Multi-objective evolutionary algorithms for optimal scheduling

Wang, Y.

Multi-objective Evolutionary Algorithms for Optimal Scheduling

Proefschrift

ter verkrijging van
de graad van doctor aan de Universiteit Leiden,
op gezag van rector magnificus prof.dr.ir. H. Bijl,
volgens besluit van het college voor promoties
te verdedigen op woensdag 19 januari 2022
klokke 13.45 uur

door

Yali Wang

Promotiecomissie

Promotors:	Dr. Michael Emmerich	
	Prof. Dr. Thomas Bäck	
Overige leden:	Prof. Dr. Hisao Ishibuchi	(SUSTech, China)
	Prof. Dr. Kaisa Miettinen	(JYU, Finland)
	Prof. Dr. Aske Plaat	(voorzitter)
	Prof. Dr. Marcello Bonsangue	(secretaris)
	Dr. Anna Kononova	
	Dr. Koen van der Blom	

This work is part of the research programme Smart Industry SI2016 with project name CIMPLO and project number 15465, which is (partly) financed by the Netherlands Organisation for Scientific Research (NWO).

Contents

Chapter 1

Introduction

1.1 Background

Optimization aims to make the most effective or functional use of resources. Mathematically speaking, optimization is the problem of finding the best feasible solution with respect to an objective function which is used to evaluate solutions for determining the best one. Most real-world optimization problems necessitate optimization of multiple, conflicting objectives, they are referred to as multi-objective optimization which is much more challenging but has extremely practical importance. We can find multi-objective optimization applications in every aspect of our real world, such as engineering, computer science, ecology, sociology, economics, agriculture, aviation, automotive, medicine, and so on.

Classically multi-objective optimization problems were handled by deterministic methods [77]. These methods have a limited scope and require functions to have certain properties, such as convexity or differentiability, and often computationally-intensive to find an exact solution. The more flexible metaheuristic approaches were introduced in some early work by Schaffer [92], Fonesca and Fleming [46], Srinivas and Deb [99], Horn et al. [59], Kursawe [70], etc. They were popularized by Deb's book "Multi-Objective Optimization using Evolutionary Algorithms" [22]. Among powerful metaheuristic techniques, evolutionary algorithms (EAs) have been used extensively and proven to be suited for solving complex optimization problems. Multi-objective evolutionary algorithm (MOEA) has already become the major approach to solve multi-objective optimization (MOO) problems.

The process of evolutionary computing is inspired by biological evolution. EA

mimics evolutionary processes in nature, such as nature selection and variation (mutation, crossover). Candidate solutions are represented as chromosomes, for instance, integer vectors or real vectors, depending on the real world application. The steps of an EA can be described as follows: a set of solutions (population) is created usually randomly (initialization). In each iteration (generation), new solutions (offspring) are created by genetic operators: selecting top members by the quality (fitness) function as parents (mating selection); recombining portions of these parents to create offspring solutions (crossover); producing a small modification to offspring solutions (mutation). From the set of old and new solutions (parents and offspring), a new set of solutions (new generation) is chosen based on the quality function, where better solutions are preferred (survivor selection). With the iteration of this process (generational loop), the solutions become better and better, and approach optimal values closer and closer (evolution). Unlike single-objective optimization, when dealing with multiple conflicting objectives, the ranking mechanism of EA needs to be designed because it is no longer obvious which solutions are better or worse than others. Moreover, to present a wide variety of trade–off solutions, diversity maintenance in the population is required.

Due to the historical development, one distinguishes different methods of evolutionary computing to: genetic algorithms (GA), evolution strategies (ES), evolutionary programming (EP), and genetic programming (GP) [5]. The major technical difference between them is the preferred representation. For example, GAs work better at binary representation, ES concentrate on real-valued representation, EP focuses on finite state machines, and GP relies on tree structured representations in order to evolve mathematical expressions. Today, these representatives are converging and the distinction between them is getting vague. It was suggested by Bäck to unify these branches under the common term “evolutionary algorithms”. Therefore, EAs will be used in this thesis as the general term.

