1.5E-2)	7.00E-2 * (2.0E-2)	**3.61E-2** (7.4E-5)	4.69E-2 * (5.6E-3)	8.13E-2 * (3.5E-2)
DTLZ6	4	2.03E-1 * (8.5E-2)	1.88E-1 * (4.5E-2)	**3.61E-2** (6.2E-4)	5.12E-2 * (1.2E-1)	1.83E-1 * (6.1E-2)
DTLZ7	4	**2.19E-1** (2.0E-2)	6.51E-1 * (7.4E-1)	3.91E-1 * (2.5E-4)	1.12E0 * (7.2E-1)	3.83E-1 * (9.1E-1)
DTLZ1	5	**1.57E1** (4.5E0)	1.21E2 * (2.6E1)	1.58E1 (6.4E0)	1.87E1 (2.0E1)	2.10E1 (2.6E1)
DTLZ2	5	2.12E-1 * (1.3E-5)	2.12E-1 * (1.5E-4)	**2.12E-1** (1.2E-4)	2.31E-1 * (4.8E-3)	2.12E-1 * (1.4E-4)
DTLZ3	5	5.01E1 (2.3E1)	3.61E2 * (1.8E2)	5.76E1 (1.5E1)	8.63E1 (4.5E1)	**4.41E1** (8.4E1)
DTLZ4	5	**2.12E-1** (8.9E-5)	2.12E-1 * (2.9E-4)	6.46E-1 * (4.4E-1)	4.30E-1 * (2.1E-1)	2.12E-1 * (1.7E-4)
DTLZ5	5	1.26E-1 * (3.7E-2)	1.61E-1 * (5.0E-2)	**2.60E-2** (1.2E-3)	3.48E-2 * (8.2E-3)	1.45E-1 * (4.0E-2)
DTLZ6	5	1.26E0 * (8.9E-1)	7.19E-1 * (2.4E-1)	**2.99E-2** (2.3E-3)	3.57E-2 * (8.3E-3)	4.52E-1 * (1.9E-1)
DTLZ7	5	**3.89E-1** (2.8E-2)	4.67E-1 * (5.5E-1)	1.14E0 * (2.0E-1)	9.09E-1 * (6.5E-1)	3.97E-1 (5.8E-2)
n = 1000						
DTLZ1	2	4.88E3 * (2.1E2)	4.99E3 * (1.6E3)	5.72E3 * (8.2E2)	8.86E2 (5.7E2)	**7.98E2** (9.7E2)
DTLZ2	2	2.12E0 * (1.9E-1)	1.03E0 * (5.7E-1)	7.89E0 * (1.6E0)	1.03E-1 * (1.1E-1)	**4.56E-3** (1.0E-3)
DTLZ3	2	1.29E4 * (2.8E2)	1.45E4 * (3.8E3)	1.66E4 * (1.7E3)	2.80E3 * (1.2E3)	**1.51E3** (2.7E3)
DTLZ4	2	2.53E0 * (6.2E-1)	9.13E-1 * (5.7E-1)	6.02E0 * (1.1E1)	1.34E-2 * (7.7E-3)	**8.13E-3** (4.8E-3)
DTLZ5	2	2.05E0 * (2.9E-1)	9.48E-1 * (9.7E-1)	7.21E0 * (1.4E0)	1.36E-1 * (6.2E-2)	**4.77E-3** (2.3E-3)
DTLZ6	2	5.40E2 * (1.6E1)	8.79E1 * (3.6E1)	6.37E2 * (1.8E1)	1.32E-2 * (2.7E-2)	**3.96E-3** (6.2E-8)
DTLZ7	2	1.32E0 * (1.1E-1)	**4.42E-1** (4.3E-1)	4.38E0 * (4.1E-1)	4.56E-1 * (2.9E-2)	4.42E-1 (4.3E-1)
DTLZ1	3	8.15E3 * (1.2E3)	5.30E3 * (1.6E3)	6.30E3 * (1.7E3)	**6.50E2** (5.9E2)	1.02E3 (1.1E3)
DTLZ2	3	4.94E0 * (5.0E-1)	1.53E0 * (1.3E0)	7.69E0 * (2.3E0)	1.03E-1 * (1.7E-1)	**5.89E-2** (4.1E-3)
DTLZ3	3	1.74E4 * (1.1E3)	1.50E4 * (4.8E3)	2.46E4 * (3.9E3)	**2.35E3** (1.7E3)	3.45E3 (2.9E3)
DTLZ4	3	6.52E0 * (1.0E0)	1.62E0 * (1.0E0)	1.97E0 * (1.9E0)	5.43E-1 * (4.0E-1)	**6.55E-2** (1.0E-2)
DTLZ5	3	5.86E0 * (3.9E-1)	1.85E0 * (1.0E0)	7.16E0 * (1.4E0)	7.28E-1 * (4.9E-1)	**2.50E-2** (1.2E-2)
DTLZ6	3	6.23E2 * (8.3E0)	1.98E2 * (9.0E1)	6.74E2 * (1.6E1)	4.95E-2 * (2.1E-1)	**2.43E-2** (3.3E-3)
DTLZ7	3	3.26E0 * (2.2E-1)	8.03E-1 (5.3E-3)	3.12E0 * (3.0E-1)	**8.02E-1** (2.0E-3)	8.03E-1 (4.2E-1)
DTLZ1	4	8.23E3 * (1.0E3)	5.02E3 * (1.4E3)	6.20E3 * (8.4E2)	**4.04E2** (4.5E2)	7.55E2 * (1.0E3)
DTLZ2	4	7.62E0 * (7.6E-1)	3.03E0 * (1.6E0)	1.05E1 * (4.2E0)	1.04E0 (8.8E-1)	**2.26E-1** (2.1E-1)
DTLZ3	4	2.87E4 * (2.8E3)	1.72E4 * (4.8E3)	2.91E4 * (5.5E3)	**1.03E3** (1.5E3)	4.10E3 * (3.9E3)
DTLZ4	4	9.33E0 * (1.2E0)	6.96E0 * (3.5E0)	1.97E0 * (1.0E1)	7.53E-1 (5.8E-1)	**7.17E-1** (9.0E-1)
DTLZ5	4	8.37E0 * (9.4E-1)	5.08E0 * (1.9E0)	7.85E0 * (2.5E0)	**7.41E-1** (1.4E-1)	2.34E0 * (4.5E0)
DTLZ6	4	6.92E2 * (3.5E0)	4.46E2 * (6.2E1)	6.95E2 * (1.4E1)	7.42E-1 * (—)	**3.06E-1** (3.1E-1)
DTLZ7	4	6.00E0 * (4.6E-1)	1.13E0 (9.0E-1)	4.76E0 * (4.6E-1)	**1.12E0** (5.6E-3)	1.13E0 (4.9E-1)
DTLZ1	5	9.81E3 * (1.1E3)	5.31E3 * (1.6E3)	7.44E3 * (9.4E2)	**1.72E3** (1.7E3)	1.96E3 (1.3E3)
DTLZ2	5	9.90E0 * (1.2E0)	1.00E1 * (2.4E0)	1.24E1 * (2.5E0)	**3.13E-1** (3.6E-1)	6.81E0 * (7.0E0)
DTLZ3	5	3.77E4 * (3.2E3)	2.33E4 * (1.0E4)	3.19E4 * (3.9E3)	**4.53E3** (3.7E3)	6.62E3 (4.4E3)
DTLZ4	5	1.12E1 * (1.0E1)	1.65E1 * (4.5E0)	1.26E0 * (4.9E0)	**6.56E-1** (4.0E-1)	1.18E1 * (5.6E0)
DTLZ5	5	1.24E1 * (1.1E0)	2.03E1 * (6.7E0)	1.08E1 * (4.7E0)	**7.46E-1** (1.1E-2)	1.61E1 * (2.0E1)
DTLZ6	5	7.41E2 * (8.3E0)	4.80E2 * (7.8E1)	7.23E2 * (1.0E1)	**1.14E2** (1.4E2)	2.28E2 * (1.3E2)
DTLZ7	5	5.40E0 * (8.4E-1)	1.49E0 (6.2E-1)	6.71E0 * (4.4E-1)	1.55E0 * (3.2E-2)	**9.18E-1** (1.0E0)

Table B.9: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100, 000 function evaluations.

		NSGA-II	GroupedNSGA-II	LinkedNSGA-II	GroupLinkNSGA-II	HighProbNSGA-II
n = 200						
LSMOP1	2	3.33E-1 * (1.0E-1)	3.42E-1 * (1.4E-1)	3.13E-1 * (2.8E-2)	**1.63E-1** (7.0E-2)	3.65E-1 * (2.5E-2)
LSMOP2	2	9.48E-2 * (5.9E-3)	1.01E-1 * (6.2E-3)	9.75E-2 * (5.0E-3)	**3.96E-2** (2.0E-3)	8.96E-2 * (2.8E-3)
LSMOP3	2	1.30E1 * (4.3E0)	1.37E1 * (3.1E0)	1.08E1 * (5.5E0)	**1.39E0** (3.1E-1)	1.19E1 * (2.3E0)
LSMOP4	2	1.44E-1 * (4.3E-3)	1.26E-1 * (3.0E-3)	1.46E-1 * (4.7E-3)	**9.29E-2** (2.0E-3)	1.39E-1 * (2.3E-3)
LSMOP5	2	3.86E-1 * (5.5E-2)	6.47E-1 * (2.2E-2)	3.56E-1 * (4.8E-2)	**3.43E-1** (2.0E-3)	7.93E-1 * (5.4E-2)
LSMOP6	2	8.80E-1 * (3.5E-2)	8.91E-1 * (1.3E-2)	8.73E-1 * (3.2E-2)	**4.29E-1** (8.4E-3)	8.26E-1 * (1.3E-2)
LSMOP7	2	4.29E0 * (2.7E0)	2.76E1 * (3.3E1)	3.82E0 * (1.5E0)	**1.48E0** (1.9E-3)	5.38E1 * (1.3E1)
LSMOP8	2	4.03E-1 * (6.9E-2)	5.58E-1 * (5.2E-2)	3.73E-1 * (6.1E-2)	**3.52E-1** (3.0E-2)	6.68E-1 * (4.8E-2)
LSMOP9	2	1.37E0 * (1.8E-1)	9.60E-1 * (1.1E-2)	1.16E0 * (1.9E-1)	**8.10E-1** (6.6E-16)	1.93E0 * (4.0E-2)
n = 300						
LSMOP1	3	2.31E0 * (4.3E-1)	7.10E-1 * (7.6E-2)	2.24E0 * (5.2E-1)	**3.87E-1** (6.2E-2)	8.82E-1 * (9.3E-2)
LSMOP2	3	1.05E-1 (5.8E-3)	1.05E-1 * (4.3E-3)	1.06E-1 * (5.6E-3)	**1.03E-1** (2.8E-3)	1.07E-1 * (2.5E-3)
LSMOP3	3	7.71E0 * (1.4E0)	5.43E0 * (7.7E-1)	7.75E0 * (1.6E0)	**1.91E0** (1.2E0)	7.54E0 * (9.1E-1)
LSMOP4	3	**2.62E-1** (9.7E-3)	2.77E-1 * (7.3E-3)	2.65E-1 (1.0E-2)	2.81E-1 * (9.1E-3)	2.81E-1 * (7.1E-3)
LSMOP5	3	6.36E0 * (9.4E-1)	2.05E0 * (3.0E-1)	6.04E0 * (1.1E0)	**4.76E-1** (2.5E-2)	2.42E0 * (3.8E-1)
LSMOP6	3	6.79E0 * (1.2E2)	5.76E1 * (3.1E1)	5.75E0 * (1.1E2)	**1.26E0** (6.6E0)	1.44E2 * (4.3E1)
LSMOP7	3	1.59E0 * (5.2E-2)	1.53E0 * (3.5E-2)	1.59E0 * (5.3E-2)	**9.02E-1** (3.1E-3)	1.50E0 * (4.0E-2)
LSMOP8	3	4.72E-1 * (7.7E-2)	3.53E-1 * (4.9E-2)	4.46E-1 * (6.9E-2)	**2.42E-1** (1.9E-2)	5.02E-1 * (4.5E-2)
LSMOP9	3	2.95E0 * (1.0E-1)	1.88E0 * (1.1E-1)	2.91E0 * (8.9E-2)	**1.14E0** (3.4E-3)	3.53E0 * (1.5E-1)
n = 400						
LSMOP1	4	5.84E0 * (7.3E-1)	5.78E0 * (1.0E0)	6.13E0 * (6.6E-1)	**7.65E-1** (2.5E-1)	6.05E0 * (1.6E0)
LSMOP2	4	1.85E-1 * (1.1E-2)	1.81E-1 (5.6E-3)	1.85E-1 * (1.1E-2)	**1.77E-1** (8.0E-3)	1.82E-1 * (5.6E-3)
LSMOP3	4	1.37E1 * (1.6E0)	1.57E1 * (2.0E0)	1.32E1 * (1.6E0)	**7.53E0** (1.7E0)	1.64E1 * (1.3E0)
LSMOP4	4	2.60E-1 (1.8E-2)	2.59E-1 (1.1E-2)	2.63E-1 (1.7E-2)	2.55E-1 (1.2E-2)	2.61E-1 (1.1E-2)
LSMOP5	4	1.85E1 * (1.9E0)	2.05E1 * (9.5E-1)	1.84E1 * (1.2E0)	**4.68E-1** (1.0E-2)	2.16E1 * (1.3E0)
LSMOP6	4	1.27E0 * (7.4E-3)	1.27E0 * (6.7E-3)	1.27E0 * (7.5E-3)	**8.87E-1** (7.6E-3)	1.27E0 * (6.2E-3)
LSMOP7	4	1.97E4 * (9.4E3)	4.80E3 * (4.6E3)	1.80E4 * (7.3E3)	**1.22E0** (4.5E-3)	5.18E3 * (3.5E3)
LSMOP8	4	9.33E0 * (1.0E0)	8.13E0 * (1.5E0)	9.37E0 * (8.9E-1)	**4.82E-1** (2.1E-1)	1.01E1 * (1.6E0)
LSMOP9	4	5.86E0 * (7.2E-1)	3.81E0 * (4.6E-1)	5.59E0 * (9.4E-1)	**1.46E0** (6.1E-1)	7.01E0 * (2.7E-1)
n = 500						
LSMOP1	5	9.04E0 * (1.1E0)	8.55E0 * (2.7E0)	9.08E0 * (8.7E-1)	**2.09E0** (1.1E0)	9.12E0 * (2.3E0)
LSMOP2	5	2.16E-1 * (1.1E-2)	2.04E-1 (1.0E-2)	2.16E-1 * (1.2E-2)	**2.03E-1** (7.8E-3)	2.09E-1 * (8.8E-3)
LSMOP3	5	1.94E1 * (2.4E0)	1.79E1 * (4.2E0)	1.90E1 * (2.8E0)	**8.30E0** (5.3E0)	1.90E1 * (4.5E0)
LSMOP4	5	3.68E-1 * (1.8E-2)	3.56E-1 * (3.1E-2)	3.70E-1 * (1.9E-2)	**3.44E-1** (2.2E-2)	3.64E-1 * (2.4E-2)
LSMOP5	5	1.95E1 * (4.1E0)	2.15E1 * (4.2E0)	2.14E1 * (3.2E0)	**1.20E0** (1.6E0)	2.12E1 * (4.9E0)
LSMOP6	5	4.75E4 * (1.6E4)	1.42E4 * (1.4E4)	4.95E4 * (1.8E4)	**1.33E0** (1.4E-1)	5.66E4 * (3.0E4)
LSMOP7	5	3.47E0 * (1.4E-1)	3.44E0 * (1.5E-1)	3.50E0 * (1.5E-1)	**1.26E0** (7.4E-2)	3.42E0 * (1.2E-1)
LSMOP8	5	1.21E0 * (1.9E-2)	1.21E0 * (1.4E-2)	1.21E0 * (2.2E-2)	**5.02E-1** (1.9E-1)	1.21E0 * (1.6E-2)
LSMOP9	5	3.02E1 * (7.5E0)	3.45E1 * (2.8E1)	3.01E1 * (1.0E1)	**6.44E-1** (2.2E-2)	2.29E1 * (1.1E1)
n = 1000						
LSMOP1	2	3.55E0 * (3.9E-1)	1.16E0 * (7.8E-2)	3.59E0 * (3.8E-1)	**3.06E-1** (3.4E-2)	1.51E0 * (9.2E-2)
LSMOP2	2	3.60E-2 * (8.2E-4)	3.93E-2 * (2.2E-4)	3.61E-2 * (3.9E-4)	**1.81E-2** (4.7E-4)	3.94E-2 * (1.4E-4)
LSMOP3	2	2.10E1 * (1.0E0)	2.05E1 * (1.3E0)	2.08E1 * (1.1E0)	**1.57E0** (1.6E-4)	1.87E1 * (1.0E0)
LSMOP4	2	6.07E-2 * (1.2E-3)	6.54E-2 * (9.3E-4)	6.07E-2 * (1.0E-3)	**3.07E-2** (8.3E-4)	6.74E-2 * (8.7E-4)
LSMOP5	2	1.04E1 * (9.4E-1)	3.55E0 * (2.2E-1)	1.03E1 * (6.9E-1)	**5.07E-1** (2.4E-2)	4.53E0 * (1.8E-1)
LSMOP6	2	7.74E-1 * (6.4E-4)	7.73E-1 * (5.1E-4)	7.74E-1 * (6.4E-4)	**3.83E-1** (1.1E-2)	7.73E-1 * (8.1E-4)
LSMOP7	2	2.20E3 * (5.0E3)	2.48E3 * (2.9E2)	3.48E3 * (4.4E3)	**1.51E0** (5.7E-4)	1.34E3 * (1.8E2)
LSMOP8	2	4.94E0 * (8.0E-1)	1.04E0 * (4.5E-2)	4.77E0 * (7.1E-1)	**5.71E-1** (1.3E-1)	1.54E0 * (5.2E-2)
LSMOP9	2	1.40E0 * (1.5E-1)	9.38E-1 * (1.3E-2)	1.37E0 * (1.2E-1)	**6.00E-1** (1.4E-1)	1.75E0 * (5.2E-2)
LSMOP1	3	6.13E0 * (6.1E-1)	7.31E0 * (7.5E-1)	6.08E0 * (5.1E-1)	**5.53E-1** (1.8E-1)	7.53E0 * (1.0E0)
LSMOP2	3	7.11E-2 * (7.2E-3)	**6.65E-2** (4.1E-3)	7.10E-2 * (5.6E-3)	6.69E-2 (2.9E-3)	6.66E-2 (4.4E-3)
LSMOP3	3	1.72E1 * (8.5E0)	2.36E1 * (8.4E0)	1.78E1 * (6.2E0)	**3.06E0** (1.2E0)	2.38E1 * (7.8E0)
LSMOP4	3	1.30E-1 * (8.1E-3)	1.27E-1 (4.3E-3)	1.30E-1 * (8.1E-3)	**1.27E-1** (4.4E-3)	1.29E-1 * (4.1E-3)
LSMOP5	3	1.57E1 * (7.9E-1)	1.72E1 * (1.0E0)	1.56E1 * (1.1E0)	**5.23E-1** (4.4E-3)	1.89E1 * (1.2E0)
LSMOP6	3	1.20E4 * (3.0E3)	2.71E3 * (3.1E3)	1.02E4 * (4.4E3)	**3.60E0** (1.5E1)	3.73E3 * (3.3E3)
LSMOP7	3	1.10E0 * (5.0E-3)	1.10E0 * (4.3E-3)	1.10E0 * (4.5E-3)	**8.53E-1** (4.2E-3)	1.10E0 * (6.2E-3)
LSMOP8	3	9.58E-1 * (1.3E-2)	9.58E-1 * (3.5E-2)	9.58E-1 * (6.5E-2)	**2.13E-1** (2.2E-2)	9.57E-1 * (1.0E-1)
LSMOP9	3	1.42E1 * (2.1E0)	1.85E0 * (2.1E-1)	1.41E1 * (1.5E0)	**1.14E0** (8.9E-4)	5.89E0 * (3.8E-1)
LSMOP1	4	8.01E0 * (8.0E-1)	8.97E0 * (8.3E-1)	8.22E0 * (9.3E-1)	**9.39E-1** (4.0E-1)	9.35E0 * (7.4E-1)
LSMOP2	4	1.47E-1 * (1.0E-2)	1.38E-1 * (4.6E-3)	1.48E-1 * (9.8E-3)	**1.34E-1** (8.3E-3)	1.38E-1 * (4.3E-3)
LSMOP3	4	2.11E1 * (1.9E0)	2.29E1 * (1.4E0)	2.10E1 * (2.0E0)	**9.43E0** (8.4E-1)	2.31E1 * (1.1E0)
LSMOP4	4	1.79E-1 * (1.0E-2)	1.73E-1 (1.1E-2)	1.79E-1 * (9.0E-3)	**1.70E-1** (9.0E-3)	1.74E-1 * (6.3E-3)
LSMOP5	4	2.10E1 * (1.2E0)	2.15E1 * (1.0E0)	2.10E1 * (1.0E0)	**4.66E-1** (8.7E-3)	2.25E1 * (1.2E0)
LSMOP6	4	1.12E0 * (6.5E-4)	1.12E0 * (6.6E-4)	1.12E0 * (4.7E-4)	**9.04E-1** (4.0E-3)	1.12E0 * (5.5E-4)
LSMOP7	4	4.14E4 * (6.8E3)	4.59E4 * (1.2E4)	4.02E4 * (9.7E3)	**1.24E0** (4.7E-3)	4.86E4 * (1.1E4)
LSMOP8	4	1.29E1 * (7.7E-1)	1.35E1 * (1.1E0)	1.25E1 * (6.4E-1)	**4.72E-1** (2.1E-1)	1.51E1 * (1.2E0)
LSMOP9	4	2.43E1 * (4.7E0)	8.90E0 * (1.6E0)	2.29E1 * (2.9E0)	**8.51E-1** (6.1E-1)	1.21E1 * (1.2E0)
LSMOP1	5	9.48E0 * (1.1E0)	1.03E1 * (7.3E-1)	9.71E0 * (1.1E0)	**2.43E0** (1.5E0)	1.04E1 * (1.0E0)
LSMOP2	5	1.94E-1 * (7.0E-3)	1.84E-1 (7.3E-3)	1.92E-1 * (1.1E-2)	**1.80E-1** (9.6E-3)	1.83E-1 (7.3E-3)
LSMOP3	5	2.38E1 * (1.7E0)	2.41E1 * (1.8E0)	2.35E1 * (2.5E0)	**1.14E1** (1.5E0)	2.43E1 * (2.4E0)
LSMOP4	5	2.80E-1 * (1.5E-2)	2.76E-1 * (1.7E-2)	2.80E-1 * (1.2E-2)	**2.62E-1** (1.2E-2)	2.71E-1 * (2.1E-2)
LSMOP5	5	2.06E1 * (2.2E0)	2.25E1 * (2.8E0)	2.09E1 * (2.3E0)	**2.16E0** (2.2E0)	2.20E1 * (2.9E0)
LSMOP6	5	5.20E4 * (1.3E4)	6.40E4 * (1.6E4)	5.48E4 * (1.2E4)	**1.29E0** (1.3E-1)	6.44E4 * (1.2E4)
LSMOP7	5	2.09E0 * (4.8E-2)	2.08E0 * (2.6E-2)	2.08E0 * (5.4E-2)	**1.20E0** (1.0E-1)	2.06E0 * (4.5E-2)
LSMOP8	5	1.15E0 * (8.9E-4)	1.15E0 * (2.7E-3)	1.15E0 * (4.3E-3)	**4.64E-1** (1.5E-1)	1.15E0 * (9.1E-4)
LSMOP9	5	7.36E1 * (6.6E0)	7.52E1 * (3.2E1)	7.37E1 * (7.8E0)	**6.43E-1** (2.4E-2)	6.13E1 * (2.5E1)

Table B.10: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100,000 function evaluations.

		NSGA-III	GroupedNSGA-III	LinkedNSGA-III	GroupLinkNSGA-III	HighProbNSGA-III
n = 200						
LSMOP1	2	5.09E-1 * (1.1E-1)	5.27E-1 * (6.3E-2)	5.12E-1 * (1.3E-1)	**1.91E-1** (5.2E-2)	5.83E-1 * (1.6E-1)
LSMOP2	2	9.34E-2 * (4.2E-3)	1.01E-1 * (7.5E-3)	9.38E-2 * (4.7E-3)	**3.85E-2** (1.6E-3)	9.45E-2 * (4.2E-3)
LSMOP3	2	1.22E1 * (3.4E0)	1.28E1 * (2.7E0)	1.19E1 * (3.3E0)	**1.54E0** (4.8E-2)	1.13E1 * (2.9E0)
LSMOP4	2	1.41E-1 * (2.8E-3)	1.28E-1 * (2.1E-3)	1.41E-1 * (3.2E-3)	**9.44E-2** (1.6E-3)	1.41E-1 * (1.6E-3)
LSMOP5	2	4.72E-1 * (1.2E-1)	7.48E-1 * (6.1E-2)	4.66E-1 * (1.3E-1)	**3.46E-1** (6.6E-3)	8.89E-1 * (4.8E-2)
LSMOP6	2	8.89E-1 * (7.6E-3)	8.78E-1 * (1.4E-2)	8.93E-1 * (1.2E-2)	**4.32E-1** (1.3E-1)	8.45E-1 * (1.6E-2)
LSMOP7	2	8.01E0 * (2.0E1)	7.56E1 * (2.0E1)	1.88E1 * (1.5E1)	**1.48E0** (1.8E-3)	8.73E1 * (2.0E1)
LSMOP8	2	4.56E-1 * (1.1E-1)	5.78E-1 * (6.9E-2)	5.07E-1 * (1.3E-1)	**3.55E-1** (1.7E-2)	6.77E-1 * (5.8E-2)
LSMOP9	2	1.52E0 * (1.3E-1)	9.94E-1 * (2.4E-2)	1.34E0 * (1.2E-1)	**8.10E-1** (6.6E-16)	1.96E0 * (2.5E-2)
n = 300						
LSMOP1	3	9.20E-1 * (2.7E-1)	1.25E0 * (2.4E-1)	9.11E-1 * (2.7E-1)	**3.89E-1** (1.0E-2)	1.29E0 * (1.8E-1)
LSMOP2	3	8.31E-2 * (1.2E-3)	8.88E-2 * (8.3E-4)	8.30E-2 * (1.1E-3)	**6.94E-2** (2.0E-3)	8.87E-2 * (1.0E-3)
LSMOP3	3	4.08E0 (1.3E0)	8.52E0 * (1.1E0)	**3.75E0** (6.7E-1)	5.41E0 * (1.3E0)	8.85E0 * (7.4E-1)
LSMOP4	3	1.97E-1 * (4.1E-3)	2.18E-1 * (4.1E-3)	1.99E-1 * (5.4E-3)	**1.66E-1** (5.5E-3)	2.19E-1 * (3.7E-3)
LSMOP5	3	2.57E0 * (8.2E-1)	3.10E0 * (4.5E-1)	2.92E0 * (1.1E0)	**4.90E-1** (5.2E-2)	3.28E0 * (1.3E0)
LSMOP6	3	6.18E0 * (5.2E0)	1.88E1 * (9.7E0)	4.83E0 * (4.1E0)	**3.24E0** (4.1E0)	5.77E1 * (1.8E1)
LSMOP7	3	1.53E0 * (2.7E-2)	1.49E0 * (2.8E-2)	1.53E0 * (2.6E-2)	**8.99E-1** (4.0E-3)	1.49E0 * (3.6E-2)
LSMOP8	3	3.80E-1 * (4.7E-2)	3.59E-1 * (4.5E-2)	3.64E-1 * (4.1E-2)	**2.13E-1** (5.2E-2)	4.86E-1 * (4.5E-2)
LSMOP9	3	2.91E0 * (7.2E-2)	1.95E0 * (1.3E-1)	2.84E0 * (9.3E-2)	**1.14E0** (1.8E-1)	3.58E0 * (9.0E-2)
n = 400						
LSMOP1	4	3.25E0 * (6.4E-1)	3.58E0 * (5.7E-1)	3.25E0 * (3.1E-1)	**5.21E-1** (5.7E-2)	3.46E0 * (7.8E-1)
LSMOP2	4	1.51E-1 * (1.5E-3)	1.59E-1 * (1.3E-3)	1.51E-1 * (1.5E-3)	**1.36E-1** (2.4E-3)	1.59E-1 * (2.1E-3)
LSMOP3	4	1.78E1 * (3.7E0)	1.73E1 * (2.4E0)	1.69E1 * (2.5E0)	**4.86E0** (3.1E0)	1.71E1 * (2.9E0)
LSMOP4	4	2.01E-1 * (5.1E-3)	2.12E-1 * (4.0E-3)	2.01E-1 * (4.5E-3)	**1.80E-1** (4.6E-3)	2.16E-1 * (4.2E-3)
LSMOP5	4	5.93E0 * (1.1E0)	6.23E0 * (1.0E0)	5.98E0 * (1.2E0)	**4.56E-1** (2.7E-3)	7.78E0 * (9.9E-1)
LSMOP6	4	1.22E0 * (9.6E-3)	1.22E0 * (8.7E-3)	1.22E0 * (1.0E-2)	**8.93E-1** (1.2E-2)	1.21E0 * (9.7E-3)
LSMOP7	4	4.74E2 * (3.3E2)	5.07E2 * (3.1E2)	5.28E2 * (4.1E2)	**9.29E0** (1.2E1)	6.83E2 * (2.2E2)
LSMOP8	4	2.30E0 * (5.1E-1)	1.70E0 * (4.1E-1)	2.24E0 * (4.3E-1)	**4.36E-1** (4.0E-2)	2.57E0 * (3.9E-1)
LSMOP9	4	7.32E0 * (7.8E-1)	3.78E0 * (3.5E-1)	7.08E0 * (9.3E-1)	**1.83E0** (1.3E-1)	7.07E0 * (1.2E0)
n = 500						
LSMOP1	5	2.94E0 * (5.3E-1)	3.41E0 * (4.9E-1)	3.16E0 * (6.8E-1)	**7.24E-1** (1.2E-1)	3.61E0 * (7.0E-1)
LSMOP2	5	1.74E-1 * (4.6E-4)	1.76E-1 * (4.0E-4)	1.74E-1 * (6.8E-4)	**1.72E-1** (1.6E-3)	1.77E-1 * (3.5E-4)
LSMOP3	5	1.03E1 * (2.4E0)	1.31E1 * (2.9E0)	1.02E1 * (2.6E0)	**8.11E0** (1.1E0)	1.37E1 * (3.0E0)
LSMOP4	5	2.91E-1 * (5.2E-3)	3.02E-1 * (4.7E-3)	2.92E-1 * (5.4E-3)	**2.79E-1** (3.8E-3)	3.09E-1 * (3.4E-3)
LSMOP5	5	7.11E0 * (1.1E0)	9.17E0 * (2.1E0)	6.82E0 * (9.6E-1)	**4.29E-1** (5.4E-3)	9.11E0 * (1.2E0)
LSMOP6	5	1.88E1 * (3.0E1)	2.77E1 * (5.1E1)	2.84E1 * (6.8E1)	**4.55E0** (5.1E0)	4.68E1 * (2.2E2)
LSMOP7	5	2.69E0 * (1.8E-1)	2.45E0 * (1.5E-1)	2.69E0 * (1.7E-1)	**1.16E0** (2.4E-2)	2.41E0 * (2.1E-1)
LSMOP8	5	1.15E0 * (8.1E-3)	1.15E0 * (1.3E-2)	1.15E0 * (1.3E-2)	**3.41E-1** (8.6E-3)	1.15E0 * (9.1E-3)
LSMOP9	5	1.36E1 * (4.2E0)	4.47E0 * (7.1E-1)	1.32E1 * (3.0E0)	**2.19E0** (9.0E-1)	1.14E1 * (9.8E-1)
n = 1000						
LSMOP1	2	3.05E0 * (3.9E-1)	1.23E0 * (6.6E-2)	2.90E0 * (2.9E-1)	**3.29E-1** (1.4E-2)	1.58E0 * (1.0E-1)
LSMOP2	2	3.41E-2 * (3.4E-4)	3.66E-2 * (2.9E-4)	3.41E-2 * (4.5E-4)	**1.09E-2** (3.2E-4)	3.71E-2 * (1.8E-4)
LSMOP3	2	2.16E1 * (1.7E0)	2.19E1 * (2.2E0)	2.14E1 * (2.7E0)	**1.57E0** (2.7E-4)	2.02E1 * (1.6E0)
LSMOP4	2	4.85E-2 * (1.0E-3)	5.12E-2 * (5.8E-4)	4.81E-2 * (7.0E-4)	**2.54E-2** (3.9E-4)	5.23E-2 * (6.2E-4)
LSMOP5	2	8.91E0 * (1.0E0)	4.52E0 * (3.6E-1)	9.03E0 * (1.0E0)	**6.28E-1** (5.0E-2)	5.63E0 * (3.8E-1)
LSMOP6	2	7.73E-1 * (4.7E-4)	7.73E-1 * (4.3E-4)	7.73E-1 * (5.8E-4)	**3.90E-1** (1.1E-2)	7.73E-1 * (5.6E-4)
LSMOP7	2	6.04E3 * (2.2E3)	3.14E3 * (3.8E2)	6.50E3 * (2.5E3)	**1.51E0** (4.3E-4)	2.83E3 * (6.1E2)
LSMOP8	2	3.83E0 * (4.2E-1)	1.42E0 * (8.6E-2)	3.67E0 * (3.3E-1)	**6.78E-1** (8.1E-2)	1.92E0 * (1.2E-1)
LSMOP9	2	1.41E0 * (1.6E-1)	1.01E0 * (1.0E-2)	1.35E0 * (1.0E-1)	**5.33E-1** (6.5E-2)	1.90E0 * (5.0E-2)
LSMOP1	3	2.98E0 * (6.8E-1)	3.94E0 * (3.1E-1)	2.91E0 * (5.0E-1)	**3.99E-1** (1.6E-2)	4.49E0 * (4.3E-1)
LSMOP2	3	5.18E-2 * (9.0E-5)	5.23E-2 * (7.4E-5)	5.18E-2 * (1.0E-4)	**4.84E-2** (7.6E-4)	5.24E-2 * (6.7E-5)
LSMOP3	3	1.07E1 * (1.2E0)	1.40E1 * (3.6E-1)	1.06E1 * (1.3E0)	**6.09E0** (1.4E0)	1.41E1 * (6.4E-1)
LSMOP4	3	9.91E-2 * (1.1E-3)	1.05E-1 * (1.1E-3)	9.85E-2 * (8.0E-4)	**8.08E-2** (3.0E-3)	1.07E-1 * (1.0E-3)
LSMOP5	3	7.14E0 * (7.9E-1)	9.13E0 * (9.7E-1)	7.45E0 * (9.4E-1)	**5.36E-1** (1.6E-2)	1.19E1 * (9.0E-1)
LSMOP6	3	1.98E3 * (4.0E2)	6.57E2 * (1.9E2)	1.93E3 * (7.9E2)	**5.48E0** (3.8E0)	1.07E3 * (3.1E2)
LSMOP7	3	1.09E0 * (4.3E-3)	1.08E0 * (2.6E-3)	1.09E0 * (3.0E-3)	**8.50E-1** (4.7E-4)	1.08E0 * (3.7E-3)
LSMOP8	3	7.55E-1 * (3.1E-2)	6.38E-1 * (4.0E-2)	7.57E-1 * (5.1E-2)	**1.94E-1** (3.2E-3)	6.35E-1 * (3.6E-2)
LSMOP9	3	1.35E1 * (1.8E0)	2.59E0 * (1.9E-1)	1.27E1 * (2.4E0)	**1.14E0** (1.5E-2)	6.07E0 * (3.7E-1)
LSMOP1	4	5.53E0 * (5.7E-1)	6.55E0 * (5.3E-1)	5.61E0 * (5.5E-1)	**5.08E-1** (3.9E-2)	6.62E0 * (8.3E-1)
LSMOP2	4	1.18E-1 * (2.2E-4)	1.19E-1 * (2.6E-4)	1.18E-1 * (3.2E-4)	**1.10E-1** (7.7E-4)	1.19E-1 * (1.3E-4)
LSMOP3	4	1.98E1 * (2.0E0)	2.29E1 * (2.7E0)	2.05E1 * (3.0E0)	**3.99E0** (1.0E0)	2.28E1 * (2.6E0)
LSMOP4	4	1.45E-1 * (1.3E-3)	1.50E-1 * (1.4E-3)	1.45E-1 * (1.6E-3)	**1.32E-1** (2.0E-3)	1.51E-1 * (1.0E-3)
LSMOP5	4	1.01E1 * (8.0E-1)	1.40E1 * (1.6E0)	1.00E1 * (8.4E-1)	**4.56E-1** (2.1E-3)	1.48E1 * (8.6E-1)
LSMOP6	4	1.11E0 * (1.7E-3)	1.11E0 * (1.6E-3)	1.11E0 * (1.3E-3)	**9.01E-1** (5.8E-3)	1.11E0 * (1.1E-3)
LSMOP7	4	4.51E3 * (1.7E3)	5.96E3 * (1.6E3)	4.13E3 * (1.2E3)	**4.89E0** (1.1E1)	5.81E3 * (2.1E3)
LSMOP8	4	4.53E0 * (4.6E-1)	5.19E0 * (6.6E-1)	4.35E0 * (5.5E-1)	**4.51E-1** (4.4E-2)	5.66E0 * (7.8E-1)
LSMOP9	4	2.18E1 * (4.3E0)	3.34E0 * (1.3E0)	2.15E1 * (3.7E0)	**1.76E0** (3.0E-1)	9.13E0 * (9.9E-1)
LSMOP1	5	4.86E0 * (6.2E-1)	5.99E0 * (5.2E-1)	5.06E0 * (9.1E-1)	**6.72E-1** (9.3E-2)	6.30E0 * (4.9E-1)
LSMOP2	5	1.53E-1 * (1.2E-4)	1.54E-1 * (1.2E-4)	1.53E-1 * (1.7E-4)	**1.51E-1** (1.4E-3)	1.54E-1 * (1.0E-4)
LSMOP3	5	1.38E1 * (3.7E0)	1.49E1 * (2.6E0)	1.29E1 * (2.9E0)	**8.83E0** (1.2E0)	1.66E1 * (5.4E0)
LSMOP4	5	2.24E-1 * (2.2E-3)	2.31E-1 * (1.5E-3)	2.24E-1 * (2.6E-3)	**2.14E-1** (3.9E-3)	2.33E-1 * (1.7E-3)
LSMOP5	5	1.00E1 * (1.2E0)	1.23E1 * (1.7E0)	9.71E0 * (1.0E0)	**4.28E-1** (3.1E-3)	1.34E1 * (1.4E0)
LSMOP6	5	3.17E2 * (5.6E2)	1.03E2 * (1.8E2)	2.83E2 * (2.2E2)	**4.33E0** (7.2E0)	2.40E2 * (3.9E2)
LSMOP7	5	1.85E0 * (2.8E-2)	1.80E0 * (4.6E-2)	1.85E0 * (5.5E-2)	**1.09E0** (1.4E-2)	1.80E0 * (3.3E-2)
LSMOP8	5	1.15E0 * (2.5E-3)	1.14E0 * (2.1E-3)	1.15E0 * (2.9E-3)	**3.31E-1** (5.6E-3)	1.14E0 * (1.4E-3)
LSMOP9	5	4.58E1 * (9.9E0)	8.31E0 * (1.9E0)	4.69E1 * (5.2E0)	**2.12E0** (1.0E0)	2.06E1 * (2.0E0)

Table B.11: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100,000 function evaluations.