This thesis focuses on the application of scheduling optimization which is a typical and important branch of optimization problem. Thinking of many tasks, such as production tasks, maintenance tasks or service tasks, how to allocate these tasks to the workers or machines, for example, when to perform a task and what is the execution sequence, is usually an NP-hard problem, especially when multiple objectives are pursued, like resource consumption, completion time, economic cost, etc. The features of scheduling optimization make it a meaningful and interesting research topic. This thesis discusses two types of scheduling problems: flexible job shop scheduling and dynamic prediction-based maintenance scheduling problem. The flexible job shop scheduling problem is one of the best known combinatorial optimization problems.

It is realistic for modeling a wide range of real-life applications because it captures key features of modern manufacturing and service systems. Dynamic prediction-based maintenance scheduling problem comes from a real world application that was studied in this PhD thesis. Its goal is to optimize the maintenance schedule based on the predicted life-span of components and the condition of available workshops. Especially, the process of optimizing the schedule needs to be performed in a rolling horizon fashion.

1.2 Research Questions

To thoroughly explore MOEA and apply them in the domain of scheduling optimization, the following research questions are investigated in this thesis.

RQ1 (Chapter 3) How can an MOEA be developed that generates uniformly distributed sets on the Pareto front regardless of the shape of the Pareto front?

Many MOEAs have been suggested in the literature since the first real implementation of an MOEA in 1984 [91]. But none of these algorithms is perfect and can behave well in all MOO problems. One special open challenge is to devise MOEA that are invariant to the shape of the Pareto front [40, 64]. In this thesis, we want to therefore study methods and techniques which can generate uniformly distributed sets on the Pareto front regardless of the shape of the Pareto front.

RQ2 (Chapters 3) How can the performance of MOEAs be improved generally?

MOEAs need to consider convergence and diversity properties of the obtained solution set. For this reason, most MOEAs have two design criteria: to find a solution set, the solutions of which are close to the Pareto front and also well-spread across the Pareto front. Although it can be non-trivial, we study whether there can be a method which can improve MOEA in general with respect to these design criteria as compared to existing EAs.

RQ3 (Chapter 4) Instead of the whole Pareto front, how can preferred solutions which are of real interest to the decision maker (DM) be obtained?

Over the past decade, the research on preference-based multi-objective optimization has been strongly motivated by real-world applications. In reality, the DM is often not interested in discovering the whole Pareto front, but rather in approximating the portion of the front that best matches his/her preferences.

Incorporating the preference information in MOEA allows the algorithm to focus on the part of the objective space, which is most interesting to the DM. The question arises of how the algorithms can adapt the search to focus on the interesting regions.

RQ4 (Chapter 5) How to solve multi-objective flexible job shop scheduling optimization problems?

Due to the NP-hard characteristic of flexible job shop scheduling problems, it is difficult to propose exact algorithms with satisfactory running time for them. Moreover, the consideration of multiple objectives further complicates the situation. However, research work on these problems is essential, and can be used as the foundation for solving our real-world scheduling optimization problems. This thesis seeks to design MOEAs to tackle these important combinatorial optimisation problems.

RQ5 (Chapter 5) How to represent and solve a real-world scheduling optimization problem in the EA world?

To solve a problem with an MOEA, the representation of the problem for evolutionary computation is an important step and defining a good representation can have a substantial impact on the performance of MOEAs. Solving an MOO problem involves the formulation of the real-world problem, the choice of the data structure used for representing solutions as the chromosomes and the genetic operators, and also many other problem specific issues.

RQ6 (Chapter 6) How to apply and adapt the developed algorithms to the dynamic prediction-based maintenance scheduling optimization problem?

In the process of generating the prediction-based maintenance schedule periodically, the time-varying characteristics, i.e., the dynamics of the optimization, are considered to find the optimal solutions at different moments. This should also lead to dynamic optimization.

1.3 Outline

This thesis is structured as follows. The content, corresponding research question(s) and publication(s) are introduced for each chapter.

Chapter 2 gives a brief introduction on optimization, multi-objective optimization, evolutionary computation. Especially, different order relations for multi-objective opti-

mization have been discussed. The work presented in this chapter has been (partially) published in:

1) André Deutz, Michael Emmerich and Yali Wang. Many-Criteria Dominance Relations. In Dimo Brockhoff, Michael Emmerich, Boris Naujoks, and Robin Purshouse, editors, Many-Criteria Optimization and Decision Analysis, Springer, Natural Computing Series (2022). [33]

In Chapter 3, to answer RQ1 and RQ2, first, a diversity-indicator based multi-objective evolutionary algorithm is proposed. After that, the performance of the standard Pareto dominance relation is improved to enhance the behaviour of general multi-objective evolutionary algorithms [114]. The work presented in this chapter has been (partially) published in:

2) Yali Wang, Michael Emmerich, André Deutz, and Thomas Bäck. Diversity-indicator Based Multi-Objective Evolutionary Algorithm: DI-MOEA. In International Conference on Evolutionary Multi-Criterion Optimization, pages 346–358. Springer, 2019. [116].

3) Yali Wang, André Deutz, Thomas Bäck, and Michael Emmerich. Edge-Rotated Cone Orders in Multi-Objective Evolutionary Algorithms for Improved Convergence and Preference Articulation. In 2020 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 165-172. IEEE, 2020. [115].

4) Yali Wang, André Deutz, Thomas Bäck, and Michael Emmerich. Improving Many-Objective Evolutionary Algorithms by Means of Edge-rotated Cones. In International Conference on Parallel Problem Solving from Nature, pages 313–326. Springer, 2020. [114].

Chapter 4 extends the basic static multi-objective optimization to preference-based multi-objective optimization. The corresponding preference-based multi-objective evolutionary algorithms are proposed with the aim of answering RQ3 and have been published in:

5) Yali Wang, Longmei Li, Kaifeng Yang, and Michael Emmerich. A New Approach to Target Region Based Multiobjective Evolutionary Algorithms. In 2017 IEEE Congress on Evolutionary Computation (CEC), pages 1757–1764. IEEE, 2017. [117].

6) Longmei Li, Yali Wang, Heike Trautmann, Ning Jing, and Michael Emmerich. Multiobjective Evolutionary Algorithms Based on Target Region Preferences. Swarm and Evolutionary Computation, 40:196–215, 2018. [73].

7) Yali Wang, Steffen Limmer, Markus Olhofer, Michael Emmerich, and Thomas Bäck. Automatic Preference Based Multi-Objective Evolutionary Algorithm on Vehicle Fleet Maintenance Scheduling Optimization. Swarm and Evolutionary Computation, p.100933, 2021. [118].

In Chapter 5 the flexible job shop scheduling problem is introduced due to its practical importance. A multi-objective evolutionary approach is developed to address multi-objective flexible job shop scheduling problems with three considered objectives: minimizing makespan, total workload and critical workload [121], this answers RQ4. After the analysis of the multi-objective flexible job shop scheduling problem, the problem of vehicle fleet maintenance scheduling optimization is formulated for our real-world application. On the basis of this formulation, the representation and corresponding algorithm are developed to solve this multi-objective optimization problem [119], this answers RQ5. The work presented in this chapter has been (partially) published in:

8) Yali Wang, Steffen Limmer, Markus Olhofer, Michael Emmerich, and Thomas Bäck. Vehicle Fleet Maintenance Scheduling Optimization by Multi-Objective Evolutionary Algorithms. In 2019 IEEE Congress on Evolutionary Computation (CEC), pages 442–449. IEEE, 2019. [119].

9) Yali Wang, Bas van Stein, Thomas Bäck, and Michael Emmerich. Improving NSGA-III for Flexible Job Shop Scheduling Using Automatic Configuration, Smart Initialization and Local Search. In Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, pages 181–182, 2020. [121].

10) Yali Wang, Bas van Stein, Thomas Bäck, and Michael Emmerich. A Tailored NSGA-III for Multi-Objective Flexible Job Shop Scheduling. In 2020 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 2746-2753, IEEE, 2020. [122].