		SMPSO	GroupedSMPSO	LinkedSMPSO	GroupLinkSMPSO	HighProbSMPSO
n = 200						
LSMOP1	2	8.75E-1 * (1.4E-1)	3.78E-1 * (9.3E-2)	8.73E-1 * (1.7E-1)	**7.89E-2** (1.7E-2)	4.18E-1 * (6.7E-2)
LSMOP2	2	9.08E-2 * (4.2E-3)	9.00E-2 * (3.7E-3)	9.00E-2 * (3.6E-3)	**1.63E-2** (1.8E-3)	8.86E-2 * (3.1E-3)
LSMOP3	2	2.41E1 * (2.1E0)	3.48E0 * (4.1E0)	2.41E1 * (1.8E0)	**1.07E0** (3.5E-1)	7.31E0 * (5.2E0)
LSMOP4	2	1.13E-1 * (2.5E-2)	**6.29E-2** (9.9E-3)	1.13E-1 * (2.1E-2)	6.33E-2 (9.3E-3)	8.62E-2 * (2.7E-2)
LSMOP5	2	1.02E0 * (2.6E-1)	7.31E-1 * (9.5E-2)	1.11E0 * (1.4E-1)	**6.55E-1** (1.0E-1)	7.35E-1 * (7.2E-2)
LSMOP6	2	5.00E-1 * (9.0E-2)	4.78E-1 * (1.3E-1)	4.87E-1 * (8.5E-2)	**9.44E-2** (2.3E-2)	4.79E-1 * (1.5E-1)
LSMOP7	2	5.55E1 * (2.6E1)	9.59E0 * (3.7E0)	5.28E1 * (2.1E1)	**1.48E0** (2.1E-4)	1.34E1 * (4.7E0)
LSMOP8	2	7.73E-1 * (1.6E-1)	7.35E-1 * (1.1E-2)	8.26E-1 * (1.4E-1)	**1.38E-1** (4.4E-2)	7.32E-1 * (1.0E-2)
LSMOP9	2	**4.23E-1** (8.8E-3)	6.03E-1 (2.7E-1)	4.23E-1 (1.2E-2)	4.99E-1 (4.1E-1)	4.42E-1 * (3.1E-2)
n = 300						
LSMOP1	3	2.20E0 * (4.2E-1)	1.81E0 * (3.5E-1)	2.14E0 * (4.5E-1)	**3.34E-1** (6.1E-2)	1.86E0 * (3.3E-1)
LSMOP2	3	9.80E-2 (5.0E-3)	9.80E-2 * (5.0E-3)	9.73E-2 (6.3E-3)	**9.60E-2** (4.0E-3)	9.88E-2 * (4.5E-3)
LSMOP3	3	1.35E1 * (4.9E0)	1.20E1 * (1.7E0)	1.32E1 * (3.9E0)	**1.52E0** (2.6E0)	1.26E1 * (1.7E0)
LSMOP4	3	2.59E-1 * (1.0E-2)	2.57E-1 * (7.5E-3)	2.61E-1 * (9.2E-3)	**2.10E-1** (1.7E-2)	2.61E-1 * (8.1E-3)
LSMOP5	3	6.11E0 * (7.0E-1)	6.24E0 * (1.6E0)	6.11E0 * (1.1E0)	**4.86E-1** (2.7E-2)	6.65E0 * (1.1E0)
LSMOP6	3	2.60E3 * (1.7E3)	1.44E3 * (1.0E3)	2.14E3 * (1.9E3)	**8.50E-1** (5.1E-1)	1.19E3 * (1.4E3)
LSMOP7	3	1.52E0 * (2.2E-2)	1.54E0 * (3.5E-2)	1.53E0 * (2.9E-2)	**7.64E-1** (1.7E-1)	1.54E0 * (3.3E-2)
LSMOP8	3	9.81E-1 * (4.4E-3)	9.81E-1 * (1.2E-2)	9.64E-1 * (7.5E-2)	**1.19E-1** (1.7E-2)	9.80E-1 * (1.1E-1)
LSMOP9	3	1.67E1 * (3.0E0)	1.53E0 * (8.5E-1)	1.66E1 * (3.0E0)	**5.02E-1** (5.2E-1)	7.61E0 * (1.5E0)
n = 400						
LSMOP1	4	3.73E0 * (1.4E0)	4.20E0 * (2.0E0)	4.39E0 * (1.3E0)	**1.29E0** (5.3E-1)	4.47E0 * (1.6E0)
LSMOP2	4	1.70E-1 (9.9E-3)	1.69E-1 (8.2E-3)	1.72E-1 (1.0E-2)	1.69E-1 (1.1E-2)	1.69E-1 (7.4E-3)
LSMOP3	4	1.89E1 * (2.7E0)	1.92E1 * (2.5E0)	1.94E1 * (1.7E0)	**1.20E1** (1.9E0)	1.93E1 * (2.0E0)
LSMOP4	4	2.47E-1 (1.7E-2)	2.49E-1 * (1.6E-2)	2.45E-1 * (2.0E-2)	**2.36E-1** (1.8E-2)	2.48E-1 * (2.8E-2)
LSMOP5	4	1.81E1 * (6.9E0)	1.94E1 * (6.2E0)	1.81E1 * (9.4E0)	**1.35E1** (5.5E0)	1.83E1 * (6.2E0)
LSMOP6	4	1.27E0 * (3.3E-4)	1.27E0 * (5.5E-3)	1.27E0 * (6.1E-3)	**1.25E0** (5.5E-2)	1.27E0 * (4.8E-3)
LSMOP7	4	4.70E4 * (4.8E4)	6.20E4 * (4.8E4)	4.98E4 * (5.6E4)	**1.46E0** (1.6E-1)	5.43E4 * (4.8E4)
LSMOP8	4	9.84E0 * (3.7E0)	1.16E1 * (5.8E0)	1.08E1 * (5.3E0)	**8.19E0** (3.0E0)	1.03E1 * (3.7E0)
LSMOP9	4	1.35E1 * (2.1E0)	1.43E1 * (2.4E0)	1.34E1 * (2.3E0)	**5.69E-1** (1.8E-1)	1.43E1 * (2.3E0)
n = 500						
LSMOP1	5	4.80E0 (1.4E0)	5.08E0 * (2.3E0)	5.63E0 * (1.7E0)	**3.99E0** (2.1E0)	5.18E0 * (1.5E0)
LSMOP2	5	2.02E-1 (1.0E-2)	2.06E-1 (9.7E-3)	2.04E-1 (1.0E-2)	2.02E-1 (1.3E-2)	2.06E-1 (1.0E-2)
LSMOP3	5	2.02E1 * (2.6E0)	2.01E1 * (1.7E0)	2.06E1 * (2.9E0)	**1.76E1** (4.2E0)	1.98E1 * (1.6E0)
LSMOP4	5	3.41E-1 (2.0E-2)	3.53E-1 (3.0E-2)	3.45E-1 (2.0E-2)	3.35E-1 (1.7E-2)	3.44E-1 (2.0E-2)
LSMOP5	5	2.39E1 (8.1E0)	2.26E1 (8.0E0)	1.86E1 (9.6E0)	2.06E1 (5.9E0)	2.08E1 (9.9E0)
LSMOP6	5	8.04E4 (3.6E4)	9.60E4 (3.7E4)	9.22E4 (5.3E4)	8.19E4 (3.4E4)	8.47E4 (4.1E4)
LSMOP7	5	3.32E0 (1.6E-1)	3.28E0 (1.4E-1)	3.31E0 (1.4E-1)	3.30E0 (3.7E-1)	3.35E0 (2.3E-1)
LSMOP8	5	1.21E0 (9.0E-3)	1.21E0 (1.4E-2)	1.21E0 (1.1E-2)	1.21E0 (1.2E-2)	1.21E0 (4.5E-3)
LSMOP9	5	5.44E1 * (4.3E0)	5.48E1 * (6.9E0)	5.36E1 * (4.2E0)	**1.23E0** (1.6E1)	5.58E1 * (5.6E0)
n = 1000						
LSMOP1	2	1.73E0 * (1.1E-1)	1.43E0 * (1.0E-1)	1.73E0 * (1.1E-1)	**9.20E-2** (1.4E-2)	1.51E0 * (1.2E-1)
LSMOP2	2	2.55E-2 * (9.3E-4)	2.67E-2 * (8.2E-4)	2.55E-2 * (8.6E-4)	**8.60E-3** (7.5E-4)	2.70E-2 * (7.1E-4)
LSMOP3	2	2.80E1 * (7.2E-1)	2.76E1 * (1.0E0)	2.79E1 * (6.2E-1)	**1.36E0** (3.2E-1)	2.76E1 * (9.6E-1)
LSMOP4	2	5.34E-2 * (7.1E-4)	5.16E-2 * (1.7E-3)	5.31E-2 * (6.9E-4)	**2.10E-2** (2.5E-3)	5.30E-2 * (1.1E-3)
LSMOP5	2	3.88E0 * (1.8E-1)	3.18E0 * (2.7E-1)	3.87E0 * (4.0E-1)	**7.42E-1** (—)	3.50E0 * (3.1E-1)
LSMOP6	2	7.58E-1 * (2.2E-3)	2.03E-1 * (1.0E-2)	7.58E-1 * (2.4E-3)	**1.33E-1** (1.1E-2)	4.31E-1 * (7.9E-2)
LSMOP7	2	2.11E3 * (3.9E2)	1.04E3 * (1.2E2)	2.02E3 * (4.4E2)	**1.30E0** (2.0E-1)	1.04E3 * (1.6E2)
LSMOP8	2	2.94E0 * (2.0E-1)	2.29E0 * (1.9E-1)	2.96E0 * (2.9E-1)	**1.80E-1** (1.2E-1)	2.42E0 * (2.4E-1)
LSMOP9	2	2.78E0 * (5.0E-1)	8.57E-2 (5.8E-3)	2.60E0 * (8.8E-1)	**2.52E-2** (4.3E-1)	1.48E-1 (2.7E-2)
LSMOP1	3	2.65E0 * (5.6E-1)	2.72E0 * (4.5E-1)	2.67E0 * (7.0E-1)	**3.28E-1** (1.0E-1)	2.66E0 * (4.5E-1)
LSMOP2	3	6.43E-2 * (3.2E-3)	6.38E-2 (3.2E-3)	6.37E-2 (3.5E-3)	**6.23E-2** (4.0E-3)	6.34E-2 (2.9E-3)
LSMOP3	3	1.84E1 * (8.3E0)	1.59E1 * (6.7E0)	1.60E1 * (6.3E0)	**3.04E0** (1.2E0)	1.62E1 * (4.6E0)
LSMOP4	3	1.19E-1 * (4.4E-3)	1.17E-1 * (4.5E-3)	1.18E-1 * (4.0E-3)	**9.42E-2** (4.5E-3)	1.19E-1 * (4.1E-3)
LSMOP5	3	6.11E0 * (9.7E-1)	6.90E0 * (8.4E-1)	5.96E0 * (9.5E-1)	**5.17E-1** (2.9E-2)	7.13E0 * (9.2E-1)
LSMOP6	3	2.75E3 * (2.0E3)	3.27E3 * (1.6E3)	3.60E3 * (2.2E3)	**7.21E-1** (2.2E-1)	3.55E3 * (2.7E3)
LSMOP7	3	1.08E0 * (2.0E-3)	1.08E0 * (3.9E-3)	1.08E0 * (1.7E-3)	**7.41E-1** (4.6E-2)	1.08E0 * (3.0E-3)
LSMOP8	3	9.33E-1 * (1.3E-1)	9.57E-1 * (9.0E-3)	8.98E-1 * (1.2E-1)	**1.03E-1** (7.7E-3)	9.57E-1 * (4.1E-2)
LSMOP9	3	2.58E1 * (2.3E0)	2.01E1 * (2.4E0)	2.65E1 * (2.6E0)	**4.67E-1** (5.9E-2)	2.22E1 * (1.5E0)
LSMOP1	4	5.07E0 * (2.2E0)	6.07E0 * (2.4E0)	5.60E0 * (1.8E0)	**2.06E0** (1.1E0)	5.22E0 * (2.3E0)
LSMOP2	4	1.34E-1 (7.2E-3)	1.31E-1 (8.1E-3)	1.34E-1 (6.0E-3)	1.33E-1 (7.9E-3)	1.35E-1 (7.0E-3)
LSMOP3	4	1.98E1 * (2.9E0)	2.01E1 * (2.0E0)	2.05E1 * (2.6E0)	**1.15E1** (2.7E0)	1.98E1 * (1.6E0)
LSMOP4	4	1.67E-1 * (8.3E-3)	1.72E-1 * (9.9E-3)	1.70E-1 * (9.3E-3)	**1.60E-1** (1.0E-2)	1.69E-1 * (1.4E-2)
LSMOP5	4	2.16E1 * (9.1E0)	1.76E1 (6.8E0)	1.99E1 * (6.3E0)	**1.54E1** (4.3E0)	1.95E1 * (8.3E0)
LSMOP6	4	1.12E0 (2.9E-12)	1.12E0 (2.7E-12)	1.12E0 (3.3E-12)	1.12E0 (1.8E-3)	1.12E0 (3.3E-12)
LSMOP7	4	5.34E4 * (4.5E4)	6.32E4 * (4.8E4)	5.93E4 * (4.5E4)	**1.55E0** (4.2E3)	5.89E4 * (5.6E4)
LSMOP8	4	1.21E1 * (4.6E0)	1.13E1 (4.4E0)	1.17E1 * (3.2E0)	**8.41E0** (5.5E0)	1.16E1 * (5.2E0)
LSMOP9	4	1.51E1 * (2.5E0)	1.67E1 * (3.2E0)	1.46E1 * (3.9E0)	**5.67E-1** (2.7E-1)	1.63E1 * (3.7E0)
LSMOP1	5	6.87E0 * (3.0E0)	6.59E0 * (2.0E0)	6.95E0 * (2.8E0)	**4.71E0** (1.1E0)	6.41E0 * (1.1E0)
LSMOP2	5	**1.80E-1** (9.3E-3)	1.85E-1 * (9.3E-3)	1.81E-1 (6.8E-3)	1.84E-1 * (9.7E-3)	1.82E-1 (1.1E-2)
LSMOP3	5	2.09E1 * (2.1E0)	2.14E1 * (2.1E0)	2.01E1 * (1.4E0)	**1.84E1** (2.2E0)	2.14E1 * (2.7E0)
LSMOP4	5	2.53E-1 (1.6E-2)	2.56E-1 (1.1E-2)	2.54E-1 (1.5E-2)	2.52E-1 (1.3E-2)	2.53E-1 (1.1E-2)
LSMOP5	5	1.83E1 (6.9E0)	2.02E1 (6.1E0)	1.95E1 (7.1E0)	2.16E1 (6.2E0)	1.95E1 (5.4E0)
LSMOP6	5	8.63E4 (5.4E4)	8.70E4 (3.2E4)	8.77E4 (4.5E4)	8.73E4 (3.1E4)	9.88E4 (2.8E4)
LSMOP7	5	2.05E0 (7.5E-2)	2.06E0 * (6.0E-2)	2.05E0 * (4.9E-2)	**2.01E0** (6.7E-2)	2.06E0 (7.5E-2)
LSMOP8	5	1.15E0 (2.5E-3)	1.15E0 (2.0E-3)	1.15E0 (2.3E-3)	1.15E0 (3.1E-3)	1.15E0 (2.0E-3)
LSMOP9	5	6.22E1 * (3.3E0)	6.31E1 * (5.1E0)	6.31E1 * (4.5E0)	**9.23E-1** (1.5E1)	6.29E1 * (4.2E0)

Table B.12: Performance comparison using the IGD indicator on the UF benchmarks using 100,000 function evaluations.

		NSGA-II	GroupedNSGA-II	LinkedNSGA-II	GroupLinkNSGA-II	HighProbNSGA-II
n = 40						
UF1	2	1.10E-1 * (2.6E-2)	**8.62E-2** (1.0E-2)	1.08E-1 * (2.2E-2)	9.58E-2 * (1.7E-2)	8.75E-2 (1.7E-2)
UF2	2	4.14E-2 (1.0E-2)	3.82E-2 (6.8E-3)	4.25E-2 (1.2E-2)	3.80E-2 (7.7E-3)	4.33E-2 (1.0E-2)
UF3	2	2.42E-1 * (6.5E-2)	2.51E-1 * (1.6E-2)	2.70E-1 * (9.4E-2)	**1.61E-1** (2.9E-2)	2.37E-1 * (1.7E-2)
UF4	2	4.90E-2 * (1.8E-3)	5.45E-2 * (3.1E-3)	4.91E-2 * (2.0E-3)	**4.71E-2** (4.0E-3)	5.49E-2 * (2.1E-3)
UF5	2	**2.75E-1** (9.1E-2)	4.58E-1 * (1.8E-1)	2.97E-1 (1.7E-1)	4.80E-1 * (1.7E-1)	3.67E-1 * (1.0E-1)
UF6	2	**1.12E-1** (7.4E-2)	2.41E-1 * (4.3E-2)	1.19E-1 (2.8E-2)	2.44E-1 * (2.4E-1)	2.46E-1 * (2.7E-2)
UF7	2	5.73E-2 (2.4E-1)	4.36E-2 (2.5E-1)	9.88E-2 (2.3E-1)	5.96E-2 (2.5E-1)	4.54E-2 (2.6E-1)
UF8	3	3.06E-1 (1.0E-1)	4.77E-1 * (1.3E-1)	3.05E-1 (7.3E-2)	**2.77E-1** (1.6E-1)	4.10E-1 * (8.8E-2)
UF9	3	3.16E-1 (7.7E-2)	5.46E-1 * (1.4E-1)	**3.12E-1** (8.4E-2)	3.16E-1 (1.8E-1)	5.94E-1 * (1.4E-1)
UF10	3	4.83E-1 (1.6E-1)	2.68E0 * (6.0E-1)	**4.57E-1** (1.3E-1)	6.12E-1 (6.7E-1)	2.33E0 * (8.1E-1)
n = 1000						
UF1	2	2.54E-1 * (6.1E-2)	2.09E-1 * (2.0E-2)	2.65E-1 * (5.1E-2)	**1.24E-1** (4.9E-3)	1.82E-1 * (1.6E-2)
UF2	2	2.28E-1 * (1.0E-2)	2.32E-1 * (7.7E-3)	2.31E-1 * (1.2E-2)	**8.51E-2** (9.2E-3)	2.79E-1 * (7.5E-3)
UF3	2	2.72E-1 * (7.6E-3)	2.93E-1 * (4.8E-3)	2.74E-1 * (9.7E-3)	**1.31E-1** (5.2E-3)	2.92E-1 * (5.9E-3)
UF4	2	1.62E-1 * (4.7E-3)	1.67E-1 * (4.1E-3)	1.60E-1 * (4.2E-3)	**1.19E-1** (1.5E-2)	1.79E-1 * (3.8E-3)
UF5	2	1.84E0 (3.0E-1)	3.15E0 * (6.1E-1)	1.95E0 * (7.6E-1)	**1.46E0** (8.3E-1)	3.20E0 * (4.9E-1)
UF6	2	9.09E-1 * (2.8E-1)	6.40E-1 * (4.8E-2)	8.36E-1 * (1.6E-1)	**1.81E-1** (7.6E-3)	5.46E-1 * (2.9E-2)
UF7	2	3.55E-1 * (1.0E-1)	1.92E-1 * (5.7E-3)	3.30E-1 * (1.8E-1)	**9.93E-2** (2.0E-2)	1.64E-1 * (3.8E-3)
UF8	3	9.72E-1 * (1.1E-1)	1.41E0 * (1.3E-1)	9.50E-1 * (1.5E-1)	**5.02E-1** (1.3E-2)	2.07E0 * (1.1E-1)
UF9	3	7.66E-1 * (3.4E-2)	1.51E0 * (1.3E-1)	7.90E-1 * (5.4E-2)	**5.43E-1** (1.0E-2)	2.04E0 * (1.9E-1)
UF10	3	4.36E0 * (1.7E0)	8.81E0 * (4.2E-1)	3.14E0 * (2.2E0)	**4.86E-1** (4.3E-2)	1.14E1 * (6.4E-1)

Table B.13: Performance comparison using the IGD indicator on the UF benchmarks using 100,000 function evaluations.

		NSGA-III	GroupedNSGA-III	LinkedNSGA-III	GroupLinkNSGA-III	HighProbNSGA-III
n = 40						
UF1	2	1.10E-1 * (1.3E-2)	**8.37E-2** (1.6E-2)	1.06E-1 * (1.6E-2)	9.83E-2 * (1.3E-2)	8.49E-2 (1.3E-2)
UF2	2	4.23E-2 * (8.8E-3)	4.32E-2 * (5.4E-3)	4.16E-2 (1.0E-2)	**3.81E-2** (7.1E-3)	4.43E-2 * (7.6E-3)
UF3	2	2.48E-1 * (8.8E-2)	2.70E-1 * (1.3E-2)	2.23E-1 * (8.4E-2)	**1.86E-1** (2.8E-2)	2.57E-1 * (1.4E-2)
UF4	2	5.00E-2 * (2.0E-3)	5.49E-2 * (1.8E-3)	5.01E-2 * (2.3E-3)	**4.56E-2** (2.4E-3)	5.56E-2 * (2.8E-3)
UF5	2	**2.73E-1** (7.8E-2)	4.85E-1 * (1.6E-1)	2.78E-1 (9.1E-2)	4.13E-1 * (1.2E-1)	4.10E-1 * (2.2E-1)
UF6	2	**1.11E-1** (3.0E-2)	2.82E-1 * (3.2E-2)	1.13E-1 (1.7E-2)	1.28E-1 * (1.0E-1)	2.58E-1 * (2.0E-2)
UF7	2	5.38E-2 (1.9E-1)	4.41E-2 (1.0E-2)	5.83E-2 * (8.1E-2)	6.07E-2 * (2.7E-2)	**4.30E-2** (1.2E-2)
UF8	3	5.32E-1 * (1.3E-2)	**4.54E-1** (9.9E-2)	5.35E-1 * (1.2E-2)	5.35E-1 * (8.1E-3)	4.57E-1 (7.3E-2)
UF9	3	3.63E-1 * (1.9E-1)	3.12E-1 * (1.0E-1)	3.93E-1 * (2.2E-1)	**1.35E-1** (2.1E-1)	2.73E-1 * (1.1E-1)
UF10	3	**3.72E-1** (7.6E-2)	8.54E-1 * (1.8E-1)	3.97E-1 (5.1E-2)	5.22E-1 * (4.7E-2)	1.07E0 * (1.3E-1)
n = 1000						
UF1	2	2.79E-1 * (4.8E-2)	2.35E-1 * (2.4E-2)	3.00E-1 * (7.3E-2)	**1.30E-1** (1.7E-2)	2.02E-1 * (1.6E-2)
UF2	2	2.22E-1 * (1.1E-2)	2.36E-1 * (8.7E-3)	2.23E-1 * (1.2E-2)	**8.50E-2** (5.2E-4)	2.85E-1 * (7.0E-3)
UF3	2	2.82E-1 * (8.3E-3)	3.08E-1 * (6.6E-3)	2.82E-1 * (1.2E-2)	**1.42E-1** (6.5E-3)	3.10E-1 * (3.9E-3)
UF4	2	1.61E-1 * (3.0E-3)	1.75E-1 * (3.4E-3)	1.61E-1 * (4.0E-3)	**1.12E-1** (2.5E-3)	1.79E-1 * (2.0E-3)
UF5	2	**1.87E0** (4.1E-1)	3.45E0 * (4.8E-1)	1.91E0 (4.8E-1)	2.08E0 (7.2E-1)	3.45E0 * (4.6E-1)
UF6	2	9.82E-1 * (1.6E-1)	7.29E-1 * (9.0E-2)	1.01E0 * (4.0E-1)	**2.11E-1** (1.3E-2)	6.30E-1 * (3.8E-2)
UF7	2	3.37E-1 * (2.3E-1)	2.05E-1 * (5.4E-3)	3.09E-1 * (1.4E-1)	**9.87E-2** (6.2E-3)	1.75E-1 * (5.4E-3)
UF8	3	8.50E-1 * (6.1E-2)	1.15E0 * (1.0E-1)	8.56E-1 * (3.9E-2)	**5.38E-1** (3.6E-3)	1.46E0 * (7.6E-2)
UF9	3	8.44E-1 * (5.2E-2)	9.98E-1 * (3.2E-2)	8.45E-1 * (2.4E-2)	**5.85E-1** (1.9E-1)	1.28E0 * (4.8E-2)
UF10	3	2.97E0 * (5.0E-1)	6.47E0 * (2.3E-1)	3.04E0 * (5.8E-1)	**5.27E-1** (1.5E-2)	7.96E0 * (3.3E-1)

Table B.14: Performance comparison using the IGD indicator on the UF benchmarks using 100,000 function evaluations.

		SMPSO	GroupedSMPSO	LinkedSMPSO	GroupLinkSMPSO	HighProbSMPSO
n = 40						
UF1	2	1.23E-1 * (2.9E-2)	1.03E-1 (2.1E-2)	1.34E-1 * (3.7E-2)	**9.78E-2** (1.0E-2)	9.89E-2 (1.7E-2)
UF2	2	6.00E-2 * (7.6E-3)	**4.37E-2** (2.4E-3)	6.23E-2 * (7.5E-3)	5.03E-2 * (5.6E-3)	4.65E-2 * (4.7E-3)
UF3	2	2.41E-1 * (1.2E-1)	**1.46E-1** (4.4E-2)	1.62E-1 (3.5E-2)	1.91E-1 * (2.2E-2)	1.49E-1 (1.2E-1)
UF4	2	1.04E-1 (2.0E-2)	1.02E-1 (1.2E-2)	1.01E-1 (1.7E-2)	9.80E-2 (1.5E-2)	9.95E-2 (2.0E-2)
UF5	2	1.27E0 (9.4E-1)	1.91E0 (9.6E-1)	1.47E0 (1.4E0)	1.37E0 (6.0E-1)	1.45E0 (7.8E-1)
UF6	2	4.13E-1 (1.5E-1)	**3.61E-1** (2.1E-1)	3.75E-1 (1.2E-1)	5.01E-1 * (1.1E-1)	3.93E-1 (9.8E-2)
UF7	2	5.96E-2 * (2.7E-1)	**3.82E-2** (2.8E-1)	4.87E-2 (2.5E-1)	5.21E-2 (2.2E-1)	3.92E-2 (2.4E-1)
UF8	3	3.89E-1 * (1.4E-1)	5.25E-1 * (1.1E-1)	4.22E-1 * (9.1E-2)	**2.43E-1** (4.3E-2)	4.37E-1 * (1.2E-1)
UF9	3	5.68E-1 * (6.0E-2)	6.28E-1 * (1.1E-1)	5.49E-1 * (9.4E-2)	**4.79E-1** (5.9E-2)	5.73E-1 * (1.5E-1)
UF10	3	2.52E0 * (5.6E-1)	2.41E0 * (5.9E-1)	2.51E0 * (7.0E-1)	**1.32E0** (8.0E-1)	2.01E0 * (8.3E-1)
n = 1000						
UF1	2	1.39E0 * (1.8E-2)	1.21E0 * (9.7E-2)	1.38E0 * (2.2E-2)	**2.78E-1** (7.0E-3)	1.27E0 * (5.8E-2)
UF2	2	1.70E-1 * (7.4E-3)	1.63E-1 * (6.4E-3)	1.71E-1 * (5.5E-3)	**8.50E-2** (5.3E-4)	1.67E-1 * (7.6E-3)
UF3	2	3.28E-1 * (7.6E-3)	2.64E-1 * (5.1E-3)	3.25E-1 * (9.2E-3)	**2.85E-2** (5.2E-3)	2.67E-1 * (5.4E-3)
UF4	2	1.36E-1 * (7.6E-4)	1.36E-1 * (1.2E-3)	1.36E-1 * (1.0E-3)	**1.20E-1** (1.2E-2)	1.36E-1 * (8.6E-4)
UF5	2	5.54E0 * (9.0E-2)	5.31E0 * (2.4E-1)	5.53E0 * (9.0E-2)	**2.79E0** (2.1E-1)	5.40E0 * (1.3E-1)
UF6	2	5.58E0 * (2.2E-1)	3.67E0 * (2.1E0)	5.52E0 * (2.4E-1)	**1.11E0** (1.4E-1)	4.15E0 * (1.9E0)
UF7	2	1.42E0 * (2.8E-2)	6.36E-1 * (1.2E-1)	1.38E0 * (3.4E-2)	**2.71E-1** (6.5E-3)	9.40E-1 * (9.5E-2)
UF8	3	5.61E-1 * (2.3E-2)	5.16E-1 * (3.5E-2)	5.64E-1 * (4.3E-2)	**3.55E-1** (4.1E-2)	5.37E-1 * (2.8E-2)
UF9	3	8.10E-1 * (3.4E-2)	7.26E-1 * (2.8E-2)	7.94E-1 * (2.2E-2)	**5.65E-1** (3.0E-2)	7.55E-1 * (3.7E-2)
UF10	3	5.01E0 * (4.7E-1)	4.78E0 * (2.8E-1)	4.89E0 * (4.0E-1)	**5.33E-1** (1.7E0)	4.78E0 * (3.7E-1)

Table B.15: Performance comparison using the IGD indicator on the WFG benchmarks using 100,000 function evaluations.

		NSGA-II	GroupedNSGA-II	LinkedNSGA-II	GroupLinkNSGA-II	HighProbNSGA-II
n = 40						
WFG1	2	4.65E-2 (2.6E-2)	6.91E-1 * (1.1E-1)	**4.16E-2** (4.2E-2)	6.44E-1 * (1.2E-1)	8.19E-1 * (3.0E-2)
WFG2	2	6.65E-1 * (8.4E-3)	6.64E-1 * (4.4E-1)	6.66E-1 * (1.0E-2)	**2.20E-1** (8.0E-3)	6.69E-1 * (1.7E-3)
WFG3	2	3.00E-2 (7.2E-3)	3.42E-2 * (3.9E-3)	**2.67E-2** (4.0E-3)	2.92E-2 * (3.5E-3)	4.80E-2 * (2.5E-3)
WFG4	2	2.06E-2 (1.8E-3)	3.60E-2 * (3.2E-3)	**2.05E-2** (1.3E-3)	3.26E-2 * (4.6E-3)	5.28E-2 * (2.9E-3)
WFG5	2	7.20E-2 (6.9E-4)	8.02E-2 * (1.2E-3)	**7.16E-2** (1.3E-3)	7.25E-2 * (8.6E-4)	8.27E-2 * (1.4E-3)
WFG6	2	6.18E-2 (1.4E-2)	7.07E-2 * (2.0E-2)	**5.62E-2** (7.8E-3)	6.05E-2 (3.4E-2)	1.15E-1 * (1.5E-2)
WFG7	2	1.77E-2 (1.0E-3)	2.65E-2 * (1.3E-3)	**1.74E-2** (1.1E-3)	2.11E-2 * (1.3E-3)	3.78E-2 * (2.6E-3)
WFG8	2	2.41E-1 * (1.0E-2)	2.27E-1 (7.0E-3)	2.40E-1 * (6.9E-3)	**2.24E-1** (1.4E-2)	2.40E-1 * (6.5E-3)
WFG9	2	**9.29E-2** (2.0E-3)	9.51E-2 * (3.1E-3)	9.33E-2 (2.4E-3)	9.53E-2 * (3.3E-2)	9.65E-2 * (2.4E-3)
WFG1	3	6.55E-1 (7.4E-2)	1.34E0 * (1.2E-1)	**6.47E-1** (6.4E-2)	1.25E0 * (9.6E-2)	1.39E0 * (4.8E-2)
WFG2	3	5.15E-1 * (1.6E-2)	5.34E-1 * (1.4E-2)	5.16E-1 * (1.6E-2)	**2.36E-1** (1.3E-2)	5.41E-1 * (1.6E-2)
WFG3	3	**1.26E-1** (2.8E-2)	1.75E-1 * (3.0E-2)	1.31E-1 (3.4E-2)	1.47E-1 * (2.8E-2)	1.79E-1 * (1.9E-2)
WFG4	3	**2.88E-1** (1.9E-2)	3.37E-1 * (1.3E-2)	2.88E-1 (1.6E-2)	3.27E-1 * (1.8E-2)	3.40E-1 * (2.3E-2)
WFG5	3	2.94E-1 (1.4E-2)	3.40E-1 * (1.4E-2)	**2.94E-1** (1.0E-2)	3.04E-1 * (9.9E-3)	3.54E-1 * (1.3E-2)
WFG6	3	**3.02E-1** (2.1E-2)	4.33E-1 * (3.2E-2)	3.03E-1 (2.0E-2)	3.74E-1 * (1.7E-2)	4.57E-1 * (3.7E-2)
WFG7	3	**2.82E-1** (1.0E-2)	3.36E-1 * (1.9E-2)	2.83E-1 (2.3E-2)	3.10E-1 * (2.0E-2)	3.35E-1 * (1.6E-2)
WFG8	3	**4.49E-1** (2.2E-2)	5.34E-1 * (2.3E-2)	4.56E-1 (1.6E-2)	4.93E-1 * (1.8E-2)	5.29E-1 * (2.3E-2)
WFG9	3	3.25E-1 (1.4E-2)	3.39E-1 * (1.9E-2)	**3.22E-1** (2.3E-2)	3.33E-1 * (1.8E-2)	3.46E-1 * (2.4E-2)
n = 1000						
WFG1	2	1.69E0 * (4.0E-2)	1.63E0 * (4.7E-2)	1.70E0 * (5.3E-2)	**1.30E0** (2.3E-2)	1.78E0 * (2.7E-2)
WFG2	2	9.07E-1 * (5.2E-1)	8.78E-1 * (5.3E-1)	9.15E-1 * (5.2E-1)	**3.50E-1** (3.6E-3)	8.81E-1 * (3.3E-2)
WFG3	2	8.30E-1 * (3.6E-2)	6.11E-1 * (3.2E-2)	8.35E-1 * (4.2E-2)	**2.62E-1** (5.1E-3)	7.23E-1 * (2.8E-2)
WFG4	2	9.43E-1 * (4.6E-2)	7.06E-1 * (4.0E-2)	9.75E-1 * (6.2E-2)	**2.59E-1** (8.1E-3)	7.82E-1 * (5.4E-2)
WFG5	2	9.73E-1 * (5.4E-2)	6.33E-1 * (3.4E-2)	9.68E-1 * (3.0E-2)	**2.73E-1** (2.0E-2)	7.13E-1 * (2.3E-2)
WFG6	2	1.00E0 * (1.2E-1)	8.38E-1 * (3.0E-2)	1.02E0 * (7.9E-2)	**4.04E-1** (2.2E-2)	9.21E-1 * (3.5E-2)
WFG7	2	9.34E-1 * (4.8E-2)	7.52E-1 * (3.8E-2)	9.21E-1 * (2.7E-2)	**3.14E-1** (1.3E-2)	7.96E-1 * (4.6E-2)
WFG8	2	1.10E0 * (3.3E-2)	8.83E-1 * (1.9E-2)	1.09E0 * (3.4E-2)	**4.45E-1** (1.7E-2)	9.19E-1 * (3.4E-2)
WFG9	2	9.72E-1 * (6.7E-2)	7.35E-1 * (4.9E-2)	9.72E-1 * (7.9E-2)	**3.22E-1** (1.4E-1)	7.83E-1 * (3.1E-2)
WFG1	3	1.83E0 * (2.7E-2)	1.70E0 * (4.2E-2)	1.82E0 * (8.7E-2)	**1.61E0** (4.3E-2)	1.93E0 * (2.1E-2)
WFG2	3	1.79E0 * (2.1E-2)	1.77E0 * (1.4E-2)	1.78E0 * (1.8E-2)	**8.72E-1** (1.0E-1)	1.83E0 * (1.5E-2)
WFG3	3	8.81E-1 * (5.8E-2)	7.26E-1 * (4.9E-2)	8.93E-1 * (6.2E-2)	**3.78E-1** (1.4E-2)	8.18E-1 * (2.2E-2)
WFG4	3	1.89E0 * (9.6E-2)	1.70E0 * (6.6E-2)	1.90E0 * (1.0E-1)	**7.01E-1** (1.0E-1)	1.64E0 * (4.0E-2)
WFG5	3	1.67E0 * (5.4E-2)	1.43E0 * (4.0E-2)	1.65E0 * (7.6E-2)	**9.99E-1** (4.9E-2)	1.53E0 * (2.9E-2)
WFG6	3	1.60E0 * (5.7E-2)	1.48E0 * (4.8E-2)	1.60E0 * (7.0E-2)	**9.68E-1** (3.9E-2)	1.51E0 * (5.9E-2)
WFG7	3	1.78E0 * (5.6E-2)	1.59E0 * (3.9E-2)	1.78E0 * (5.1E-2)	**7.26E-1** (2.9E-2)	1.60E0 * (4.0E-2)
WFG8	3	1.83E0 * (4.6E-2)	1.65E0 * (4.1E-2)	1.83E0 * (5.4E-2)	**1.11E0** (7.9E-2)	1.67E0 * (3.8E-2)
WFG9	3	1.60E0 * (1.1E-1)	1.60E0 * (4.4E-2)	1.57E0 * (6.8E-2)	**9.15E-1** (1.6E-1)	1.63E0 * (3.7E-2)

Table B.16: Performance comparison using the IGD indicator on the WFG benchmarks using 100,000 function evaluations.

		NSGA-III	GroupedNSGA-III	LinkedNSGA-III	GroupLinkNSGA-III	HighProbNSGA-III
n = 40						
WFG1	2	1.39E-1 (6.1E-2)	6.67E-1 * (1.1E-1)	**1.36E-1** (6.5E-2)	6.68E-1 * (1.4E-1)	8.71E-1 * (4.7E-2)
WFG2	2	6.63E-1 * (4.3E-1)	6.64E-1 * (4.3E-1)	6.62E-1 * (4.3E-1)	**2.19E-1** (5.4E-3)	6.71E-1 * (2.9E-3)
WFG3	2	2.83E-2 * (7.2E-3)	3.61E-2 * (4.5E-3)	3.41E-2 * (1.1E-2)	**2.20E-2** (3.9E-3)	5.30E-2 * (3.4E-3)
WFG4	2	1.70E-2 (1.4E-3)	3.89E-2 * (3.8E-3)	**1.69E-2** (2.0E-3)	3.34E-2 * (3.6E-3)	5.67E-2 * (3.9E-3)
WFG5	2	**7.04E-2** (4.5E-4)	7.77E-2 * (2.2E-3)	7.04E-2 (5.8E-4)	7.04E-2 (1.1E-3)	8.00E-2 * (1.4E-3)
WFG6	2	5.42E-2 (1.4E-2)	7.55E-2 (2.1E-2)	5.66E-2 (1.2E-2)	**5.09E-2** (4.5E-2)	1.33E-1 * (1.9E-2)
WFG7	2	**1.50E-2** (8.4E-3)	2.59E-2 * (2.8E-3)	1.61E-2 (9.9E-3)	1.57E-2 (1.2E-3)	3.89E-2 * (3.4E-3)
WFG8	2	2.55E-1 * (2.0E-2)	2.40E-1 * (5.4E-3)	2.54E-1 * (2.0E-2)	**2.27E-1** (1.2E-2)	2.52E-1 * (5.9E-3)
WFG9	2	**9.23E-2** (1.6E-3)	9.46E-2 * (2.4E-3)	9.28E-2 (1.7E-3)	9.43E-2 (2.1E-3)	9.49E-2 * (2.1E-3)
WFG1	3	8.60E-1 (1.0E-1)	1.31E0 * (8.4E-2)	**8.41E-1** (1.2E-1)	1.24E0 * (8.8E-2)	1.36E0 * (6.2E-2)
WFG2	3	4.93E-1 * (1.2E-2)	5.05E-1 * (5.5E-3)	4.90E-1 * (8.7E-3)	**1.73E-1** (3.1E-1)	5.08E-1 * (5.2E-3)
WFG3	3	**8.85E-2** (3.0E-2)	2.07E-1 * (3.8E-2)	1.20E-1 (4.1E-2)	1.70E-1 * (4.3E-2)	2.22E-1 * (2.7E-2)
WFG4	3	**2.30E-1** (3.9E-3)	2.54E-1 * (3.5E-3)	2.31E-1 (2.6E-3)	2.48E-1 * (3.4E-3)	2.59E-1 * (5.1E-3)
WFG5	3	2.38E-1 * (2.2E-3)	2.56E-1 * (2.3E-3)	2.38E-1 * (3.6E-3)	**2.33E-1** (1.4E-3)	2.61E-1 * (4.3E-3)
WFG6	3	**2.35E-1** (6.6E-3)	3.07E-1 * (1.2E-2)	2.37E-1 (7.9E-3)	2.49E-1 * (7.2E-3)	3.10E-1 * (1.2E-2)
WFG7	3	2.30E-1 (1.0E-2)	2.48E-1 * (9.0E-3)	2.31E-1 (1.4E-2)	**2.29E-1** (2.3E-3)	2.47E-1 * (5.7E-3)
WFG8	3	**3.52E-1** (1.2E-2)	3.88E-1 * (7.4E-3)	3.57E-1 (1.1E-2)	3.53E-1 (6.5E-3)	3.93E-1 * (9.3E-3)
WFG9	3	**2.45E-1** (5.1E-3)	2.51E-1 * (4.6E-3)	2.46E-1 (4.8E-3)	2.49E-1 * (4.5E-3)	2.51E-1 * (3.6E-3)
n = 1000						
WFG1	2	1.64E0 * (4.8E-2)	1.62E0 * (4.6E-2)	1.65E0 * (6.5E-2)	**1.32E0** (2.6E-2)	1.77E0 * (2.1E-2)
WFG2	2	8.84E-1 * (4.9E-2)	8.41E-1 * (2.5E-2)	8.82E-1 * (1.8E-2)	**3.51E-1** (5.7E-3)	8.71E-1 * (2.9E-2)
WFG3	2	7.81E-1 * (2.2E-2)	6.71E-1 * (2.3E-2)	7.85E-1 * (4.2E-2)	**2.56E-1** (4.7E-3)	7.48E-1 * (3.1E-2)
WFG4	2	9.12E-1 * (5.4E-2)	7.68E-1 * (4.5E-2)	9.06E-1 * (3.5E-2)	**2.57E-1** (8.2E-3)	8.22E-1 * (5.1E-2)
WFG5	2	8.90E-1 * (5.0E-2)	6.71E-1 * (4.0E-2)	8.92E-1 * (4.1E-2)	**2.96E-1** (3.1E-2)	7.47E-1 * (3.3E-2)
WFG6	2	9.27E-1 * (8.5E-2)	9.00E-1 * (3.0E-2)	9.27E-1 * (7.1E-2)	**4.24E-1** (1.5E-2)	9.65E-1 * (2.6E-2)
WFG7	2	8.81E-1 * (4.8E-2)	7.82E-1 * (3.5E-2)	8.82E-1 * (5.8E-2)	**3.18E-1** (6.0E-3)	8.19E-1 * (4.2E-2)
WFG8	2	1.05E0 * (4.6E-2)	9.35E-1 * (3.6E-2)	1.04E0 * (4.8E-2)	**4.98E-1** (1.8E-2)	9.78E-1 * (2.4E-2)
WFG9	2	9.04E-1 * (8.5E-2)	8.04E-1 * (3.8E-2)	9.08E-1 * (8.1E-2)	**3.63E-1** (1.1E-1)	8.45E-1 * (3.4E-2)
WFG1	3	1.85E0 * (3.6E-2)	1.83E0 * (2.8E-2)	1.85E0 * (4.0E-2)	**1.61E0** (2.7E-2)	1.99E0 * (2.9E-2)
WFG2	3	1.77E0 * (2.9E-2)	1.76E0 * (3.5E-2)	1.77E0 * (2.3E-2)	**9.16E-1** (5.6E-2)	1.81E0 * (2.8E-2)
WFG3	3	1.20E0 * (1.8E-1)	9.26E-1 * (8.1E-2)	1.17E0 * (1.2E-1)	**4.33E-1** (3.4E-2)	1.01E0 * (6.8E-2)
WFG4	3	2.85E0 * (1.1E-1)	2.56E0 * (1.4E-1)	2.83E0 * (1.6E-1)	**5.64E-1** (6.0E-2)	2.59E0 * (1.5E-1)
WFG5	3	1.62E0 * (7.6E-2)	1.49E0 * (6.8E-2)	1.59E0 * (9.8E-2)	**9.14E-1** (8.7E-2)	1.65E0 * (1.1E-1)
WFG6	3	1.46E0 * (4.8E-2)	1.44E0 * (4.3E-2)	1.47E0 * (3.8E-2)	**7.27E-1** (2.9E-2)	1.45E0 * (3.7E-2)
WFG7	3	2.56E0 * (3.9E-1)	2.44E0 * (2.1E-1)	2.54E0 * (1.4E-1)	**1.46E0** (9.1E-1)	2.48E0 * (1.8E-1)
WFG8	3	1.91E0 * (9.5E-2)	1.77E0 * (5.3E-2)	1.90E0 * (9.9E-2)	**1.28E0** (4.1E-1)	1.82E0 * (7.0E-2)
WFG9	3	1.37E0 * (7.1E-2)	1.54E0 * (7.8E-2)	1.35E0 * (8.9E-2)	**8.52E-1** (1.2E-1)	1.55E0 * (7.9E-2)

Table B.17: Performance comparison using the IGD indicator on the WFG benchmarks using 100, 000 function evaluations.

		SMPSO	GroupedSMPSO	LinkedSMPSO	GroupLinkSMPSO	HighProbSMPSO
n = 40						
WFG1	2	1.22E0 * (3.4E-2)	1.23E0 * (2.6E-2)	1.23E0 * (3.9E-2)	**1.20E0** (2.3E-2)	1.23E0 * (1.3E-2)
WFG2	2	2.54E-1 * (2.2E-2)	**2.32E-1** (7.9E-3)	2.40E-1 * (4.3E-1)	2.45E-1 (1.8E-1)	2.39E-1 * (1.7E-2)
WFG3	2	5.20E-2 * (1.7E-2)	**3.70E-2** (7.2E-3)	4.53E-2 * (1.7E-2)	7.05E-2 * (1.6E-2)	3.98E-2 (1.0E-2)
WFG4	2	1.03E-1 * (1.4E-2)	1.01E-1 * (9.7E-3)	1.02E-1 * (1.8E-2)	**8.70E-2** (9.1E-3)	9.81E-2 * (8.5E-3)
WFG5	2	8.26E-2 (1.6E-2)	8.17E-2 (1.9E-2)	9.29E-2 (3.1E-2)	9.40E-2 (3.8E-2)	8.61E-2 (3.2E-2)
WFG6	2	4.79E-2 * (1.6E-3)	4.75E-2 * (2.0E-3)	4.73E-2 * (1.7E-3)	**3.03E-2** (9.3E-3)	5.14E-2 * (3.6E-3)
WFG7	2	2.37E-2 (8.8E-3)	2.51E-2 (4.7E-3)	**2.26E-2** (7.6E-3)	5.28E-2 * (1.0E-2)	2.68E-2 * (7.9E-3)
WFG8	2	3.45E-1 * (3.8E-2)	2.85E-1 * (3.7E-2)	3.37E-1 * (3.7E-2)	**2.31E-1** (2.8E-2)	2.88E-1 * (3.1E-2)
WFG9	2	9.60E-2 (3.3E-2)	**7.71E-2** (3.1E-2)	8.96E-2 * (2.1E-2)	8.34E-2 (2.2E-2)	8.83E-2 (2.2E-2)
WFG1	3	1.54E0 (1.2E-2)	1.54E0 (2.0E-2)	1.53E0 (1.8E-2)	1.53E0 (1.7E-2)	1.54E0 (1.4E-2)
WFG2	3	3.41E-1 * (1.4E-1)	2.79E-1 * (4.1E-2)	2.85E-1 * (5.9E-2)	**2.46E-1** (1.9E-2)	2.76E-1 * (4.2E-2)
WFG3	3	2.71E-1 * (5.7E-2)	2.12E-1 * (6.3E-2)	2.73E-1 * (5.2E-2)	**1.73E-1** (2.7E-2)	2.24E-1 * (6.4E-2)
WFG4	3	4.16E-1 * (4.9E-2)	3.97E-1 * (1.9E-2)	4.16E-1 * (3.5E-2)	**3.67E-1** (2.3E-2)	3.90E-1 * (2.9E-2)
WFG5	3	4.37E-1 (7.2E-2)	4.69E-1 * (7.8E-2)	4.85E-1 * (1.5E-1)	**4.19E-1** (1.0E-1)	4.82E-1 * (9.0E-2)
WFG6	3	3.53E-1 * (4.6E-2)	3.47E-1 (3.9E-2)	3.39E-1 (2.5E-2)	**3.31E-1** (2.8E-2)	3.39E-1 (4.1E-2)
WFG7	3	4.70E-1 * (9.3E-2)	4.70E-1 * (4.9E-2)	4.75E-1 * (8.5E-2)	**3.42E-1** (3.1E-2)	4.76E-1 * (3.5E-2)
WFG8	3	6.87E-1 * (6.2E-2)	7.05E-1 * (7.3E-2)	7.16E-1 * (7.6E-2)	**6.00E-1** (7.7E-2)	7.09E-1 * (5.8E-2)
WFG9	3	3.91E-1 (3.5E-2)	3.69E-1 (5.2E-2)	3.85E-1 (6.4E-2)	3.75E-1 (4.3E-2)	3.72E-1 (3.1E-2)
n = 1000						
WFG1	2	1.31E0 * (1.2E-2)	1.29E0 * (6.4E-3)	1.31E0 * (1.0E-2)	**1.26E0** (8.1E-3)	1.31E0 * (1.6E-2)
WFG2	2	8.99E-1 * (5.5E-1)	8.12E-1 * (6.0E-1)	9.03E-1 * (5.4E-1)	**2.95E-1** (1.4E-1)	8.68E-1 * (5.7E-1)
WFG3	2	8.97E-1 * (3.9E-2)	5.12E-1 * (4.2E-2)	8.97E-1 * (4.5E-2)	**1.57E-1** (1.3E-2)	6.02E-1 * (2.6E-2)
WFG4	2	4.95E-1 * (1.4E-2)	3.50E-1 * (1.8E-2)	4.96E-1 * (1.6E-2)	**1.29E-1** (6.9E-3)	4.68E-1 * (1.2E-2)
WFG5	2	6.12E-1 * (2.0E-2)	5.11E-1 * (2.4E-2)	6.09E-1 * (1.5E-2)	**1.95E-1** (4.7E-2)	5.74E-1 * (1.6E-2)
WFG6	2	2.89E-1 * (1.8E-2)	1.84E-2 * (2.5E-3)	2.91E-1 * (1.9E-2)	**1.58E-2** (9.2E-4)	2.83E-1 * (2.1E-2)
WFG7	2	9.32E-1 * (2.7E-2)	6.34E-1 * (3.9E-2)	9.40E-1 * (4.0E-2)	**1.34E-1** (1.1E-2)	6.48E-1 * (4.3E-2)
WFG8	2	1.28E0 * (7.8E-2)	1.11E0 * (1.4E-1)	1.28E0 * (9.5E-2)	**3.60E-1** (4.3E-2)	8.96E-1 * (3.8E-1)
WFG9	2	4.56E-1 * (7.5E-2)	3.26E-1 * (5.3E-2)	4.74E-1 * (7.3E-2)	**8.08E-2** (2.5E-2)	3.92E-1 * (4.5E-2)
WFG1	3	1.74E0 * (3.6E-2)	1.66E0 * (4.7E-2)	1.74E0 * (4.0E-2)	**1.53E0** (1.2E-2)	1.69E0 * (4.9E-2)
WFG2	3	1.76E0 * (1.0E0)	1.68E0 * (1.6E-2)	1.76E0 * (3.5E-2)	**3.49E-1** (1.4E-1)	1.70E0 * (2.0E-2)
WFG3	3	8.77E-1 * (1.0E-1)	4.62E-1 * (3.5E-2)	8.83E-1 * (9.9E-2)	**1.41E-1** (2.3E-2)	6.36E-1 * (2.5E-2)
WFG4	3	1.36E0 * (8.0E-2)	1.06E0 * (1.1E-1)	1.37E0 * (7.2E-2)	**4.46E-1** (9.9E-2)	1.30E0 * (8.6E-2)
WFG5	3	1.56E0 * (5.2E-2)	1.47E0 * (5.3E-2)	1.56E0 * (3.9E-2)	**7.51E-1** (1.5E-1)	1.52E0 * (5.8E-2)
WFG6	3	1.18E0 * (7.7E-2)	8.59E-1 * (8.4E-2)	1.17E0 * (6.2E-2)	**3.88E-1** (7.2E-2)	1.10E0 * (7.5E-2)
WFG7	3	1.60E0 * (8.1E-2)	1.34E0 * (6.5E-2)	1.63E0 * (8.5E-2)	**5.10E-1** (2.4E-2)	1.43E0 * (5.5E-2)
WFG8	3	1.80E0 * (7.4E-2)	1.50E0 * (7.9E-2)	1.77E0 * (9.6E-2)	**8.78E-1** (5.2E-2)	1.62E0 * (5.1E-2)
WFG9	3	1.21E0 * (4.7E-2)	1.12E0 * (8.6E-2)	1.22E0 * (9.4E-2)	**4.57E-1** (5.9E-2)	1.11E0 * (9.1E-2)

Table B.18: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100,000 function evaluations.

		NSGA-II	GroupedNSGA-II	LinkedNSGA-II	GroupLinkNSGA-II	HighProbNSGA-II
n = 40						
DTLZ1	2	7.59E-1 * (1.0E0)	2.19E2 * (3.0E1)	**3.98E-1** (7.1E-1)	9.16E-1 * (1.3E0)	2.21E2 * (2.8E1)
DTLZ2	2	5.11E-3 (2.0E-4)	1.24E-2 * (9.0E-4)	**5.06E-3** (3.4E-4)	5.31E-3 * (1.9E-4)	1.34E-2 * (1.2E-3)
DTLZ3	2	2.14E0 * (2.0E0)	5.69E2 * (1.4E2)	**1.08E0** (1.9E0)	2.34E0 * (2.4E0)	5.81E2 * (5.8E1)
DTLZ4	2	**5.12E-3** (6.0E-4)	1.37E-2 * (1.8E-3)	5.22E-3 (3.0E-4)	5.25E-3 (7.3E-1)	1.40E-2 * (1.2E-3)
DTLZ5	2	5.12E-3 (2.8E-4)	1.27E-2 * (1.0E-3)	**5.04E-3** (2.7E-4)	5.17E-3 * (2.3E-4)	1.35E-2 * (7.3E-4)
DTLZ6	2	8.86E-3 * (9.9E-3)	1.12E0 * (4.3E-1)	6.60E-3 * (9.7E-3)	**5.29E-3** (5.0E-4)	1.34E0 * (4.2E-1)
DTLZ7	2	5.32E-3 (3.4E-4)	8.15E-3 * (9.0E-4)	**5.22E-3** (2.7E-4)	5.22E-3 (2.5E-4)	8.02E-3 * (3.9E-4)
DTLZ1	3	9.07E0 * (4.4E0)	2.22E2 * (3.3E1)	**5.63E0** (4.0E0)	1.18E1 * (1.1E1)	2.37E2 * (3.2E1)
DTLZ2	3	7.32E-2 (4.4E-3)	1.11E-1 * (4.1E-3)	**7.30E-2** (5.4E-3)	7.93E-2 * (2.6E-3)	1.19E-1 * (6.1E-3)
DTLZ3	3	1.56E1 * (8.9E0)	5.86E2 * (6.5E1)	**7.33E0** (4.9E0)	1.00E1 (1.0E1)	6.07E2 * (7.5E1)
DTLZ4	3	7.31E-2 (4.1E-3)	1.09E-1 * (8.5E-3)	**7.21E-2** (3.9E-3)	7.81E-2 * (3.3E-3)	1.15E-1 * (8.1E-3)
DTLZ5	3	6.52E-3 (4.8E-4)	2.59E-2 * (2.9E-3)	**6.35E-3** (4.6E-4)	7.05E-3 * (7.4E-4)	2.91E-2 * (3.9E-3)
DTLZ6	3	1.04E-2 * (9.7E-3)	1.43E0 * (5.3E-1)	6.51E-3 (5.6E-3)	**6.08E-3** (5.3E-4)	4.79E0 * (2.0E0)
DTLZ7	3	8.15E-2 (5.1E-3)	1.22E-1 * (1.2E-2)	8.30E-2 (7.3E-3)	**8.04E-2** (6.7E-3)	1.48E-1 * (1.5E-2)
DTLZ1	4	3.57E1 (1.5E1)	4.08E2 * (6.4E1)	**3.09E1** (1.7E1)	1.20E2 * (3.4E1)	4.09E2 * (5.4E1)
DTLZ2	4	**1.63E-1** (4.6E-3)	2.51E-1 * (1.2E-2)	1.67E-1 (7.2E-3)	1.96E-1 * (8.0E-3)	2.61E-1 * (1.3E-2)
DTLZ3	4	6.70E1 * (4.5E1)	9.04E2 * (1.3E2)	**4.49E1** (2.7E1)	2.29E2 * (8.3E1)	9.58E2 * (1.2E2)
DTLZ4	4	**1.62E-1** (6.5E-3)	2.49E-1 * (1.3E-2)	1.63E-1 (8.4E-3)	1.94E-1 * (1.3E-2)	2.56E-1 * (1.0E-2)
DTLZ5	4	**1.19E-1** (3.1E-2)	2.26E-1 * (5.9E-2)	1.29E-1 (4.4E-2)	1.27E-1 (2.5E-2)	2.18E-1 * (4.7E-2)
DTLZ6	4	1.47E1 * (1.2E0)	2.20E1 * (1.7E0)	1.45E1 * (1.6E0)	**1.10E1** (2.6E0)	2.46E1 * (1.2E0)
DTLZ7	4	2.35E-1 (1.2E-2)	3.69E-1 * (3.2E-2)	**2.32E-1** (1.9E-2)	2.60E-1 * (2.8E-2)	4.30E-1 * (2.6E-2)
DTLZ1	5	**5.57E1** (2.1E1)	5.41E2 * (6.5E1)	6.04E1 (1.5E1)	1.98E2 * (3.2E1)	5.18E2 * (7.6E1)
DTLZ2	5	2.62E-1 (1.2E-2)	4.36E-1 * (2.6E-2)	**2.61E-1** (1.0E-2)	3.44E-1 * (2.0E-2)	4.55E-1 * (2.8E-2)
DTLZ3	5	**1.90E2** (4.3E1)	1.41E3 * (1.6E2)	2.02E2 (5.2E1)	5.78E2 * (1.1E2)	1.39E3 * (1.6E2)
DTLZ4	5	2.64E-1 (9.8E-3)	4.23E-1 * (2.3E-2)	**2.62E-1** (1.0E-2)	3.46E-1 * (2.8E-2)	4.32E-1 * (2.1E-2)
DTLZ5	5	1.15E0 (3.4E-1)	1.18E0 (4.0E-1)	1.15E0 (4.7E-1)	1.05E0 (4.4E-1)	1.13E0 (3.1E-1)
DTLZ6	5	2.40E1 * (1.9E0)	2.65E1 * (1.4E0)	2.34E1 * (1.6E0)	**2.16E1** (2.7E0)	2.71E1 * (1.5E0)
DTLZ7	5	**4.02E-1** (2.2E-2)	6.02E-1 * (3.9E-2)	4.08E-1 (1.6E-2)	4.37E-1 * (3.5E-2)	6.66E-1 * (4.2E-2)
n = 1000						
DTLZ1	2	4.34E3 * (1.3E2)	2.63E4 * (2.1E3)	**4.18E3** (1.8E2)	4.23E3 (9.3E2)	3.00E4 * (2.8E2)
DTLZ2	2	1.97E0 * (3.1E-1)	8.18E0 * (1.5E-1)	2.16E0 * (3.2E-1)	**5.56E-3** (2.2E-4)	1.22E1 * (2.6E-1)
DTLZ3	2	1.16E4 * (4.2E2)	7.27E4 * (4.1E3)	1.11E4 * (3.5E2)	**8.54E3** (1.2E3)	8.30E4 * (7.6E2)
DTLZ4	2	2.32E0 * (3.0E-1)	8.70E0 * (3.0E-1)	2.87E0 * (3.8E-1)	**5.39E-3** (4.9E-4)	1.24E1 * (2.3E-1)
DTLZ5	2	1.91E0 * (3.5E-1)	8.17E0 * (2.1E-1)	2.22E0 * (4.0E-1)	**5.45E-3** (3.1E-4)	1.22E1 * (4.0E-1)
DTLZ6	2	5.97E2 * (1.7E1)	3.77E2 * (5.9E0)	5.87E2 * (1.0E1)	**1.03E1** (6.7E-1)	7.50E2 * (6.9E0)
DTLZ7	2	1.67E0 * (2.6E-1)	2.79E-1 * (1.1E-2)	1.54E0 * (1.5E-1)	**5.23E-3** (2.9E-4)	7.22E-1 * (1.5E-2)
DTLZ1	3	**7.38E3** (4.8E2)	2.79E4 * (1.3E3)	7.50E3 (3.2E2)	1.45E4 * (1.9E3)	2.89E4 * (7.1E2)
DTLZ2	3	8.29E0 * (8.1E-1)	1.65E1 * (7.6E-1)	8.32E0 * (7.6E-1)	**8.23E-2** (3.4E-3)	2.30E1 * (1.0E0)
DTLZ3	3	1.80E4 (7.4E2)	8.64E4 * (1.4E3)	**1.77E4** (9.5E2)	4.01E4 * (4.9E3)	9.00E4 * (1.1E3)
DTLZ4	3	1.02E1 * (1.9E0)	1.71E1 * (8.8E-1)	1.00E1 * (1.8E0)	**8.31E-2** (4.6E-1)	2.21E1 * (9.2E-1)
DTLZ5	3	9.91E0 * (7.0E-1)	2.14E1 * (9.5E-1)	9.96E0 * (1.0E0)	**8.60E-3** (7.4E-4)	2.80E1 * (1.5E0)
DTLZ6	3	7.53E2 * (8.6E0)	7.47E2 * (6.3E0)	7.55E2 * (7.4E0)	**1.41E1** (3.7E-1)	8.41E2 * (4.8E0)
DTLZ7	3	2.79E0 * (2.1E-1)	8.73E-1 * (4.7E-2)	2.81E0 * (2.2E-1)	**8.01E-2** (6.4E-3)	1.94E0 * (9.5E-2)
DTLZ1	4	**1.09E4** (7.8E2)	2.63E4 * (1.3E3)	1.10E4 (4.8E2)	1.65E4 * (1.6E3)	2.69E4 * (6.4E2)
DTLZ2	4	2.31E1 * (1.4E0)	3.80E1 * (2.1E0)	2.30E1 * (1.3E0)	**2.29E-1** (1.7E-2)	4.78E1 * (2.1E0)
DTLZ3	4	**3.64E4** (2.0E3)	8.95E4 * (1.7E3)	3.67E4 (2.0E3)	4.85E4 * (3.5E3)	9.15E4 * (1.9E3)
DTLZ4	4	2.44E1 * (2.2E0)	3.81E1 * (1.7E0)	2.40E1 * (2.4E0)	**2.55E-1** (2.9E-2)	4.68E1 * (1.5E0)
DTLZ5	4	5.67E1 * (2.0E0)	6.54E1 * (2.7E0)	5.74E1 * (3.7E0)	**3.81E1** (7.7E0)	6.93E1 * (1.4E0)
DTLZ6	4	8.70E2 * (3.6E0)	8.69E2 * (6.5E0)	8.69E2 * (4.2E0)	**6.49E2** (1.0E2)	8.80E2 * (3.6E0)
DTLZ7	4	5.12E0 * (3.7E-1)	2.68E0 * (2.6E-1)	5.11E0 * (3.6E-1)	**2.45E-1** (2.7E-2)	4.74E0 * (2.1E-1)
DTLZ1	5	1.25E4 (1.1E3)	2.48E4 * (1.3E3)	**1.25E4** (9.1E2)	1.77E4 * (2.6E3)	2.53E4 * (1.4E3)
DTLZ2	5	4.53E1 * (4.5E0)	6.12E1 * (3.4E0)	4.53E1 * (2.3E0)	**8.14E-1** (1.5E-1)	6.79E1 * (2.6E0)
DTLZ3	5	6.32E4 * (3.1E3)	9.15E4 * (1.5E3)	6.36E4 * (3.2E3)	**5.25E4** (4.8E3)	9.28E4 * (1.8E3)
DTLZ4	5	4.79E1 * (5.4E0)	6.01E1 * (7.8E0)	4.73E1 * (5.8E0)	**2.43E0** (1.0E0)	7.12E1 * (4.4E0)
DTLZ5	5	7.24E1 (2.4E0)	7.13E1 (4.7E0)	7.24E1 (3.0E0)	**7.02E1** (6.9E0)	7.49E1 * (2.1E0)
DTLZ6	5	8.79E2 * (4.9E0)	8.74E2 * (5.3E0)	8.79E2 * (5.2E0)	**7.39E2** (4.4E1)	8.83E2 * (3.5E0)
DTLZ7	5	9.60E0 * (8.3E-1)	8.14E0 * (2.4E0)	9.36E0 * (5.7E-1)	**4.39E-1** (2.1E-2)	9.31E0 * (9.4E-1)

Table B.19: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100, 000 function evaluations.

		NSGA-III	GroupedNSGA-III	LinkedNSGA-III	GroupLinkNSGA-III	HighProbNSGA-III
n = 40						
DTLZ1	2	2.57E0 * (1.4E0)	2.82E2 * (4.0E1)	**1.13E0** (1.0E0)	1.54E1 * (9.7E0)	2.72E2 * (3.0E1)
DTLZ2	2	3.96E-3 * (6.1E-8)	1.40E-2 * (9.6E-4)	**3.96E-3** (5.7E-8)	3.97E-3 * (1.1E-5)	1.49E-2 * (1.0E-3)
DTLZ3	2	6.78E0 * (4.1E0)	7.10E2 * (7.4E1)	**2.77E0** (2.8E0)	4.00E1 * (1.9E1)	7.21E2 * (6.6E1)
DTLZ4	2	**3.96E-3** (2.3E-6)	1.50E-2 * (2.4E-3)	3.96E-3 (3.9E-5)	4.00E-3 * (5.6E-5)	1.58E-2 * (1.2E-3)
DTLZ5	2	3.96E-3 * (4.7E-8)	1.39E-2 * (8.1E-4)	**3.96E-3** (7.9E-8)	3.97E-3 * (1.3E-5)	1.48E-2 * (1.2E-3)
DTLZ6	2	3.96E-3 * (3.8E-8)	1.33E0 * (4.2E-1)	**3.96E-3** (3.7E-8)	3.96E-3 * (2.3E-8)	2.12E0 * (7.1E-1)
DTLZ7	2	**5.06E-3** (8.5E-5)	1.03E-2 * (9.4E-4)	5.07E-3 (1.0E-4)	5.16E-3 * (1.5E-4)	1.00E-2 * (1.3E-3)
DTLZ1	3	6.59E0 * (4.0E0)	2.44E2 * (2.7E1)	**4.25E0** (2.3E0)	2.54E1 * (1.2E1)	2.38E2 * (3.3E1)
DTLZ2	3	5.44E-2 (1.2E-6)	6.47E-2 * (1.5E-3)	**5.44E-2** (6.9E-7)	5.45E-2 * (2.6E-5)	6.32E-2 * (9.7E-4)
DTLZ3	3	1.48E1 * (9.0E0)	6.83E2 * (5.5E1)	**6.61E0** (5.6E0)	6.29E1 * (3.1E1)	6.54E2 * (6.1E1)
DTLZ4	3	**5.44E-2** (1.2E-4)	7.18E-2 * (4.0E-3)	5.44E-2 (4.8E-1)	5.65E-2 * (4.8E-1)	6.83E-2 * (3.2E-3)
DTLZ5	3	1.36E-2 * (2.1E-3)	3.16E-2 * (3.1E-3)	1.34E-2 * (2.5E-3)	**1.22E-2** (1.3E-3)	3.42E-2 * (3.2E-3)
DTLZ6	3	2.01E-2 * (3.7E-3)	1.37E0 * (4.3E-1)	2.12E-2 * (3.6E-3)	**1.68E-2** (3.4E-3)	3.28E0 * (1.1E0)
DTLZ7	3	7.72E-2 (4.4E-3)	1.20E-1 * (1.6E-2)	7.80E-2 (6.8E-3)	**7.63E-2** (4.8E-3)	1.29E-1 * (1.7E-2)
DTLZ1	4	1.10E1 (7.3E0)	2.52E2 * (2.0E1)	**8.53E0** (4.5E0)	7.56E1 * (1.8E1)	2.28E2 * (2.7E1)
DTLZ2	4	1.40E-1 (2.4E-6)	1.51E-1 * (1.0E-3)	**1.40E-1** (2.7E-6)	1.40E-1 * (8.6E-5)	1.48E-1 * (1.2E-3)
DTLZ3	4	2.72E1 (1.2E1)	6.88E2 * (1.0E2)	**2.34E1** (1.4E1)	1.54E2 * (8.7E1)	7.00E2 * (9.6E1)
DTLZ4	4	1.40E-1 (3.1E-1)	1.65E-1 * (8.2E-3)	**1.40E-1** (7.1E-5)	1.42E-1 * (3.0E-3)	1.58E-1 * (5.0E-3)
DTLZ5	4	**6.16E-2** (1.2E-2)	1.27E-1 * (4.4E-2)	6.53E-2 (1.6E-2)	9.05E-2 * (2.8E-2)	1.49E-1 * (5.8E-2)
DTLZ6	4	**1.88E-1** (4.6E-2)	3.97E0 * (6.0E-1)	1.94E-1 (6.1E-2)	2.08E-1 (5.7E-2)	9.25E0 * (1.1E0)
DTLZ7	4	2.17E-1 (2.2E-2)	3.24E-1 * (4.2E-2)	2.11E-1 (2.0E-2)	**2.09E-1** (1.5E-2)	3.81E-1 * (5.1E-2)
DTLZ1	5	**1.44E1** (7.8E0)	2.63E2 * (3.0E1)	1.67E1 (7.9E0)	1.00E2 * (2.1E1)	2.52E2 * (2.7E1)
DTLZ2	5	2.12E-1 (1.6E-5)	2.29E-1 * (2.1E-3)	**2.12E-1** (8.2E-6)	2.13E-1 * (5.1E-4)	2.26E-1 * (1.8E-3)
DTLZ3	5	4.84E1 * (2.5E1)	7.99E2 * (1.1E2)	**3.64E1** (1.7E1)	2.66E2 * (6.6E1)	8.11E2 * (1.0E2)
DTLZ4	5	2.12E-1 (2.1E-1)	2.51E-1 * (4.9E-3)	**2.12E-1** (2.1E-1)	2.16E-1 * (2.9E-1)	2.42E-1 * (7.6E-3)
DTLZ5	5	**1.26E-1** (3.8E-2)	2.97E-1 * (7.0E-2)	1.36E-1 (4.8E-2)	2.36E-1 * (8.0E-2)	3.01E-1 * (1.0E-1)
DTLZ6	5	1.25E0 * (8.5E-1)	5.41E0 * (1.0E0)	5.45E-1 (6.6E-1)	**4.90E-1** (1.4E-1)	1.07E1 * (1.2E0)
DTLZ7	5	3.87E-1 (2.6E-2)	4.23E-1 * (3.7E-2)	**3.83E-1** (2.7E-2)	3.94E-1 (2.7E-2)	4.32E-1 * (2.5E-2)
n = 1000						
DTLZ1	2	4.81E3 (2.3E2)	2.80E4 * (7.5E2)	**4.78E3** (2.4E2)	1.03E4 * (2.7E3)	3.03E4 * (2.6E2)
DTLZ2	2	2.12E0 * (3.0E-1)	8.95E0 * (3.0E-1)	2.23E0 * (2.4E-1)	**4.58E-3** (1.9E-4)	1.32E1 * (2.9E-1)
DTLZ3	2	1.27E4 * (5.1E2)	7.67E4 * (2.6E3)	**1.25E4** (6.4E2)	2.45E4 * (4.3E3)	8.41E4 * (1.0E3)
DTLZ4	2	2.52E0 * (3.8E-1)	9.61E0 * (2.3E-1)	2.70E0 * (5.6E-1)	**4.72E-3** (7.3E-1)	1.35E1 * (3.0E-1)
DTLZ5	2	2.14E0 * (2.9E-1)	8.90E0 * (4.4E-1)	2.23E0 * (2.5E-1)	**4.56E-3** (3.0E-4)	1.32E1 * (2.6E-1)
DTLZ6	2	5.39E2 * (1.4E1)	4.83E2 * (1.0E1)	5.42E2 * (9.9E0)	**1.00E1** (5.6E-1)	7.49E2 * (4.9E0)
DTLZ7	2	1.37E0 * (1.4E-1)	3.25E-1 * (1.2E-2)	1.24E0 * (8.2E-2)	**5.04E-3** (1.1E-4)	7.65E-1 * (1.5E-2)
DTLZ1	3	8.44E3 (1.1E3)	2.46E4 * (4.8E2)	**8.15E3** (1.2E3)	1.25E4 * (1.1E3)	2.59E4 * (4.4E2)
DTLZ2	3	5.24E0 * (8.1E-1)	1.32E1 * (8.4E-1)	5.39E0 * (5.0E-1)	**5.59E-2** (7.4E-4)	1.76E1 * (5.1E-1)
DTLZ3	3	1.76E4 (9.6E2)	8.23E4 * (1.5E3)	**1.71E4** (9.1E2)	3.70E4 * (3.8E3)	8.64E4 * (1.0E3)
DTLZ4	3	6.31E0 * (9.1E-1)	1.51E1 * (5.3E-1)	6.56E0 * (1.7E0)	**5.81E-2** (4.8E-1)	1.84E1 * (6.4E-1)
DTLZ5	3	5.68E0 * (7.8E-1)	1.53E1 * (5.1E-1)	6.00E0 * (8.4E-1)	**1.51E-2** (1.7E-3)	1.96E1 * (4.6E-1)
DTLZ6	3	6.23E2 * (8.2E0)	6.56E2 * (5.6E0)	6.22E2 * (8.9E0)	**1.32E1** (6.8E-1)	7.81E2 * (3.3E0)
DTLZ7	3	3.16E0 * (3.1E-1)	9.33E-1 * (6.2E-2)	3.16E0 * (1.4E-1)	**7.67E-2** (5.6E-3)	2.11E0 * (1.2E-1)
DTLZ1	4	**8.32E3** (7.7E2)	2.27E4 * (3.9E2)	8.36E3 (1.0E3)	1.22E4 * (9.5E2)	2.37E4 * (4.6E2)
DTLZ2	4	7.51E0 * (6.0E-1)	2.07E1 * (5.3E-1)	7.46E0 * (5.1E-1)	**1.47E-1** (1.8E-3)	2.34E1 * (6.2E-1)
DTLZ3	4	2.93E4 (3.8E3)	8.37E4 * (9.6E2)	**2.86E4** (3.2E3)	4.15E4 * (5.2E3)	8.77E4 * (7.5E2)
DTLZ4	4	8.93E0 * (9.7E-1)	2.64E1 * (1.5E0)	9.59E0 * (1.8E0)	**1.66E-1** (3.7E-1)	2.68E1 * (1.0E0)
DTLZ5	4	8.62E0 * (9.9E-1)	2.55E1 * (8.6E-1)	8.63E0 * (5.1E-1)	**3.18E-1** (1.3E-1)	2.84E1 * (4.9E-1)
DTLZ6	4	6.93E2 * (5.3E0)	7.18E2 * (4.4E0)	6.90E2 * (6.1E0)	**1.11E2** (6.3E1)	8.09E2 * (4.0E0)
DTLZ7	4	5.87E0 * (4.7E-1)	3.30E0 * (2.7E-1)	5.81E0 * (5.8E-1)	**2.14E-1** (2.5E-2)	4.93E0 * (4.0E-1)
DTLZ1	5	9.79E3 (1.0E3)	2.19E4 * (5.6E2)	**9.61E3** (9.9E2)	1.18E4 * (8.0E2)	2.28E4 * (4.2E2)
DTLZ2	5	9.76E0 * (5.9E-1)	2.80E1 * (1.3E0)	9.78E0 * (8.9E-1)	**2.36E-1** (3.5E-3)	3.02E1 * (1.3E0)
DTLZ3	5	3.75E4 (3.2E3)	8.49E4 * (1.0E3)	**3.62E4** (3.6E3)	4.49E4 * (4.1E3)	8.83E4 * (1.0E3)
DTLZ4	5	1.16E1 * (1.3E1)	3.41E1 * (2.1E0)	1.14E1 * (1.0E1)	**3.01E-1** (6.1E-1)	3.46E1 * (2.2E0)
DTLZ5	5	1.19E1 * (7.6E-1)	3.38E1 * (1.7E0)	1.17E1 * (1.0E0)	**6.41E-1** (1.2E-1)	3.61E1 * (1.1E0)
DTLZ6	5	7.42E2 * (7.1E0)	7.62E2 * (1.0E1)	7.40E2 * (8.2E0)	**2.91E2** (1.8E1)	8.28E2 * (2.9E0)
DTLZ7	5	5.42E0 * (5.4E-1)	2.32E0 * (3.5E-1)	5.35E0 * (6.7E-1)	**3.85E-1** (2.3E-2)	3.80E0 * (4.6E-1)

Table B.20: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100,000 function evaluations.

		SMPSO	GroupedSMPSO	LinkedSMPSO	GroupLinkSMPSO	HighProbSMPSO
n = 40						
DTLZ1	2	**2.22E1** (4.6E1)	3.86E1 (7.0E1)	4.48E1 (4.2E1)	5.49E1 (4.3E1)	6.93E1 * (6.4E1)
DTLZ2	2	5.21E-3 (4.4E-4)	5.17E-3 (2.8E-4)	5.23E-3 (1.9E-4)	**5.12E-3** (2.8E-4)	5.21E-3 * (1.6E-4)
DTLZ3	2	1.04E2 (2.8E2)	7.61E1 (2.0E2)	1.29E2 (2.3E2)	1.07E2 (1.2E2)	8.11E1 (1.7E2)
DTLZ4	2	5.32E-3 (7.3E-1)	7.42E-1 (7.3E-1)	5.46E-3 (7.3E-1)	5.90E-3 (7.3E-1)	7.42E-1 (7.3E-1)
DTLZ5	2	5.13E-3 (2.3E-4)	5.21E-3 (2.0E-4)	5.16E-3 (2.7E-4)	5.10E-3 (3.6E-4)	5.13E-3 (2.2E-4)
DTLZ6	2	5.21E-3 (3.1E-4)	5.24E-3 (3.6E-4)	5.16E-3 (3.1E-4)	5.29E-3 (3.1E-4)	5.14E-3 (2.3E-4)
DTLZ7	2	5.39E-3 (3.1E-4)	5.39E-3 (5.8E-4)	**5.32E-3** (5.8E-4)	4.42E-1 * (4.3E-1)	5.44E-3 (4.3E-1)
DTLZ1	3	1.57E1 (1.7E1)	2.34E1 (2.2E1)	1.66E1 (1.8E1)	1.60E1 (2.8E1)	2.25E1 (2.3E1)
DTLZ2	3	9.47E-2 * (8.0E-3)	9.09E-2 * (8.4E-3)	9.90E-2 * (1.4E-2)	**7.64E-2** (4.6E-3)	9.66E-2 * (1.0E-2)
DTLZ3	3	2.63E1 (7.3E1)	6.05E1 (9.7E1)	5.01E1 (5.7E1)	5.01E1 (7.3E1)	2.51E1 (5.3E1)
DTLZ4	3	2.58E-1 (2.8E-1)	3.81E-1 * (2.8E-1)	**2.57E-1** (2.6E-1)	5.41E-1 * (2.7E-1)	5.41E-1 * (2.7E-1)
DTLZ5	3	6.74E-3 * (9.5E-4)	6.11E-3 * (4.0E-4)	6.55E-3 * (6.3E-4)	**5.79E-3** (3.3E-4)	6.10E-3 * (3.9E-4)
DTLZ6	3	3.42E-1 * (2.2E0)	**5.99E-3** (4.7E-4)	6.05E-3 (1.9E-1)	6.03E-3 (4.1E-4)	3.00E0 * (2.4E0)
DTLZ7	3	1.00E-1 (1.7E-2)	1.06E-1 (1.2E-2)	9.95E-2 (1.1E-2)	9.60E-2 (2.1E-2)	1.01E-1 (1.3E-2)
DTLZ1	4	1.90E1 (4.7E1)	2.34E1 (3.3E1)	2.11E1 (2.7E1)	1.80E1 (2.5E1)	2.12E1 (2.6E1)
DTLZ2	4	8.00E-1 (2.0E-1)	8.83E-1 (1.6E-1)	8.01E-1 (1.2E-1)	8.58E-1 (1.5E-1)	8.20E-1 (1.9E-1)
DTLZ3	4	2.52E1 (9.5E1)	5.03E1 (1.0E2)	2.58E1 (7.4E1)	4.82E1 (6.3E1)	5.14E1 (6.7E1)
DTLZ4	4	4.04E-1 (7.9E-2)	4.18E-1 (4.4E-2)	4.04E-1 (6.4E-2)	4.02E-1 (3.1E-2)	4.21E-1 (4.0E-2)
DTLZ5	4	2.86E-1 (8.7E-2)	2.81E-1 (5.8E-2)	2.79E-1 (9.1E-2)	2.79E-1 (1.5E-1)	3.24E-1 (1.2E-1)
DTLZ6	4	1.51E1 (4.8E0)	1.51E1 (4.9E0)	1.48E1 (2.7E0)	1.48E1 (3.9E0)	1.63E1 (5.8E0)
DTLZ7	4	3.34E-1 * (8.1E-2)	3.57E-1 * (3.5E-2)	3.33E-1 * (4.3E-2)	**3.09E-1** (3.0E-2)	3.56E-1 * (2.7E-2)
DTLZ1	5	1.25E2 (1.6E2)	1.53E2 (1.7E2)	1.12E2 (2.1E2)	1.33E2 (2.0E2)	1.81E2 (8.7E1)
DTLZ2	5	1.15E0 (3.7E-1)	1.11E0 (3.4E-1)	1.08E0 (2.6E-1)	1.20E0 (4.4E-1)	1.27E0 (3.7E-1)
DTLZ3	5	2.62E2 (3.9E2)	3.54E2 * (2.9E2)	2.25E2 (2.9E2)	**1.69E2** (2.0E2)	2.51E2 (3.0E2)
DTLZ4	5	7.29E-1 (2.1E-1)	7.76E-1 (1.5E-1)	8.16E-1 (2.4E-1)	7.81E-1 (1.9E-1)	8.15E-1 (2.1E-1)
DTLZ5	5	7.32E-1 (6.8E-1)	4.53E-1 (4.9E-1)	4.12E-1 (6.9E-1)	4.53E-1 (6.1E-1)	5.85E-1 (6.3E-1)
DTLZ6	5	1.53E1 (3.9E0)	1.73E1 (4.1E0)	1.53E1 (4.0E0)	1.52E1 (6.0E0)	1.68E1 (5.9E0)
DTLZ7	5	7.29E-1 * (2.5E-1)	6.41E-1 * (1.2E-1)	6.18E-1 * (1.9E-1)	**5.16E-1** (3.5E-2)	6.85E-1 * (2.0E-1)
n = 1000						
DTLZ1	2	2.59E3 * (1.5E3)	2.48E3 * (1.4E3)	1.86E3 (2.1E3)	**1.67E3** (1.1E3)	1.85E3 (1.9E3)
DTLZ2	2	3.12E0 * (3.9E-1)	2.05E0 * (3.0E-1)	3.39E0 * (9.1E-1)	**5.40E-3** (4.1E-4)	2.77E0 * (6.2E-1)
DTLZ3	2	5.03E3 (5.4E3)	6.02E3 (3.5E3)	4.32E3 (5.9E3)	3.91E3 (3.4E3)	4.66E3 (6.3E3)
DTLZ4	2	3.53E0 * (1.1E0)	2.26E0 * (1.5E0)	4.31E0 * (1.5E0)	**5.41E-3** (5.1E-4)	3.24E0 * (1.0E0)
DTLZ5	2	3.09E0 * (7.5E-1)	1.99E0 * (3.2E-1)	3.21E0 * (5.7E-1)	**5.29E-3** (3.8E-4)	2.90E0 * (5.0E-1)
DTLZ6	2	4.23E2 * (2.7E1)	1.23E2 * (3.5E1)	4.13E2 * (3.1E1)	**6.28E-3** (7.3E-1)	4.75E2 * (1.3E1)
DTLZ7	2	4.88E0 * (3.1E-1)	**6.13E-1** (9.2E-2)	4.74E0 * (3.2E-1)	8.10E-1 (3.6E-1)	4.39E0 * (2.1E-1)
DTLZ1	3	6.03E2 (8.8E2)	8.54E2 (8.7E2)	5.79E2 (1.1E3)	8.91E2 (6.3E2)	9.74E2 (1.1E3)
DTLZ2	3	4.83E0 * (1.1E0)	5.69E0 * (1.6E0)	4.91E0 * (9.5E-1)	**8.62E-2** (1.1E-2)	5.36E0 * (1.1E0)
DTLZ3	3	1.92E3 (1.7E3)	1.04E3 (1.3E3)	1.49E3 (1.6E3)	1.49E3 (1.2E3)	1.63E3 (2.1E3)
DTLZ4	3	2.36E0 * (1.3E0)	1.96E0 * (8.9E-1)	2.19E0 * (9.5E-1)	**5.41E-1** (1.7E-1)	2.54E0 * (1.0E0)
DTLZ5	3	5.46E0 * (1.5E0)	6.90E0 * (1.2E0)	5.37E0 * (1.4E0)	**6.38E-3** (1.1E-3)	7.00E0 * (1.5E0)
DTLZ6	3	4.84E2 * (1.4E1)	2.02E2 * (6.2E1)	4.78E2 * (1.7E1)	**1.83E0** (7.4E0)	4.81E2 * (2.0E1)
DTLZ7	3	7.90E0 * (5.0E-1)	1.97E0 * (8.3E-1)	7.88E0 * (2.5E-1)	**1.53E0** (1.4E0)	7.92E0 * (5.4E-1)
DTLZ1	4	8.47E2 (1.2E3)	7.06E2 (9.9E2)	9.07E2 (9.7E2)	1.11E3 (3.1E3)	8.64E2 (1.1E3)
DTLZ2	4	1.80E1 (4.7E0)	1.91E1 * (7.0E0)	1.74E1 (5.4E0)	**1.65E1** (5.5E0)	1.91E1 (5.6E0)
DTLZ3	4	1.62E3 (2.0E3)	2.16E3 (2.0E3)	1.78E3 (2.2E3)	1.67E3 (1.8E3)	1.73E3 (1.0E3)
DTLZ4	4	6.23E0 * (2.1E0)	7.69E0 * (2.5E0)	6.17E0 * (3.1E0)	**2.07E0** (1.0E0)	7.38E0 * (4.2E0)
DTLZ5	4	2.67E1 (1.3E1)	3.63E1 (2.4E1)	3.54E1 (1.9E1)	3.52E1 (2.8E1)	3.50E1 (1.3E1)
DTLZ6	4	5.00E2 (2.4E1)	5.03E2 * (1.8E1)	4.96E2 (2.6E1)	**4.88E2** (3.5E1)	4.94E2 (2.4E1)
DTLZ7	4	1.16E1 * (6.5E-1)	8.40E0 * (1.3E0)	1.18E1 * (7.6E-1)	**3.43E-1** (7.0E-2)	1.16E1 * (3.3E-1)
DTLZ1	5	2.97E3 (5.9E3)	4.49E3 (7.1E3)	4.28E3 (5.1E3)	5.44E3 (6.5E3)	4.31E3 (3.0E3)
DTLZ2	5	2.90E1 (1.5E1)	3.05E1 (1.3E1)	2.67E1 (1.4E1)	2.76E1 (1.4E1)	2.67E1 (1.2E1)
DTLZ3	5	8.22E3 (7.3E3)	9.66E3 (1.0E4)	9.19E3 (6.7E3)	9.11E3 (5.8E3)	1.07E4 (6.8E3)
DTLZ4	5	1.48E1 (6.0E0)	1.56E1 (6.0E0)	1.41E1 (6.9E0)	1.45E1 (8.8E0)	1.70E1 (4.7E0)
DTLZ5	5	4.56E1 (1.6E1)	4.62E1 (1.6E1)	4.05E1 (3.1E1)	4.83E1 (1.7E1)	4.11E1 (1.5E1)
DTLZ6	5	4.97E2 (2.8E1)	5.00E2 (2.5E1)	4.97E2 (1.5E1)	4.99E2 (2.9E1)	5.02E2 (1.8E1)
DTLZ7	5	1.55E1 * (7.6E-1)	1.24E1 * (1.9E0)	1.53E1 * (4.4E-1)	**5.90E-1** (1.0E-1)	1.55E1 * (5.5E-1)

Table B.21: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100, 000 function evaluations.

		SMPSO	xSMPSO	NSGA-II	xNSGA-II	NSGA-III	xNSGA-III
n = 200							
LSMOP1	2	8.99E-1 * (1.5E-1)	**2.12E-1** (1.6E-2)	3.13E-1 * (2.8E-2)	5.67E-1 * (3.3E-1)	5.03E-1 * (1.0E-1)	6.87E-1 * (8.3E-3)
LSMOP2	2	9.09E-2 * (3.7E-3)	**2.53E-2** (1.4E-3)	9.68E-2 * (2.9E-3)	7.73E-2 * (1.1E-3)	9.30E-2 * (3.7E-3)	7.03E-2 * (1.1E-3)
LSMOP3	2	2.43E1 * (2.5E0)	1.49E0 * (1.0E-2)	1.25E1 * (3.1E0)	**9.39E-1** (9.6E-2)	1.33E1 * (2.9E0)	1.03E0 * (1.3E-1)
LSMOP4	2	1.10E-1 * (2.3E-2)	**6.80E-2** (6.6E-3)	1.45E-1 * (4.7E-3)	1.43E-1 * (3.1E-3)	1.40E-1 * (3.5E-3)	1.42E-1 * (3.7E-3)
LSMOP5	2	1.10E0 * (3.0E-1)	7.42E-1 * (1.4E-2)	**3.96E-1** (9.4E-2)	7.42E-1 * (—)	4.76E-1 * (1.0E-1)	7.42E-1 * (—)
LSMOP6	2	5.17E-1 * (8.1E-2)	**1.15E-1** (1.9E-2)	8.86E-1 * (3.2E-2)	6.84E-1 * (7.5E-3)	8.88E-1 * (1.1E-2)	6.88E-1 * (1.4E-2)
LSMOP7	2	6.26E1 * (2.2E1)	1.48E0 * (3.8E-3)	3.62E0 * (1.2E0)	**1.47E0** (2.7E-3)	8.22E0 * (1.9E1)	1.47E0 (5.0E-3)
LSMOP8	2	7.85E-1 * (1.5E-1)	**2.31E-1** (8.6E-2)	3.86E-1 * (6.2E-2)	7.42E-1 * (—)	4.51E-1 * (9.8E-2)	7.42E-1 * (—)
LSMOP9	2	**4.26E-1** (1.2E-2)	8.10E-1 * (—)	1.34E0 * (1.2E-1)	8.10E-1 * (—)	1.50E0 * (1.2E-1)	8.10E-1 * (—)
n = 300							
LSMOP1	3	2.21E0 * (4.0E-1)	**3.66E-1** (1.1E-2)	2.21E0 * (3.8E-1)	4.66E-1 * (5.8E-2)	8.65E-1 * (1.9E-1)	5.27E-1 * (3.3E-2)
LSMOP2	3	9.71E-2 * (4.6E-3)	**7.89E-2** (5.2E-3)	1.07E-1 * (4.1E-3)	1.15E-1 * (4.7E-3)	8.32E-2 * (1.2E-3)	8.69E-2 * (7.8E-4)
LSMOP3	3	1.48E1 * (4.1E0)	8.49E-1 (1.0E-1)	7.51E0 * (2.3E0)	**7.90E-1** (4.9E-2)	3.95E0 * (1.2E0)	8.37E-1 * (2.4E-2)
LSMOP4	3	2.60E-1 * (8.6E-3)	2.03E-1 (1.1E-2)	2.64E-1 * (1.3E-2)	2.94E-1 * (1.0E-2)	**1.97E-1** (5.0E-3)	2.10E-1 * (5.0E-3)
LSMOP5	3	6.19E0 * (1.5E0)	**5.18E-1** (2.0E-2)	5.96E0 * (9.8E-1)	5.41E-1 * (8.1E-5)	3.27E0 * (9.6E-1)	5.42E-1 * (3.2E-4)
LSMOP6	3	2.20E3 * (1.8E3)	1.25E0 (3.3E-1)	7.22E1 * (1.5E2)	1.24E0 (3.3E-3)	6.25E0 * (6.8E0)	**1.24E0** (3.4E-3)
LSMOP7	3	1.52E0 * (2.6E-2)	**9.38E-1** (1.8E-2)	1.55E0 * (7.4E-2)	9.71E-1 * (9.9E-3)	1.53E0 * (2.5E-2)	9.63E-1 * (1.0E-2)
LSMOP8	3	9.80E-1 * (7.5E-2)	**3.59E-1** (2.9E-2)	4.62E-1 * (7.5E-2)	3.62E-1 * (9.9E-4)	3.67E-1 * (4.8E-2)	3.63E-1 * (4.2E-4)
LSMOP9	3	1.56E1 * (2.1E0)	1.53E0 * (—)	2.94E0 * (1.1E-1)	1.53E0 * (—)	2.91E0 * (8.3E-2)	**1.49E0** (2.3E-1)
n = 400							
LSMOP1	4	4.49E0 * (1.6E0)	8.14E-1 (1.5E-1)	5.59E0 * (6.8E-1)	7.73E-1 * (3.7E-2)	3.15E0 * (3.7E-1)	**7.29E-1** (6.8E-2)
LSMOP2	4	1.70E-1 * (5.6E-3)	**1.50E-1** (8.2E-3)	1.81E-1 * (1.1E-2)	1.89E-1 * (9.3E-3)	1.52E-1 (1.8E-3)	1.51E-1 (1.8E-3)
LSMOP3	4	1.96E1 * (2.7E0)	1.79E0 * (3.7E-3)	1.45E1 * (2.0E0)	1.74E0 * (1.0E-1)	1.71E1 * (4.5E0)	**1.66E0** (1.3E-1)
LSMOP4	4	2.52E-1 * (1.8E-2)	2.35E-1 * (1.3E-2)	2.62E-1 * (1.5E-2)	2.80E-1 * (1.0E-2)	**2.01E-1** (3.8E-3)	2.19E-1 * (3.9E-3)
LSMOP5	4	1.83E1 * (6.0E0)	4.60E-1 * (4.9E-3)	1.82E1 * (1.6E0)	4.67E-1 * (7.2E-3)	5.76E0 * (1.0E0)	**4.57E-1** (1.1E-3)
LSMOP6	4	1.27E0 * (1.4E-4)	8.83E-1 (3.5E-2)	1.26E0 * (7.9E-3)	8.92E-1 * (3.5E-3)	1.22E0 * (1.0E-2)	**8.79E-1** (3.5E-3)
LSMOP7	4	5.34E4 * (5.1E4)	1.22E0 * (1.3E-2)	2.05E4 * (1.0E4)	1.23E0 * (6.6E-3)	5.21E2 * (2.9E2)	**1.21E0** (2.0E-3)
LSMOP8	4	1.21E1 * (3.6E0)	4.59E-1 (1.1E-2)	9.32E0 * (1.2E0)	4.66E-1 * (4.8E-3)	2.10E0 * (6.3E-1)	**4.57E-1** (1.4E-3)
LSMOP9	4	1.35E1 * (2.4E0)	2.24E0 * (—)	5.77E0 * (7.9E-1)	2.24E0 * (—)	7.19E0 * (1.0E0)	**6.40E-1** (6.1E-1)
n = 500							
LSMOP1	5	4.97E0 * (1.8E0)	9.33E-1 (1.8E-2)	8.76E0 * (1.1E0)	9.33E-1 (2.1E-2)	3.00E0 * (5.8E-1)	**9.32E-1** (1.4E-2)
LSMOP2	5	2.06E-1 * (1.1E-2)	1.93E-1 * (9.2E-3)	2.17E-1 * (1.3E-2)	2.20E-1 * (9.8E-3)	1.74E-1 (6.3E-4)	**1.74E-1** (9.8E-4)
LSMOP3	5	1.98E1 * (2.5E0)	**9.58E-1** (—)	1.87E1 * (2.7E0)	9.58E-1 (—)	1.04E1 * (1.5E0)	9.58E-1 (—)
LSMOP4	5	3.44E-1 * (2.5E-2)	2.96E-1 * (8.1E-3)	3.75E-1 * (3.5E-2)	3.51E-1 * (1.2E-2)	2.93E-1 (5.6E-3)	**2.90E-1** (3.7E-3)
LSMOP5	5	2.21E1 * (9.3E0)	6.01E-1 * (1.0E-1)	2.03E1 * (3.8E0)	5.05E-1 * (4.0E-2)	6.97E0 * (1.5E0)	**3.81E-1** (1.3E-2)
LSMOP6	5	7.96E4 * (4.2E4)	1.32E0 * (1.1E-1)	5.14E4 * (9.8E3)	1.26E0 * (5.8E-2)	2.31E1 * (3.5E1)	**1.11E0** (5.0E-3)
LSMOP7	5	3.34E0 * (2.0E-1)	1.26E0 * (5.9E-2)	3.46E0 * (1.1E-1)	1.12E0 * (3.5E-2)	2.66E0 * (1.8E-1)	**1.05E0** (1.4E-2)
LSMOP8	5	1.21E0 * (1.6E-2)	5.18E-1 * (5.0E-2)	1.21E0 * (1.6E-2)	4.11E-1 * (2.3E-2)	1.15E0 * (1.1E-2)	**3.61E-1** (2.1E-3)
LSMOP9	5	5.42E1 * (4.9E0)	3.00E0 * (6.6E-1)	3.07E1 * (1.0E1)	3.00E0 * (—)	1.33E1 * (2.4E0)	**6.01E-1** (3.3E-2)
n = 1000							
LSMOP1	2	1.74E0 * (9.9E-2)	**2.99E-1** (3.7E-3)	3.56E0 * (4.3E-1)	5.75E-1 * (1.4E-1)	2.97E0 * (3.5E-1)	6.89E-1 * (5.9E-3)
LSMOP2	2	2.54E-2 * (6.2E-4)	**8.91E-3** (3.0E-4)	3.61E-2 * (5.1E-4)	1.91E-2 * (3.2E-4)	3.42E-2 * (4.1E-4)	1.65E-2 * (8.5E-5)
LSMOP3	2	2.78E1 * (9.8E-1)	**1.56E0** (1.0E-3)	2.12E1 * (2.3E0)	1.58E0 * (4.0E-4)	2.09E1 * (2.3E0)	1.58E0 * (3.4E-4)
LSMOP4	2	5.33E-2 * (7.0E-4)	**2.25E-2** (3.2E-4)	6.07E-2 * (9.9E-4)	5.91E-2 * (1.9E-3)	4.83E-2 * (8.5E-4)	5.37E-2 * (8.2E-4)
LSMOP5	2	3.92E0 * (2.8E-1)	**7.42E-1** (—)	1.06E1 * (1.0E0)	7.42E-1 (—)	8.85E0 * (1.0E0)	7.42E-1 (—)
LSMOP6	2	7.58E-1 * (1.8E-3)	**1.74E-1** (4.8E-3)	7.74E-1 * (9.6E-4)	6.77E-1 * (3.1E-4)	7.73E-1 * (4.8E-4)	6.77E-1 * (3.6E-4)
LSMOP7	2	1.90E3 * (3.6E2)	**1.51E0** (3.2E-4)	2.54E3 * (2.6E3)	1.51E0 * (4.0E-4)	5.61E3 * (1.8E3)	1.51E0 * (4.2E-4)
LSMOP8	2	2.98E0 * (3.5E-1)	**7.42E-1** (—)	4.89E0 * (5.9E-1)	7.42E-1 (—)	3.69E0 * (4.0E-1)	7.42E-1 (—)
LSMOP9	2	2.52E0 * (4.8E-1)	**5.61E-1** (1.5E-2)	1.43E0 * (1.3E-1)	8.09E-1 * (5.8E-4)	1.37E0 * (1.6E-1)	8.09E-1 * (5.9E-4)
LSMOP1	3	2.55E0 * (6.8E-1)	**3.75E-1** (1.8E-2)	6.17E0 * (8.7E-1)	6.71E-1 * (3.9E-2)	2.97E0 * (3.0E-1)	7.35E-1 * (2.7E-2)
LSMOP2	3	6.35E-2 * (4.1E-3)	6.13E-2 * (4.8E-3)	7.11E-2 * (3.7E-3)	7.49E-2 * (6.6E-3)	5.17E-2 * (1.1E-4)	**5.11E-2** (3.0E-4)
LSMOP3	3	1.64E1 * (6.4E0)	**8.57E-1** (1.9E-2)	2.00E1 * (6.7E0)	8.60E-1 * (—)	1.05E1 * (1.0E0)	8.60E-1 * (—)
LSMOP4	3	1.18E-1 * (5.2E-3)	**8.96E-2** (5.3E-3)	1.29E-1 * (6.9E-3)	1.45E-1 * (7.6E-3)	9.91E-2 * (1.1E-3)	1.05E-1 * (1.7E-3)
LSMOP5	3	6.08E0 * (5.4E-1)	**5.34E-1** (9.3E-3)	1.56E1 * (1.7E0)	5.41E-1 * (6.6E-5)	7.34E0 * (8.7E-1)	5.42E-1 * (3.6E-4)
LSMOP6	3	3.20E3 * (9.8E2)	**1.30E0** (4.7E-1)	1.08E4 * (3.9E3)	1.31E0 * (1.1E-3)	2.08E3 * (9.7E2)	1.31E0 * (7.6E-4)
LSMOP7	3	1.08E0 * (2.3E-3)	**8.58E-1** (4.4E-3)	1.10E0 * (5.1E-3)	8.69E-1 * (1.2E-3)	1.09E0 * (1.9E-3)	8.67E-1 * (1.3E-3)
LSMOP8	3	9.23E-1 * (1.2E-1)	**3.27E-1** (6.2E-2)	9.58E-1 * (4.2E-4)	3.43E-1 * (2.8E-2)	7.65E-1 * (3.8E-2)	3.59E-1 * (3.1E-4)
LSMOP9	3	2.63E1 * (1.9E0)	1.53E0 (9.4E-1)	1.37E1 * (1.1E0)	1.53E0 * (—)	1.27E1 * (2.0E0)	**1.52E0** (3.7E-1)
LSMOP1	4	5.50E0 * (2.9E0)	**7.13E-1** (1.7E-1)	8.18E0 * (7.9E-1)	8.64E-1 * (2.0E-2)	5.60E0 * (6.7E-1)	8.62E-1 * (4.2E-2)
LSMOP2	4	1.34E-1 * (5.6E-3)	1.29E-1 * (5.9E-3)	1.44E-1 * (9.9E-3)	1.44E-1 * (7.9E-3)	1.18E-1 * (3.4E-4)	**1.16E-1** (8.0E-4)
LSMOP3	4	1.97E1 * (2.2E0)	**1.81E0** (4.2E-3)	2.07E1 * (2.1E0)	1.81E0 (2.6E-5)	2.08E1 * (2.3E0)	1.81E0 (4.3E-4)
LSMOP4	4	1.72E-1 * (8.8E-3)	1.52E-1 * (8.6E-3)	1.79E-1 * (9.0E-3)	1.91E-1 * (1.0E-2)	**1.45E-1** (1.5E-3)	1.50E-1 * (1.8E-3)
LSMOP5	4	1.76E1 * (5.0E0)	4.58E-1 (7.5E-3)	2.11E1 * (9.2E-1)	4.66E-1 * (5.3E-3)	1.02E1 * (9.0E-1)	**4.57E-1** (1.2E-3)
LSMOP6	4	1.12E0 * (3.1E-12)	9.04E-1 (8.9E-2)	1.12E0 * (4.8E-4)	9.05E-1 * (3.4E-3)	1.11E0 * (1.6E-3)	**8.99E-1** (9.4E-4)
LSMOP7	4	5.46E4 * (4.1E4)	1.25E0 * (1.1E-2)	3.88E4 * (1.0E4)	1.25E0 * (6.3E-3)	4.33E3 * (1.9E3)	**1.23E0** (1.0E-3)
LSMOP8	4	1.26E1 * (4.6E0)	**4.54E-1** (1.0E-2)	1.25E1 * (8.4E-1)	4.64E-1 * (4.4E-3)	4.34E0 * (5.3E-1)	4.57E-1 (8.8E-4)
LSMOP9	4	1.47E1 * (3.1E0)	2.24E0 * (—)	2.28E1 * (4.2E0)	2.24E0 * (—)	2.30E1 * (2.9E0)	**6.54E-1** (1.6E0)
LSMOP1	5	6.42E0 * (1.7E0)	**9.32E-1** (1.5E-2)	9.61E0 * (1.1E0)	9.39E-1 * (4.8E-3)	5.08E0 * (7.1E-1)	9.33E-1 (6.1E-3)
LSMOP2	5	1.85E-1 * (8.3E-3)	1.83E-1 * (7.1E-3)	1.91E-1 * (9.9E-3)	1.93E-1 * (1.5E-2)	1.53E-1 * (1.3E-4)	**1.52E-1** (4.5E-4)
LSMOP3	5	2.10E1 * (3.0E0)	**9.58E-1** (—)	2.35E1 * (2.2E0)	9.58E-1 (—)	1.30E1 * (3.0E0)	9.58E-1 (—)
LSMOP4	5	2.54E-1 * (1.6E-2)	2.35E-1 * (1.1E-2)	2.86E-1 * (2.8E-2)	2.72E-1 * (1.1E-2)	**2.24E-1** (1.6E-3)	2.26E-1 * (2.0E-3)
LSMOP5	5	2.04E1 * (6.4E0)	5.69E-1 * (9.9E-2)	2.10E1 * (2.4E0)	4.88E-1 * (2.5E-2)	9.91E0 * (7.3E-1)	**3.78E-1** (1.1E-2)
LSMOP6	5	9.26E4 * (3.4E4)	1.37E0 * (1.2E-1)	5.58E4 * (1.1E4)	1.29E0 * (1.2E-1)	2.15E2 * (2.5E2)	**1.17E0** (1.7E-3)
LSMOP7	5	2.05E0 * (5.0E-2)	1.13E0 * (4.3E-2)	2.09E0 * (3.7E-2)	1.05E0 * (3.2E-2)	1.84E0 * (3.5E-2)	**9.99E-1** (6.0E-3)
LSMOP8	5	1.15E0 * (2.1E-3)	5.18E-1 * (9.0E-2)	1.15E0 * (1.5E-3)	3.99E-1 * (1.5E-2)	1.15E0 * (2.8E-3)	**3.59E-1** (3.2E-3)
LSMOP9	5	6.17E1 * (4.0E0)	3.00E0 * (—)	7.37E1 * (9.5E0)	3.00E0 * (—)	4.83E1 * (7.7E0)	**6.75E-1** (4.0E-1)

Table B.22: Performance comparison using the IGD indicator on the UF benchmarks using 100,000 function evaluations.

		SMPSO	xSMPSO	NSGA-II	xNSGA-II	NSGA-III	xNSGA-III
n = 40							
UF1	2	1.17E-1 * (3.6E-2)	1.53E-1 * (3.6E-2)	1.09E-1 (2.9E-2)	1.05E-1 (1.4E-2)	**1.04E-1** (2.3E-2)	1.06E-1 (3.7E-2)
UF2	2	6.40E-2 * (9.7E-3)	6.55E-2 * (9.1E-3)	4.35E-2 (9.5E-3)	4.36E-2 (1.6E-2)	**4.21E-2** (1.0E-2)	4.39E-2 (1.0E-2)
UF3	2	2.14E-1 * (1.3E-1)	**1.64E-1** (2.4E-2)	2.33E-1 * (7.4E-2)	1.74E-1 (2.0E-2)	2.26E-1 * (1.0E-1)	1.88E-1 * (2.0E-2)
UF4	2	1.01E-1 * (1.9E-2)	7.45E-2 * (1.1E-2)	4.90E-2 (2.0E-3)	4.74E-2 (2.4E-3)	4.98E-2 * (2.6E-3)	**4.71E-2** (4.1E-3)
UF5	2	1.48E0 * (1.2E0)	1.45E0 * (1.4E0)	3.20E-1 * (1.4E-1)	3.22E-1 (1.1E-1)	**2.68E-1** (8.3E-2)	3.26E-1 (1.2E-1)
UF6	2	3.80E-1 * (1.4E-1)	4.01E-1 * (1.2E-1)	1.18E-1 (9.3E-2)	1.10E-1 (2.1E-2)	**1.08E-1** (1.9E-2)	1.10E-1 (2.9E-2)
UF7	2	5.21E-2 (2.4E-1)	5.03E-2 (2.2E-2)	5.36E-2 (2.8E-1)	5.19E-2 (1.7E-2)	5.60E-2 (4.0E-2)	5.05E-2 (1.4E-2)
UF8	3	4.55E-1 * (1.4E-1)	4.19E-1 * (1.1E-1)	3.08E-1 (3.4E-2)	**3.05E-1** (7.6E-2)	5.29E-1 * (1.4E-2)	5.37E-1 * (1.3E-2)
UF9	3	5.18E-1 * (1.0E-1)	5.75E-1 * (7.2E-2)	**3.31E-1** (1.4E-1)	3.38E-1 (6.3E-2)	3.54E-1 (1.8E-1)	3.84E-1 (1.5E-1)
UF10	3	2.51E0 * (1.0E0)	2.63E0 * (7.4E-1)	4.59E-1 * (8.6E-2)	4.31E-1 * (1.6E-1)	**3.71E-1** (8.7E-2)	4.02E-1 (1.2E-1)
n = 1000							
UF1	2	1.39E0 * (1.5E-2)	1.39E0 * (1.9E-2)	**2.74E-1** (4.5E-2)	3.66E-1 * (7.1E-2)	2.85E-1 (5.6E-2)	3.46E-1 * (7.4E-2)
UF2	2	1.69E-1 * (6.0E-3)	**1.66E-1** (2.7E-2)	2.31E-1 * (1.4E-2)	2.57E-1 * (1.9E-2)	2.25E-1 * (1.2E-2)	2.37E-1 * (1.0E-2)
UF3	2	3.28E-1 * (6.2E-3)	**1.23E-1** (1.1E-3)	2.71E-1 * (8.2E-3)	2.25E-1 * (2.7E-2)	2.80E-1 * (6.1E-3)	2.38E-1 * (9.3E-3)
UF4	2	1.36E-1 * (9.4E-4)	1.30E-1 * (3.4E-3)	1.63E-1 * (3.8E-3)	**8.73E-2** (1.0E-2)	1.62E-1 * (2.4E-3)	8.99E-2 (7.7E-3)
UF5	2	5.55E0 * (6.9E-2)	5.54E0 * (1.3E-1)	1.96E0 (3.2E-1)	2.16E0 * (4.8E-1)	**1.91E0** (7.0E-1)	2.30E0 * (6.4E-1)
UF6	2	5.48E0 * (3.3E-1)	5.61E0 * (2.2E-1)	**8.65E-1** (2.3E-1)	1.20E0 * (2.3E-1)	9.21E-1 (2.8E-1)	1.28E0 * (2.3E-1)
UF7	2	1.41E0 * (2.9E-2)	1.42E0 * (2.2E-2)	3.35E-1 (9.3E-2)	4.21E-1 * (2.5E-1)	**2.95E-1** (7.9E-2)	3.85E-1 * (1.0E-1)
UF8	3	5.64E-1 * (4.5E-2)	**4.89E-1** (2.9E-2)	9.60E-1 * (7.6E-2)	1.08E0 * (9.3E-2)	8.49E-1 * (7.5E-2)	9.18E-1 * (6.3E-2)
UF9	3	7.96E-1 * (2.8E-2)	**7.08E-1** (1.7E-2)	7.78E-1 * (3.9E-2)	9.01E-1 * (7.1E-2)	8.50E-1 * (3.5E-2)	9.08E-1 * (4.4E-2)
UF10	3	4.85E0 * (4.7E-1)	**2.07E0** (2.2E0)	3.61E0 * (1.3E0)	4.23E0 * (1.6E0)	2.94E0 (4.5E-1)	3.78E0 * (3.7E-1)

Table B.23: Performance comparison using the IGD indicator on the WFG benchmarks using 100,000 function evaluations.

		SMPSO	xSMPSO	NSGA-II	xNSGA-II	NSGA-III	xNSGA-III
n = 40							
WFG1	2	1.23E0 * (2.4E-2)	1.23E0 * (3.0E-2)	**4.80E-2** (5.6E-2)	9.29E-2 * (5.4E-2)	1.11E-1 * (5.1E-2)	2.02E-1 * (7.4E-2)
WFG2	2	2.61E-1 * (3.2E-2)	1.84E-1 * (5.7E-2)	6.66E-1 * (8.8E-3)	3.65E-2 (1.4E-2)	6.65E-1 * (4.4E-1)	**3.25E-2** (1.9E-2)
WFG3	2	6.08E-2 * (1.9E-2)	6.97E-2 * (2.8E-2)	**2.86E-2** (1.0E-2)	3.34E-2 (7.5E-3)	2.87E-2 (1.0E-2)	3.41E-2 (9.6E-3)
WFG4	2	9.72E-2 * (1.0E-2)	1.13E-1 * (1.6E-2)	2.09E-2 * (1.6E-3)	2.17E-2 * (1.7E-3)	**1.72E-2** (2.1E-3)	1.90E-2 * (2.2E-3)
WFG5	2	8.04E-2 * (3.2E-2)	8.78E-2 * (3.6E-2)	7.21E-2 * (7.3E-4)	7.20E-2 * (7.6E-4)	**7.05E-2** (5.4E-4)	7.08E-2 (1.5E-3)
WFG6	2	4.72E-2 * (1.3E-3)	**2.87E-2** (4.1E-3)	5.76E-2 * (1.1E-2)	5.73E-2 * (3.0E-2)	5.73E-2 * (1.8E-2)	8.09E-2 * (2.1E-6)
WFG7	2	2.30E-2 * (5.9E-3)	3.63E-2 * (1.5E-2)	1.75E-2 (1.0E-3)	1.82E-2 * (2.5E-3)	**1.50E-2** (6.1E-3)	1.71E-2 (8.2E-3)
WFG8	2	3.38E-1 * (4.9E-2)	3.61E-1 * (5.5E-2)	**2.37E-1** (9.1E-3)	2.43E-1 * (6.5E-3)	2.54E-1 * (1.7E-2)	2.49E-1 * (1.0E-2)
WFG9	2	9.42E-2 * (1.5E-2)	**4.82E-2** (1.1E-2)	9.28E-2 * (1.4E-3)	9.33E-2 * (1.4E-3)	9.27E-2 * (2.2E-3)	9.47E-2 * (5.3E-3)
WFG1	3	1.54E0 * (1.8E-2)	1.53E0 * (1.8E-2)	**6.33E-1** (7.6E-2)	7.79E-1 * (9.8E-2)	8.90E-1 * (1.3E-1)	1.03E0 * (8.6E-2)
WFG2	3	3.13E-1 * (9.2E-2)	3.01E-1 * (1.0E-1)	5.13E-1 * (1.6E-2)	2.27E-1 * (2.0E-2)	4.88E-1 * (5.7E-3)	**1.74E-1** (8.4E-3)
WFG3	3	2.36E-1 * (6.3E-2)	1.44E-1 * (5.4E-2)	1.33E-1 * (2.9E-2)	1.44E-1 * (2.7E-2)	**9.94E-2** (3.6E-2)	1.57E-1 * (3.9E-2)
WFG4	3	4.16E-1 * (3.5E-2)	4.24E-1 * (4.4E-2)	2.85E-1 * (1.2E-2)	2.88E-1 * (9.2E-3)	**2.30E-1** (3.6E-3)	2.36E-1 * (5.0E-3)
WFG5	3	4.73E-1 * (9.9E-2)	4.68E-1 * (1.3E-1)	2.92E-1 * (1.0E-2)	2.92E-1 * (1.7E-2)	2.38E-1 (2.5E-3)	**2.37E-1** (3.1E-3)
WFG6	3	3.38E-1 * (4.2E-2)	3.29E-1 * (3.5E-2)	3.05E-1 * (1.7E-2)	3.35E-1 * (2.9E-2)	**2.36E-1** (8.9E-3)	2.40E-1 * (2.3E-3)
WFG7	3	4.55E-1 * (6.2E-2)	4.91E-1 * (6.1E-2)	2.87E-1 * (2.0E-2)	2.90E-1 * (3.0E-2)	**2.29E-1** (1.2E-2)	2.32E-1 (1.3E-2)
WFG8	3	7.19E-1 * (4.5E-2)	7.18E-1 * (4.1E-2)	4.52E-1 * (1.5E-2)	4.55E-1 * (2.4E-2)	**3.53E-1** (9.8E-3)	3.64E-1 * (1.2E-2)
WFG9	3	3.96E-1 * (4.0E-2)	3.80E-1 * (6.2E-2)	3.22E-1 * (1.3E-2)	3.28E-1 * (2.2E-2)	**2.46E-1** (4.7E-3)	2.49E-1 * (4.4E-3)
n = 1000							
WFG1	2	1.31E0 * (1.2E-2)	**1.26E0** (6.6E-3)	1.70E0 * (5.4E-2)	1.30E0 * (1.3E-3)	1.64E0 * (4.6E-2)	1.29E0 * (1.6E-3)
WFG2	2	8.91E-1 * (5.4E-1)	**4.39E-1** (8.5E-3)	8.97E-1 * (2.5E-2)	6.51E-1 * (2.5E-2)	8.84E-1 * (2.5E-2)	6.54E-1 * (1.4E-2)
WFG3	2	9.00E-1 * (5.4E-2)	**4.76E-1** (7.8E-2)	8.37E-1 * (4.5E-2)	5.17E-1 * (1.7E-2)	8.01E-1 * (3.1E-2)	6.10E-1 * (1.6E-2)
WFG4	2	4.92E-1 * (1.4E-2)	5.03E-1 * (1.5E-2)	9.72E-1 * (6.3E-2)	**3.89E-1** (1.3E-2)	9.14E-1 * (5.7E-2)	4.16E-1 * (1.7E-2)
WFG5	2	6.07E-1 * (1.9E-2)	5.31E-1 (1.9E-2)	9.75E-1 * (5.1E-2)	5.24E-1 (5.5E-2)	8.88E-1 * (3.9E-2)	**5.15E-1** (5.9E-2)
WFG6	2	2.87E-1 * (1.6E-2)	1.83E-2 * (2.0E-3)	1.04E0 * (7.6E-2)	2.16E-2 * (3.0E-3)	9.52E-1 * (6.3E-2)	**1.33E-2** (1.4E-4)
WFG7	2	9.37E-1 * (3.4E-2)	5.96E-1 * (1.5E-2)	9.17E-1 * (2.7E-2)	**4.52E-1** (1.1E-2)	8.83E-1 * (6.6E-2)	5.11E-1 * (2.1E-2)
WFG8	2	1.27E0 * (8.1E-2)	**3.69E-1** (6.5E-2)	1.09E0 * (3.4E-2)	6.07E-1 * (1.6E-2)	1.04E0 * (3.9E-2)	6.29E-1 * (1.2E-2)
WFG9	2	4.66E-1 * (7.4E-2)	**3.07E-1** (5.0E-2)	9.72E-1 * (7.1E-2)	3.56E-1 * (8.1E-2)	9.24E-1 * (7.9E-2)	3.84E-1 * (4.0E-2)
WFG1	3	1.75E0 * (4.5E-2)	1.68E0 * (3.5E-2)	1.81E0 * (4.8E-2)	1.63E0 * (7.3E-2)	1.84E0 * (4.7E-2)	**1.54E0** (2.9E-3)
WFG2	3	1.75E0 * (2.3E-2)	1.14E0 * (7.6E-2)	1.79E0 * (2.1E-2)	1.17E0 * (5.2E-2)	1.77E0 * (3.2E-2)	**1.00E0** (3.3E-2)
WFG3	3	8.63E-1 * (8.2E-2)	**6.80E-1** (6.7E-2)	8.73E-1 * (3.8E-2)	1.15E0 * (1.1E-2)	1.19E0 * (1.6E-1)	9.88E-1 * (1.6E-2)
WFG4	3	1.36E0 * (6.4E-2)	1.36E0 (7.3E-2)	1.87E0 * (6.3E-2)	**1.34E0** (4.6E-2)	2.87E0 * (1.0E-1)	1.66E0 * (5.9E-2)
WFG5	3	1.57E0 * (4.7E-2)	1.45E0 (4.5E-2)	1.67E0 * (5.9E-2)	**1.43E0** (6.1E-2)	1.62E0 * (8.5E-2)	2.68E0 * (6.3E-2)
WFG6	3	1.16E0 * (8.7E-2)	3.61E-1 * (4.0E-2)	1.58E0 * (5.8E-2)	3.56E-1 * (3.8E-2)	1.45E0 * (3.2E-2)	**2.23E-1** (1.0E-3)
WFG7	3	1.60E0 * (4.4E-2)	**1.23E0** (6.2E-2)	1.78E0 * (5.3E-2)	1.44E0 * (2.8E-2)	2.52E0 * (2.2E-1)	1.70E0 * (9.3E-2)
WFG8	3	1.79E0 * (7.8E-2)	**1.12E0** (6.6E-2)	1.83E0 * (5.7E-2)	1.44E0 * (2.9E-2)	1.89E0 * (7.5E-2)	1.34E0 * (2.4E-2)
WFG9	3	1.21E0 * (8.1E-2)	1.17E0 * (7.2E-2)	1.57E0 * (6.3E-2)	1.11E0 (1.2E-1)	1.35E0 * (9.7E-2)	**1.06E0** (2.0E-1)

Table B.24: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100, 000 function evaluations.

		SMPSO	xSMPSO	NSGA-II	xNSGA-II	NSGA-III	xNSGA-III
n = 40							
DTLZ1	2	2.63E1 * (8.9E1)	3.96E1 * (5.8E1)	**8.08E-1** (1.0E0)	2.64E0 * (1.0E0)	2.65E0 * (8.2E-1)	5.98E0 * (4.2E0)
DTLZ2	2	5.21E-3 * (1.8E-4)	5.25E-3 * (2.5E-4)	5.08E-3 * (2.6E-4)	5.13E-3 * (1.6E-4)	**3.96E-3** (7.7E-8)	3.96E-3 * (1.6E-7)
DTLZ3	2	1.08E2 * (1.6E2)	4.54E1 * (1.2E2)	**2.15E0** (1.1E0)	6.99E0 * (4.5E0)	5.29E0 * (4.0E0)	1.35E1 * (8.4E0)
DTLZ4	2	5.40E-3 * (7.3E-1)	5.30E-3 * (6.3E-4)	5.08E-3 * (3.8E-4)	5.09E-3 * (1.7E-4)	**3.96E-3** (3.2E-6)	3.96E-3 (2.9E-7)
DTLZ5	2	5.10E-3 * (3.0E-4)	5.40E-3 * (4.7E-4)	5.12E-3 * (1.3E-4)	5.02E-3 * (3.4E-4)	**3.96E-3** (6.7E-8)	3.96E-3 * (1.4E-7)
DTLZ6	2	5.20E-3 * (2.8E-4)	5.13E-3 * (3.7E-4)	6.04E-3 * (8.6E-3)	5.68E-3 * (3.6E-4)	**3.96E-3** (4.5E-8)	3.96E-3 (2.5E-8)
DTLZ7	2	5.49E-3 * (4.3E-1)	5.33E-3 * (3.6E-4)	5.30E-3 * (2.1E-4)	5.35E-3 * (3.4E-4)	**5.08E-3** (1.0E-4)	5.19E-3 * (1.4E-4)
DTLZ1	3	1.76E1 * (3.3E1)	4.17E1 * (4.7E1)	8.59E0 * (5.0E0)	1.54E1 * (5.1E0)	**6.45E0** (3.2E0)	1.10E1 * (5.3E0)
DTLZ2	3	9.53E-2 * (1.2E-2)	1.03E-1 * (2.0E-2)	7.26E-2 * (4.0E-3)	7.18E-2 * (3.1E-3)	**5.44E-2** (7.7E-7)	5.44E-2 * (2.5E-6)
DTLZ3	3	5.29E1 * (6.7E1)	2.51E1 (4.9E1)	1.63E1 (1.0E1)	2.67E1 * (1.7E1)	**1.33E1** (7.3E0)	2.85E1 * (1.0E1)
DTLZ4	3	2.58E-1 (2.4E-1)	2.64E-1 (4.9E-2)	7.22E-2 (3.4E-3)	7.18E-2 (2.7E-3)	5.44E-2 (4.8E-1)	5.44E-2 (6.7E-6)
DTLZ5	3	6.49E-3 (6.5E-4)	7.55E-3 * (1.5E-3)	**6.42E-3** (4.1E-4)	6.44E-3 (4.4E-4)	1.37E-2 * (2.2E-3)	1.37E-2 * (3.0E-3)
DTLZ6	3	6.35E-3 * (2.0E0)	**5.92E-3** (3.9E-4)	8.70E-3 * (8.5E-3)	6.46E-3 * (5.7E-4)	2.10E-2 * (4.8E-3)	2.08E-2 * (4.2E-3)
DTLZ7	3	1.00E-1 * (9.7E-3)	1.03E-1 * (1.5E-2)	8.19E-2 (5.9E-3)	7.96E-2 (7.3E-3)	**7.78E-2** (3.9E-3)	7.83E-2 (5.5E-3)
DTLZ1	4	1.54E1 (3.1E1)	2.07E1 * (2.4E1)	3.61E1 * (1.8E1)	2.08E2 * (5.1E1)	**1.18E1** (7.2E0)	1.67E1 * (8.0E0)
DTLZ2	4	8.92E-1 * (1.6E-1)	8.70E-1 * (1.8E-1)	1.64E-1 * (7.4E-3)	1.74E-1 * (8.6E-3)	**1.40E-1** (3.9E-6)	1.40E-1 * (9.7E-6)
DTLZ3	4	4.49E1 (7.6E1)	**2.53E1** (5.9E1)	7.18E1 * (4.1E1)	9.92E1 * (4.2E1)	2.80E1 (1.0E1)	5.24E1 * (2.9E1)
DTLZ4	4	4.10E-1 * (3.4E-2)	5.28E-1 * (8.1E-2)	1.64E-1 * (8.6E-3)	1.71E-1 * (8.8E-3)	**1.40E-1** (9.0E-5)	1.40E-1 (4.2E-5)
DTLZ5	4	3.06E-1 * (6.4E-2)	2.80E-1 * (7.9E-2)	1.32E-1 * (3.7E-2)	1.52E-1 * (4.3E-2)	**6.23E-2** (1.4E-2)	1.33E-1 * (5.3E-2)
DTLZ6	4	1.48E1 * (3.1E0)	**9.62E-2** (4.1E-2)	1.48E1 * (2.8E0)	2.00E-1 * (8.2E-2)	1.99E-1 * (7.7E-2)	2.63E-1 * (6.2E-2)
DTLZ7	4	3.36E-1 * (7.7E-2)	3.25E-1 * (1.7E-2)	2.33E-1 * (1.9E-2)	2.32E-1 * (1.2E-2)	**2.10E-1** (1.9E-2)	2.14E-1 (2.6E-2)
DTLZ1	5	1.12E2 * (2.1E2)	1.01E2 * (1.0E2)	5.75E1 * (2.0E1)	2.59E2 * (3.7E1)	**1.64E1** (7.2E0)	1.35E2 * (6.5E1)
DTLZ2	5	1.28E0 * (4.5E-1)	1.16E0 * (3.9E-1)	2.63E-1 * (8.3E-3)	3.01E-1 * (2.1E-2)	**2.12E-1** (1.2E-5)	2.12E-1 * (2.5E-5)
DTLZ3	5	2.74E2 * (3.8E2)	9.48E1 * (7.6E1)	1.97E2 * (5.4E1)	8.14E2 * (2.6E1)	**4.82E1** (2.8E1)	1.45E2 * (3.2E2)
DTLZ4	5	7.54E-1 * (2.6E-1)	9.51E-1 * (1.9E-1)	2.62E-1 (1.3E-2)	3.03E-1 (1.4E-2)	**2.12E-1** (2.2E-1)	2.12E-1 (7.9E-5)
DTLZ5	5	5.36E-1 * (4.9E-1)	4.96E-1 * (4.2E-1)	1.06E0 * (5.0E-1)	1.34E0 * (3.5E-1)	**1.21E-1** (3.6E-2)	2.90E-1 * (1.0E-1)
DTLZ6	5	1.53E1 * (4.0E0)	**1.10E-1** (5.8E-2)	2.34E1 * (1.1E0)	3.42E-1 * (1.0E-1)	1.24E0 * (8.1E-1)	3.30E-1 * (9.5E-2)
DTLZ7	5	7.67E-1 * (1.2E-1)	5.14E-1 * (2.3E-2)	4.06E-1 * (2.4E-2)	4.00E-1 (2.3E-2)	**3.85E-1** (3.4E-2)	3.88E-1 (2.3E-2)
n = 1000							
DTLZ1	2	2.05E3 * (1.4E3)	**1.28E3** (1.1E3)	4.37E3 * (1.9E2)	2.63E3 * (7.6E3)	4.85E3 * (2.5E2)	2.57E3 * (4.9E3)
DTLZ2	2	3.22E0 * (7.8E-1)	3.60E0 * (8.0E-1)	**1.97E0** (2.3E-1)	3.87E0 * (6.7E-1)	2.04E0 (2.2E-1)	3.82E0 * (5.2E-1)
DTLZ3	2	5.90E3 * (5.8E3)	**2.20E3** (3.6E3)	1.16E4 * (5.3E2)	2.39E4 * (2.3E2)	1.29E4 * (5.6E2)	4.23E3 * (6.5E3)
DTLZ4	2	3.91E0 * (1.9E0)	4.36E0 * (1.4E0)	**2.42E0** (4.9E-1)	4.67E0 * (8.9E-1)	2.55E0 (3.8E-1)	4.51E0 * (7.5E-1)
DTLZ5	2	3.20E0 * (4.7E-1)	3.53E0 * (4.9E-1)	**1.94E0** (2.4E-1)	3.88E0 * (5.2E-1)	2.05E0 (3.3E-1)	3.72E0 * (4.9E-1)
DTLZ6	2	4.40E2 * (1.5E1)	5.30E-3 * (3.1E-4)	5.87E2 * (1.1E1)	5.82E-3 * (3.4E-4)	5.39E2 * (1.7E1)	**3.96E-3** (2.4E-8)
DTLZ7	2	4.88E0 * (3.2E-1)	5.38E-3 * (3.0E-4)	1.64E0 * (1.5E-1)	5.40E-3 * (2.8E-4)	1.36E0 * (1.6E-1)	**5.12E-3** (1.5E-4)
DTLZ1	3	7.94E2 (8.9E2)	**6.64E2** (1.0E3)	7.41E3 * (3.9E2)	7.31E3 * (2.1E2)	8.42E3 * (1.2E3)	6.53E3 * (2.8E3)
DTLZ2	3	4.80E0 * (1.5E0)	**4.36E0** (8.0E-1)	8.28E0 * (7.6E-1)	1.03E1 * (8.4E-1)	5.06E0 * (5.5E-1)	7.42E0 * (5.4E-1)
DTLZ3	3	1.80E3 (1.8E3)	**1.26E3** (1.4E3)	1.78E4 * (8.2E2)	8.82E3 * (1.4E4)	1.75E4 * (1.3E3)	7.72E3 * (1.0E4)
DTLZ4	3	1.82E0 * (1.4E0)	**6.70E-1** (3.8E-1)	1.00E1 * (1.6E0)	1.27E1 * (1.0E0)	6.25E0 * (1.0E0)	8.72E0 * (8.4E-1)
DTLZ5	3	5.13E0 (1.2E0)	**5.03E0** (1.8E0)	9.67E0 * (8.6E-1)	1.21E1 * (1.1E0)	5.87E0 * (6.2E-1)	8.21E0 * (7.1E-1)
DTLZ6	3	4.85E2 * (2.4E1)	**5.82E-3** (4.9E-4)	7.52E2 * (5.8E0)	6.52E-3 * (5.5E-4)	6.23E2 * (5.5E0)	2.27E-2 * (4.6E-3)
DTLZ7	3	8.12E0 * (3.6E-1)	1.53E0 * (1.4E0)	2.88E0 * (2.4E-1)	8.16E-2 (5.7E-3)	3.23E0 * (3.4E-1)	**7.89E-2** (5.6E-3)
DTLZ1	4	6.65E2 (1.1E3)	**4.40E2** (7.3E2)	1.09E4 * (9.0E2)	7.35E3 * (3.3E2)	8.34E3 * (9.6E2)	6.49E3 * (2.1E2)
DTLZ2	4	1.90E1 * (5.6E0)	1.82E1 * (8.3E0)	2.24E1 * (1.2E0)	2.64E1 * (1.9E0)	**7.38E0** (9.1E-1)	9.54E0 * (6.6E-1)
DTLZ3	4	1.85E3 * (2.4E3)	**2.29E2** (2.1E2)	3.61E4 * (1.7E3)	2.48E4 * (2.3E1)	2.84E4 * (3.1E3)	1.32E4 * (1.2E4)
DTLZ4	4	**5.84E0** (2.1E0)	6.13E0 (2.6E0)	2.45E1 * (1.4E0)	2.76E1 * (3.0E0)	9.79E0 * (2.8E0)	1.20E1 * (1.8E0)
DTLZ5	4	3.53E1 * (1.3E1)	9.64E0 (2.0E1)	5.70E1 * (3.4E0)	6.67E1 * (2.9E0)	**8.58E0** (4.7E-1)	1.32E1 * (1.6E0)
DTLZ6	4	5.04E2 * (1.7E1)	**1.41E-1** (8.0E-2)	8.71E2 * (5.9E0)	3.42E-1 * (2.4E-2)	6.92E2 * (5.7E0)	3.35E-1 * (6.8E-2)
DTLZ7	4	1.15E1 * (4.0E-1)	2.24E0 * (1.0E0)	5.06E0 * (4.5E-1)	2.24E0 * (2.0E0)	5.89E0 * (4.1E-1)	**2.09E-1** (2.2E-2)
DTLZ1	5	3.26E3 (4.8E3)	**2.29E3** (2.1E3)	1.23E4 * (7.8E2)	7.77E3 * (6.2E2)	1.01E4 * (1.2E3)	6.22E3 * (7.3E2)
DTLZ2	5	3.30E1 * (1.4E1)	2.78E1 * (1.0E1)	4.60E1 * (4.0E0)	6.41E1 * (7.3E0)	**9.69E0** (1.0E0)	1.22E1 * (1.0E0)
DTLZ3	5	8.66E3 * (7.8E3)	**2.23E3** (2.3E3)	6.30E4 * (3.4E3)	2.48E4 * (1.0E1)	3.75E4 * (3.9E3)	1.78E4 * (9.9E3)
DTLZ4	5	1.45E1 * (4.4E0)	1.58E1 (8.0E0)	4.65E1 * (3.1E0)	6.00E1 * (3.6E0)	**1.14E1** (1.2E0)	1.40E1 * (1.0E1)
DTLZ5	5	4.84E1 * (1.2E1)	2.08E1 (2.0E1)	7.23E1 * (2.2E0)	7.41E1 * (2.4E0)	**1.19E1** (1.0E0)	2.14E1 * (2.4E0)
DTLZ6	5	4.98E2 * (1.9E1)	**1.27E-1** (1.4E-1)	8.79E2 * (7.0E0)	7.42E-1 * (3.9E-1)	7.41E2 * (5.8E0)	6.08E-1 * (4.3E-1)
DTLZ7	5	1.53E1 * (7.8E-1)	3.00E0 * (1.2E0)	9.32E0 * (6.4E-1)	3.00E0 * (—)	5.37E0 * (5.7E-1)	**3.90E-1** (2.6E-2)

Table B.25: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100,000 function evaluations.

		WOF-NSGA-II	ReNSGA-II	WOF-MOEA/D	ReMOEA/D	WOF-Randomised
n = 200						
LSMOP1	2	5.78E-1 * (8.4E-2)	1.38E0 * (2.8E0)	2.13E-1 (3.1E-2)	1.83E1 * (4.6E0)	**1.80E-1** (1.0E-1)
LSMOP2	2	4.78E-2 * (9.0E-3)	1.56E-1 * (7.9E-3)	6.83E-2 * (7.1E-3)	1.71E-1 * (1.2E-3)	**2.08E-2** (5.6E-3)
LSMOP3	2	1.06E0 * (2.4E-1)	2.29E1 * (4.7E0)	**7.08E-1** (3.3E-2)	3.13E1 * (4.0E0)	1.45E0 * (5.1E-1)
LSMOP4	2	1.16E-1 * (6.5E-3)	2.09E-1 * (3.0E-2)	1.46E-1 * (7.7E-3)	2.90E-1 * (9.8E-3)	**8.34E-2** (1.6E-2)
LSMOP5	2	3.43E-1 * (2.6E-2)	3.11E0 * (3.0E-1)	**2.01E-1** (1.1E-2)	4.42E1 * (4.4E0)	3.94E-1 * (8.6E-2)
LSMOP6	2	6.57E-1 * (4.7E-2)	9.72E-1 * (1.3E-2)	7.47E-1 * (1.9E-3)	9.72E-1 * (1.5E-12)	**3.93E-1** (2.0E-1)
LSMOP7	2	1.47E0 * (5.0E-3)	2.10E3 * (8.7E2)	**1.46E0** (2.6E-3)	2.46E5 * (3.6E4)	1.47E0 * (1.0E-2)
LSMOP8	2	7.42E-1 * (3.9E-1)	2.67E0 * (5.1E-1)	**1.14E-1** (8.9E-3)	3.81E1 * (3.7E0)	2.53E-1 * (1.7E-1)
LSMOP9	2	8.10E-1 * (1.5E-9)	2.81E1 * (2.0E1)	**1.74E-1** (4.1E-2)	1.38E2 * (1.6E1)	8.10E-1 * (—)
n = 300						
LSMOP1	3	3.09E-1 (5.8E-2)	6.07E0 * (2.0E0)	4.36E-1 * (8.2E-2)	1.56E1 * (4.2E0)	**3.03E-1** (1.7E-1)
LSMOP2	3	1.15E-1 * (4.5E-3)	1.10E-1 * (3.0E-3)	9.23E-2 * (1.8E-3)	1.17E-1 * (5.1E-3)	**8.34E-2** (5.2E-3)
LSMOP3	3	**8.26E-1** (8.8E-2)	1.89E1 * (5.2E0)	8.60E-1 * (8.0E-6)	2.53E1 * (2.2E0)	8.60E-1 * (1.5E-4)
LSMOP4	3	2.76E-1 * (1.0E-2)	3.22E-1 * (1.1E-2)	2.60E-1 * (2.2E-2)	3.38E-1 * (4.7E-3)	**1.84E-1** (1.2E-2)
LSMOP5	3	5.21E-1 * (2.2E-2)	1.11E1 * (6.7E0)	**2.51E-1** (2.3E-1)	2.30E1 * (2.8E0)	4.26E-1 * (7.7E-2)
LSMOP6	3	1.24E0 (4.2E-3)	1.32E4 * (1.3E4)	1.58E0 * (2.6E-1)	8.42E4 * (1.1E4)	**1.24E0** (1.7E-2)
LSMOP7	3	9.15E-1 * (1.1E-2)	1.64E0 * (5.4E-2)	9.46E-1 * (1.1E-4)	1.61E0 * (3.5E-2)	**8.70E-1** (1.7E-2)
LSMOP8	3	3.61E-1 * (1.8E-2)	9.82E-1 * (1.2E-3)	2.28E-1 (3.2E-2)	9.79E-1 * (1.9E-3)	**2.25E-1** (1.5E-1)
LSMOP9	3	**1.14E0** (4.6E-4)	1.06E2 * (5.3E1)	1.15E0 * (1.1E-2)	3.28E2 * (2.8E1)	1.15E0 * (2.1E-1)
n = 400						
LSMOP1	4	6.54E-1 * (3.8E-2)	9.45E0 * (3.8E0)	6.07E-1 * (7.4E-2)	1.33E1 * (5.5E0)	**4.89E-1** (1.7E-1)
LSMOP2	4	1.85E-1 * (7.9E-3)	1.86E-1 * (7.3E-3)	1.52E-1 * (2.3E-3)	1.69E-1 * (5.0E-3)	**1.48E-1** (4.7E-3)
LSMOP3	4	1.77E0 * (5.4E-2)	2.43E1 * (5.0E0)	**9.43E-1** (2.2E-2)	2.41E1 * (5.7E0)	1.52E0 * (4.6E-1)
LSMOP4	4	2.76E-1 * (1.1E-2)	2.79E-1 * (1.5E-2)	2.58E-1 * (7.3E-2)	2.57E-1 * (7.2E-3)	**2.21E-1** (9.3E-3)
LSMOP5	4	4.66E-1 * (7.0E-3)	3.35E1 * (5.4E0)	1.04E0 * (2.2E-5)	2.89E1 * (4.3E0)	**4.54E-1** (1.7E-2)
LSMOP6	4	8.89E-1 * (7.0E-3)	1.27E0 * (3.4E-12)	1.05E0 * (1.2E-3)	1.27E0 * (1.8E-12)	**8.69E-1** (8.2E-3)
LSMOP7	4	1.23E0 (1.1E-2)	1.05E5 * (4.4E4)	1.75E0 * (2.5E-3)	1.16E5 * (1.5E4)	**1.22E0** (2.6E-1)
LSMOP8	4	4.66E-1 * (4.1E-3)	2.61E1 * (4.2E0)	1.04E0 * (5.9E-1)	2.76E1 * (2.7E0)	**4.49E-1** (5.1E-2)
LSMOP9	4	**1.46E0** (1.5E-3)	1.53E2 * (1.2E2)	1.47E0 * (3.2E-2)	5.08E2 * (3.3E1)	1.77E0 * (4.6E-1)
n = 500						
LSMOP1	5	9.11E-1 * (1.6E-2)	1.25E1 * (5.0E0)	**5.37E-1** (2.5E-1)	1.61E1 * (7.6E0)	7.80E-1 * (2.4E-1)
LSMOP2	5	2.19E-1 * (1.5E-2)	2.12E-1 * (9.6E-3)	**1.67E-1** (8.3E-3)	1.88E-1 * (8.0E-3)	1.73E-1 * (2.3E-3)
LSMOP3	5	9.59E-1 * (5.5E-3)	2.76E1 * (8.0E0)	**9.58E-1** (2.1E-8)	2.29E1 * (4.2E0)	9.58E-1 * (—)
LSMOP4	5	3.44E-1 * (1.9E-2)	3.73E-1 * (2.4E-2)	**2.52E-1** (8.1E-3)	3.51E-1 * (4.3E-2)	2.85E-1 * (5.2E-3)
LSMOP5	5	5.47E-1 (6.7E-2)	3.20E1 * (6.3E0)	**3.58E-1** (6.3E-1)	1.08E1 * (4.0E0)	4.28E-1 (2.8E-2)
LSMOP6	5	1.36E0 * (1.1E-1)	1.38E5 * (3.4E4)	1.40E0 * (8.6E-2)	2.41E4 * (9.5E3)	**1.12E0** (1.4E-1)
LSMOP7	5	1.28E0 * (1.5E-1)	3.36E0 * (1.3E-1)	1.11E0 (7.6E-4)	2.58E0 * (4.3E-1)	**1.03E0** (2.0E-1)
LSMOP8	5	4.13E-1 * (2.7E-2)	1.21E0 * (1.1E-2)	**3.27E-1** (2.2E-2)	1.17E0 * (1.1E-2)	3.42E-1 * (1.4E-2)
LSMOP9	5	**1.82E0** (5.4E-3)	2.67E2 * (2.0E2)	1.87E0 * (1.4E-2)	8.14E2 * (5.6E1)	1.90E0 * (6.1E-1)
n = 1000						
LSMOP1	2	6.46E-1 * (9.1E-2)	1.50E0 * (6.0E-2)	2.71E-1 * (1.5E-2)	2.42E1 * (1.2E0)	**1.74E-1** (1.1E-1)
LSMOP2	2	1.90E-2 * (5.2E-4)	4.09E-2 * (8.8E-4)	1.91E-2 * (1.0E-3)	4.44E-2 * (1.0E-3)	**1.03E-2** (3.0E-3)
LSMOP3	2	1.57E0 * (2.5E-3)	2.84E1 * (1.3E0)	**1.47E0** (2.1E-1)	4.27E1 * (2.1E0)	1.57E0 * (3.7E-3)
LSMOP4	2	4.13E-2 * (4.3E-3)	5.97E-2 * (3.9E-3)	5.86E-2 * (1.6E-3)	9.46E-2 * (9.7E-4)	**2.37E-2** (2.5E-3)
LSMOP5	2	7.42E-1 (1.6E-1)	3.22E0 * (2.4E-1)	**7.34E-1** (6.2E-3)	4.99E1 * (3.0E0)	7.41E-1 (4.7E-2)
LSMOP6	2	6.71E-1 * (1.3E-3)	7.67E-1 * (3.0E-3)	7.47E-1 * (1.2E-4)	7.75E-1 * (6.8E-13)	**3.49E-1** (1.8E-1)
LSMOP7	2	1.51E0 * (2.4E-3)	1.51E3 * (3.2E2)	1.51E0 (6.2E-4)	3.10E5 * (2.9E4)	**1.51E0** (2.8E-3)
LSMOP8	2	7.42E-1 * (4.0E-2)	2.47E0 * (1.6E-1)	4.78E-1 * (5.9E-2)	4.53E1 * (2.1E0)	**2.14E-1** (1.8E-1)
LSMOP9	2	8.08E-1 * (1.6E-3)	8.04E0 * (2.1E0)	**8.94E-2** (5.7E-1)	1.54E2 * (8.6E0)	4.90E-1 (7.5E-1)
LSMOP1	3	5.93E-1 * (3.9E-2)	4.48E0 * (1.4E0)	4.32E-1 * (7.3E-2)	2.37E1 * (1.7E0)	**3.24E-1** (1.4E-1)
LSMOP2	3	7.58E-2 * (8.1E-3)	6.84E-2 * (5.1E-3)	5.46E-2 * (1.2E-3)	8.34E-2 * (6.6E-3)	**5.16E-2** (5.7E-4)
LSMOP3	3	8.60E-1 * (4.4E-4)	1.64E1 * (7.8E0)	**8.60E-1** (8.4E-6)	2.99E1 * (1.7E0)	8.60E-1 * (2.6E-8)
LSMOP4	3	1.43E-1 * (5.2E-3)	1.36E-1 * (5.1E-3)	1.19E-1 * (3.8E-3)	1.51E-1 * (2.9E-3)	**9.31E-2** (7.0E-3)
LSMOP5	3	5.41E-1 * (5.1E-4)	9.70E0 * (2.4E0)	**3.98E-1** (1.7E-1)	3.34E1 * (3.4E0)	4.51E-1 (7.3E-2)
LSMOP6	3	1.31E0 (1.8E-3)	7.46E3 * (5.3E3)	1.67E0 * (2.0E-3)	1.04E5 * (7.6E3)	**1.31E0** (9.8E-2)
LSMOP7	3	8.56E-1 * (4.3E-3)	1.09E0 * (8.9E-3)	9.47E-1 * (6.0E-4)	1.11E0 * (3.3E-3)	**8.47E-1** (3.3E-3)
LSMOP8	3	3.43E-1 * (5.7E-2)	9.58E-1 * (6.7E-4)	2.12E-1 (3.4E-2)	9.58E-1 * (1.9E-1)	**1.86E-1** (7.8E-2)
LSMOP9	3	**1.14E0** (3.4E-4)	4.22E1 * (9.9E0)	1.16E0 * (2.7E-2)	3.51E2 * (2.1E1)	1.14E0 * (1.7E-2)
LSMOP1	4	8.33E-1 * (3.1E-2)	1.11E1 * (2.8E0)	6.37E-1 (4.7E-2)	2.13E1 * (1.9E0)	**5.98E-1** (1.7E-1)
LSMOP2	4	1.46E-1 * (9.9E-3)	1.38E-1 * (7.7E-3)	1.18E-1 * (2.7E-3)	1.35E-1 * (4.9E-3)	**1.16E-1** (1.2E-3)
LSMOP3	4	1.81E0 * (2.4E-3)	2.97E1 * (5.0E0)	**1.07E0** (4.6E-2)	2.91E1 * (3.5E0)	1.81E0 * (2.4E-2)
LSMOP4	4	1.87E-1 * (9.4E-3)	1.80E-1 * (1.3E-2)	1.61E-1 * (6.7E-3)	1.75E-1 * (6.2E-3)	**1.48E-1** (4.2E-3)
LSMOP5	4	4.65E-1 * (6.2E-3)	3.00E1 * (6.3E0)	1.04E0 * (2.2E-4)	3.53E1 * (3.6E0)	**4.57E-1** (1.3E-2)
LSMOP6	4	9.05E-1 * (4.9E-3)	1.12E0 * (3.8E-12)	1.05E0 * (1.2E-3)	1.12E0 * (3.4E-12)	**8.97E-1** (7.3E-3)
LSMOP7	4	1.25E0 (1.5E-2)	9.12E4 * (6.5E4)	1.78E0 * (7.6E-1)	1.34E5 * (1.0E4)	**1.24E0** (1.1E-1)
LSMOP8	4	4.65E-1 * (4.0E-3)	2.50E1 * (3.6E0)	4.57E-1 (6.3E-1)	3.08E1 * (1.9E0)	**4.54E-1** (2.4E-2)
LSMOP9	4	**1.46E0** (7.1E-4)	7.48E1 * (4.8E1)	1.46E0 * (1.0E-2)	5.33E2 * (2.6E1)	1.74E0 * (4.5E-1)
LSMOP1	5	9.17E-1 * (1.4E-2)	1.50E1 * (2.2E0)	**6.44E-1** (1.8E-1)	2.11E1 * (1.9E0)	8.33E-1 * (1.7E-1)
LSMOP2	5	1.91E-1 * (9.2E-3)	1.86E-1 * (1.1E-2)	1.55E-1 (1.7E-2)	1.73E-1 * (7.9E-3)	**1.52E-1** (1.3E-3)
LSMOP3	5	9.62E-1 * (6.7E-2)	3.49E1 * (4.9E0)	**9.58E-1** (8.1E-9)	2.77E1 * (2.3E0)	9.58E-1 * (—)
LSMOP4	5	2.71E-1 * (1.5E-2)	2.80E-1 * (1.4E-2)	**2.09E-1** (7.0E-3)	2.57E-1 * (2.4E-2)	2.22E-1 * (5.6E-3)
LSMOP5	5	5.23E-1 * (4.6E-2)	3.47E1 * (5.3E0)	**3.72E-1** (1.3E-1)	1.65E1 * (5.3E0)	4.29E-1 (6.7E-2)
LSMOP6	5	1.48E0 * (2.2E-1)	1.26E5 * (5.8E4)	1.95E0 * (2.5E-1)	4.05E4 * (5.5E3)	**1.26E0** (4.0E-1)
LSMOP7	5	1.16E0 * (1.4E-1)	2.05E0 * (4.2E-2)	1.11E0 (6.6E-4)	1.91E0 * (8.6E-2)	**1.02E0** (1.4E-1)
LSMOP8	5	4.08E-1 * (2.5E-2)	1.15E0 * (1.4E-3)	**3.29E-1** (2.3E-2)	1.14E0 * (9.4E-3)	3.32E-1 (1.1E-2)
LSMOP9	5	**1.81E0** (7.4E-3)	1.53E2 * (5.6E1)	1.88E0 * (2.5E-2)	8.39E2 * (3.3E1)	1.86E0 * (3.4E-1)

Table B.26: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100, 000 function evaluations.

		WOF-SMPSO	LS-SMPSO	WOF-NSGA-II	LS-NSGA-II	WOF-Randomised
n = 200						
LSMOP1	2	**5.34E-2** (3.1E-2)	2.46E-1 * (1.3E-2)	5.78E-1 * (8.4E-2)	5.60E-1 * (7.2E-2)	1.80E-1 * (1.0E-1)
LSMOP2	2	**1.31E-2** (8.7E-4)	2.64E-2 * (1.1E-3)	4.78E-2 * (9.0E-3)	3.72E-2 * (1.8E-3)	2.08E-2 * (5.6E-3)
LSMOP3	2	1.49E0 * (1.0E-2)	1.51E0 * (6.0E-3)	**1.06E0** (2.4E-1)	1.53E0 * (3.5E-3)	1.45E0 * (5.1E-1)
LSMOP4	2	**5.75E-2** (9.2E-3)	7.62E-2 * (6.0E-3)	1.16E-1 * (6.5E-3)	9.98E-2 * (2.9E-3)	8.34E-2 * (1.6E-2)
LSMOP5	2	3.52E-1 (2.9E-1)	7.42E-1 * (—)	**3.43E-1** (2.6E-2)	7.42E-1 * (—)	3.94E-1 (8.6E-2)
LSMOP6	2	**1.08E-1** (1.5E-2)	1.98E-1 * (5.4E-2)	6.57E-1 * (4.7E-2)	3.58E-1 * (2.3E-3)	3.93E-1 * (2.0E-1)
LSMOP7	2	1.47E0 * (4.7E-3)	1.48E0 * (3.0E-3)	1.47E0 * (5.0E-3)	1.47E0 (2.1E-3)	**1.47E0** (1.0E-2)
LSMOP8	2	**1.18E-1** (1.0E-1)	7.42E-1 * (—)	7.42E-1 * (3.9E-1)	7.42E-1 * (—)	2.53E-1 * (1.7E-1)
LSMOP9	2	**8.10E-1** (2.5E-1)	8.10E-1 (—)	8.10E-1 * (1.5E-9)	8.10E-1 (—)	8.10E-1 (—)
n = 300						
LSMOP1	3	**1.78E-1** (4.8E-2)	3.86E-1 * (3.8E-2)	3.09E-1 * (5.8E-2)	4.51E-1 * (3.2E-2)	3.03E-1 * (1.7E-1)
LSMOP2	3	**7.44E-2** (6.3E-3)	8.29E-2 * (7.8E-3)	1.15E-1 * (4.5E-3)	1.17E-1 * (5.4E-3)	8.34E-2 * (5.2E-3)
LSMOP3	3	8.60E-1 * (3.7E-4)	8.60E-1 * (6.7E-3)	8.26E-1 (8.8E-2)	**7.93E-1** (3.0E-2)	8.60E-1 * (1.5E-4)
LSMOP4	3	1.98E-1 * (8.8E-3)	2.09E-1 * (1.1E-2)	2.76E-1 * (1.0E-2)	2.79E-1 * (7.5E-3)	**1.84E-1** (1.2E-2)
LSMOP5	3	4.50E-1 (1.6E-1)	5.17E-1 * (3.0E-1)	5.21E-1 * (2.2E-2)	5.38E-1 * (2.0E-2)	**4.26E-1** (7.7E-2)
LSMOP6	3	7.52E-1 (2.1E-1)	7.65E-1 (1.7E-1)	1.24E0 * (4.2E-3)	**7.20E-1** (4.0E-2)	1.24E0 * (1.7E-2)
LSMOP7	3	8.98E-1 * (3.4E-3)	9.48E-1 * (6.7E-2)	9.15E-1 * (1.1E-2)	9.51E-1 * (2.6E-2)	**8.70E-1** (1.7E-2)
LSMOP8	3	**1.05E-1** (2.1E-2)	3.13E-1 * (6.8E-2)	3.61E-1 * (1.8E-2)	3.61E-1 * (3.0E-2)	2.25E-1 * (1.5E-1)
LSMOP9	3	1.53E0 * (4.0E-1)	1.53E0 * (—)	**1.14E0** (4.6E-4)	1.53E0 * (—)	1.15E0 * (2.1E-1)
n = 400						
LSMOP1	4	4.98E-1 (1.0E-1)	8.50E-1 * (5.9E-2)	6.54E-1 * (3.8E-2)	7.28E-1 * (3.8E-2)	**4.89E-1** (1.7E-1)
LSMOP2	4	1.50E-1 (7.6E-3)	1.51E-1 * (8.0E-3)	1.85E-1 * (7.9E-3)	1.88E-1 * (9.0E-3)	**1.48E-1** (4.7E-3)
LSMOP3	4	1.78E0 * (2.7E-3)	1.73E0 (3.5E-2)	1.77E0 * (5.4E-2)	1.78E0 * (1.2E-3)	**1.52E0** (4.6E-1)
LSMOP4	4	2.24E-1 * (1.4E-2)	2.34E-1 * (8.8E-3)	2.76E-1 * (1.1E-2)	2.82E-1 * (1.4E-2)	**2.21E-1** (9.3E-3)
LSMOP5	4	5.44E-1 (3.3E-1)	7.90E-1 * (3.8E-1)	4.66E-1 * (7.0E-3)	4.66E-1 * (5.0E-3)	**4.54E-1** (1.7E-2)
LSMOP6	4	8.86E-1 (1.0E-1)	**8.45E-1** (1.8E-1)	8.89E-1 (7.0E-3)	1.06E0 * (2.2E-1)	8.69E-1 (8.2E-3)
LSMOP7	4	1.23E0 (4.5E-2)	1.27E0 (2.5E-1)	1.23E0 (1.1E-2)	1.23E0 (4.4E-2)	1.22E0 (2.6E-1)
LSMOP8	4	4.89E-1 * (1.2E-1)	6.72E-1 * (2.1E-1)	4.66E-1 * (4.1E-3)	4.64E-1 * (4.4E-3)	**4.49E-1** (5.1E-2)
LSMOP9	4	2.24E0 * (—)	2.24E0 * (—)	**1.46E0** (1.5E-3)	2.24E0 * (—)	1.77E0 * (4.6E-1)
n = 500						
LSMOP1	5	8.35E-1 * (7.0E-2)	9.34E-1 * (1.9E-2)	9.11E-1 * (1.6E-2)	9.18E-1 * (4.9E-2)	**7.80E-1** (2.4E-1)
LSMOP2	5	1.91E-1 * (5.8E-3)	1.97E-1 * (1.0E-2)	2.19E-1 * (1.5E-2)	2.19E-1 * (1.0E-2)	**1.73E-1** (2.3E-3)
LSMOP3	5	**9.58E-1** (—)	9.58E-1 (—)	9.59E-1 * (5.5E-3)	9.58E-1 (—)	9.58E-1 (—)
LSMOP4	5	2.95E-1 * (1.3E-2)	2.98E-1 * (1.2E-2)	3.44E-1 * (1.9E-2)	3.49E-1 * (1.5E-2)	**2.85E-1** (5.2E-3)
LSMOP5	5	9.52E-1 * (3.1E-1)	7.09E-1 * (2.1E-1)	5.47E-1 * (6.7E-2)	5.41E-1 * (4.9E-2)	**4.28E-1** (2.8E-2)
LSMOP6	5	1.80E0 * (2.6E-1)	1.50E0 * (1.7E-1)	1.36E0 * (1.1E-1)	1.69E0 * (4.9E-1)	**1.12E0** (1.4E-1)
LSMOP7	5	1.65E0 * (2.9E-1)	1.24E0 * (1.1E-1)	1.28E0 * (1.5E-1)	1.26E0 * (1.4E-1)	**1.03E0** (2.0E-1)
LSMOP8	5	7.66E-1 * (1.7E-1)	9.58E-1 * (3.0E-2)	4.13E-1 * (2.7E-2)	4.17E-1 * (6.6E-2)	**3.42E-1** (1.4E-2)
LSMOP9	5	3.00E0 * (2.5E-4)	3.00E0 * (—)	**1.82E0** (5.4E-3)	3.00E0 * (—)	1.90E0 * (6.1E-1)
n = 1000						
LSMOP1	2	**7.17E-2** (6.2E-3)	3.06E-1 * (3.4E-3)	6.46E-1 * (9.1E-2)	6.27E-1 * (3.5E-2)	1.74E-1 * (1.1E-1)
LSMOP2	2	**7.29E-3** (3.9E-4)	9.43E-3 * (1.6E-3)	1.90E-2 * (5.2E-4)	1.86E-2 * (9.3E-4)	1.03E-2 * (3.0E-3)
LSMOP3	2	**1.56E0** (1.9E-3)	1.56E0 * (1.1E-3)	1.57E0 * (2.5E-3)	1.57E0 * (6.7E-4)	1.57E0 * (3.7E-3)
LSMOP4	2	**2.04E-2** (5.9E-4)	2.28E-2 * (6.1E-4)	4.13E-2 * (4.3E-3)	3.31E-2 * (1.5E-3)	2.37E-2 * (2.5E-3)
LSMOP5	2	7.42E-1 (2.9E-1)	7.42E-1 * (—)	7.42E-1 (1.6E-1)	7.42E-1 * (—)	**7.41E-1** (4.7E-2)
LSMOP6	2	**1.73E-1** (3.4E-3)	1.90E-1 * (1.4E-1)	6.71E-1 * (1.3E-3)	3.12E-1 * (1.3E-4)	3.49E-1 * (1.8E-1)
LSMOP7	2	1.51E0 * (6.0E-4)	1.51E0 * (3.9E-4)	1.51E0 * (2.4E-3)	**1.50E0** (7.8E-4)	1.51E0 * (2.8E-3)
LSMOP8	2	**2.12E-1** (5.9E-1)	7.42E-1 * (—)	7.42E-1 * (4.0E-2)	7.42E-1 * (—)	2.14E-1 (1.8E-1)
LSMOP9	2	**4.67E-1** (9.5E-3)	8.04E-1 * (1.3E-3)	8.08E-1 * (1.6E-3)	8.07E-1 * (2.9E-3)	4.90E-1 * (7.5E-1)
LSMOP1	3	**2.00E-1** (4.1E-2)	4.00E-1 * (3.6E-2)	5.93E-1 * (3.9E-2)	5.96E-1 * (1.3E-2)	3.24E-1 * (1.4E-1)
LSMOP2	3	6.12E-2 * (5.2E-3)	6.16E-2 * (4.4E-3)	7.58E-2 * (8.1E-3)	7.40E-2 * (5.9E-3)	**5.16E-2** (5.7E-4)
LSMOP3	3	8.60E-1 * (—)	8.60E-1 * (—)	8.60E-1 * (4.4E-4)	**8.60E-1** (3.1E-4)	8.60E-1 * (2.6E-8)
LSMOP4	3	**8.91E-2** (4.9E-3)	9.56E-2 * (4.7E-3)	1.43E-1 * (5.2E-3)	1.45E-1 * (5.6E-3)	9.31E-2 * (7.0E-3)
LSMOP5	3	**4.29E-1** (2.3E-1)	8.92E-1 * (3.7E-1)	5.41E-1 * (5.1E-4)	5.40E-1 * (5.8E-5)	4.51E-1 (7.3E-2)
LSMOP6	3	9.11E-1 (5.8E-1)	7.88E-1 (1.0E-1)	1.31E0 * (1.8E-3)	**7.43E-1** (4.0E-2)	1.31E0 * (9.8E-2)
LSMOP7	3	8.49E-1 (1.0E-1)	8.63E-1 * (9.5E-2)	8.56E-1 * (4.3E-3)	8.64E-1 * (5.4E-3)	**8.47E-1** (3.3E-3)
LSMOP8	3	**9.22E-2** (1.5E-2)	2.33E-1 * (1.8E-1)	3.43E-1 * (5.7E-2)	3.39E-1 * (1.6E-2)	1.86E-1 * (7.8E-2)
LSMOP9	3	**1.11E0** (5.7E-1)	1.53E0 * (3.9E-1)	1.14E0 * (3.4E-4)	1.49E0 * (3.9E-1)	1.14E0 * (1.7E-2)
LSMOP1	4	**4.91E-1** (1.3E-1)	8.53E-1 * (3.7E-2)	8.33E-1 * (3.1E-2)	7.68E-1 * (2.4E-2)	5.98E-1 * (1.7E-1)
LSMOP2	4	1.26E-1 * (5.7E-3)	1.29E-1 * (8.1E-3)	1.46E-1 * (9.9E-3)	1.47E-1 * (1.0E-2)	**1.16E-1** (1.2E-3)
LSMOP3	4	1.81E0 (3.3E-3)	**1.77E0** (5.4E-2)	1.81E0 * (2.4E-3)	1.80E0 * (5.8E-4)	1.81E0 (2.4E-2)
LSMOP4	4	1.55E-1 * (6.9E-3)	1.56E-1 * (6.6E-3)	1.87E-1 * (9.4E-3)	1.90E-1 * (9.4E-3)	**1.48E-1** (4.2E-3)
LSMOP5	4	4.94E-1 (2.5E-1)	7.80E-1 * (3.0E-1)	4.65E-1 * (6.2E-3)	4.64E-1 * (5.1E-3)	**4.57E-1** (1.3E-2)
LSMOP6	4	9.09E-1 (1.1E-1)	**8.53E-1** (2.0E-1)	9.05E-1 * (4.9E-3)	1.04E0 * (6.0E-2)	8.97E-1 * (7.3E-3)
LSMOP7	4	**1.20E0** (2.5E-2)	1.32E0 * (2.2E-1)	1.25E0 * (1.5E-2)	1.22E0 (1.1E-1)	1.24E0 * (1.1E-1)
LSMOP8	4	4.89E-1 * (1.3E-1)	7.56E-1 * (1.3E-1)	4.65E-1 * (4.0E-3)	4.62E-1 * (5.0E-3)	**4.54E-1** (2.4E-2)
LSMOP9	4	2.24E0 (1.4E0)	2.24E0 * (—)	**1.46E0** (7.1E-4)	2.24E0 * (—)	1.74E0 * (4.5E-1)
LSMOP1	5	8.46E-1 (1.0E-1)	9.27E-1 * (1.0E-2)	9.17E-1 * (1.4E-2)	9.19E-1 * (3.8E-2)	**8.33E-1** (1.7E-1)
LSMOP2	5	1.82E-1 * (1.1E-2)	1.85E-1 * (9.8E-3)	1.91E-1 * (9.2E-3)	1.97E-1 * (1.3E-2)	**1.52E-1** (1.3E-3)
LSMOP3	5	**9.58E-1** (—)	9.58E-1 (—)	9.62E-1 * (6.7E-2)	9.58E-1 (—)	9.58E-1 (—)
LSMOP4	5	2.33E-1 * (1.3E-2)	2.36E-1 * (8.1E-3)	2.71E-1 * (1.5E-2)	2.74E-1 * (9.2E-3)	**2.22E-1** (5.6E-3)
LSMOP5	5	9.83E-1 * (2.2E-1)	8.32E-1 * (7.7E-2)	5.23E-1 * (4.6E-2)	5.05E-1 * (3.8E-2)	**4.29E-1** (6.7E-2)
LSMOP6	5	1.83E0 * (2.8E-1)	1.54E0 (4.2E-1)	1.48E0 * (2.2E-1)	1.42E0 (6.0E-1)	**1.26E0** (4.0E-1)
LSMOP7	5	1.35E0 * (1.8E-1)	1.15E0 * (8.8E-2)	1.16E0 * (1.4E-1)	1.18E0 * (5.4E-2)	**1.02E0** (1.4E-1)
LSMOP8	5	6.59E-1 * (1.5E-1)	9.39E-1 * (2.9E-3)	4.08E-1 * (2.5E-2)	4.83E-1 * (2.8E-1)	**3.32E-1** (1.1E-2)
LSMOP9	5	3.00E0 (2.2E0)	3.00E0 * (—)	**1.81E0** (7.4E-3)	3.00E0 * (—)	1.86E0 * (3.4E-1)

Table B.27: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100,000 function evaluations.

		WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	randomLMEA	randomMOEA/DVA	randomS3-CMA-ES
n = 200							
LSMOP1	2	**5.34E-2** (3.1E-2)	5.78E-1 * (8.4E-2)	1.80E-1 * (1.0E-1)	6.13E-1 * (4.3E-2)	6.39E-1 * (8.2E-2)	7.54E-1 * (1.2E0)
LSMOP2	2	**1.31E-2** (8.7E-4)	4.78E-2 * (9.0E-3)	2.08E-2 * (5.6E-3)	1.03E-1 * (5.2E-3)	1.53E-1 * (1.0E-3)	5.70E-2 * (6.4E-3)
LSMOP3	2	1.49E0 * (1.0E-2)	**1.06E0** (2.4E-1)	1.45E0 * (5.1E-1)	2.06E1 * (2.9E0)	1.20E1 * (3.0E1)	5.70E0 * (6.8E-1)
LSMOP4	2	**5.75E-2** (9.2E-3)	1.16E-1 * (6.5E-3)	8.34E-2 * (1.6E-2)	1.67E-1 * (4.2E-3)	1.73E-1 * (3.0E-3)	2.09E-1 * (1.3E-2)
LSMOP5	2	3.52E-1 (2.9E-1)	**3.43E-1** (2.6E-2)	3.94E-1 (8.6E-2)	7.24E-1 * (1.4E0)	8.52E-1 * (8.0E-1)	7.50E-1 * (1.0E-3)
LSMOP6	2	**1.08E-1** (1.5E-2)	6.57E-1 * (4.7E-2)	3.93E-1 * (2.0E-1)	9.04E-1 * (1.8E-2)	5.07E0 * (1.1E1)	7.66E-1 * (3.5E-3)
LSMOP7	2	1.47E0 * (4.7E-3)	1.47E0 * (5.0E-3)	**1.47E0** (1.0E-2)	6.15E0 * (4.1E0)	1.95E1 * (3.4E1)	5.79E2 * (1.1E3)
LSMOP8	2	**1.18E-1** (1.0E-1)	7.42E-1 * (3.9E-1)	2.53E-1 * (1.7E-1)	1.93E0 * (2.7E-1)	8.02E-1 * (1.0E-2)	1.90E0 * (1.5E0)
LSMOP9	2	**8.10E-1** (2.5E-1)	8.10E-1 * (1.5E-9)	8.10E-1 (—)	1.82E0 * (6.4E-2)	1.95E0 * (7.3E-2)	9.87E-1 * (2.4E-2)
n = 300							
LSMOP1	3	**1.78E-1** (4.8E-2)	3.09E-1 * (5.8E-2)	3.03E-1 * (1.7E-1)	4.73E-1 * (6.4E-2)	6.49E-1 * (9.7E-2)	2.09E0 * (1.5E0)
LSMOP2	3	7.44E-2 (6.3E-3)	1.15E-1 * (4.5E-3)	8.34E-2 * (5.2E-3)	8.05E-2 * (1.7E-3)	7.95E-2 * (1.4E-3)	**7.16E-2** (4.0E-3)
LSMOP3	3	8.60E-1 * (3.7E-4)	**8.26E-1** (8.8E-2)	8.60E-1 * (1.5E-4)	1.15E1 * (1.9E0)	3.77E1 * (2.3E1)	9.69E0 * (1.6E0)
LSMOP4	3	1.98E-1 * (8.8E-3)	2.76E-1 * (1.0E-2)	**1.84E-1** (1.2E-2)	1.91E-1 * (5.1E-3)	2.32E-1 * (5.4E-3)	2.40E-1 * (1.3E-2)
LSMOP5	3	4.50E-1 (1.6E-1)	5.21E-1 * (2.2E-2)	**4.26E-1** (7.7E-2)	2.26E0 * (1.3E0)	2.61E0 * (1.0E0)	9.87E-1 * (2.0E-2)
LSMOP6	3	**7.52E-1** (2.1E-1)	1.24E0 * (4.2E-3)	1.24E0 * (1.7E-2)	2.25E0 * (1.7E0)	2.98E1 * (4.5E1)	1.90E1 * (1.8E1)
LSMOP7	3	8.98E-1 * (3.4E-3)	9.15E-1 * (1.1E-2)	**8.70E-1** (1.7E-2)	1.79E0 * (9.7E-3)	5.53E1 * (7.1E1)	1.03E0 * (8.0E-3)
LSMOP8	3	**1.05E-1** (2.1E-2)	3.61E-1 * (1.8E-2)	2.25E-1 * (1.5E-1)	3.35E-1 * (5.2E-2)	2.77E-1 * (6.6E-2)	9.56E-1 * (8.5E-3)
LSMOP9	3	1.53E0 * (4.0E-1)	**1.14E0** (4.6E-4)	1.15E0 * (2.1E-1)	3.47E0 * (2.9E-1)	1.80E0 * (7.0E-2)	1.22E1 * (1.1E1)
n = 400							
LSMOP1	4	4.98E-1 (1.0E-1)	6.54E-1 * (3.8E-2)	**4.89E-1** (1.7E-1)	8.91E-1 * (1.6E-1)	8.70E-1 * (5.1E-1)	1.02E0 * (5.2E-1)
LSMOP2	4	1.50E-1 * (7.6E-3)	1.85E-1 * (7.9E-3)	1.48E-1 * (4.7E-3)	1.37E-1 * (2.9E-3)	1.64E-1 * (1.4E-3)	**1.33E-1** (4.9E-3)
LSMOP3	4	1.78E0 * (2.7E-3)	1.77E0 * (5.4E-2)	**1.52E0** (4.6E-1)	2.31E1 * (4.0E0)	3.77E2 * (6.1E2)	1.35E1 * (8.5E0)
LSMOP4	4	2.24E-1 * (1.4E-2)	2.76E-1 * (1.1E-2)	2.21E-1 * (9.3E-3)	**1.83E-1** (4.8E-3)	2.25E-1 * (5.1E-3)	1.95E-1 * (1.7E-2)
LSMOP5	4	5.44E-1 (3.3E-1)	4.66E-1 * (7.0E-3)	**4.54E-1** (1.7E-2)	3.88E0 * (1.5E0)	6.73E0 * (3.6E0)	1.22E0 * (7.6E-2)
LSMOP6	4	8.86E-1 (1.0E-1)	8.89E-1 * (7.0E-3)	**8.69E-1** (8.2E-3)	1.27E0 * (1.9E-4)	1.61E1 * (3.8E1)	1.09E0 * (7.2E-3)
LSMOP7	4	1.23E0 (4.5E-2)	1.23E0 (1.1E-2)	**1.22E0** (2.6E-1)	1.62E3 * (1.4E3)	9.26E3 * (2.2E3)	2.89E2 * (3.1E3)
LSMOP8	4	4.89E-1 * (1.2E-1)	4.66E-1 * (4.1E-3)	**4.49E-1** (5.1E-2)	1.57E0 * (8.3E-1)	1.11E0 * (9.3E-1)	1.34E0 * (1.8E0)
LSMOP9	4	2.24E0 * (—)	**1.46E0** (1.5E-3)	1.77E0 * (4.6E-1)	7.79E0 * (7.0E-1)	4.25E0 * (8.5E-2)	4.57E0 * (6.0E-1)
n = 500							
LSMOP1	5	8.35E-1 * (7.0E-2)	9.11E-1 * (1.6E-2)	**7.80E-1** (2.4E-1)	1.29E0 * (3.0E-1)	8.84E-1 * (7.1E-2)	1.82E0 * (1.0E0)
LSMOP2	5	1.91E-1 * (5.8E-3)	2.19E-1 * (1.5E-2)	1.73E-1 * (2.3E-3)	1.69E-1 * (3.3E-3)	1.88E-1 * (2.1E-3)	**1.65E-1** (2.8E-3)
LSMOP3	5	**9.58E-1** (—)	9.59E-1 * (5.5E-3)	9.58E-1 (—)	1.74E1 * (5.0E0)	4.50E2 * (6.8E2)	6.55E0 * (3.2E0)
LSMOP4	5	2.95E-1 * (1.3E-2)	3.44E-1 * (1.9E-2)	2.85E-1 * (5.2E-3)	2.62E-1 * (8.0E-3)	3.29E-1 * (3.7E-3)	**2.54E-1** (1.3E-2)
LSMOP5	5	9.52E-1 * (3.1E-1)	5.47E-1 * (6.7E-2)	**4.28E-1** (2.8E-2)	3.85E0 * (3.2E0)	1.10E0 * (3.2E-2)	1.14E0 * (2.0E-2)
LSMOP6	5	1.80E0 * (2.6E-1)	1.36E0 * (1.1E-1)	**1.12E0** (1.4E-1)	1.23E2 * (5.0E2)	1.49E2 * (1.9E2)	6.43E0 * (2.4E0)
LSMOP7	5	1.65E0 * (2.9E-1)	1.28E0 * (1.5E-1)	**1.03E0** (2.0E-1)	3.69E0 * (8.9E-2)	5.42E2 * (1.2E3)	1.30E0 * (3.7E-2)
LSMOP8	5	7.66E-1 * (1.7E-1)	4.13E-1 * (2.7E-2)	**3.42E-1** (1.4E-2)	1.23E0 * (2.2E-1)	1.12E0 * (2.0E-2)	1.11E0 * (4.2E-3)
LSMOP9	5	3.00E0 * (2.5E-4)	**1.82E0** (5.4E-3)	1.90E0 * (6.1E-1)	1.48E1 * (3.5E0)	8.10E0 * (6.5E0)	5.62E1 * (2.2E1)
n = 1000							
LSMOP1	2	**7.17E-2** (6.2E-3)	6.46E-1 * (9.1E-2)	1.74E-1 * (1.1E-1)	2.78E0 * (2.7E-1)	7.06E0 * (3.5E-1)	7.83E0 * (5.6E-1)
LSMOP2	2	**7.29E-3** (3.9E-4)	1.90E-2 * (5.2E-4)	1.03E-2 * (3.0E-3)	3.44E-2 * (6.6E-4)	3.98E-2 * (5.0E-4)	3.63E-2 * (3.8E-4)
LSMOP3	2	**1.56E0** (1.9E-3)	1.57E0 * (2.5E-3)	1.57E0 * (3.7E-3)	3.15E1 * (1.2E0)	7.65E2 * (1.0E3)	2.76E1 * (2.2E0)
LSMOP4	2	**2.04E-2** (5.9E-4)	4.13E-2 * (4.3E-3)	2.37E-2 * (2.5E-3)	6.18E-2 * (9.1E-4)	6.67E-2 * (1.1E-3)	6.09E-2 * (1.7E-3)
LSMOP5	2	7.42E-1 (2.9E-1)	7.42E-1 (1.6E-1)	**7.41E-1** (4.7E-2)	6.62E0 * (9.2E-1)	1.55E1 * (1.3E0)	1.48E1 * (6.6E-1)
LSMOP6	2	**1.73E-1** (3.4E-3)	6.71E-1 * (1.3E-3)	3.49E-1 * (1.8E-1)	7.74E-1 * (5.4E-4)	2.02E3 * (2.2E3)	7.71E-1 * (1.3E-3)
LSMOP7	2	1.51E0 (6.0E-4)	1.51E0 * (2.4E-3)	**1.51E0** (2.8E-3)	4.73E3 * (1.4E3)	5.22E4 * (8.3E3)	3.84E4 * (3.8E3)
LSMOP8	2	**2.12E-1** (5.9E-1)	7.42E-1 * (4.0E-2)	2.14E-1 (1.8E-1)	8.78E0 * (6.2E-1)	1.26E1 * (1.0E0)	1.26E1 * (7.6E-1)
LSMOP9	2	**4.67E-1** (9.5E-3)	8.08E-1 * (1.6E-3)	4.90E-1 * (7.5E-1)	6.88E0 * (1.3E0)	2.82E1 * (1.7E0)	1.30E1 * (1.6E0)
LSMOP1	3	**2.00E-1** (4.1E-2)	5.93E-1 * (3.9E-2)	3.24E-1 * (1.4E-1)	2.52E0 * (3.2E-1)	6.73E0 * (5.1E-1)	7.95E0 * (5.5E-1)
LSMOP2	3	6.12E-2 * (5.2E-3)	7.58E-2 * (8.1E-3)	**5.16E-2** (5.7E-4)	5.32E-2 * (6.0E-4)	6.59E-2 * (4.0E-3)	5.23E-2 * (6.3E-4)
LSMOP3	3	**8.60E-1** (—)	8.60E-1 * (4.4E-4)	8.60E-1 (2.6E-8)	1.69E1 * (1.4E0)	2.74E2 * (3.4E2)	1.95E1 * (3.7E0)
LSMOP4	3	**8.91E-2** (4.9E-3)	1.43E-1 * (5.2E-3)	9.31E-2 * (7.0E-3)	9.51E-2 * (1.9E-3)	1.21E-1 * (3.1E-3)	1.10E-1 * (2.5E-3)
LSMOP5	3	**4.29E-1** (2.3E-1)	5.41E-1 * (5.1E-4)	4.51E-1 (7.3E-2)	6.85E0 * (8.2E-1)	1.28E1 * (1.2E0)	1.24E1 * (1.5E0)
LSMOP6	3	**9.11E-1** (5.8E-1)	1.31E0 * (1.8E-3)	1.31E0 * (9.8E-2)	1.63E2 * (6.3E1)	2.14E4 * (6.6E3)	1.51E4 * (3.1E3)
LSMOP7	3	8.49E-1 (1.0E-1)	8.56E-1 * (4.3E-3)	**8.47E-1** (3.3E-3)	1.12E0 * (4.2E-3)	6.01E2 * (1.0E3)	1.07E0 * (6.6E-3)
LSMOP8	3	**9.22E-2** (1.5E-2)	3.43E-1 * (5.7E-2)	1.86E-1 * (7.8E-2)	9.59E-1 * (5.1E-4)	6.86E-1 * (7.6E-2)	6.25E-1 * (6.0E-2)
LSMOP9	3	**1.11E0** (5.7E-1)	1.14E0 * (3.4E-4)	1.14E0 * (1.7E-2)	2.50E1 * (3.3E0)	7.28E1 * (6.2E0)	7.51E1 * (7.2E0)
LSMOP1	4	**4.91E-1** (1.3E-1)	8.33E-1 * (3.1E-2)	5.98E-1 * (1.7E-1)	2.89E0 * (2.4E-1)	6.05E0 * (7.3E-1)	7.05E0 * (2.9E-1)
LSMOP2	4	1.26E-1 * (5.7E-3)	1.46E-1 * (9.9E-3)	1.16E-1 * (1.2E-3)	1.13E-1 * (2.2E-3)	1.42E-1 * (1.0E-2)	**1.09E-1** (1.7E-3)
LSMOP3	4	**1.81E0** (3.3E-3)	1.81E0 * (2.4E-3)	1.81E0 (2.4E-2)	2.51E1 * (2.3E0)	3.11E3 * (2.8E3)	3.26E1 * (2.3E0)
LSMOP4	4	1.55E-1 * (6.9E-3)	1.87E-1 * (9.4E-3)	1.48E-1 * (4.2E-3)	**1.36E-1** (1.7E-3)	1.70E-1 * (1.6E-2)	1.44E-1 * (4.3E-3)
LSMOP5	4	4.94E-1 (2.5E-1)	4.65E-1 * (6.2E-3)	**4.57E-1** (1.3E-2)	1.05E1 * (2.5E0)	1.23E1 * (1.4E0)	1.22E1 * (1.2E0)
LSMOP6	4	9.09E-1 (1.1E-1)	9.05E-1 * (4.9E-3)	**8.97E-1** (7.3E-3)	1.12E0 * (2.7E-12)	7.21E2 * (1.3E3)	1.11E0 * (2.2E-3)
LSMOP7	4	**1.20E0** (2.5E-2)	1.25E0 * (1.5E-2)	1.24E0 * (1.1E-1)	5.94E3 * (2.4E3)	2.48E4 * (6.5E3)	2.03E4 * (3.5E3)
LSMOP8	4	4.89E-1 * (1.3E-1)	4.65E-1 * (4.0E-3)	**4.54E-1** (2.4E-2)	4.75E0 * (1.2E0)	8.85E0 * (1.0E0)	8.86E0 * (1.2E0)
LSMOP9	4	2.24E0 (1.4E0)	**1.46E0** (7.1E-4)	1.74E0 * (4.5E-1)	2.97E1 * (6.0E0)	9.10E1 * (1.3E1)	6.77E1 * (1.6E1)
LSMOP1	5	8.46E-1 (1.0E-1)	9.17E-1 * (1.4E-2)	**8.33E-1** (1.7E-1)	2.49E0 * (4.7E-1)	9.25E-1 * (3.8E-2)	6.84E0 * (6.7E-1)
LSMOP2	5	1.82E-1 * (1.1E-2)	1.91E-1 * (9.2E-3)	1.52E-1 * (1.3E-3)	1.54E-1 * (2.2E-3)	1.62E-1 * (1.3E-3)	**1.46E-1** (2.9E-3)
LSMOP3	5	**9.58E-1** (—)	9.62E-1 * (6.7E-2)	9.58E-1 (—)	2.64E1 * (9.8E0)	1.43E3 * (1.9E3)	1.81E1 * (6.0E0)
LSMOP4	5	2.33E-1 * (1.3E-2)	2.71E-1 * (1.5E-2)	2.22E-1 * (5.6E-3)	**2.07E-1** (5.6E-3)	2.52E-1 * (5.2E-3)	2.20E-1 * (4.8E-3)
LSMOP5	5	9.83E-1 * (2.2E-1)	5.23E-1 * (4.6E-2)	**4.29E-1** (6.7E-2)	6.15E0 * (2.5E0)	1.19E0 * (1.9E-1)	8.34E0 * (2.1E0)
LSMOP6	5	1.83E0 * (2.8E-1)	1.48E0 * (2.2E-1)	**1.26E0** (4.0E-1)	2.26E2 * (5.7E2)	1.02E3 * (1.3E3)	4.47E3 * (3.9E3)
LSMOP7	5	1.35E0 * (1.8E-1)	1.16E0 * (1.4E-1)	**1.02E0** (1.4E-1)	2.14E0 * (2.4E-2)	9.50E2 * (2.1E3)	1.66E0 * (1.1E-1)
LSMOP8	5	6.59E-1 * (1.5E-1)	4.08E-1 * (2.5E-2)	**3.32E-1** (1.1E-2)	1.16E0 * (1.9E-3)	1.13E0 * (2.9E-2)	1.13E0 * (8.1E-3)
LSMOP9	5	3.00E0 (2.2E0)	**1.81E0** (7.4E-3)	1.86E0 * (3.4E-1)	4.45E1 * (8.6E0)	1.70E2 * (1.7E1)	1.77E2 * (2.2E1)

Table B.28: Performance comparison using the IGD indicator on the UF benchmarks using 100, 000 function evaluations.

		WOF-NSGA-II	ReNSGA-II	WOF-MOEA/D	ReMOEA/D	WOF-Randomised
n = 40						
UF1	2	1.01E-1 * (2.6E-2)	2.12E-1 * (8.4E-2)	**6.98E-2** (1.2E-2)	9.23E-1 * (4.6E-1)	9.04E-2 * (2.2E-2)
UF2	2	**4.11E-2** (7.4E-3)	1.47E-1 * (1.3E-1)	4.77E-2 * (1.1E-2)	1.34E0 * (1.3E-1)	4.21E-2 (8.2E-3)
UF3	2	1.69E-1 (2.0E-2)	6.29E-1 * (1.4E-1)	2.24E-1 * (4.3E-2)	1.69E0 * (3.0E-1)	**1.68E-1** (4.0E-2)
UF4	2	**4.72E-2** (1.1E-3)	7.55E-2 * (7.7E-3)	8.27E-2 * (5.4E-3)	1.02E-1 * (7.0E-3)	4.88E-2 (5.6E-3)
UF5	2	3.86E-1 (2.0E-1)	2.04E0 * (5.6E-1)	5.16E-1 * (1.7E-1)	3.72E0 * (9.5E-1)	**3.48E-1** (2.0E-1)
UF6	2	**1.14E-1** (1.8E-1)	9.92E-1 * (5.0E-1)	1.88E-1 * (2.8E-1)	3.80E0 * (1.8E0)	1.27E-1 (2.4E-1)
UF7	2	5.56E-2 (1.4E-2)	2.20E-1 * (1.6E-1)	3.43E-1 (3.0E-1)	7.96E-1 * (3.0E-1)	**5.08E-2** (1.6E-2)
UF8	3	**2.24E-1** (1.2E-1)	1.20E0 * (4.9E-1)	2.91E-1 (1.6E-2)	5.77E0 * (3.3E0)	5.28E-1 * (5.4E-2)
UF9	3	**3.06E-1** (1.4E-1)	1.38E0 * (7.0E-1)	3.47E-1 * (1.1E-1)	5.50E0 * (2.6E0)	3.13E-1 (2.3E-1)
UF10	3	**4.30E-1** (1.4E-1)	7.46E0 * (2.6E0)	7.12E-1 * (1.3E-1)	2.75E1 * (5.0E0)	5.20E-1 * (3.9E-2)
n = 1000						
UF1	2	**1.57E-1** (3.7E-2)	1.35E0 * (1.6E-2)	2.67E-1 * (1.4E-2)	3.01E0 * (1.7E-1)	1.96E-1 * (1.6E-2)
UF2	2	1.13E-1 * (1.3E-2)	1.56E-1 * (7.0E-3)	9.84E-2 * (4.6E-3)	2.08E0 * (4.0E-1)	**8.69E-2** (3.2E-3)
UF3	2	1.38E-1 * (1.0E-2)	3.46E-1 * (2.5E-3)	2.02E-1 * (1.8E-2)	2.44E0 * (9.5E-2)	**3.50E-2** (1.0E-2)
UF4	2	7.40E-2 (5.0E-3)	1.39E-1 * (1.2E-3)	1.04E-1 * (6.7E-3)	1.43E-1 * (2.4E-3)	**7.08E-2** (1.8E-2)
UF5	2	**1.20E0** (1.9E-1)	6.31E0 * (3.7E-1)	1.41E0 * (1.6E-1)	8.54E0 * (2.5E-1)	1.61E0 * (5.3E-1)
UF6	2	4.74E-1 (2.6E-1)	5.55E0 * (1.5E-1)	7.74E-1 * (1.1E-1)	1.20E1 * (4.4E-1)	**4.64E-1** (3.1E-1)
UF7	2	**1.50E-1** (2.6E-2)	1.41E0 * (1.9E-2)	4.71E-1 * (2.4E-1)	3.00E0 * (1.5E-1)	1.67E-1 * (2.0E-2)
UF8	3	4.67E-1 (2.9E-2)	6.14E-1 (5.9E-2)	**3.31E-1** (6.4E-1)	9.02E0 * (1.3E0)	5.42E-1 (2.1E-3)
UF9	3	6.52E-1 (8.4E-2)	7.81E-1 * (2.2E-2)	7.14E-1 * (1.0E-1)	1.03E1 * (7.8E-1)	**6.20E-1** (9.6E-2)
UF10	3	1.90E0 * (4.1E-1)	5.46E0 * (4.8E-1)	**8.59E-1** (9.8E-2)	4.20E1 * (2.5E0)	8.97E-1 (2.9E-1)

Table B.29: Performance comparison using the IGD indicator on the UF benchmarks using 100, 000 function evaluations.

		WOF-SMPSO	LS-SMPSO	WOF-NSGA-II	LS-NSGA-II	WOF-Randomised
n = 40						
UF1	2	9.14E-2 (2.2E-2)	1.22E-1 * (3.1E-2)	1.01E-1 * (2.6E-2)	1.35E-1 * (3.8E-2)	**9.04E-2** (2.2E-2)
UF2	2	5.41E-2 * (5.7E-3)	6.53E-2 * (5.0E-3)	**4.11E-2** (7.4E-3)	6.93E-2 * (1.0E-2)	4.21E-2 (8.2E-3)
UF3	2	**1.33E-1** (3.6E-2)	1.91E-1 * (2.5E-2)	1.69E-1 * (2.0E-2)	2.03E-1 * (1.9E-2)	1.68E-1 * (4.0E-2)
UF4	2	5.19E-2 * (8.4E-3)	5.76E-2 * (3.4E-4)	4.72E-2 (1.1E-3)	**4.69E-2** (1.2E-3)	4.88E-2 (5.6E-3)
UF5	2	6.24E-1 * (1.6E-1)	8.98E-1 * (2.8E-1)	3.86E-1 (2.0E-1)	4.38E-1 (1.4E-1)	**3.48E-1** (2.0E-1)
UF6	2	3.22E-1 * (1.7E-1)	4.83E-1 * (4.7E-2)	**1.14E-1** (1.8E-1)	4.07E-1 * (3.2E-2)	1.27E-1 (2.4E-1)
UF7	2	**4.68E-2** (9.4E-3)	3.71E-1 * (4.3E-2)	5.56E-2 * (1.4E-2)	3.60E-1 * (3.9E-3)	5.08E-2 (1.6E-2)
UF8	3	2.52E-1 (5.0E-2)	3.29E-1 * (8.9E-2)	**2.24E-1** (1.2E-1)	2.33E-1 (6.2E-2)	5.28E-1 * (5.4E-2)
UF9	3	4.81E-1 * (8.9E-2)	4.94E-1 * (7.0E-2)	**3.06E-1** (1.4E-1)	4.78E-1 * (7.0E-2)	3.13E-1 (2.3E-1)
UF10	3	1.28E0 * (2.9E-1)	1.91E0 * (5.4E-1)	**4.30E-1** (1.4E-1)	4.81E-1 (1.7E-2)	5.20E-1 * (3.9E-2)
n = 1000						
UF1	2	2.72E-1 * (5.9E-3)	2.85E-1 * (9.5E-3)	**1.57E-1** (3.7E-2)	2.34E-1 * (1.6E-2)	1.96E-1 * (1.6E-2)
UF2	2	**8.50E-2** (4.1E-4)	8.61E-2 * (1.1E-3)	1.13E-1 * (1.3E-2)	9.50E-2 * (2.3E-3)	8.69E-2 * (3.2E-3)
UF3	2	**2.54E-2** (3.8E-3)	1.24E-1 * (1.0E-3)	1.38E-1 * (1.0E-2)	1.31E-1 * (2.7E-3)	3.50E-2 * (1.0E-2)
UF4	2	**5.49E-2** (9.0E-3)	5.91E-2 * (8.9E-5)	7.40E-2 * (5.0E-3)	7.22E-2 * (1.7E-3)	7.08E-2 * (1.8E-2)
UF5	2	2.81E0 * (1.1E-1)	3.01E0 * (1.0E-1)	**1.20E0** (1.9E-1)	1.64E0 * (2.1E-1)	1.61E0 * (5.3E-1)
UF6	2	1.02E0 * (5.9E-2)	1.14E0 * (1.3E-1)	4.74E-1 (2.6E-1)	1.14E0 * (2.1E-1)	**4.64E-1** (3.1E-1)
UF7	2	2.77E-1 * (7.4E-3)	4.69E-1 * (1.5E-1)	**1.50E-1** (2.6E-2)	4.40E-1 * (8.1E-3)	1.67E-1 * (2.0E-2)
UF8	3	**3.56E-1** (4.4E-3)	5.02E-1 * (1.3E-1)	4.67E-1 * (2.9E-2)	6.08E-1 * (2.5E-2)	5.42E-1 * (2.1E-3)
UF9	3	**5.70E-1** (7.2E-3)	5.95E-1 * (1.6E-2)	6.52E-1 * (8.4E-2)	5.85E-1 * (1.1E-2)	6.20E-1 * (9.6E-2)
UF10	3	2.08E0 * (7.6E-1)	4.06E0 * (3.6E-1)	1.90E0 * (4.1E-1)	1.81E0 * (2.2E-1)	**8.97E-1** (2.9E-1)

Table B.30: Performance comparison using the IGD indicator on the UF benchmarks using 100,000 function evaluations.

		WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	randomLMEA	randomMOEA/DVA	randomS3-CMA-ES
n = 40							
UF1	2	9.14E-2 * (2.2E-2)	1.01E-1 * (2.6E-2)	9.04E-2 * (2.2E-2)	1.16E-1 * (1.2E-2)	**3.14E-2** (2.3E-3)	3.39E-1 * (1.5E-1)
UF2	2	5.41E-2 * (5.7E-3)	4.11E-2 * (7.4E-3)	4.21E-2 * (8.2E-3)	8.28E-2 * (8.7E-3)	**3.37E-2** (1.4E-3)	7.79E-2 * (3.1E-2)
UF3	2	**1.33E-1** (3.6E-2)	1.69E-1 * (2.0E-2)	1.68E-1 * (4.0E-2)	3.19E-1 * (2.8E-3)	2.26E-1 * (6.8E-3)	3.60E-1 * (3.5E-2)
UF4	2	5.19E-2 * (8.4E-3)	**4.72E-2** (1.1E-3)	4.88E-2 (5.6E-3)	6.33E-2 * (7.5E-3)	6.32E-2 * (2.9E-3)	9.98E-2 * (3.8E-3)
UF5	2	6.24E-1 * (1.6E-1)	3.86E-1 * (2.0E-1)	3.48E-1 * (2.0E-1)	**2.49E-1** (4.1E-2)	1.49E0 * (6.5E-2)	1.24E0 * (5.2E-1)
UF6	2	3.22E-1 * (1.7E-1)	**1.14E-1** (1.8E-1)	1.27E-1 (2.4E-1)	1.31E-1 (1.9E-2)	3.87E-1 * (1.1E-2)	1.21E0 * (1.3E0)
UF7	2	**4.68E-2** (9.4E-3)	5.56E-2 * (1.4E-2)	5.08E-2 (1.6E-2)	7.21E-2 * (2.5E-1)	7.87E-2 * (2.3E-3)	4.81E-1 * (1.4E-1)
UF8	3	2.52E-1 * (5.0E-2)	2.24E-1 * (1.2E-1)	5.28E-1 * (5.4E-2)	5.34E-1 * (5.7E-3)	**1.66E-1** (9.0E-3)	4.89E-1 * (1.8E-1)
UF9	3	4.81E-1 * (8.9E-2)	3.06E-1 * (1.4E-1)	3.13E-1 * (2.3E-1)	6.01E-1 * (5.8E-2)	**1.92E-1** (1.2E-2)	4.86E-1 * (1.2E-1)
UF10	3	1.28E0 * (2.9E-1)	**4.30E-1** (1.4E-1)	5.20E-1 * (3.9E-2)	6.32E-1 * (1.6E-1)	2.97E0 * (2.1E-1)	1.93E0 * (2.9E-1)
n = 1000							
UF1	2	2.72E-1 * (5.9E-3)	**1.57E-1** (3.7E-2)	1.96E-1 * (1.6E-2)	5.34E-1 * (8.0E-2)	1.64E0 * (5.1E-2)	5.76E-1 * (4.0E-1)
UF2	2	**8.50E-2** (4.1E-4)	1.13E-1 * (1.3E-2)	8.69E-2 * (3.2E-3)	3.23E-1 * (2.1E-2)	7.20E-1 * (4.3E-2)	9.51E-1 * (8.4E-2)
UF3	2	**2.54E-2** (3.8E-3)	1.38E-1 * (1.0E-2)	3.50E-2 * (1.0E-2)	3.70E-1 * (1.2E-2)	8.63E-1 * (6.8E-2)	7.65E-1 * (1.5E-1)
UF4	2	**5.49E-2** (9.0E-3)	7.40E-2 * (5.0E-3)	7.08E-2 * (1.8E-2)	1.79E-1 * (2.6E-3)	1.31E-1 * (9.2E-4)	1.45E-1 * (7.8E-3)
UF5	2	2.81E0 * (1.1E-1)	**1.20E0** (1.9E-1)	1.61E0 * (5.3E-1)	2.57E0 * (1.1E-1)	6.21E0 * (1.0E-1)	3.53E0 * (1.0E0)
UF6	2	1.02E0 * (5.9E-2)	4.74E-1 (2.6E-1)	**4.64E-1** (3.1E-1)	2.18E0 * (2.8E-1)	6.68E0 * (2.1E-1)	1.46E0 * (8.6E-2)
UF7	2	2.77E-1 * (7.4E-3)	**1.50E-1** (2.6E-2)	1.67E-1 * (2.0E-2)	6.78E-1 * (1.8E-1)	1.72E0 * (5.2E-2)	8.22E-1 * (3.9E-1)
UF8	3	**3.56E-1** (4.4E-3)	4.67E-1 * (2.9E-2)	5.42E-1 * (2.1E-3)	1.01E0 * (5.9E-2)	2.73E0 * (5.9E-1)	1.82E0 * (2.6E-1)
UF9	3	**5.70E-1** (7.2E-3)	6.52E-1 * (8.4E-2)	6.20E-1 * (9.6E-2)	1.21E0 * (8.9E-2)	3.17E0 * (3.5E-1)	1.86E0 * (1.0E0)
UF10	3	2.08E0 * (7.6E-1)	1.90E0 * (4.1E-1)	**8.97E-1** (2.9E-1)	5.80E0 * (4.4E-1)	1.57E1 * (2.0E0)	1.12E1 * (1.8E0)

Table B.31: Performance comparison using the IGD indicator on the WFG benchmarks using 100,000 function evaluations.

		WOF-NSGA-II	ReNSGA-II	WOF-MOEA/D	ReMOEA/D	WOF-Randomised
n = 40						
WFG1	2	**1.71E-1** (6.1E-2)	1.22E0 * (2.9E-2)	1.01E0 * (6.5E-2)	1.30E0 * (7.0E-3)	3.49E-1 * (9.5E-2)
WFG2	2	2.87E-2 (1.1E-2)	7.08E-1 * (7.0E-2)	9.68E-2 * (9.2E-3)	8.02E-1 * (5.8E-2)	**2.44E-2** (6.1E-3)
WFG3	2	2.98E-2 * (8.7E-3)	5.20E-1 * (8.9E-2)	4.81E-2 * (4.8E-3)	7.81E-1 * (3.5E-2)	**2.16E-2** (3.7E-3)
WFG4	2	2.24E-2 * (3.4E-3)	1.85E-1 * (6.7E-2)	7.19E-2 * (8.5E-3)	1.02E0 * (2.1E-1)	**1.85E-2** (2.4E-3)
WFG5	2	7.22E-2 * (7.1E-4)	2.50E-1 * (2.0E-1)	7.28E-2 * (2.4E-3)	6.28E-1 * (2.6E-1)	**6.94E-2** (1.2E-4)
WFG6	2	8.32E-2 * (1.2E-3)	8.23E-2 * (1.8E-3)	8.54E-2 * (5.4E-3)	1.53E-1 * (3.1E-1)	**4.13E-2** (4.9E-2)
WFG7	2	1.83E-2 * (1.2E-3)	2.47E-1 * (6.1E-2)	4.67E-2 * (1.5E-2)	8.24E-1 * (7.5E-2)	**1.38E-2** (5.7E-4)
WFG8	2	2.15E-1 * (1.6E-2)	4.41E-1 * (5.1E-2)	**1.67E-1** (1.8E-2)	8.99E-1 * (7.3E-2)	2.09E-1 * (5.9E-2)
WFG9	2	9.15E-2 (4.6E-2)	9.43E-2 * (6.7E-3)	9.31E-2 * (2.9E-3)	1.38E-1 * (4.6E-2)	**8.28E-2** (5.4E-2)
WFG1	3	**9.77E-1** (7.2E-2)	1.53E0 * (1.2E-2)	1.38E0 * (7.5E-2)	1.55E0 * (1.4E-2)	1.18E0 * (1.0E-1)
WFG2	3	2.24E-1 * (9.0E-3)	7.90E-1 * (1.2E-1)	3.02E-1 * (3.0E-2)	8.91E-1 * (3.4E-2)	**1.66E-1** (4.8E-3)
WFG3	3	1.35E-1 (2.7E-2)	8.26E-1 * (1.8E-1)	1.74E-1 * (3.0E-2)	1.03E0 * (4.6E-2)	**1.20E-1** (3.0E-2)
WFG4	3	2.91E-1 * (1.2E-2)	4.86E-1 * (5.4E-2)	2.97E-1 * (2.5E-2)	1.09E0 * (1.0E-1)	**2.35E-1** (4.5E-3)
WFG5	3	2.92E-1 * (1.4E-2)	4.23E-1 * (7.7E-2)	2.91E-1 * (1.0E-2)	8.04E-1 * (2.9E-1)	**2.36E-1** (2.5E-3)
WFG6	3	3.13E-1 * (1.9E-2)	3.38E-1 * (3.4E-2)	3.28E-1 * (1.9E-2)	4.89E-1 * (5.7E-2)	**2.41E-1** (4.1E-3)
WFG7	3	3.13E-1 * (1.1E-1)	6.15E-1 * (8.1E-2)	2.95E-1 * (3.8E-2)	1.09E0 * (9.4E-2)	**2.28E-1** (4.0E-2)
WFG8	3	5.02E-1 * (3.6E-2)	8.90E-1 * (8.3E-2)	4.93E-1 * (8.8E-2)	1.12E0 * (5.2E-2)	**3.53E-1** (6.4E-2)
WFG9	3	3.18E-1 * (1.0E-2)	3.30E-1 * (2.6E-2)	3.15E-1 * (2.1E-2)	4.81E-1 * (4.2E-2)	**2.46E-1** (6.3E-3)
n = 1000						
WFG1	2	1.27E0 * (1.2E-2)	1.32E0 * (1.1E-2)	1.26E0 * (8.7E-3)	1.38E0 * (1.3E-2)	**1.22E0** (2.7E-2)
WFG2	2	2.35E-1 * (7.5E-2)	1.22E0 * (5.4E-3)	2.17E-1 * (3.6E-2)	1.27E0 * (1.9E-2)	**1.40E-1** (2.9E-2)
WFG3	2	2.12E-1 * (2.9E-2)	1.12E0 * (4.6E-2)	1.71E-1 * (5.1E-2)	1.19E0 * (4.3E-2)	**1.25E-1** (3.1E-2)
WFG4	2	1.77E-1 * (2.3E-2)	5.74E-1 * (2.0E-2)	1.64E-1 * (3.1E-2)	1.68E0 * (1.1E-1)	**1.27E-1** (1.8E-2)
WFG5	2	9.28E-2 * (6.6E-2)	1.34E0 * (9.4E-2)	1.01E-1 * (1.6E-2)	1.58E0 * (5.5E-2)	**6.97E-2** (3.3E-3)
WFG6	2	2.34E-2 * (5.4E-3)	2.52E0 * (1.6E-2)	7.35E-2 * (3.5E-2)	2.44E0 * (2.0E-2)	**1.35E-2** (3.6E-3)
WFG7	2	2.41E-1 * (2.7E-2)	9.61E-1 * (4.3E-2)	2.23E-1 * (9.9E-2)	1.56E0 * (1.5E-1)	**1.39E-1** (2.9E-2)
WFG8	2	3.59E-1 * (4.3E-2)	1.36E0 * (6.4E-3)	3.77E-1 * (5.6E-2)	1.65E0 * (6.9E-2)	**2.81E-1** (7.8E-2)
WFG9	2	1.04E-1 * (5.2E-2)	4.88E-1 * (7.8E-2)	8.67E-2 * (3.6E-2)	8.82E-1 * (1.7E-1)	**5.05E-2** (2.9E-2)
WFG1	3	1.55E0 * (3.6E-2)	1.68E0 * (5.1E-2)	1.62E0 * (5.5E-2)	1.70E0 * (1.3E-1)	**1.50E0** (1.8E-2)
WFG2	3	5.15E-1 * (1.1E-1)	2.11E0 * (1.3E-2)	7.63E-1 * (1.0E-1)	2.15E0 * (2.0E-2)	**4.04E-1** (1.0E-1)
WFG3	3	3.97E-1 (1.5E-1)	1.08E0 * (2.4E-1)	**3.39E-1** (9.8E-2)	1.33E0 * (8.7E-2)	3.98E-1 (1.4E-1)
WFG4	3	6.48E-1 * (1.2E-1)	1.34E0 * (5.2E-2)	5.25E-1 * (7.4E-2)	2.02E0 * (2.0E-1)	**4.22E-1** (9.3E-2)
WFG5	3	6.22E-1 * (1.0E-1)	2.61E0 * (2.4E-1)	4.63E-1 * (4.0E-1)	3.09E0 * (1.3E-1)	**4.01E-1** (1.1E-1)
WFG6	3	4.03E-1 * (8.8E-2)	3.92E0 * (3.6E-2)	3.85E-1 * (6.4E-2)	3.75E0 * (7.9E-2)	**2.46E-1** (1.8E-2)
WFG7	3	7.19E-1 * (3.7E-2)	1.81E0 * (1.1E-1)	6.61E-1 * (8.3E-2)	2.31E0 * (9.1E-2)	**5.29E-1** (5.5E-2)
WFG8	3	1.07E0 * (1.2E-1)	1.97E0 * (8.8E-2)	1.35E0 * (2.1E-1)	2.31E0 * (1.6E-1)	**7.72E-1** (1.3E-1)
WFG9	3	7.51E-1 * (1.2E-1)	1.12E0 * (8.6E-2)	5.88E-1 * (7.6E-2)	1.69E0 * (1.8E-1)	**4.31E-1** (1.1E-1)

Table B.32: Performance comparison using the IGD indicator on the WFG benchmarks using 100, 000 function evaluations.

		WOF-SMPSO	LS-SMPSO	WOF-NSGA-II	LS-NSGA-II	WOF-Randomised
n = 40						
WFG1	2	1.12E0 * (7.3E-2)	1.23E0 * (2.4E-2)	**1.71E-1** (6.1E-2)	3.07E-1 * (8.1E-2)	3.49E-1 * (9.5E-2)
WFG2	2	2.80E-2 * (7.5E-3)	2.44E-1 * (4.4E-2)	2.87E-2 * (1.1E-2)	**1.90E-2** (1.4E-2)	2.44E-2 * (6.1E-3)
WFG3	2	3.00E-2 * (6.4E-3)	3.64E-2 * (2.1E-2)	2.98E-2 * (8.7E-3)	2.69E-2 * (6.9E-3)	**2.16E-2** (3.7E-3)
WFG4	2	7.68E-2 * (3.4E-2)	9.48E-2 * (2.2E-2)	2.24E-2 * (3.4E-3)	**1.61E-2** (8.1E-4)	1.85E-2 * (2.4E-3)
WFG5	2	**6.79E-2** (4.6E-3)	1.29E-1 * (4.9E-2)	7.22E-2 * (7.1E-4)	7.17E-2 (4.4E-2)	6.94E-2 (1.2E-4)
WFG6	2	**2.28E-2** (9.8E-3)	3.53E-2 * (7.8E-3)	8.32E-2 * (1.2E-3)	4.49E-2 * (4.2E-2)	4.13E-2 * (4.9E-2)
WFG7	2	1.88E-2 * (1.6E-3)	7.66E-2 * (1.0E-1)	1.83E-2 * (1.2E-3)	3.37E-2 * (1.4E-2)	**1.38E-2** (5.7E-4)
WFG8	2	**1.07E-1** (8.7E-2)	3.04E-1 * (2.5E-2)	2.15E-1 * (1.6E-2)	2.92E-1 * (1.0E-1)	2.09E-1 * (5.9E-2)
WFG9	2	**4.27E-2** (1.3E-2)	6.68E-2 * (2.5E-2)	9.15E-2 * (4.6E-2)	4.39E-2 (8.4E-3)	8.28E-2 (5.4E-2)
WFG1	3	1.51E0 * (2.1E-2)	1.54E0 * (1.8E-2)	**9.77E-1** (7.2E-2)	1.07E0 * (8.0E-2)	1.18E0 * (1.0E-1)
WFG2	3	2.36E-1 * (1.7E-2)	5.78E-1 * (1.9E-1)	2.24E-1 * (9.0E-3)	2.15E-1 * (1.2E-2)	**1.66E-1** (4.8E-3)
WFG3	3	1.43E-1 * (2.6E-2)	1.16E-1 (1.1E-1)	1.35E-1 * (2.7E-2)	**8.72E-2** (1.9E-2)	1.20E-1 * (3.0E-2)
WFG4	3	3.46E-1 * (3.2E-2)	5.12E-1 * (8.9E-2)	2.91E-1 * (1.2E-2)	2.79E-1 * (1.7E-2)	**2.35E-1** (4.5E-3)
WFG5	3	3.04E-1 * (2.5E-2)	5.15E-1 * (2.1E-1)	2.92E-1 * (1.4E-2)	3.01E-1 * (2.5E-2)	**2.36E-1** (2.5E-3)
WFG6	3	3.48E-1 * (6.2E-2)	3.76E-1 * (5.0E-2)	3.13E-1 * (1.9E-2)	3.16E-1 * (1.9E-2)	**2.41E-1** (4.1E-3)
WFG7	3	2.99E-1 * (1.9E-2)	5.31E-1 * (3.1E-2)	3.13E-1 * (1.1E-1)	4.35E-1 * (2.4E-2)	**2.28E-1** (4.0E-2)
WFG8	3	5.60E-1 * (3.4E-2)	7.72E-1 * (1.0E-1)	5.02E-1 * (3.6E-2)	5.98E-1 * (5.4E-2)	**3.53E-1** (6.4E-2)
WFG9	3	3.38E-1 * (3.1E-2)	4.34E-1 * (7.8E-2)	3.18E-1 * (1.0E-2)	3.31E-1 * (2.5E-2)	**2.46E-1** (6.3E-3)
n = 1000						
WFG1	2	**1.20E0** (5.1E-2)	1.27E0 * (8.9E-3)	1.27E0 * (1.2E-2)	1.29E0 * (4.3E-3)	1.22E0 * (2.7E-2)
WFG2	2	**7.25E-2** (2.1E-2)	4.49E-1 * (4.7E-3)	2.35E-1 * (7.5E-2)	4.57E-1 * (5.1E-4)	1.40E-1 * (2.9E-2)
WFG3	2	**8.94E-2** (1.3E-2)	4.16E-1 * (7.1E-3)	2.12E-1 * (2.9E-2)	4.44E-1 * (6.2E-3)	1.25E-1 * (3.1E-2)
WFG4	2	**1.11E-1** (8.2E-3)	5.51E-1 * (1.8E-2)	1.77E-1 * (2.3E-2)	1.20E-1 * (1.1E-2)	1.27E-1 * (1.8E-2)
WFG5	2	**6.78E-2** (4.8E-3)	5.49E-1 * (2.0E-2)	9.28E-2 * (6.6E-2)	1.69E-1 * (8.5E-2)	6.97E-2 (3.3E-3)
WFG6	2	1.69E-2 * (1.4E-3)	1.75E-2 * (3.9E-3)	2.34E-2 * (5.4E-3)	1.95E-1 * (1.7E-1)	**1.35E-2** (3.6E-3)
WFG7	2	**7.70E-2** (1.1E-2)	6.56E-1 * (1.1E-2)	2.41E-1 * (2.7E-2)	2.57E-1 * (1.9E-2)	1.39E-1 * (2.9E-2)
WFG8	2	**1.03E-1** (1.0E-1)	4.17E-1 * (9.9E-3)	3.59E-1 * (4.3E-2)	4.72E-1 * (4.2E-3)	2.81E-1 * (7.8E-2)
WFG9	2	**3.83E-2** (7.2E-3)	3.48E-1 * (7.8E-2)	1.04E-1 * (5.2E-2)	1.33E-1 * (2.8E-2)	5.05E-2 * (2.9E-2)
WFG1	3	1.51E0 * (1.9E-2)	1.73E0 * (4.8E-2)	1.55E0 * (3.6E-2)	1.70E0 * (2.6E-2)	**1.50E0** (1.8E-2)
WFG2	3	**2.43E-1** (1.6E-2)	1.31E0 * (1.0E-2)	5.15E-1 * (1.1E-1)	1.32E0 * (9.6E-3)	4.04E-1 * (1.0E-1)
WFG3	3	**1.13E-1** (3.0E-2)	7.16E-1 * (1.9E-3)	3.97E-1 * (1.5E-1)	5.98E-1 * (5.0E-2)	3.98E-1 * (1.4E-1)
WFG4	3	**3.70E-1** (2.4E-2)	1.42E0 * (6.1E-2)	6.48E-1 * (1.2E-1)	1.08E0 * (4.8E-2)	4.22E-1 * (9.3E-2)
WFG5	3	4.28E-1 (1.4E-1)	1.49E0 * (5.4E-2)	6.22E-1 * (1.0E-1)	1.11E0 * (5.8E-2)	**4.01E-1** (1.1E-1)
WFG6	3	3.78E-1 * (5.9E-2)	1.20E0 * (5.8E-2)	4.03E-1 * (8.8E-2)	6.78E-1 * (2.5E-1)	**2.46E-1** (1.8E-2)
WFG7	3	**3.58E-1** (4.1E-2)	1.42E0 * (9.6E-2)	7.19E-1 * (3.7E-2)	1.31E0 * (3.2E-2)	5.29E-1 * (5.5E-2)
WFG8	3	**6.19E-1** (7.5E-2)	1.32E0 * (2.8E-1)	1.07E0 * (1.2E-1)	1.58E0 * (1.8E-1)	7.72E-1 * (1.3E-1)
WFG9	3	**3.68E-1** (3.6E-2)	1.25E0 * (1.0E-1)	7.51E-1 * (1.2E-1)	7.00E-1 * (4.2E-1)	4.31E-1 * (1.1E-1)

Table B.33: Performance comparison using the IGD indicator on the WFG benchmarks using 100,000 function evaluations.

		WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	randomLMEA	randomMOEA/DVA	randomS3-CMA-ES
n = 40							
WFG1	2	1.12E0 * (7.3E-2)	**1.71E-1** (6.1E-2)	3.49E-1 * (9.5E-2)	6.46E-1 * (1.5E-1)	1.13E0 * (1.0E-1)	1.52E0 * (1.4E-1)
WFG2	2	2.80E-2 * (7.5E-3)	2.87E-2 (1.1E-2)	**2.44E-2** (6.1E-3)	6.66E-1 * (4.3E-1)	2.39E-1 * (2.5E-2)	1.08E0 * (1.2E-1)
WFG3	2	3.00E-2 * (6.4E-3)	2.98E-2 * (8.7E-3)	**2.16E-2** (3.7E-3)	7.24E-2 * (3.6E-2)	3.02E-2 * (1.1E-2)	1.06E0 * (2.1E-1)
WFG4	2	7.68E-2 * (3.4E-2)	2.24E-2 * (3.4E-3)	**1.85E-2** (2.4E-3)	2.79E-2 * (1.7E-2)	7.88E-1 * (8.7E-2)	1.44E0 * (3.9E-1)
WFG5	2	**6.79E-2** (4.6E-3)	7.22E-2 * (7.1E-4)	6.94E-2 (1.2E-4)	8.66E-2 * (1.0E-2)	5.18E-1 * (9.6E-2)	9.11E-1 * (1.9E-1)
WFG6	2	**2.28E-2** (9.8E-3)	8.32E-2 * (1.2E-3)	4.13E-2 * (4.9E-2)	5.97E-2 * (1.1E-2)	6.07E-1 * (1.1E-1)	2.20E0 * (2.9E-1)
WFG7	2	1.88E-2 * (1.6E-3)	1.83E-2 * (1.2E-3)	**1.38E-2** (5.7E-4)	8.49E-2 * (7.3E-2)	1.70E-2 * (1.7E-3)	1.42E0 * (4.1E-1)
WFG8	2	**1.07E-1** (8.7E-2)	2.15E-1 * (1.6E-2)	2.09E-1 * (5.9E-2)	3.51E-1 * (2.6E-2)	6.94E-1 * (4.7E-2)	1.20E0 * (2.5E-1)
WFG9	2	**4.27E-2** (1.3E-2)	9.15E-2 * (4.6E-2)	8.28E-2 (5.4E-2)	9.57E-2 * (6.3E-3)	9.11E-2 * (2.5E-3)	1.06E0 * (3.1E-1)
WFG1	3	1.51E0 * (2.1E-2)	9.77E-1 * (7.2E-2)	1.18E0 * (1.0E-1)	**7.26E-1** (1.5E-1)	1.46E0 * (6.2E-2)	1.59E0 * (7.5E-2)
WFG2	3	2.36E-1 * (1.7E-2)	2.24E-1 * (9.0E-3)	**1.66E-1** (4.8E-3)	5.17E-1 * (1.6E-2)	1.05E0 * (6.3E-1)	7.94E-1 * (2.5E-1)
WFG3	3	1.43E-1 * (2.6E-2)	1.35E-1 * (2.7E-2)	1.20E-1 * (3.0E-2)	**8.54E-2** (2.2E-2)	4.50E-1 * (6.3E-2)	1.29E0 * (3.9E-1)
WFG4	3	3.46E-1 * (3.2E-2)	2.91E-1 * (1.2E-2)	2.35E-1 * (4.5E-3)	**2.18E-1** (3.7E-3)	1.54E0 * (1.7E-1)	1.45E0 * (2.6E-1)
WFG5	3	3.04E-1 * (2.5E-2)	2.92E-1 * (1.4E-2)	2.36E-1 * (2.5E-3)	**2.33E-1** (3.4E-3)	9.23E-1 * (9.5E-2)	1.19E0 * (3.5E-1)
WFG6	3	3.48E-1 * (6.2E-2)	3.13E-1 * (1.9E-2)	2.41E-1 * (4.1E-3)	**2.23E-1** (5.4E-3)	9.45E-1 * (1.0E-1)	2.09E0 * (6.9E-1)
WFG7	3	2.99E-1 * (1.9E-2)	3.13E-1 * (1.1E-1)	**2.28E-1** (4.0E-2)	2.54E-1 * (4.5E-2)	3.97E-1 * (3.6E-2)	1.58E0 * (2.2E-1)
WFG8	3	5.60E-1 * (3.4E-2)	5.02E-1 * (3.6E-2)	3.53E-1 (6.4E-2)	**3.45E-1** (1.0E-2)	1.10E0 * (1.4E-1)	1.63E0 * (3.2E-1)
WFG9	3	3.38E-1 * (3.1E-2)	3.18E-1 * (1.0E-2)	**2.46E-1** (6.3E-3)	2.96E-1 * (3.8E-2)	3.46E-1 * (1.8E-2)	1.22E0 * (2.4E-1)
n = 1000							
WFG1	2	**1.20E0** (5.1E-2)	1.27E0 * (1.2E-2)	1.22E0 * (2.7E-2)	1.77E0 * (2.1E-2)	1.32E0 * (1.3E-2)	1.67E0 * (1.5E-1)
WFG2	2	**7.25E-2** (2.1E-2)	2.35E-1 * (7.5E-2)	1.40E-1 * (2.9E-2)	8.95E-1 * (5.4E-1)	9.55E-1 * (3.8E-2)	3.13E0 * (2.9E-3)
WFG3	2	**8.94E-2** (1.3E-2)	2.12E-1 * (2.9E-2)	1.25E-1 * (3.1E-2)	8.13E-1 * (2.2E-2)	8.48E-1 * (4.6E-2)	2.48E0 * (2.4E-2)
WFG4	2	**1.11E-1** (8.2E-3)	1.77E-1 * (2.3E-2)	1.27E-1 * (1.8E-2)	1.06E0 * (4.6E-2)	1.50E0 * (2.3E-2)	1.99E0 * (5.3E-2)
WFG5	2	**6.78E-2** (4.8E-3)	9.28E-2 * (6.6E-2)	6.97E-2 (3.3E-3)	9.84E-1 * (2.1E-2)	1.17E0 * (3.3E-2)	1.65E0 * (3.9E-2)
WFG6	2	1.69E-2 * (1.4E-3)	2.34E-2 * (5.4E-3)	**1.35E-2** (3.6E-3)	9.50E-1 * (3.5E-2)	1.59E0 * (3.7E-2)	1.60E0 * (3.5E-2)
WFG7	2	**7.70E-2** (1.1E-2)	2.41E-1 * (2.7E-2)	1.39E-1 * (2.9E-2)	9.75E-1 * (8.9E-2)	9.91E-1 * (2.7E-2)	1.91E0 * (5.5E-2)
WFG8	2	**1.03E-1** (1.0E-1)	3.59E-1 * (4.3E-2)	2.81E-1 * (7.8E-2)	1.10E0 * (3.2E-2)	1.46E0 * (2.0E-2)	2.10E0 * (3.6E-2)
WFG9	2	**3.83E-2** (7.2E-3)	1.04E-1 * (5.2E-2)	5.05E-2 * (2.9E-2)	1.08E0 * (1.7E-1)	1.58E0 * (3.7E-2)	1.65E0 * (1.6E-1)
WFG1	3	1.51E0 * (1.9E-2)	1.55E0 * (3.6E-2)	**1.50E0** (1.8E-2)	1.93E0 * (4.8E-2)	1.60E0 * (2.4E-2)	1.94E0 * (5.9E-2)
WFG2	3	**2.43E-1** (1.6E-2)	5.15E-1 * (1.1E-1)	4.04E-1 * (1.0E-1)	1.74E0 * (3.6E-2)	1.93E0 * (1.2E-2)	5.35E0 * (8.5E-3)
WFG3	3	**1.13E-1** (3.0E-2)	3.97E-1 * (1.5E-1)	3.98E-1 * (1.4E-1)	9.46E-1 * (5.4E-2)	1.59E0 * (4.3E-2)	3.13E0 * (6.4E-2)
WFG4	3	**3.70E-1** (2.4E-2)	6.48E-1 * (1.2E-1)	4.22E-1 * (9.3E-2)	2.04E0 * (7.0E-2)	2.85E0 * (3.6E-2)	2.70E0 * (9.5E-2)
WFG5	3	4.28E-1 (1.4E-1)	6.22E-1 * (1.0E-1)	**4.01E-1** (1.1E-1)	1.62E0 * (8.1E-2)	2.10E0 * (5.2E-2)	3.19E0 * (7.0E-2)
WFG6	3	3.78E-1 * (5.9E-2)	4.03E-1 * (8.8E-2)	**2.46E-1** (1.8E-2)	1.29E0 * (7.2E-2)	2.32E0 * (4.1E-2)	2.97E0 * (7.6E-2)
WFG7	3	**3.58E-1** (4.1E-2)	7.19E-1 * (3.7E-2)	5.29E-1 * (5.5E-2)	1.75E0 * (9.4E-2)	2.04E0 * (7.2E-2)	2.56E0 * (5.7E-2)
WFG8	3	**6.19E-1** (7.5E-2)	1.07E0 * (1.2E-1)	7.72E-1 * (1.3E-1)	1.75E0 * (7.5E-2)	2.48E0 * (5.8E-2)	3.83E0 * (5.3E-2)
WFG9	3	**3.68E-1** (3.6E-2)	7.51E-1 * (1.2E-1)	4.31E-1 * (1.1E-1)	1.65E0 * (1.4E-1)	1.59E0 * (8.1E-1)	3.16E0 * (2.7E-1)

Table B.34: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100, 000 function evaluations.

		WOF-NSGA-II	ReNSGA-II	WOF-MOEA/D	ReMOEA/D	WOF-Randomised
n = 40						
DTLZ1	2	6.22E0 (4.4E0)	3.18E2 * (6.7E0)	4.40E0 (2.2E0)	3.24E2 * (8.3E0)	**3.33E0** (1.1E1)
DTLZ2	2	5.05E-3 * (2.4E-4)	1.19E-1 * (2.9E-1)	4.42E-3 * (1.9E-3)	5.22E0 * (9.6E-1)	**3.96E-3** (3.4E-7)
DTLZ3	2	2.00E1 * (7.5E0)	8.75E2 * (3.0E1)	1.61E1 (2.0E1)	9.08E2 * (1.4E1)	**1.03E1** (1.9E1)
DTLZ4	2	5.19E-3 * (4.4E-4)	6.02E-1 * (3.8E-1)	5.87E-3 * (2.4E-3)	5.61E0 * (9.1E-1)	**3.96E-3** (2.8E-6)
DTLZ5	2	5.03E-3 * (2.0E-4)	2.88E-1 * (2.4E-1)	4.46E-3 * (2.0E-3)	4.97E0 * (8.4E-1)	**3.96E-3** (1.9E-7)
DTLZ6	2	5.62E-3 * (3.7E-4)	3.00E0 * (1.0E0)	6.52E-3 * (7.4E-4)	6.00E0 * (2.0E0)	**3.96E-3** (5.3E-8)
DTLZ7	2	**5.44E-3** (4.3E-1)	2.57E-1 * (2.8E-1)	4.46E-1 * (4.0E-4)	2.23E0 * (1.5E0)	4.42E-1 (4.3E-1)
DTLZ1	3	2.77E1 * (1.2E1)	2.75E2 * (1.5E0)	**1.08E1** (1.1E1)	2.66E2 * (7.4E0)	1.11E1 (2.4E1)
DTLZ2	3	7.32E-2 * (3.4E-3)	3.86E-1 * (5.2E-1)	5.50E-2 * (2.9E-3)	4.62E0 * (5.2E-1)	**5.44E-2** (6.1E-6)
DTLZ3	3	6.04E1 * (4.2E1)	8.64E2 * (2.7E1)	2.52E1 (3.1E1)	8.81E2 * (1.5E1)	**2.41E1** (3.3E1)
DTLZ4	3	7.10E-2 * (3.2E-3)	8.73E-1 * (4.2E-1)	5.47E-2 * (2.1E-3)	4.98E0 * (7.3E-1)	**5.44E-2** (8.3E-6)
DTLZ5	3	**6.48E-3** (5.1E-4)	3.85E-1 * (4.8E-1)	3.16E-2 * (2.0E-3)	4.34E0 * (9.9E-1)	1.37E-2 * (2.9E-3)
DTLZ6	3	**6.46E-3** (7.0E-4)	3.98E0 * (2.5E0)	3.52E-2 * (4.2E-3)	5.01E0 * (3.4E0)	2.29E-2 * (3.5E-3)
DTLZ7	3	**8.54E-2** (7.1E-1)	5.86E-1 * (5.7E-1)	8.02E-1 * (6.5E-1)	2.59E0 * (1.5E0)	8.01E-1 (7.2E-1)
DTLZ1	4	1.60E2 * (4.9E1)	2.45E2 * (2.0E1)	**5.74E0** (6.6E0)	2.28E2 * (5.9E0)	1.02E1 (1.4E1)
DTLZ2	4	1.64E-1 * (8.1E-3)	1.56E0 * (5.4E-1)	1.48E-1 * (3.9E-3)	4.51E0 * (1.0E0)	**1.40E-1** (2.0E-5)
DTLZ3	4	2.99E2 * (1.4E2)	8.35E2 * (3.6E1)	**2.32E1** (4.8E1)	8.50E2 * (2.0E1)	3.25E1 (7.3E1)
DTLZ4	4	1.63E-1 * (5.7E-3)	1.15E0 * (5.5E-1)	4.58E-1 * (6.0E-1)	4.66E0 * (5.4E-1)	**1.40E-1** (7.5E-5)
DTLZ5	4	1.52E-1 * (4.1E-2)	1.92E0 * (1.1E0)	**4.69E-2** (5.6E-3)	4.07E0 * (1.1E0)	8.13E-2 * (3.5E-2)
DTLZ6	4	1.02E1 * (2.2E0)	7.40E0 * (2.8E0)	**5.12E-2** (1.2E-1)	3.11E0 * (2.0E0)	1.83E-1 * (6.1E-2)
DTLZ7	4	**2.41E-1** (8.9E-1)	7.87E-1 * (5.2E-1)	1.12E0 * (7.2E-1)	4.83E0 * (3.6E0)	3.83E-1 (9.1E-1)
DTLZ1	5	2.02E2 * (5.6E1)	2.89E2 * (3.7E1)	**1.87E1** (2.0E1)	2.15E2 * (1.8E1)	2.10E1 (2.6E1)
DTLZ2	5	2.71E-1 * (1.3E-2)	3.03E0 * (9.2E-1)	2.31E-1 * (4.8E-3)	3.83E0 * (6.7E-1)	**2.12E-1** (1.4E-4)
DTLZ3	5	6.24E2 * (1.6E2)	8.16E2 * (2.7E1)	8.63E1 (4.5E1)	8.23E2 * (1.4E1)	**4.41E1** (8.4E1)
DTLZ4	5	2.73E-1 * (1.3E-2)	1.81E0 * (4.3E-1)	4.30E-1 * (2.1E-1)	4.52E0 * (8.8E-1)	**2.12E-1** (1.7E-4)
DTLZ5	5	1.47E0 * (6.6E-1)	2.55E0 * (9.0E-1)	**3.48E-2** (8.2E-3)	3.92E0 * (7.6E-1)	1.45E-1 * (4.0E-2)
DTLZ6	5	1.96E1 * (3.8E0)	7.40E0 * (2.9E0)	**3.57E-2** (8.3E-3)	3.45E0 * (1.9E0)	4.52E-1 * (1.9E-1)
DTLZ7	5	4.01E-1 (1.9E-2)	9.47E-1 * (1.5E0)	9.09E-1 * (6.5E-1)	6.23E0 * (2.6E0)	**3.97E-1** (5.8E-2)
n = 1000						
DTLZ1	2	5.15E3 * (1.7E3)	1.10E4 * (3.9E2)	8.86E2 (5.7E2)	1.16E4 * (3.4E2)	**7.98E2** (9.7E2)
DTLZ2	2	7.18E-1 * (6.3E-1)	5.95E-1 * (1.1E-1)	1.03E-1 * (1.1E-1)	1.73E2 * (9.2E1)	**4.56E-3** (1.0E-3)
DTLZ3	2	1.38E4 * (4.5E3)	3.02E4 * (8.4E2)	2.80E3 * (1.2E3)	3.18E4 * (1.5E3)	**1.51E3** (2.7E3)
DTLZ4	2	6.91E-1 * (4.0E-1)	2.46E0 * (2.9E-1)	1.34E-2 * (7.7E-3)	1.89E2 * (1.6E1)	**8.13E-3** (4.8E-3)
DTLZ5	2	7.07E-1 * (5.2E-1)	6.51E-1 * (1.3E-1)	1.36E-1 * (6.2E-2)	1.56E2 * (1.2E2)	**4.77E-3** (2.3E-3)
DTLZ6	2	7.65E1 * (4.7E1)	4.90E2 * (2.2E1)	1.32E-2 * (2.7E-2)	4.91E2 * (2.7E1)	**3.96E-3** (6.2E-8)
DTLZ7	2	4.42E-1 * (1.1E-4)	5.91E0 * (3.0E-1)	4.56E-1 * (2.9E-2)	7.05E0 * (3.9E-1)	**4.42E-1** (4.3E-1)
DTLZ1	3	5.40E3 * (1.8E3)	1.10E4 * (7.6E2)	**6.50E2** (5.9E2)	9.74E3 * (4.9E2)	1.02E3 (1.1E3)
DTLZ2	3	1.85E0 * (1.3E0)	4.45E0 * (1.2E0)	1.03E-1 * (1.7E-1)	1.18E0 * (4.9E-1)	**5.89E-2** (4.1E-3)
DTLZ3	3	1.74E4 * (4.4E3)	3.26E4 * (1.1E3)	**2.35E3** (1.7E3)	3.11E4 * (1.1E3)	3.45E3 (2.9E3)
DTLZ4	3	1.82E0 * (1.5E0)	6.00E0 * (3.7E0)	5.43E-1 * (4.0E-1)	1.73E2 * (1.8E1)	**6.55E-2** (1.0E-2)
DTLZ5	3	2.25E0 * (2.1E0)	5.71E0 * (1.4E0)	7.28E-1 * (4.9E-1)	1.40E0 * (9.8E0)	**2.50E-2** (1.2E-2)
DTLZ6	3	2.33E2 * (9.5E1)	5.10E2 * (2.0E1)	4.95E-2 * (2.1E-1)	4.67E2 * (1.3E1)	**2.43E-2** (3.3E-3)
DTLZ7	3	**7.99E-1** (7.1E-1)	8.75E0 * (3.9E-1)	8.02E-1 * (2.0E-3)	1.04E1 * (5.3E-1)	8.03E-1 * (4.2E-1)
DTLZ1	4	6.32E3 * (3.2E3)	1.31E4 * (1.3E3)	**4.04E2** (4.5E2)	8.88E3 * (4.2E2)	7.55E2 * (1.0E3)
DTLZ2	4	5.22E0 * (2.3E0)	4.83E1 * (2.0E1)	1.04E0 (8.8E-1)	1.33E0 * (8.9E-1)	**2.26E-1** (2.1E-1)
DTLZ3	4	2.08E4 * (6.4E3)	3.40E4 * (7.3E2)	**1.03E3** (1.5E3)	3.07E4 * (6.2E2)	4.10E3 * (3.9E3)
DTLZ4	4	8.91E0 * (5.0E0)	2.37E1 * (1.2E1)	7.53E-1 (5.8E-1)	1.40E2 * (2.9E1)	**7.17E-1** (9.0E-1)
DTLZ5	4	4.84E1 * (1.7E1)	6.54E1 * (2.5E1)	**7.41E-1** (1.4E-1)	1.03E0 * (6.4E-1)	2.34E0 * (4.5E0)
DTLZ6	4	6.26E2 * (8.2E1)	5.22E2 * (1.7E1)	7.42E-1 * (—)	4.57E2 * (1.5E1)	**3.06E-1** (3.1E-1)
DTLZ7	4	**1.11E0** (4.9E-1)	1.16E1 * (5.4E-1)	1.12E0 * (5.6E-3)	1.42E1 * (9.6E-1)	1.13E0 * (4.9E-1)
DTLZ1	5	8.15E3 * (3.2E3)	1.53E4 * (1.6E3)	**1.72E3** (1.7E3)	8.49E3 * (5.1E2)	1.96E3 (1.3E3)
DTLZ2	5	3.44E1 * (7.2E0)	9.68E1 * (1.2E1)	**3.13E-1** (3.6E-1)	1.51E0 * (7.1E-1)	6.81E0 * (7.0E0)
DTLZ3	5	2.46E4 * (1.1E4)	3.42E4 * (1.8E3)	**4.53E3** (3.7E3)	3.13E4 * (1.5E3)	6.62E3 (4.4E3)
DTLZ4	5	3.44E1 * (8.4E0)	6.34E1 * (2.7E1)	**6.56E-1** (4.0E-1)	1.16E2 * (8.6E1)	1.18E1 * (5.6E0)
DTLZ5	5	6.70E1 * (1.8E1)	8.35E1 * (2.6E1)	**7.46E-1** (1.1E-2)	1.51E0 * (1.1E0)	1.61E1 * (2.0E1)
DTLZ6	5	6.48E2 * (8.3E1)	5.21E2 * (1.6E1)	**1.14E2** (1.4E2)	4.54E2 * (2.4E1)	2.28E2 * (1.3E2)
DTLZ7	5	1.46E0 (1.0E0)	1.49E1 * (4.5E-1)	1.55E0 * (3.2E-2)	1.82E1 * (1.2E0)	**9.18E-1** (1.0E0)

Table B.35: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100,000 function evaluations.

		WOF-SMPSO	LS-SMPSO	WOF-NSGA-II	LS-NSGA-II	WOF-Randomised
n = 40						
DTLZ1	2	**2.25E-3** (1.1E-4)	2.52E-3 * (2.8E-4)	6.22E0 * (4.4E0)	2.81E-3 * (2.6E-4)	3.33E0 * (1.1E1)
DTLZ2	2	5.11E-3 * (2.9E-4)	5.15E-3 * (2.1E-4)	5.05E-3 * (2.4E-4)	5.12E-3 * (2.9E-4)	**3.96E-3** (3.4E-7)
DTLZ3	2	**4.98E-3** (6.0E-4)	5.05E-3 (1.6E-4)	2.00E1 * (7.5E0)	5.64E-3 * (3.0E-4)	1.03E1 * (1.9E1)
DTLZ4	2	5.31E-3 * (7.3E-1)	5.22E-3 * (2.8E-4)	5.19E-3 * (4.4E-4)	5.12E-3 * (2.1E-4)	**3.96E-3** (2.8E-6)
DTLZ5	2	5.07E-3 * (2.0E-4)	5.19E-3 * (3.5E-4)	5.03E-3 * (2.0E-4)	5.17E-3 * (2.7E-4)	**3.96E-3** (1.9E-7)
DTLZ6	2	5.15E-3 * (3.3E-4)	5.16E-3 * (2.8E-4)	5.62E-3 * (3.7E-4)	5.67E-3 * (2.8E-4)	**3.96E-3** (5.3E-8)
DTLZ7	2	4.42E-1 (4.3E-1)	4.42E-1 * (9.3E-5)	**5.44E-3** (4.3E-1)	4.42E-1 * (4.7E-5)	4.42E-1 (4.3E-1)
DTLZ1	3	7.27E-2 * (3.8E-1)	1.87E-1 * (3.5E-1)	2.77E1 * (1.2E1)	**2.89E-2** (1.8E-3)	1.11E1 * (2.4E1)
DTLZ2	3	8.41E-2 * (6.4E-3)	1.26E-1 * (3.7E-2)	7.32E-2 * (3.4E-3)	7.32E-2 * (4.1E-3)	**5.44E-2** (6.1E-6)
DTLZ3	3	1.12E-1 * (2.7E-1)	4.25E-1 * (4.0E-1)	6.04E1 * (4.2E1)	**7.31E-2** (3.9E-3)	2.41E1 * (3.3E1)
DTLZ4	3	2.46E-1 * (1.8E-1)	2.82E-1 * (2.7E-1)	7.10E-2 * (3.2E-3)	7.16E-2 * (3.8E-3)	**5.44E-2** (8.3E-6)
DTLZ5	3	**5.80E-3** (2.6E-4)	5.88E-3 (5.3E-4)	6.48E-3 * (5.1E-4)	6.63E-3 * (4.8E-4)	1.37E-2 * (2.9E-3)
DTLZ6	3	5.98E-3 (5.0E-4)	**5.84E-3** (4.5E-4)	6.46E-3 * (7.0E-4)	6.61E-3 * (5.0E-4)	2.29E-2 * (3.5E-3)
DTLZ7	3	1.04E-1 (2.5E-2)	1.09E-1 (2.5E-1)	**8.54E-2** (7.1E-1)	7.99E-1 * (1.1E-3)	8.01E-1 (7.2E-1)
DTLZ1	4	2.06E1 * (1.9E1)	1.95E0 * (3.5E1)	1.60E2 * (4.9E1)	**5.93E-2** (2.6E-3)	1.02E1 * (1.4E1)
DTLZ2	4	6.12E-1 * (1.2E-1)	1.08E0 * (3.2E-1)	1.64E-1 * (8.1E-3)	1.64E-1 * (7.2E-3)	**1.40E-1** (2.0E-5)
DTLZ3	4	2.29E1 * (4.5E1)	8.88E-1 * (9.2E0)	2.99E2 * (1.4E2)	**1.76E-1** (1.4E-2)	3.25E1 * (7.3E1)
DTLZ4	4	4.10E-1 * (6.6E-2)	9.06E-1 * (3.2E-1)	1.63E-1 * (5.7E-3)	1.68E-1 * (9.4E-3)	**1.40E-1** (7.5E-5)
DTLZ5	4	2.71E-1 * (1.8E-1)	2.93E-1 * (1.0E-1)	1.52E-1 * (4.1E-2)	1.41E-1 * (4.1E-2)	**8.13E-2** (3.5E-2)
DTLZ6	4	**1.50E-1** (2.3E-1)	3.42E-1 * (4.1E-1)	1.02E1 * (2.2E0)	2.90E-1 (1.0E-1)	1.83E-1 (6.1E-2)
DTLZ7	4	3.26E-1 (2.6E-2)	3.29E-1 (3.9E-2)	**2.41E-1** (8.9E-1)	1.11E0 * (1.7E-3)	3.83E-1 (9.1E-1)
DTLZ1	5	1.95E1 (3.1E1)	**4.51E0** (5.1E1)	2.02E2 * (5.6E1)	1.42E2 * (1.5E2)	2.10E1 (2.6E1)
DTLZ2	5	8.86E-1 * (1.7E-1)	1.55E0 * (7.8E-1)	2.71E-1 * (1.3E-2)	2.87E-1 * (2.0E-2)	**2.12E-1** (1.4E-4)
DTLZ3	5	5.18E1 (4.9E1)	6.57E1 (2.8E2)	6.24E2 * (1.6E2)	1.12E2 (2.2E2)	**4.41E1** (8.4E1)
DTLZ4	5	7.12E-1 * (1.0E-1)	1.52E0 * (7.0E-1)	2.73E-1 * (1.3E-2)	4.46E-1 * (4.8E-1)	**2.12E-1** (1.7E-4)
DTLZ5	5	4.61E-1 * (2.7E-1)	4.85E-1 * (5.9E-1)	1.47E0 * (6.6E-1)	1.73E0 * (1.5E0)	**1.45E-1** (4.0E-2)
DTLZ6	5	3.29E0 * (5.3E0)	7.42E-1 * (7.1E-2)	1.96E1 * (3.8E0)	**3.42E-1** (1.9E-2)	4.52E-1 * (1.9E-1)
DTLZ7	5	5.25E-1 * (2.5E-2)	5.34E-1 * (5.1E-2)	4.01E-1 (1.9E-2)	4.05E-1 (1.8E-2)	**3.97E-1** (5.8E-2)
n = 1000						
DTLZ1	2	**2.32E-3** (5.1E-1)	1.76E-1 (1.1E-15)	5.15E3 * (1.7E3)	2.74E-3 (7.0E-4)	7.98E2 * (9.7E2)
DTLZ2	2	5.89E-3 * (8.1E-4)	5.46E-3 * (5.5E-4)	7.18E-1 * (6.3E-1)	5.26E-3 * (2.5E-4)	**4.56E-3** (1.0E-3)
DTLZ3	2	1.14E-2 (2.1E0)	4.32E-1 * (4.2E-1)	1.38E4 * (4.5E3)	**5.56E-3** (5.3E-4)	1.51E3 * (2.7E3)
DTLZ4	2	8.93E-3 * (4.3E-3)	5.43E-3 (1.1E-2)	6.91E-1 * (4.0E-1)	**5.16E-3** (2.8E-4)	8.13E-3 * (4.8E-3)
DTLZ5	2	5.87E-3 (5.8E-4)	5.43E-3 (6.4E-4)	7.07E-1 * (5.2E-1)	5.20E-3 (2.7E-4)	**4.77E-3** (2.3E-3)
DTLZ6	2	5.18E-3 * (3.7E-4)	5.22E-3 * (3.8E-4)	7.65E1 * (4.7E1)	5.75E-3 * (3.4E-4)	**3.96E-3** (6.2E-8)
DTLZ7	2	4.42E-1 * (2.5E-4)	4.42E-1 * (3.9E-4)	4.42E-1 * (1.1E-4)	4.42E-1 * (6.6E-5)	**4.42E-1** (4.3E-1)
DTLZ1	3	2.40E0 * (4.1E1)	2.66E-1 * (9.3E0)	5.40E3 * (1.8E3)	**3.03E-2** (2.4E-3)	1.02E3 * (1.1E3)
DTLZ2	3	1.62E-1 * (9.0E-2)	9.86E-1 * (1.0E0)	1.85E0 * (1.3E0)	7.43E-2 * (6.2E-3)	**5.89E-2** (4.1E-3)
DTLZ3	3	5.62E-1 * (1.0E1)	5.62E-1 * (2.4E-2)	1.74E4 * (4.4E3)	**7.57E-2** (8.3E-3)	3.45E3 * (2.9E3)
DTLZ4	3	2.78E-1 * (2.2E-1)	8.43E-1 * (3.8E-1)	1.82E0 * (1.5E0)	7.72E0 * (2.1E1)	**6.55E-2** (1.0E-2)
DTLZ5	3	5.14E-2 * (5.1E-2)	2.71E-1 * (3.7E-1)	2.25E0 * (2.1E0)	**6.99E-3** (4.2E-4)	2.50E-2 * (1.2E-2)
DTLZ6	3	5.89E-3 (2.5E-4)	**5.86E-3** (4.7E-4)	2.33E2 * (9.5E1)	6.49E-3 * (3.9E-4)	2.43E-2 * (3.3E-3)
DTLZ7	3	1.53E0 * (1.1E0)	1.53E0 * (—)	**7.99E-1** (7.1E-1)	1.53E0 * (7.3E-1)	8.03E-1 * (4.2E-1)
DTLZ1	4	5.24E2 * (8.9E2)	3.05E-1 * (1.5E1)	6.32E3 * (3.2E3)	**6.12E-2** (4.1E-3)	7.55E2 * (1.0E3)
DTLZ2	4	1.06E1 * (4.1E0)	2.68E1 * (1.7E1)	5.22E0 * (2.3E0)	3.05E1 * (2.4E1)	**2.26E-1** (2.1E-1)
DTLZ3	4	1.02E3 * (1.7E3)	7.59E-1 * (5.0E1)	2.08E4 * (6.4E3)	**1.74E-1** (1.2E-2)	4.10E3 * (3.9E3)
DTLZ4	4	3.78E0 * (1.6E0)	1.34E1 * (4.8E0)	8.91E0 * (5.0E0)	3.02E1 * (2.8E1)	**7.17E-1** (9.0E-1)
DTLZ5	4	2.25E1 * (2.0E1)	2.73E1 * (2.5E1)	4.84E1 * (1.7E1)	5.94E1 * (2.2E1)	**2.34E0** (4.5E0)
DTLZ6	4	4.30E0 * (2.2E1)	7.42E-1 (1.2E-1)	6.26E2 * (8.2E1)	3.42E-1 (3.7E-3)	**3.06E-1** (3.1E-1)
DTLZ7	4	2.24E0 * (1.9E0)	2.24E0 * (—)	**1.11E0** (4.9E-1)	2.24E0 * (—)	1.13E0 * (4.9E-1)
DTLZ1	5	9.53E2 * (6.4E2)	**1.29E1** (1.3E2)	8.15E3 * (3.2E3)	1.10E3 * (1.6E3)	1.96E3 * (1.3E3)
DTLZ2	5	1.93E1 * (5.7E0)	4.71E1 * (2.6E1)	3.44E1 * (7.2E0)	6.03E1 * (1.5E1)	**6.81E0** (7.0E0)
DTLZ3	5	1.66E3 * (2.5E3)	**8.04E-1** (2.4E1)	2.46E4 * (1.1E4)	1.80E4 * (1.1E4)	6.62E3 * (4.4E3)
DTLZ4	5	**8.66E0** (3.6E0)	2.98E1 * (2.5E1)	3.44E1 * (8.4E0)	9.75E1 * (2.2E1)	1.18E1 (5.6E0)
DTLZ5	5	3.12E1 * (1.7E1)	6.07E1 * (2.9E1)	6.70E1 * (1.8E1)	9.65E1 * (2.0E1)	**1.61E1** (2.0E1)
DTLZ6	5	9.17E1 * (1.4E2)	7.42E-1 * (—)	6.48E2 * (8.3E1)	**7.22E-1** (2.7E-1)	2.28E2 * (1.3E2)
DTLZ7	5	3.00E0 * (—)	3.00E0 * (—)	1.46E0 (1.0E0)	3.00E0 * (—)	**9.18E-1** (1.0E0)

Table B.36: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100, 000 function evaluations.

		WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	randomLMEA	randomMOEA/DVA	randomS3-CMA-ES
n = 40							
DTLZ1	2	**2.25E-3** (1.1E-4)	6.22E0 * (4.4E0)	3.33E0 * (1.1E1)	1.87E0 * (1.4E0)	1.33E2 * (3.4E1)	4.71E2 * (8.2E1)
DTLZ2	2	5.11E-3 * (2.9E-4)	5.05E-3 * (2.4E-4)	**3.96E-3** (3.4E-7)	4.45E-3 * (1.0E-4)	4.43E-2 * (6.4E-3)	9.07E-2 * (3.7E-3)
DTLZ3	2	**4.98E-3** (6.0E-4)	2.00E1 * (7.5E0)	1.03E1 * (1.9E1)	5.74E0 * (4.0E0)	3.21E2 * (7.1E1)	1.28E3 * (1.9E2)
DTLZ4	2	5.31E-3 * (7.3E-1)	5.19E-3 * (4.4E-4)	**3.96E-3** (2.8E-6)	4.45E-3 * (8.4E-5)	1.89E-1 * (1.2E-2)	3.90E-1 * (5.6E-1)
DTLZ5	2	5.07E-3 * (2.0E-4)	5.03E-3 * (2.0E-4)	**3.96E-3** (1.9E-7)	4.45E-3 * (4.9E-5)	4.43E-2 * (3.1E-3)	9.04E-2 * (8.1E-3)
DTLZ6	2	5.15E-3 * (3.3E-4)	5.62E-3 * (3.7E-4)	**3.96E-3** (5.3E-8)	4.43E-3 * (7.7E-5)	2.23E0 * (1.1E0)	1.77E1 * (1.6E0)
DTLZ7	2	4.42E-1 * (4.3E-1)	5.44E-3 * (4.3E-1)	4.42E-1 * (4.3E-1)	**4.57E-3** (5.2E-6)	5.36E-2 * (1.4E-2)	2.63E-1 * (2.2E-1)
DTLZ1	3	**7.27E-2** (3.8E-1)	2.77E1 * (1.2E1)	1.11E1 * (2.4E1)	1.58E0 * (1.1E0)	1.30E2 * (3.4E1)	4.10E2 * (6.1E1)
DTLZ2	3	8.41E-2 * (6.4E-3)	7.32E-2 * (3.4E-3)	**5.44E-2** (6.1E-6)	5.60E-2 * (7.4E-4)	6.20E-2 * (1.6E-3)	1.05E-1 * (1.2E-2)
DTLZ3	3	**1.12E-1** (2.7E-1)	6.04E1 * (4.2E1)	2.41E1 * (3.3E1)	8.45E0 * (4.2E0)	3.56E2 * (7.5E1)	1.23E3 * (1.7E2)
DTLZ4	3	2.46E-1 * (1.8E-1)	7.10E-2 * (3.2E-3)	**5.44E-2** (8.3E-6)	2.55E-1 * (4.8E-1)	4.75E-1 * (1.7E-3)	4.63E-1 * (2.1E-1)
DTLZ5	3	5.80E-3 * (2.6E-4)	6.48E-3 * (5.1E-4)	1.37E-2 * (2.9E-3)	**5.27E-3** (1.5E-4)	1.83E-2 * (1.8E-3)	7.27E-2 * (1.3E-2)
DTLZ6	3	5.98E-3 * (5.0E-4)	6.46E-3 * (7.0E-4)	2.29E-2 * (3.5E-3)	**5.10E-3** (2.0E-4)	2.18E0 * (1.7E0)	1.69E1 * (1.7E0)
DTLZ7	3	1.04E-1 * (2.5E-2)	8.54E-2 * (7.1E-1)	8.01E-1 * (7.2E-1)	**6.24E-2** (1.7E-3)	1.60E-1 * (1.8E-2)	3.18E-1 * (4.4E-2)
DTLZ1	4	2.06E1 * (1.9E1)	1.60E2 * (4.9E1)	1.02E1 * (1.4E1)	**1.32E0** (7.7E-1)	1.13E2 * (2.3E1)	3.07E2 * (7.0E1)
DTLZ2	4	6.12E-1 * (1.2E-1)	1.64E-1 * (8.1E-3)	1.40E-1 * (2.0E-5)	**1.38E-1** (1.7E-3)	1.56E-1 * (2.5E-4)	1.56E-1 * (6.9E-3)
DTLZ3	4	2.29E1 (4.5E1)	2.99E2 * (1.4E2)	3.25E1 * (7.3E1)	**9.42E0** (5.8E0)	3.51E2 * (8.9E1)	1.02E3 * (2.1E2)
DTLZ4	4	4.10E-1 * (6.6E-2)	1.63E-1 * (5.7E-3)	**1.40E-1** (7.5E-5)	3.79E-1 (4.6E-1)	6.38E-1 * (9.0E-4)	5.87E-1 * (1.5E-1)
DTLZ5	4	2.71E-1 * (1.8E-1)	1.52E-1 * (4.1E-2)	8.13E-2 * (3.5E-2)	**6.16E-3** (2.4E-4)	7.74E-2 * (7.8E-2)	5.79E-2 * (9.8E-3)
DTLZ6	4	1.50E-1 * (2.3E-1)	1.02E1 * (2.2E0)	1.83E-1 * (6.1E-2)	**5.95E-3** (2.8E-4)	1.35E0 * (1.2E0)	1.67E1 * (9.2E-1)
DTLZ7	4	3.26E-1 * (2.6E-2)	2.41E-1 * (8.9E-1)	3.83E-1 * (9.1E-1)	**1.93E-1** (7.4E-3)	3.14E-1 * (2.4E-2)	4.14E-1 * (2.3E-1)
DTLZ1	5	1.95E1 * (3.1E1)	2.02E2 * (5.6E1)	2.10E1 * (2.6E1)	**1.18E0** (8.4E-1)	1.05E2 * (2.2E1)	2.76E2 * (5.9E1)
DTLZ2	5	8.86E-1 * (1.7E-1)	2.71E-1 * (1.3E-2)	2.12E-1 (1.4E-4)	**2.12E-1** (2.0E-3)	2.69E-1 * (2.2E-4)	2.32E-1 * (5.2E-3)
DTLZ3	5	5.18E1 * (4.9E1)	6.24E2 * (1.6E2)	4.41E1 * (8.4E1)	**1.08E1** (7.2E0)	3.42E2 * (7.1E1)	1.04E3 * (2.0E2)
DTLZ4	5	7.12E-1 * (1.0E-1)	2.73E-1 * (1.3E-2)	**2.12E-1** (1.7E-4)	3.50E-1 * (6.4E-1)	7.26E-1 * (5.0E-4)	7.24E-1 * (1.3E-1)
DTLZ5	5	4.61E-1 * (2.7E-1)	1.47E0 * (6.6E-1)	1.45E-1 * (4.0E-2)	**6.60E-3** (3.5E-4)	2.11E-1 * (1.3E-1)	5.73E-2 * (1.1E-2)
DTLZ6	5	3.29E0 * (5.3E0)	1.96E1 * (3.8E0)	4.52E-1 * (1.9E-1)	**6.38E-3** (4.8E-4)	1.59E0 * (1.1E0)	1.63E1 * (1.1E0)
DTLZ7	5	5.25E-1 * (2.5E-2)	4.01E-1 * (1.9E-2)	3.97E-1 * (5.8E-2)	**3.71E-1** (1.2E-2)	4.63E-1 * (3.7E-2)	5.14E-1 * (2.6E-2)
n = 1000							
DTLZ1	2	**2.32E-3** (5.1E-1)	5.15E3 * (1.7E3)	7.98E2 * (9.7E2)	7.33E3 * (2.0E2)	3.64E3 * (1.5E3)	2.63E4 * (5.2E2)
DTLZ2	2	5.89E-3 * (8.1E-4)	7.18E-1 * (6.3E-1)	**4.56E-3** (1.0E-3)	1.72E1 * (2.0E0)	5.16E1 * (8.1E0)	1.14E2 * (2.0E0)
DTLZ3	2	**1.14E-2** (2.1E0)	1.38E4 * (4.5E3)	1.51E3 * (2.7E3)	2.13E4 * (1.4E3)	9.91E3 * (4.4E3)	7.28E4 * (1.1E3)
DTLZ4	2	8.93E-3 (4.3E-3)	6.91E-1 * (4.0E-1)	**8.13E-3** (4.8E-3)	1.69E1 * (3.0E0)	5.35E1 * (6.2E0)	1.16E2 * (2.9E0)
DTLZ5	2	5.87E-3 (5.8E-4)	7.07E-1 * (5.2E-1)	**4.77E-3** (2.3E-3)	1.69E1 * (1.4E0)	5.32E1 * (8.0E0)	1.14E2 * (1.8E0)
DTLZ6	2	5.18E-3 * (3.7E-4)	7.65E1 * (4.7E1)	**3.96E-3** (6.2E-8)	6.97E2 * (1.4E1)	4.57E2 * (6.8E0)	8.83E2 * (5.2E0)
DTLZ7	2	4.42E-1 * (2.5E-4)	4.42E-1 * (1.1E-4)	**4.42E-1** (4.3E-1)	3.16E0 * (2.8E-1)	5.98E0 * (1.8E-1)	8.93E0 * (3.0E-1)
DTLZ1	3	**2.40E0** (4.1E1)	5.40E3 * (1.8E3)	1.02E3 * (1.1E3)	5.75E3 * (2.9E2)	3.94E3 * (1.2E3)	2.18E4 * (4.3E2)
DTLZ2	3	1.62E-1 * (9.0E-2)	1.85E0 * (1.3E0)	**5.89E-2** (4.1E-3)	1.66E1 * (1.5E0)	5.03E1 * (7.9E0)	1.15E2 * (2.4E0)
DTLZ3	3	**5.62E-1** (1.0E1)	1.74E4 * (4.4E3)	3.45E3 * (2.9E3)	2.25E4 * (1.1E3)	1.07E4 * (2.7E3)	7.28E4 * (1.6E3)
DTLZ4	3	2.78E-1 * (2.2E-1)	1.82E0 * (1.5E0)	**6.55E-2** (1.0E-2)	1.73E1 * (2.4E0)	5.05E1 * (8.8E0)	1.09E2 * (4.2E0)
DTLZ5	3	5.14E-2 * (5.1E-2)	2.25E0 * (2.1E0)	**2.50E-2** (1.2E-2)	1.68E1 * (1.5E0)	5.07E1 * (8.1E0)	1.15E2 * (2.4E0)
DTLZ6	3	**5.89E-3** (2.5E-4)	2.33E2 * (9.5E1)	2.43E-2 * (3.3E-3)	6.97E2 * (1.7E1)	4.56E2 * (6.4E0)	8.83E2 * (5.0E0)
DTLZ7	3	1.53E0 * (1.1E0)	**7.99E-1** (7.1E-1)	8.03E-1 * (4.2E-1)	4.27E0 * (4.1E-1)	8.61E0 * (4.2E-1)	1.28E1 * (1.1E0)
DTLZ1	4	**5.24E2** (8.9E2)	6.32E3 * (3.2E3)	7.55E2 (1.0E3)	4.75E3 * (1.6E2)	4.08E3 * (1.1E3)	1.92E4 * (4.4E2)
DTLZ2	4	1.06E1 * (4.1E0)	5.22E0 * (2.3E0)	**2.26E-1** (2.1E-1)	1.65E1 * (1.6E0)	5.02E1 * (7.5E0)	1.12E2 * (3.2E0)
DTLZ3	4	**1.02E3** (1.7E3)	2.08E4 * (6.4E3)	4.10E3 * (3.9E3)	2.18E4 * (1.6E3)	1.14E4 * (3.8E3)	7.25E4 * (1.1E3)
DTLZ4	4	3.78E0 * (1.6E0)	8.91E0 * (5.0E0)	**7.17E-1** (9.0E-1)	1.75E1 * (3.5E0)	4.90E1 * (6.4E0)	1.04E2 * (2.5E1)
DTLZ5	4	2.25E1 * (2.0E1)	4.84E1 * (1.7E1)	**2.34E0** (4.5E0)	1.66E1 * (1.9E0)	4.85E1 * (8.9E0)	1.12E2 * (2.8E0)
DTLZ6	4	4.30E0 * (2.2E1)	6.26E2 * (8.2E1)	**3.06E-1** (3.1E-1)	6.94E2 * (1.4E1)	4.56E2 * (1.5E1)	8.86E2 * (3.9E0)
DTLZ7	4	2.24E0 * (1.9E0)	**1.11E0** (4.9E-1)	1.13E0 * (4.9E-1)	5.30E0 * (6.3E-1)	1.13E1 * (5.5E-1)	1.67E1 * (3.4E-1)
DTLZ1	5	**9.53E2** (6.4E2)	8.15E3 * (3.2E3)	1.96E3 * (1.3E3)	4.34E3 * (1.5E2)	3.79E3 * (1.8E3)	1.72E4 * (5.3E2)
DTLZ2	5	1.93E1 * (5.7E0)	3.44E1 * (7.2E0)	6.81E0 (7.0E0)	1.58E1 * (1.6E0)	**6.48E0** (1.8E0)	1.13E2 * (3.0E0)
DTLZ3	5	**1.66E3** (2.5E3)	2.46E4 * (1.1E4)	6.62E3 * (4.4E3)	2.20E4 * (1.1E3)	1.24E4 * (1.0E4)	7.24E4 * (1.0E3)
DTLZ4	5	8.66E0 * (3.6E0)	3.44E1 * (8.4E0)	1.18E1 * (5.6E0)	1.67E1 * (3.2E0)	**3.55E0** (2.2E0)	4.85E1 * (2.5E1)
DTLZ5	5	3.12E1 * (1.7E1)	6.70E1 * (1.8E1)	1.61E1 * (2.0E1)	1.64E1 * (1.5E0)	**6.76E0** (2.3E0)	1.13E2 * (1.9E0)
DTLZ6	5	**9.17E1** (1.4E2)	6.48E2 * (8.3E1)	2.28E2 * (1.3E2)	6.92E2 * (1.8E1)	4.58E2 * (1.2E1)	8.86E2 * (6.6E0)
DTLZ7	5	3.00E0 * (—)	1.46E0 (1.0E0)	**9.18E-1** (1.0E0)	6.96E0 * (1.1E0)	1.57E1 * (5.9E-1)	2.11E1 * (7.3E-1)

Table B.37: Performance comparison using the IGD indicator on the LSMOP benchmarks using 10, 000, 000 function evaluations.

		WOF-SMPSO	WOF-Randomised	LMEA	MOEA/DVA
n = 1000					
LSMOP1	2	2.36E-2 * (1.3E-3)	1.08E-1 * (3.4E-2)	2.39E-1 * (3.0E-2)	**3.63E-3** (1.2E-5)
LSMOP2	2	**6.96E-3** (8.0E-4)	8.51E-3 * (5.3E-3)	3.09E-2 * (3.5E-4)	3.31E-2 * (1.5E-2)
LSMOP3	2	**3.61E-1** (6.3E-2)	8.97E-1 * (2.4E-1)	5.58E0 * (1.0E0)	9.31E-1 * (3.5E-2)
LSMOP4	2	1.88E-2 * (5.6E-4)	1.97E-2 * (3.2E-3)	3.85E-2 * (1.4E-3)	**1.11E-2** (7.6E-3)
LSMOP5	2	2.23E-2 * (3.5E-3)	2.16E-1 * (1.0E-1)	4.72E-1 * (3.1E-1)	**4.71E-3** (4.9E-4)
LSMOP6	2	**2.82E-2** (2.2E-3)	1.81E-1 * (6.2E-3)	7.67E-1 * (8.4E-4)	4.88E-1 * (4.6E-2)
LSMOP7	2	**8.24E-1** (1.1E-1)	1.11E0 * (5.4E-1)	1.71E0 * (2.3E-1)	6.89E0 * (1.7E0)
LSMOP8	2	1.70E-2 * (1.9E-3)	3.57E-2 * (1.2E-2)	1.22E-1 * (2.2E-1)	**1.53E-2** (1.1E-2)
LSMOP9	2	1.26E-2 (3.1E-3)	4.58E-1 * (2.0E-3)	6.19E-1 * (4.0E-2)	**7.64E-3** (1.9E-2)
LSMOP1	3	1.82E-1 * (1.7E-2)	1.21E-1 * (4.1E-2)	1.45E-1 * (1.9E-2)	**4.14E-2** (9.7E-5)
LSMOP2	3	5.92E-2 * (3.2E-3)	**4.41E-2** (4.0E-4)	5.13E-2 * (5.2E-4)	4.74E-2 * (1.4E-2)
LSMOP3	3	8.60E-1 * (—)	**4.60E-1** (2.0E-1)	1.86E0 * (3.7E-1)	6.62E-1 * (6.0E-2)
LSMOP4	3	8.44E-2 * (7.4E-3)	6.37E-2 * (3.7E-3)	7.31E-2 * (1.9E-3)	**5.73E-2** (1.7E-2)
LSMOP5	3	2.47E-1 * (5.1E-2)	2.17E-1 * (1.8E-1)	6.11E0 * (6.1E0)	**5.50E-2** (8.7E-5)
LSMOP6	3	**9.08E-1** (7.1E-3)	1.28E0 * (3.4E-2)	1.64E0 * (1.6E-1)	5.11E0 * (4.5E-1)
LSMOP7	3	**3.94E-1** (1.4E-2)	8.24E-1 * (1.7E-1)	5.81E-1 * (2.1E-2)	6.15E-1 * (5.3E-2)
LSMOP8	3	8.08E-2 * (5.2E-3)	1.03E-1 * (4.5E-2)	7.79E-2 * (7.3E-3)	**5.75E-2** (1.8E-2)
LSMOP9	3	4.10E-1 * (9.6E-3)	1.40E0 * (3.2E-1)	6.06E-1 * (5.1E-2)	**1.48E-1** (2.2E-2)

Table B.38: Performance comparison using the IGD indicator on the UF benchmarks using 10,000,000 function evaluations.

		WOF-SMPSO	WOF-Randomised	LMEA	MOEA/DVA
n = 1000					
UF1	2	2.19E-1 * (7.6E-3)	9.51E-2 * (1.1E-2)	6.49E-2 * (1.4E-2)	**4.18E-3** (3.1E-5)
UF2	2	7.89E-2 * (1.0E-3)	3.25E-2 * (8.5E-3)	2.53E-2 * (8.3E-4)	**7.32E-3** (1.3E-3)
UF3	2	2.46E-2 * (7.4E-4)	2.08E-2 * (1.6E-3)	9.44E-2 * (1.6E-2)	**4.13E-3** (6.4E-5)
UF4	2	**4.26E-2** (1.5E-4)	5.11E-2 * (4.3E-3)	4.46E-2 * (7.9E-4)	6.39E-2 * (5.4E-3)
UF5	2	1.39E0 * (6.6E-1)	1.90E-1 * (4.7E-2)	6.04E-1 * (2.0E-1)	**9.77E-2** (2.6E-3)
UF6	2	3.89E-1 * (2.5E-1)	1.16E-1 * (5.9E-2)	1.35E-1 * (1.1E-2)	**3.05E-2** (9.3E-3)
UF7	2	1.72E-1 * (8.5E-3)	5.14E-2 * (4.7E-3)	1.20E-1 * (1.3E-1)	**4.37E-3** (1.0E-4)
UF8	3	2.67E-1 * (1.0E-2)	2.55E-1 * (2.9E-1)	2.68E-1 * (3.2E-2)	**2.01E-1** (2.6E-1)
UF9	3	5.11E-1 * (6.4E-2)	5.85E-1 * (1.2E-1)	2.73E-1 * (3.7E-2)	**4.68E-2** (6.4E-6)
UF10	3	1.93E0 * (1.6E0)	5.45E-1 * (1.4E-1)	7.23E-1 * (1.8E-1)	**3.31E-1** (2.6E-3)

Table B.39: Performance comparison using the IGD indicator on the DTLZ benchmarks using 10,000,000 function evaluations.

		WOF-SMPSO	WOF-Randomised	LMEA	MOEA/DVA
n = 1000					
DTLZ1	2	2.23E-3 * (1.4E-4)	**1.78E-3** (5.0E-6)	6.01E2 * (2.8E1)	4.63E2 * (1.1E1)
DTLZ2	2	5.12E-3 * (2.8E-4)	**3.96E-3** (3.8E-9)	2.42E-2 * (5.5E-3)	2.19E-2 * (4.3E-4)
DTLZ3	2	4.86E-3 * (2.2E-4)	**3.96E-3** (1.3E-5)	1.63E3 * (4.9E1)	1.27E3 * (3.0E1)
DTLZ4	2	5.11E-3 * (2.6E-4)	**3.96E-3** (1.3E-7)	2.86E-2 * (1.5E-2)	2.74E-2 * (1.3E-2)
DTLZ5	2	5.07E-3 * (2.5E-4)	**3.96E-3** (2.6E-9)	2.49E-2 * (8.8E-3)	2.19E-2 * (4.0E-4)
DTLZ6	2	5.12E-3 * (3.3E-4)	**3.96E-3** (1.0E-8)	2.44E2 * (6.1E0)	1.60E2 * (3.1E0)
DTLZ7	2	5.36E-3 (3.0E-4)	5.25E-3 (4.3E-1)	9.36E-3 (1.2E-3)	2.25E-2 (5.2E-4)
DTLZ1	3	8.42E0 * (3.0E1)	**2.09E-2** (2.9E-2)	3.83E2 * (1.9E1)	3.42E2 * (1.3E1)
DTLZ2	3	7.15E-2 * (5.0E-3)	**5.44E-2** (6.0E-8)	6.23E-2 * (3.4E-3)	6.05E-2 * (1.4E-4)
DTLZ3	3	2.82E-1 * (2.1E0)	**5.44E-2** (2.2E-4)	1.27E3 * (6.0E1)	1.12E3 * (2.5E1)
DTLZ4	3	7.18E-2 * (5.3E-3)	**5.44E-2** (4.3E-6)	1.07E-1 * (1.1E-1)	4.74E-1 * (4.0E-4)
DTLZ5	3	**5.95E-3** (4.5E-4)	1.36E-2 * (2.8E-3)	2.59E-2 * (1.2E-2)	1.78E-2 * (3.3E-4)
DTLZ6	3	**5.85E-3** (5.2E-4)	2.24E-2 * (5.1E-3)	2.18E2 * (1.0E1)	1.42E2 * (3.8E0)
DTLZ7	3	9.74E-2 * (1.3E-2)	7.86E-2 * (6.7E-3)	**6.18E-2** (2.2E-3)	1.11E-1 * (2.7E-4)

Table B.40: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100,000 function evaluations.

		WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	DLS-MOEA
n = 1000					
LSMOP1	2	**7.17E-2** (6.2E-3)	6.46E-1 * (9.1E-2)	1.74E-1 * (1.1E-1)	1.31E0 * (9.9E-2)
LSMOP2	2	**7.29E-3** (3.9E-4)	1.90E-2 * (5.2E-4)	1.03E-2 * (3.0E-3)	3.16E-2 * (5.7E-4)
LSMOP3	2	**1.56E0** (1.9E-3)	1.57E0 * (2.5E-3)	1.57E0 * (3.7E-3)	1.88E1 * (1.7E0)
LSMOP4	2	**2.04E-2** (5.9E-4)	4.13E-2 * (4.3E-3)	2.37E-2 * (2.5E-3)	4.51E-2 * (6.6E-4)
LSMOP5	2	7.42E-1 (2.9E-1)	7.42E-1 (1.6E-1)	**7.41E-1** (4.7E-2)	5.25E0 * (4.9E-1)
LSMOP6	2	**1.73E-1** (3.4E-3)	6.71E-1 * (1.3E-3)	3.49E-1 * (1.8E-1)	7.62E-1 * (9.5E-2)
LSMOP7	2	1.51E0 (6.0E-4)	1.51E0 * (2.4E-3)	**1.51E0** (2.8E-3)	4.79E2 * (3.0E2)
LSMOP8	2	**2.12E-1** (5.9E-1)	7.42E-1 * (4.0E-2)	2.14E-1 (1.8E-1)	2.33E0 * (2.5E-1)
LSMOP9	2	**4.67E-1** (9.5E-3)	8.08E-1 * (1.6E-3)	4.90E-1 * (7.5E-1)	1.06E0 * (1.7E-2)
LSMOP1	3	**2.00E-1** (4.1E-2)	5.93E-1 * (3.9E-2)	3.24E-1 * (1.4E-1)	1.30E0 * (1.3E-1)
LSMOP2	3	6.12E-2 * (5.2E-3)	7.58E-2 * (8.1E-3)	5.16E-2 * (5.7E-4)	**5.08E-2** (2.5E-4)
LSMOP3	3	**8.60E-1** (—)	8.60E-1 * (4.4E-4)	8.60E-1 (2.6E-8)	6.12E0 * (7.9E-1)
LSMOP4	3	8.91E-2 (4.9E-3)	1.43E-1 * (5.2E-3)	9.31E-2 * (7.0E-3)	**8.83E-2** (9.9E-4)
LSMOP5	3	**4.29E-1** (2.3E-1)	5.41E-1 * (5.1E-4)	4.51E-1 (7.3E-2)	5.50E0 * (5.2E-1)
LSMOP6	3	**9.11E-1** (5.8E-1)	1.31E0 * (1.8E-3)	1.31E0 * (9.8E-2)	1.09E2 * (4.7E1)
LSMOP7	3	8.49E-1 (1.0E-1)	8.56E-1 * (4.3E-3)	**8.47E-1** (3.3E-3)	1.09E0 * (1.0E-2)
LSMOP8	3	**9.22E-2** (1.5E-2)	3.43E-1 * (5.7E-2)	1.86E-1 * (7.8E-2)	5.99E-1 * (7.3E-2)
LSMOP9	3	**1.11E0** (5.7E-1)	1.14E0 * (3.4E-4)	1.14E0 * (1.7E-2)	4.01E0 * (6.1E-1)

Table B.41: Performance comparison using the IGD indicator on the UF benchmarks using 100,000 function evaluations.

		WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	DLS-MOEA
n = 40					
UF1	2	9.14E-2 (2.2E-2)	1.01E-1 (2.6E-2)	9.04E-2 (2.2E-2)	8.68E-2 (3.9E-2)
UF2	2	5.41E-2 * (5.7E-3)	4.11E-2 (7.4E-3)	4.21E-2 (8.2E-3)	**2.83E-2** (3.0E-2)
UF3	2	**1.33E-1** (3.6E-2)	1.69E-1 * (2.0E-2)	1.68E-1 * (4.0E-2)	2.93E-1 * (5.0E-2)
UF4	2	5.19E-2 * (8.4E-3)	4.72E-2 (1.1E-3)	4.88E-2 * (5.6E-3)	**4.57E-2** (2.8E-3)
UF5	2	6.24E-1 * (1.6E-1)	3.86E-1 (2.0E-1)	3.48E-1 (2.0E-1)	**3.12E-1** (1.3E-1)
UF6	2	3.22E-1 * (1.7E-1)	**1.14E-1** (1.8E-1)	1.27E-1 (2.4E-1)	2.69E-1 (2.2E-1)
UF7	2	**4.68E-2** (9.4E-3)	5.56E-2 * (1.4E-2)	5.08E-2 (1.6E-2)	4.95E-2 (2.6E-1)
UF8	3	2.52E-1 (5.0E-2)	**2.24E-1** (1.2E-1)	5.28E-1 * (5.4E-2)	5.27E-1 * (1.8E-1)
UF9	3	4.81E-1 * (8.9E-2)	3.06E-1 * (1.4E-1)	3.13E-1 * (2.3E-1)	**2.44E-1** (1.7E-1)
UF10	3	1.28E0 * (2.9E-1)	4.30E-1 (1.4E-1)	5.20E-1 * (3.9E-2)	**4.18E-1** (1.2E-1)
n = 1000					
UF1	2	2.72E-1 * (5.9E-3)	**1.57E-1** (3.7E-2)	1.96E-1 * (1.6E-2)	1.77E-1 * (2.2E-2)
UF2	2	**8.50E-2** (4.1E-4)	1.13E-1 * (1.3E-2)	8.69E-2 * (3.2E-3)	1.64E-1 * (1.2E-2)
UF3	2	**2.54E-2** (3.8E-3)	1.38E-1 * (1.0E-2)	3.50E-2 * (1.0E-2)	2.67E-1 * (6.2E-3)
UF4	2	**5.49E-2** (9.0E-3)	7.40E-2 * (5.0E-3)	7.08E-2 * (1.8E-2)	1.54E-1 * (2.4E-3)
UF5	2	2.81E0 * (1.1E-1)	**1.20E0** (1.9E-1)	1.61E0 * (5.3E-1)	2.22E0 * (3.5E-1)
UF6	2	1.02E0 * (5.9E-2)	4.74E-1 (2.6E-1)	**4.64E-1** (3.1E-1)	6.59E-1 * (1.6E-1)
UF7	2	2.77E-1 * (7.4E-3)	**1.50E-1** (2.6E-2)	1.67E-1 * (2.0E-2)	1.55E-1 (1.1E-1)
UF8	3	**3.56E-1** (4.4E-3)	4.67E-1 * (2.9E-2)	5.42E-1 * (2.1E-3)	4.14E-1 * (1.8E-2)
UF9	3	**5.70E-1** (7.2E-3)	6.52E-1 * (8.4E-2)	6.20E-1 * (9.6E-2)	6.60E-1 * (2.8E-2)
UF10	3	2.08E0 * (7.6E-1)	1.90E0 * (4.1E-1)	**8.97E-1** (2.9E-1)	3.81E0 * (3.9E-1)

Table B.42: Performance comparison using the IGD indicator on the WFG benchmarks using 100,000 function evaluations.

		WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	DLS-MOEA
n = 40					
WFG1	2	1.12E0 * (7.3E-2)	**1.71E-1** (6.1E-2)	3.49E-1 * (9.5E-2)	1.11E0 * (3.0E-2)
WFG2	2	2.80E-2 * (7.5E-3)	2.87E-2 (1.1E-2)	**2.44E-2** (6.1E-3)	6.60E-1 * (3.7E-3)
WFG3	2	3.00E-2 * (6.4E-3)	2.98E-2 * (8.7E-3)	2.16E-2 * (3.7E-3)	**1.71E-2** (5.0E-3)
WFG4	2	7.68E-2 * (3.4E-2)	2.24E-2 * (3.4E-3)	**1.85E-2** (2.4E-3)	1.96E-2 (4.8E-3)
WFG5	2	**6.79E-2** (4.6E-3)	7.22E-2 * (7.1E-4)	6.94E-2 (1.2E-4)	7.26E-2 * (1.8E-3)
WFG6	2	**2.28E-2** (9.8E-3)	8.32E-2 * (1.2E-3)	4.13E-2 * (4.9E-2)	4.91E-2 * (1.6E-2)
WFG7	2	1.88E-2 * (1.6E-3)	1.83E-2 * (1.2E-3)	**1.38E-2** (5.7E-4)	2.05E-2 * (1.2E-2)
WFG8	2	**1.07E-1** (8.7E-2)	2.15E-1 * (1.6E-2)	2.09E-1 * (5.9E-2)	2.05E-1 * (1.1E-2)
WFG9	2	**4.27E-2** (1.3E-2)	9.15E-2 * (4.6E-2)	8.28E-2 (5.4E-2)	9.31E-2 * (5.1E-2)
WFG1	3	1.51E0 * (2.1E-2)	9.77E-1 * (7.2E-2)	1.18E0 * (1.0E-1)	**3.73E-1** (7.0E-2)
WFG2	3	2.36E-1 * (1.7E-2)	2.24E-1 * (9.0E-3)	**1.66E-1** (4.8E-3)	5.01E-1 * (6.1E-3)
WFG3	3	1.43E-1 * (2.6E-2)	1.35E-1 * (2.7E-2)	1.20E-1 * (3.0E-2)	**3.02E-2** (4.8E-3)
WFG4	3	3.46E-1 * (3.2E-2)	2.91E-1 * (1.2E-2)	**2.35E-1** (4.5E-3)	3.18E-1 * (9.7E-3)
WFG5	3	3.04E-1 * (2.5E-2)	2.92E-1 * (1.4E-2)	**2.36E-1** (2.5E-3)	3.25E-1 * (8.4E-3)
WFG6	3	3.48E-1 * (6.2E-2)	3.13E-1 * (1.9E-2)	**2.41E-1** (4.1E-3)	3.21E-1 * (1.6E-2)
WFG7	3	2.99E-1 * (1.9E-2)	3.13E-1 * (1.1E-1)	**2.28E-1** (4.0E-2)	3.14E-1 * (7.6E-3)
WFG8	3	5.60E-1 * (3.4E-2)	5.02E-1 * (3.6E-2)	**3.53E-1** (6.4E-2)	3.70E-1 (8.4E-3)
WFG9	3	3.38E-1 * (3.1E-2)	3.18E-1 * (1.0E-2)	**2.46E-1** (6.3E-3)	3.14E-1 * (8.2E-3)
n = 1000					
WFG1	2	**1.20E0** (5.1E-2)	1.27E0 * (1.2E-2)	1.22E0 * (2.7E-2)	2.13E0 * (1.0E-2)
WFG2	2	**7.25E-2** (2.1E-2)	2.35E-1 * (7.5E-2)	1.40E-1 * (2.9E-2)	1.42E0 * (9.0E-3)
WFG3	2	**8.94E-2** (1.3E-2)	2.12E-1 * (2.9E-2)	1.25E-1 * (3.1E-2)	8.86E-1 * (3.9E-2)
WFG4	2	**1.11E-1** (8.2E-3)	1.77E-1 * (2.3E-2)	1.27E-1 * (1.8E-2)	1.09E0 * (6.7E-2)
WFG5	2	**6.78E-2** (4.8E-3)	9.28E-2 * (6.6E-2)	6.97E-2 (3.3E-3)	9.85E-1 * (5.6E-2)
WFG6	2	1.69E-2 * (1.4E-3)	2.34E-2 * (5.4E-3)	**1.35E-2** (3.6E-3)	1.20E0 * (1.3E-1)
WFG7	2	**7.70E-2** (1.1E-2)	2.41E-1 * (2.7E-2)	1.39E-1 * (2.9E-2)	8.74E-1 * (4.1E-2)
WFG8	2	**1.03E-1** (1.0E-1)	3.59E-1 * (4.3E-2)	2.81E-1 * (7.8E-2)	1.11E0 * (3.9E-2)
WFG9	2	**3.83E-2** (7.2E-3)	1.04E-1 * (5.2E-2)	5.05E-2 * (2.9E-2)	1.02E0 * (9.0E-2)
WFG1	3	1.51E0 * (1.9E-2)	1.55E0 * (3.6E-2)	**1.50E0** (1.8E-2)	2.37E0 * (1.4E-2)
WFG2	3	**2.43E-1** (1.6E-2)	5.15E-1 * (1.1E-1)	4.04E-1 * (1.0E-1)	1.69E0 * (1.0E0)
WFG3	3	**1.13E-1** (3.0E-2)	3.97E-1 * (1.5E-1)	3.98E-1 * (1.4E-1)	7.74E-1 * (4.9E-2)
WFG4	3	**3.70E-1** (2.4E-2)	6.48E-1 * (1.2E-1)	4.22E-1 * (9.3E-2)	1.19E0 * (8.4E-2)
WFG5	3	4.28E-1 (1.4E-1)	6.22E-1 * (1.0E-1)	**4.01E-1** (1.1E-1)	9.84E-1 * (2.8E-2)
WFG6	3	3.78E-1 * (5.9E-2)	4.03E-1 * (8.8E-2)	**2.46E-1** (1.8E-2)	9.04E-1 * (2.2E-2)
WFG7	3	**3.58E-1** (4.1E-2)	7.19E-1 * (3.7E-2)	5.29E-1 * (5.5E-2)	8.93E-1 * (5.4E-2)
WFG8	3	**6.19E-1** (7.5E-2)	1.07E0 * (1.2E-1)	7.72E-1 * (1.3E-1)	1.14E0 * (4.4E-2)
WFG9	3	**3.68E-1** (3.6E-2)	7.51E-1 * (1.2E-1)	4.31E-1 * (1.1E-1)	1.06E0 * (8.3E-2)

Table B.43: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100,000 function evaluations.

		WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	DLS-MOEA
n = 40					
DTLZ1	2	**2.25E-3** (1.1E-4)	6.22E0 * (4.4E0)	3.33E0 * (1.1E1)	5.88E0 * (4.2E0)
DTLZ2	2	5.11E-3 * (2.9E-4)	5.05E-3 * (2.4E-4)	**3.96E-3** (3.4E-7)	5.63E-3 * (2.0E-4)
DTLZ3	2	**4.98E-3** (6.0E-4)	2.00E1 * (7.5E0)	1.03E1 * (1.9E1)	1.54E1 * (9.4E0)
DTLZ4	2	5.31E-3 * (7.3E-1)	5.19E-3 * (4.4E-4)	**3.96E-3** (2.8E-6)	5.82E-3 * (7.3E-1)
DTLZ5	2	5.07E-3 * (2.0E-4)	5.03E-3 * (2.0E-4)	**3.96E-3** (1.9E-7)	5.58E-3 * (2.3E-4)
DTLZ6	2	5.15E-3 * (3.3E-4)	5.62E-3 * (3.7E-4)	**3.96E-3** (5.3E-8)	2.22E-1 * (8.0E-2)
DTLZ7	2	4.42E-1 * (4.3E-1)	5.44E-3 * (4.3E-1)	4.42E-1 * (4.3E-1)	**4.19E-3** (1.4E-5)
DTLZ1	3	**7.27E-2** (3.8E-1)	2.77E1 * (1.2E1)	1.11E1 * (2.4E1)	4.61E0 * (2.9E0)
DTLZ2	3	8.41E-2 * (6.4E-3)	7.32E-2 * (3.4E-3)	**5.44E-2** (6.1E-6)	7.80E-2 * (8.2E-4)
DTLZ3	3	**1.12E-1** (2.7E-1)	6.04E1 * (4.2E1)	2.41E1 * (3.3E1)	2.18E1 * (1.1E1)
DTLZ4	3	2.46E-1 * (1.8E-1)	7.10E-2 * (3.2E-3)	**5.44E-2** (8.3E-6)	5.41E-1 * (4.6E-1)
DTLZ5	3	**5.80E-3** (2.6E-4)	6.48E-3 * (5.1E-4)	1.37E-2 * (2.9E-3)	6.09E-3 * (2.2E-4)
DTLZ6	3	**5.98E-3** (5.0E-4)	6.46E-3 * (7.0E-4)	2.29E-2 * (3.5E-3)	1.40E-1 * (6.6E-2)
DTLZ7	3	1.04E-1 (2.5E-2)	8.54E-2 (7.1E-1)	8.01E-1 (7.2E-1)	1.62E-1 (5.6E-4)
n = 1000					
DTLZ1	2	**2.32E-3** (5.1E-1)	5.15E3 * (1.7E3)	7.98E2 * (9.7E2)	4.68E3 * (1.8E2)
DTLZ2	2	5.89E-3 * (8.1E-4)	7.18E-1 * (6.3E-1)	**4.56E-3** (1.0E-3)	1.06E0 * (1.0E-1)
DTLZ3	2	**1.14E-2** (2.1E0)	1.38E4 * (4.5E3)	1.51E3 * (2.7E3)	1.32E4 * (4.3E2)
DTLZ4	2	8.93E-3 (4.3E-3)	6.91E-1 * (4.0E-1)	**8.13E-3** (4.8E-3)	1.49E0 * (4.1E-1)
DTLZ5	2	5.87E-3 (5.8E-4)	7.07E-1 * (5.2E-1)	**4.77E-3** (2.3E-3)	1.12E0 * (1.2E-1)
DTLZ6	2	5.18E-3 * (3.7E-4)	7.65E1 * (4.7E1)	**3.96E-3** (6.2E-8)	6.65E2 * (8.8E0)
DTLZ7	2	4.42E-1 * (2.5E-4)	4.42E-1 * (1.1E-4)	**4.42E-1** (4.3E-1)	9.66E-1 * (7.8E-2)
DTLZ1	3	**2.40E0** (4.1E1)	5.40E3 * (1.8E3)	1.02E3 * (1.1E3)	4.26E3 * (2.5E2)
DTLZ2	3	1.62E-1 * (9.0E-2)	1.85E0 * (1.3E0)	**5.89E-2** (4.1E-3)	9.26E-1 * (1.2E-1)
DTLZ3	3	**5.62E-1** (1.0E1)	1.74E4 * (4.4E3)	3.45E3 * (2.9E3)	1.24E4 * (5.1E2)
DTLZ4	3	2.78E-1 * (2.2E-1)	1.82E0 * (1.5E0)	**6.55E-2** (1.0E-2)	1.65E0 * (2.1E-1)
DTLZ5	3	5.14E-2 * (5.1E-2)	2.25E0 * (2.1E0)	**2.50E-2** (1.2E-2)	1.39E0 * (1.5E-1)
DTLZ6	3	**5.89E-3** (2.5E-4)	2.33E2 * (9.5E1)	2.43E-2 * (3.3E-3)	7.00E2 * (3.1E1)
DTLZ7	3	1.53E0 * (1.1E0)	**7.99E-1** (7.1E-1)	8.03E-1 * (4.2E-1)	1.58E0 * (1.7E-1)

Table B.44: Performance comparison using the IGD indicator on the LSMOP benchmarks using 10,000,000 function evaluations.

		WOF-SMPSO	WOF-Randomised	LMEA	MOEA/DVA	S3-CMA-ES
n = 1000						
LSMOP2	2	**6.96E-3** (8.0E-4)	8.51E-3 * (5.3E-3)	3.09E-2 * (3.5E-4)	3.31E-2 * (1.5E-2)	9.64E-3 * (1.3E-3)
LSMOP4	2	1.88E-2 * (5.6E-4)	1.97E-2 * (3.2E-3)	3.85E-2 * (1.4E-3)	**1.11E-2** (7.6E-3)	1.25E-2 * (2.8E-3)
LSMOP5	2	2.23E-2 * (3.5E-3)	2.16E-1 * (1.0E-1)	4.72E-1 * (3.1E-1)	**4.71E-3** (4.9E-4)	7.42E-1 * (4.8E-10)
LSMOP6	2	**2.82E-2** (2.2E-3)	1.81E-1 * (6.2E-3)	7.67E-1 * (8.4E-4)	4.88E-1 * (4.6E-2)	7.45E-1 * (1.0E-3)
LSMOP7	2	**8.24E-1** (1.1E-1)	1.11E0 * (5.4E-1)	1.71E0 * (2.3E-1)	6.89E0 * (1.7E0)	1.78E0 * (2.1E-1)
LSMOP8	2	1.70E-2 * (1.9E-3)	3.57E-2 * (1.2E-2)	1.22E-1 * (2.2E-1)	**1.53E-2** (1.1E-2)	2.75E-1 * (8.0E-2)
LSMOP9	2	1.26E-2 (3.1E-3)	4.58E-1 * (2.0E-3)	6.19E-1 * (4.0E-2)	**7.64E-3** (1.9E-2)	5.18E-1 * (1.6E-1)
LSMOP2	3	5.92E-2 * (3.2E-3)	4.41E-2 * (4.0E-4)	5.13E-2 * (5.2E-4)	4.74E-2 * (1.4E-2)	**4.37E-2** (1.1E-3)
LSMOP4	3	8.44E-2 * (7.4E-3)	6.37E-2 * (3.7E-3)	7.31E-2 * (1.9E-3)	5.73E-2 (1.7E-2)	**5.25E-2** (4.8E-3)
LSMOP5	3	2.47E-1 * (5.1E-2)	2.17E-1 * (1.8E-1)	6.11E0 * (6.1E0)	**5.50E-2** (8.7E-5)	9.45E-1 * (1.0E-1)
LSMOP6	3	**9.08E-1** (7.1E-3)	1.28E0 * (3.4E-2)	1.64E0 * (1.6E-1)	5.11E0 * (4.5E-1)	1.45E0 * (9.1E-3)
LSMOP7	3	**3.94E-1** (1.4E-2)	8.24E-1 * (1.7E-1)	5.81E-1 * (2.1E-2)	6.15E-1 * (5.3E-2)	9.45E-1 * (1.0E-8)
LSMOP8	3	8.08E-2 * (5.2E-3)	1.03E-1 * (4.5E-2)	7.79E-2 * (7.3E-3)	**5.75E-2** (1.8E-2)	9.45E-1 * (1.4E-9)
LSMOP9	3	4.10E-1 * (9.6E-3)	1.40E0 * (3.2E-1)	6.06E-1 * (5.1E-2)	**1.48E-1** (2.2E-2)	6.55E-1 * (3.1E-2)

Table B.45: Performance comparison using the IGD indicator on the DTLZ benchmarks using 10,000,000 function evaluations.

		WOF-SMPSO	WOF-Randomised	LMEA	MOEA/DVA	S3-CMA-ES
n = 1000						
DTLZ1	2	2.23E-3 * (1.4E-4)	**1.78E-3** (5.0E-6)	6.01E2 * (2.8E1)	4.63E2 * (1.1E1)	1.00E4 * (1.5E2)
DTLZ2	2	5.12E-3 * (2.8E-4)	**3.96E-3** (3.8E-9)	2.42E-2 * (5.5E-3)	2.19E-2 * (4.3E-4)	5.56E-3 * (5.9E-4)
DTLZ3	2	4.86E-3 * (2.2E-4)	**3.96E-3** (1.3E-5)	1.63E3 * (4.9E1)	1.27E3 * (3.0E1)	2.75E4 * (3.8E2)
DTLZ4	2	5.11E-3 * (2.6E-4)	**3.96E-3** (1.3E-7)	2.86E-2 * (1.5E-2)	2.74E-2 * (1.3E-2)	1.70E-1 * (5.4E-1)
DTLZ5	2	5.07E-3 * (2.5E-4)	**3.96E-3** (2.6E-9)	2.49E-2 * (8.8E-3)	2.19E-2 * (4.0E-4)	5.53E-3 * (3.4E-4)
DTLZ6	2	5.12E-3 * (3.3E-4)	**3.96E-3** (1.0E-8)	2.44E2 * (6.1E0)	1.60E2 * (3.1E0)	7.45E2 * (9.4E0)
DTLZ7	2	5.36E-3 (3.0E-4)	**5.25E-3** (4.3E-1)	9.36E-3 (1.2E-3)	2.25E-2 (5.2E-4)	2.80E-2 * (5.8E-3)
DTLZ1	3	8.42E0 * (3.0E1)	**2.09E-2** (2.9E-2)	3.83E2 * (1.9E1)	3.42E2 * (1.3E1)	8.29E3 * (2.5E2)
DTLZ2	3	7.15E-2 * (5.0E-3)	**5.44E-2** (6.0E-8)	6.23E-2 * (3.4E-3)	6.05E-2 * (1.4E-4)	5.91E-2 * (1.6E-3)
DTLZ3	3	2.82E-1 * (2.1E0)	**5.44E-2** (2.2E-4)	1.27E3 * (6.0E1)	1.12E3 * (2.5E1)	2.75E4 * (3.7E2)
DTLZ4	3	7.18E-2 * (5.3E-3)	**5.44E-2** (4.3E-6)	1.07E-1 * (1.1E-1)	4.74E-1 * (4.0E-4)	5.72E-1 * (2.7E-1)
DTLZ5	3	**5.95E-3** (4.5E-4)	1.36E-2 * (2.8E-3)	2.59E-2 * (1.2E-2)	1.78E-2 * (3.3E-4)	1.27E-2 * (1.0E-3)
DTLZ6	3	**5.85E-3** (5.2E-4)	2.24E-2 * (5.1E-3)	2.18E2 * (1.0E1)	1.42E2 * (3.8E0)	7.44E2 * (1.1E1)
DTLZ7	3	9.74E-2 * (1.3E-2)	7.86E-2 * (6.7E-3)	**6.18E-2** (2.2E-3)	1.11E-1 * (2.7E-4)	7.40E-2 * (4.2E-3)

Table B.46: Performance comparison using the IGD indicator on the LSMOP benchmarks using 100, 000 function evaluations.

		WOF-SMPSO	WOF-Randomised	groupInfLMEA	groupInfMOEA/DVA	groupInfS3-CMA-ES
n = 1000						
LSMOP2	2	**7.29E-3** (3.9E-4)	1.03E-2 * (3.0E-3)	3.88E-2 * (6.6E-4)	3.93E-2 * (1.4E-2)	3.61E-2 * (5.1E-4)
LSMOP4	2	**2.04E-2** (5.9E-4)	2.37E-2 * (2.5E-3)	5.58E-2 * (9.3E-4)	5.22E-2 * (2.0E-2)	5.51E-2 * (1.0E-3)
LSMOP5	2	7.42E-1 (2.9E-1)	**7.41E-1** (4.7E-2)	6.68E0 * (4.3E-1)	1.24E1 * (3.9E-1)	6.67E0 * (4.4E-1)
LSMOP6	2	**1.73E-1** (3.4E-3)	3.49E-1 * (1.8E-1)	1.12E2 * (4.9E2)	4.12E2 * (6.2E2)	7.70E-1 * (1.7E-3)
LSMOP7	2	1.51E0 (6.0E-4)	**1.51E0** (2.8E-3)	7.58E3 * (9.3E2)	4.31E4 * (4.1E3)	5.95E3 * (8.4E2)
LSMOP8	2	**2.12E-1** (5.9E-1)	2.14E-1 (1.8E-1)	4.53E0 * (2.2E-1)	1.03E1 * (3.4E-1)	5.94E0 * (7.0E-1)
LSMOP9	2	**4.67E-1** (9.5E-3)	4.90E-1 * (7.5E-1)	1.25E1 * (8.1E-1)	1.99E1 * (2.1E0)	2.15E0 * (3.0E-1)
LSMOP2	3	6.12E-2 (5.2E-3)	5.16E-2 (5.7E-4)	5.66E-2 (2.9E-3)	5.10E-2 (1.7E-2)	5.20E-2 (7.6E-4)
LSMOP4	3	**8.91E-2** (4.9E-3)	9.31E-2 * (7.0E-3)	1.05E-1 * (4.1E-3)	1.01E-1 * (2.5E-2)	9.95E-2 * (3.3E-3)
LSMOP5	3	**4.29E-1** (2.3E-1)	4.51E-1 (7.3E-2)	5.10E0 * (8.9E0)	1.09E1 * (5.0E-1)	6.09E0 * (9.9E-1)
LSMOP6	3	**9.11E-1** (5.8E-1)	1.31E0 * (9.8E-2)	5.20E2 * (3.1E2)	1.24E4 * (3.4E3)	5.02E3 * (2.0E3)
LSMOP7	3	8.49E-1 (1.0E-1)	**8.47E-1** (3.3E-3)	1.74E1 * (2.9E2)	1.06E3 * (8.5E2)	1.02E0 * (5.2E-3)
LSMOP8	3	**9.22E-2** (1.5E-2)	1.86E-1 * (7.8E-2)	6.02E-1 * (4.7E-2)	6.43E-1 * (4.6E-2)	5.92E-1 * (2.9E-2)
LSMOP9	3	**1.11E0** (5.7E-1)	1.14E0 * (1.7E-2)	2.84E1 * (1.6E0)	5.47E1 * (2.6E0)	3.47E1 * (7.7E0)

Table B.47: Performance comparison using the IGD indicator on the DTLZ benchmarks using 100, 000 function evaluations.

		WOF-SMPSO	WOF-Randomised	groupInfLMEA	groupInfMOEA/DVA	groupInfS3-CMA-ES
n = 1000						
DTLZ1	2	**2.32E-3** (5.1E-1)	7.98E2 * (9.7E2)	2.23E4 * (3.1E2)	2.06E4 * (4.2E2)	1.54E4 * (6.1E2)
DTLZ2	2	5.89E-3 * (8.1E-4)	**4.56E-3** (1.0E-3)	3.82E1 * (7.4E-1)	5.45E1 * (8.0E-1)	5.03E1 * (2.5E0)
DTLZ3	2	**1.14E-2** (2.1E0)	1.51E3 * (2.7E3)	6.27E4 * (9.3E2)	5.68E4 * (1.1E3)	4.31E4 * (1.5E3)
DTLZ4	2	8.93E-3 (4.3E-3)	**8.13E-3** (4.8E-3)	3.82E1 * (1.0E0)	5.48E1 * (9.9E-1)	5.08E1 * (4.0E0)
DTLZ5	2	5.87E-3 (5.8E-4)	**4.77E-3** (2.3E-3)	3.81E1 * (9.8E-1)	5.45E1 * (1.1E0)	4.94E1 * (2.0E0)
DTLZ6	2	5.18E-3 * (3.7E-4)	**3.96E-3** (6.2E-8)	8.55E2 * (1.6E0)	8.19E2 * (3.3E0)	7.67E2 * (8.7E0)
DTLZ7	2	4.42E-1 * (2.5E-4)	**4.42E-1** (4.3E-1)	4.82E0 * (1.1E-1)	6.46E0 * (1.0E-1)	3.07E0 * (1.6E-1)
DTLZ1	3	**2.40E0** (4.1E1)	1.02E3 * (1.1E3)	1.12E4 * (2.0E2)	1.15E4 * (3.1E2)	1.27E4 * (5.7E2)
DTLZ2	3	1.62E-1 * (9.0E-2)	**5.89E-2** (4.1E-3)	1.55E1 * (4.4E-1)	3.94E1 * (1.1E0)	5.03E1 * (2.1E0)
DTLZ3	3	**5.62E-1** (1.0E1)	3.45E3 * (2.9E3)	3.76E4 * (9.0E2)	3.78E4 * (6.1E2)	4.36E4 * (1.9E3)
DTLZ4	3	2.78E-1 * (2.2E-1)	**6.55E-2** (1.0E-2)	1.58E1 * (6.1E-1)	3.96E1 * (5.7E-1)	5.04E1 * (2.2E0)
DTLZ5	3	5.14E-2 * (5.1E-2)	**2.50E-2** (1.2E-2)	1.54E1 * (4.2E-1)	3.93E1 * (1.0E0)	5.04E1 * (2.6E0)
DTLZ6	3	**5.89E-3** (2.5E-4)	2.43E-2 * (3.3E-3)	8.02E2 * (1.9E0)	7.40E2 * (4.4E0)	7.67E2 * (1.3E1)
DTLZ7	3	1.53E0 * (1.1E0)	**8.03E-1** (4.2E-1)	3.53E0 * (1.8E-1)	7.67E0 * (2.0E-1)	4.18E0 * (2.5E-1)

APPENDIX C

Winning Rates using HV

Table C.1: Winning rates using the HV indicator for different problem categories using 100,000 evaluations of the SMPSO and NSGA-II algorithms and their WOF-versions as well as the randomised WOF algorithm. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	SMPSO	WOF-SMPSO	NSGA-II	WOF-NSGA-II	WOF-Randomised
SMPSO	—	1.08 | 1.78 0.00 | 1.56	18.47 | 7.14 27.17 | 0.00	2.17 | 3.57 0.00 | 1.56	1.63 | 3.57 0.00 | 0.00
WOF-SMPSO	83.69 | 73.21 84.78 | 67.18	—	69.56 | 32.14 82.60 | 59.37	50.54 | 26.78 64.13 | 26.56	30.97 | 21.42 39.13 | 0.00
NSGA-II	34.78 | 64.28 17.39 | 31.25	13.58 | 42.85 1.08 | 12.50	—	7.06 | 19.64 0.00 | 6.25	3.80 | 12.50 0.00 | 0.00
WOF-NSGA-II	77.71 | 69.64 77.17 | 68.75	27.17 | 39.28 16.30 | 37.50	57.06 | 12.50 73.91 | 50.00	—	8.69 | 8.92 5.43 | 3.12
WOF-Randomised	85.32 | 76.78 85.86 | 78.12	42.93 | 50.00 33.69 | 62.50	77.71 | 50.00 85.86 | 78.12	63.58 | 46.42 69.56 | 65.62	—

Table C.2: Winning rates using the HV indicator for different problem categories using 100,000 evaluations of the NSGA-III and MOEA/D algorithms and their WOF-versions as well as the randomised WOF algorithm. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	NSGA-III	WOF-NSGA-III	MOEA/D	WOF-MOEA/D	WOF-Randomised
NSGA-III	—	12.50 | 37.50 1.08 | 12.50	41.30 | 60.71 34.78 | 18.75	23.91 | 55.35 7.60 | 21.87	9.78 | 30.35 0.00 | 7.81
WOF-NSGA-III	62.50 | 21.42 76.08 | 56.25	—	72.28 | 53.57 77.17 | 62.50	49.45 | 62.50 36.95 | 48.43	3.80 | 1.78 4.34 | 4.68
MOEA/D	23.36 | 8.92 18.47 | 40.62	8.15 | 21.42 2.17 | 12.50	—	16.84 | 35.71 6.52 | 23.43	5.97 | 16.07 1.08 | 10.93
WOF-MOEA/D	61.95 | 17.85 80.43 | 60.93	27.17 | 8.92 40.21 | 25.00	66.30 | 30.35 82.60 | 59.37	—	10.32 | 7.14 8.69 | 18.75
WOF-Randomised	69.02 | 23.21 85.86 | 65.62	54.34 | 33.92 66.30 | 37.50	77.71 | 53.57 85.86 | 67.18	67.39 | 64.28 70.65 | 54.68	—

Table C.3: Winning rates using the HV indicator for different problem categories using 100,000 evaluations of NSGA-II and its grouped and linked versions. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	NSGA-II	GroupedNSGA-II	LinkedNSGA-II	GroupLinkNSGA-II	HighProbNSGA-II
NSGA-II	—	28.80 | 69.64 7.60 | 14.06	0.00 | 0.00 0.00 | 0.00	14.13 | 46.42 0.00 | 9.37	35.32 | 71.42 15.21 | 20.31
GroupedNSGA-II	20.65 | 3.57 31.52 | 6.25	—	21.73 | 3.57 33.69 | 6.25	0.00 | 0.00 0.00 | 0.00	33.15 | 39.28 32.60 | 12.50
LinkedNSGA-II	4.89 | 7.14 1.08 | 0.00	29.34 | 71.42 7.60 | 12.50	—	15.76 | 51.78 0.00 | 9.37	35.32 | 71.42 14.13 | 18.75
GroupLinkNSGA-II	58.15 | 14.28 76.08 | 51.56	75.00 | 64.28 76.08 | 64.06	55.97 | 8.92 75.00 | 50.00	—	76.08 | 67.85 76.08 | 64.06
HighProbNSGA-II	15.21 | 3.57 22.82 | 0.00	3.80 | 1.78 3.26 | 0.00	15.76 | 3.57 23.91 | 0.00	0.54 | 1.78 0.00 | 0.00	—

Table C.4: Winning rates using the HV indicator for different problem categories using 100,000 evaluations of NSGA-III and its grouped and linked versions. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	NSGA-III	GroupedNSGA-III	LinkedNSGA-III	GroupLinkNSGA-III	HighProbNSGA-III
NSGA-III	—	33.15 | 66.07 15.21 | 23.43	1.08 | 3.57 0.00 | 1.56	13.04 | 42.85 0.00 | 12.50	38.04 | 69.64 20.65 | 26.56
GroupedNSGA-III	25.00 | 5.35 35.86 | 15.62	—	23.91 | 3.57 34.78 | 15.62	0.00 | 0.00 0.00 | 0.00	38.58 | 37.50 41.30 | 18.75
LinkedNSGA-III	5.43 | 7.14 3.26 | 3.12	35.32 | 71.42 16.30 | 26.56	—	13.04 | 42.85 0.00 | 12.50	41.30 | 75.00 22.82 | 31.25
GroupLinkNSGA-III	63.04 | 23.21 79.34 | 57.81	77.17 | 69.64 79.34 | 71.87	61.95 | 19.64 79.34 | 57.81	—	78.80 | 75.00 79.34 | 71.87
HighProbNSGA-III	17.93 | 5.35 27.17 | 10.93	5.97 | 10.71 2.17 | 7.81	17.39 | 3.57 27.17 | 10.93	0.00 | 0.00 0.00 | 0.00	—

Table C.5: Winning rates using the HV indicator for different problem categories using 100,000 evaluations of SMPSO and its grouped and linked versions. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	SMPSO	GroupedSMPSO	LinkedSMPSO	GroupLinkSMPSO	HighProbSMPSO
SMPSO	—	4.89 | 10.71 3.26 | 0.00	0.00 | 0.00 0.00 | 0.00	1.63 | 5.35 0.00 | 0.00	5.97 | 10.71 3.26 | 0.00
GroupedSMPSO	25.54 | 17.85 33.69 | 3.12	—	25.54 | 17.85 33.69 | 3.12	2.17 | 7.14 0.00 | 0.00	21.19 | 10.71 30.43 | 3.12
LinkedSMPSO	1.63 | 3.57 1.08 | 0.00	4.34 | 8.92 3.26 | 1.56	—	2.17 | 7.14 0.00 | 0.00	7.06 | 14.28 3.26 | 1.56
GroupLinkSMPSO	59.23 | 39.28 68.47 | 29.68	58.15 | 41.07 66.30 | 31.25	59.78 | 41.07 67.39 | 31.25	—	59.78 | 41.07 67.39 | 31.25
HighProbSMPSO	21.73 | 16.07 27.17 | 0.00	0.54 | 0.00 0.00 | 0.00	19.56 | 10.71 26.08 | 0.00	1.63 | 5.35 0.00 | 0.00	—

Table C.6: Winning rates using the HV indicator for different problem categories using 100,000 evaluations. The original NSGA-II, SMPSO and NSGA-III algorithms are shown along their LCSA-enhanced versions. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	SMPSO	xSMPSO	NSGA-II	xNSGA-II	NSGA-III	xNSGA-III
SMPSO	—	5.97 | 17.85 0.00 | 3.12	19.02 | 7.14 28.26 | 0.00	7.06 | 3.57 8.69 | 1.56	16.84 | 7.14 26.08 | 0.00	6.52 | 3.57 8.69 | 0.00
xSMPSO	61.95 | 21.42 72.82 | 67.18	—	60.32 | 14.28 73.91 | 65.62	35.32 | 14.28 47.82 | 25.00	55.43 | 16.07 68.47 | 53.12	28.26 | 12.50 39.13 | 7.81
NSGA-II	36.95 | 67.85 19.56 | 32.81	21.73 | 64.28 3.26 | 10.93	—	14.67 | 25.00 9.78 | 9.37	14.13 | 23.21 8.69 | 1.56	11.41 | 23.21 4.34 | 0.00
xNSGA-II	73.36 | 75.00 65.21 | 76.56	29.89 | 67.85 10.86 | 20.31	53.26 | 17.85 64.13 | 60.93	—	53.26 | 26.78 61.95 | 53.12	22.82 | 23.21 21.73 | 4.68
NSGA-III	40.21 | 67.85 22.82 | 40.62	26.63 | 64.28 8.69 | 25.00	36.95 | 44.64 33.69 | 37.50	23.91 | 41.07 14.13 | 25.00	—	16.30 | 32.14 7.60 | 6.25
xNSGA-III	75.00 | 76.78 67.39 | 78.12	47.28 | 67.85 30.43 | 60.93	67.93 | 50.00 70.65 | 78.12	41.84 | 41.07 36.95 | 64.06	54.34 | 14.28 67.39 | 60.93	—

Table C.7: Winning rates using the HV indicator for different problem categories using 100,000 evaluations. WOF and LSMOF are compared using the NSGA-II and SMPSO algorithms. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-SMPSO	LS-SMPSO	WOF-NSGA-II	LS-NSGA-II	WOF-Randomised
WOF-SMPSO	—	63.04 | 69.64 58.69 | 35.93	51.08 | 26.78 64.13 | 28.12	47.82 | 32.14 54.34 | 25.00	30.97 | 21.42 39.13 | 0.00
LS-SMPSO	8.69 | 3.57 10.86 | 20.31	—	35.86 | 21.42 43.47 | 28.12	29.34 | 14.28 35.86 | 21.87	13.58 | 10.71 16.30 | 6.25
WOF-NSGA-II	27.17 | 39.28 16.30 | 37.50	48.91 | 64.28 40.21 | 43.75	—	32.06 | 33.92 30.43 | 28.12	8.69 | 8.92 5.43 | 3.12
LS-NSGA-II	29.89 | 42.85 25.00 | 37.50	45.65 | 76.78 35.86 | 37.50	32.06 | 33.92 33.69 | 21.87	—	16.84 | 26.78 15.21 | 9.37
WOF-Randomised	42.93 | 50.00 33.69 | 62.50	69.02 | 73.21 64.13 | 68.75	63.58 | 46.42 69.56 | 65.62	64.67 | 53.57 67.39 | 70.31	—

Table C.8: Winning rates using the HV indicator for different problem categories using 100,000 evaluations. WOF is compared with the random-group-based MOEA/DVA, LMEA and S^3-CMA-ES. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	randomLMEA	randomMOEA/DVA	randomS3-CMA-ES
WOF-SMPSO	—	51.08 | 26.78 64.13 | 28.12	30.97 | 21.42 39.13 | 0.00	70.10 | 42.85 79.34 | 46.87	83.15 | 75.00 83.69 | 64.06	80.43 | 76.78 80.43 | 51.56
WOF-NSGA-II	27.17 | 39.28 16.30 | 37.50	—	8.69 | 8.92 5.43 | 3.12	62.50 | 32.14 72.82 | 46.87	75.00 | 62.50 77.17 | 59.37	74.45 | 69.64 73.91 | 51.56
WOF-Randomised	42.93 | 50.00 33.69 | 62.50	63.58 | 46.42 69.56 | 65.62	—	76.08 | 44.64 85.86 | 70.31	84.23 | 71.42 85.86 | 78.12	85.86 | 78.57 85.86 | 76.56
randomLMEA	19.02 | 46.42 5.43 | 29.68	19.56 | 42.85 6.52 | 28.12	10.86 | 35.71 0.00 | 10.93	—	44.56 | 71.42 36.95 | 34.37	49.45 | 85.71 34.78 | 35.93
randomMOEA/DVA	4.34 | 10.71 1.08 | 7.81	4.34 | 5.35 1.08 | 9.37	0.54 | 1.78 0.00 | 0.00	14.67 | 12.50 9.78 | 15.62	—	33.15 | 58.92 17.39 | 17.18
randomS3-CMA-ES	7.60 | 8.92 4.34 | 18.75	7.06 | 5.35 5.43 | 15.62	1.08 | 1.78 0.00 | 1.56	11.95 | 1.78 11.95 | 12.50	24.45 | 16.07 25.00 | 26.56	—

Table C.9: Winning rates using the HV indicator for different problem categories using 100,000 evaluations. WOF is compared with ReMO using NSGA-II and SMPSO. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-NSGA-II	ReNSGA-II	WOF-MOEA/D	ReMOEA/D	WOF-Randomised
WOF-NSGA-II	—	80.97 | 75.00 77.17 | 70.31	42.39 | 48.21 36.95 | 35.93	83.15 | 80.35 78.26 | 70.31	8.69 | 8.92 5.43 | 3.12
ReNSGA-II	0.00 | 0.00 0.00 | 0.00	—	4.34 | 5.35 3.26 | 6.25	47.82 | 69.64 42.39 | 20.31	0.00 | 0.00 0.00 | 0.00
WOF-MOEA/D	33.15 | 16.07 43.47 | 37.50	83.15 | 75.00 84.78 | 75.00	—	86.41 | 83.92 85.86 | 76.56	10.32 | 7.14 8.69 | 18.75
ReMOEA/D	2.17 | 0.00 4.34 | 6.25	6.52 | 0.00 8.69 | 12.50	2.17 | 0.00 2.17 | 4.68	—	1.08 | 0.00 2.17 | 3.12
WOF-Randomised	63.58 | 46.42 69.56 | 65.62	88.04 | 83.92 85.86 | 79.68	67.39 | 64.28 70.65 | 54.68	88.58 | 85.71 85.86 | 79.68	—

Table C.10: Winning rates using the HV indicator for different problem categories using 100,000 evaluations. WOF is compared with DLS-MOEA. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

102 | 42 60 | 0	WOF-SMPSO	WOF-NSGA-II	WOF-Randomised	DLS-MOEA
WOF-SMPSO	—	62.74 | 30.95 85.00	47.05 | 28.57 60.00	69.60 | 42.85 88.33
WOF-NSGA-II	21.56 | 38.09 10.00	—	8.82 | 11.90 6.66	59.80 | 33.33 78.33
WOF-Randomised	34.31 | 47.61 25.00	61.76 | 40.47 76.66	—	64.70 | 33.33 86.66
DLS-MOEA	23.52 | 45.23 8.33	17.64 | 35.71 5.00	18.62 | 42.85 1.66	—

Table C.11: Winning rates using the HV indicator for different problem categories using 10,000,000 evaluations. WOF is compared with MOEA/DVA and LMEA. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

42 | 0 42 | 0	WOF-SMPSO	WOF-Randomised	LMEA	MOEA/DVA
WOF-SMPSO	—	35.71 | 35.71 |	69.04 | 69.04 |	47.61 | 47.61 |
WOF-Randomised	57.14 | 57.14 |	—	83.33 | 83.33 |	50.00 | 50.00 |
LMEA	14.28 | 14.28 |	4.76 | 4.76 |	—	23.80 | 23.80 |
MOEA/DVA	30.95 | 30.95 |	33.33 | 33.33 |	47.61 | 47.61 |	—

Table C.12: Winning rates using the HV indicator for different problem categories using $10,000,000$ evaluations. WOF is compared with MOEA/DVA, LMEA and S^3-CMA-ES. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

28 | 0 28 | 0	WOF-SMPSO	WOF-Randomised	LMEA	MOEA/DVA	S3-CMA-ES
WOF-SMPSO	—	39.28 | — 39.28	78.57 | — 78.57	64.28 | — 64.28	71.42 | — 71.42
WOF-Randomised	50.00 | — 50.00	—	85.71 | — 85.71	67.85 | — 67.85	75.00 | — 75.00
LMEA	7.14 | — 7.14	3.57 | — 3.57	—	35.71 | — 35.71	32.14 | — 32.14
MOEA/DVA	10.71 | — 10.71	17.85 | — 17.85	25.00 | — 25.00	—	28.57 | — 28.57
S3-CMA-ES	21.42 | — 21.42	21.42 | — 21.42	42.85 | — 42.85	35.71 | — 35.71	—

Table C.13: Winning rates using the HV indicator for different problem categories using 100,000 evaluations. WOF is compared with modified versions of MOEA/DVA, LMEA and S^3-CMA-ES. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

28 | 0 28 | 0	WOF-SMPSO	WOF-Randomised	groupInfLMEA	groupInfMOEA/DVA	groupInfS3-CMA-ES
WOF-SMPSO	—	42.85 42.85	96.42 96.42	92.85 92.85	92.85 92.85
WOF-Randomised	35.71 35.71	—	85.71 85.71	82.14 82.14	85.71 85.71
groupInfLMEA	0.00 0.00	0.00 0.00	—	7.14 7.14	7.14 7.14
groupInfMOEA/DVA	0.00 0.00	0.00 0.00	0.00 0.00	—	0.00 0.00
groupInfS3-CMA-ES	3.57 3.57	0.00 0.00	28.57 28.57	25.00 25.00	—

Table C.14: Winning rates using the HV indicator for different problem categories using 100,000 evaluations. The proposed WOF, GLMO and LCSA are compared using NSGA-II. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-NSGA-II		xNSGA-II		GroupLinkNSGA-II	
WOF-NSGA-II	—		39.13 50.00	16.07 25.00	38.04 32.60	46.42 31.25
xNSGA-II	16.30 17.39	14.28 18.75	—		34.23 22.82	50.00 39.06
GroupLinkNSGA-II	29.34 41.30	8.92 21.87	39.67 55.43	12.50 21.87	—	

Table C.15: Winning rates using the HV indicator for different problem categories using 100,000 evaluations. The proposed WOF, GLMO and LCSA are compared using NSGA-III. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-NSGA-III	xNSGA-III	GroupLinkNSGA-III
WOF-NSGA-III	—	37.50 | 21.42 46.73 | 12.50	38.04 | 37.50 34.78 | 42.18
xNSGA-III	33.69 | 37.50 28.26 | 53.12	—	35.86 | 37.50 29.34 | 51.56
GroupLinkNSGA-III	34.23 | 16.07 45.65 | 26.56	40.76 | 12.50 56.52 | 23.43	—

Table C.16: Winning rates using the HV indicator for different problem categories using 100,000 evaluations. The proposed WOF, GLMO and LCSA are compared using SMPSO. Each row shows the amount of wins (based on statistical significance) against the respective algorithms in the columns.

184 | 56 92 | 64	WOF-SMPSO	xSMPSO	GroupLinkSMPSO
WOF-SMPSO	—	56.52 | 69.64 53.26 | 18.75	60.32 | 57.14 56.52 | 56.25
xSMPSO	13.04 | 7.14 13.04 | 31.25	—	29.34 | 14.28 28.26 | 56.25
GroupLinkSMPSO	12.50 | 3.57 17.39 | 7.81	42.93 | 33.92 48.91 | 10.93	—

Glossary

MOP	Multi-objective Problem (Section 2.1)
LSO	Large-scale Optimisation (Section 2.3)
EA	Evolutionary Algorithm (Section 2.2)
PSO	Particle Swarm Optimisation (Section 2.2)
PF	Pareto-front of a multi-objective optimisation problem (Definition 2.4)
PS	Pareto-set of a multi-objective optimisation problem (Definition 2.3)
n	Number of decision variables of an optimisation problem
m	Number of objective functions of an optimisation problem
Ω	The decision space of an optimisation problem
$\mathcal{M}$	The objective space of an optimisation problem
Γ	A grouping mechanism (Definition 2.6)
CC	Cooperative Coevolution (Section 2.5)
GLMO	Grouped and Linked Polynomial Mutation Operator (Section 5.2.4)
LCSA	Linear Combination-based Search Algorithm (Section 5.3)
WOF	Weighted Optimisation Framework (Section 5.1)
HV	Hypervolume indicator (Section 2.7)
IGD	Inverted Generational Distance indicator (Section 2.7)
DG / DG2	Differential Grouping / Differential Grouping 2 (Section 3.3.3)

Ehrenerklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; verwendete fremde und eigene Quellen sind als solche kenntlich gemacht. Insbesondere habe ich nicht die Hilfe eines kommerziellen Promotionsberaters in Anspruch genommen. Dritte haben von mir weder unmittelbar noch mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Ich habe insbesondere nicht wissentlich:

- Ergebnisse erfunden oder widersprüchliche Ergebnisse verschwiegen,
- statistische Verfahren absichtlich missbraucht, um Daten in ungerechtfertigter Weise zu interpretieren,
- fremde Ergebnisse oder Veröffentlichungen plagiiert,
- fremde Forschungsergebnisse verzerrt wiedergegeben.

Mir ist bekannt, dass Verstöße gegen das Urheberrecht Unterlassungs- und Schadensersatzansprüche des Urhebers sowie eine strafrechtliche Ahndung durch die Strafverfolgungsbehörden begründen kann. Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form als Dissertation eingereicht und ist als Ganzes auch noch nicht veröffentlicht.

Magdeburg, den 13.09.2019

Heiner Zille