Chapter 6 gives attention to RQ6, it looks at the performance of proposed (preference-based) multi-objective evolutionary algorithms on our real-world vehicle fleet maintenance scheduling optimization problem. Especially, to apply the dynamic algorithm on this application problem, the scenario of a taxi fleet is simulated, the maintenance

schedule is based on the prediction of remaining useful life (RUL) of components in each car. The dynamic algorithm is used to update the maintenance schedule of the vehicles based on the predicted RUL which keeps changing with the execution of driving tasks. Moreover, an empirical comparison of different maintenance strategies is presented. The work presented in this chapter has been (partially) published in:

11) Yali Wang, Steffen Limmer, Duc Van Nguyen, Markus Olhofer, Thomas Bäck and Michael Emmerich. Optimizing the Maintenance Schedule for A Vehicle Fleet: A Simulation-based Case Study. Engineering optimization, pp. 1-14. 2021. [120].

Chapter 7 concludes the thesis and closes with a discussion of future work.

Chapter 2

Preliminaries

This chapter provides an introduction to critical basics of optimization, multi-objective optimization, multi-objective evolutionary algorithm, and the discussion of order relations for multi-objective optimization.

2.1 Optimization

An optimization problem is the problem of finding the best solution from all feasible solutions. All sorts of optimization problems arise in different disciplines from mathematics, computer science to engineering and economics, and so on.

2.1.1 Applications

Some applications listed below can give a rough impression on optimization problems. It is worthy noting that this is just a drop in the ocean, the real-world optimization problems go far beyond these disciplines and applications.

Agriculture: managing river basins to satisfy urban and agricultural consumptive demand, also in-stream environmental demands [126].

Architecture: designing a building with respect to thermal comfort, energy efficiency, and construction cost criteria [58].

Aviation: planning airport construction to minimize the cost of all the items influenced by the site layout; maximize the safety of airport operations dur-

ing construction; reduce construction-related security breaches; and improve the safety of construction operations [69].

Aerospace: designing satellite orbits to minimize the spatial resolution requirement (at-nadir resolution and off-nadir resolution) and temporal resolution requirement (the repeat cycle and the revisit time) [98].

Chemistry: finding desirable molecule drug which can improve solubility, metabolic stability, cell permeability, and with reduced side effects. [125].

Engineering: designing hydraulic presser to maximize the nominal pressure rating and fully loaded power while minimizing the oil injection volume [123].

Environment: designing marine protected area networks to maximize network effectiveness, species persistence, and minimize cost of protection [47].

Investment: choosing an optimal set of assets in order to minimize the risk and maximize the profit of the investment [101].

Machine learning: assisting machine learning algorithms to optimize their hyperparameters, selecting models to minimize model complexity and maximize classification accuracy [65].

Manufacturing: making an efficient supply chain plan to minimize total losses of supply chain including production cost, hiring, firing and training cost, raw material and end product inventory holding cost, transportation and shortage cost, simultaneously, minimize the sum of the maximum amount of shortages among the customers' zones in all periods to improve customer satisfaction [78].

Medical: searching for new therapeutic drugs to maximize the potency of the drug, at the same time, minimize synthesis costs and unwanted side effects [108].

Scheduling: determining the vehicle routing to minimize the total distance traveled, the total time required, the total tour cost, and the fleet size, and maximizing the quality of the service and the profit collected [66].

All these problems have in common that a software/ search algorithm framework which can support human decision makers in solving such problems is desirable due to the large number of alternative solutions.

2.1.2 Mathematical Definition

The goal of the optimization process is to find the values of decision variables that result in a maximum or minimum of a function called the objective function. In mathematical terms, optimization problems can be formulated as:

$$\begin{aligned} \text{Minimize} \quad & f(\mathbf{x}) && (2.1)\\ \text{Subject to} \quad & g_i(\mathbf{x}) = 0, \qquad i = 1, \cdots, p && (2.2)\\ & h_j(\mathbf{x}) \geq 0, \qquad j = 1, \cdots, q && (2.3)\\ & \mathbf{x} \in \mathcal{X}. && (2.4) \end{aligned}$$

Here $\mathbf{x}$ is the set of decision variables. The decision variables are the numerical quantities for which values are under our control and are to be chosen to find an optimal solution. The decision variables consist of independent variables, a vector x containing n decision variables can be represented by: $x = (x_1, x_2...., x_n)^T$. Decision variables may have continuous values which can take on any value in a specified interval or discrete values which are restricted to a specified interval of integers. This leads to continuous optimization and discrete optimization problems. In theory, continuous optimization problems tend to be easier to solve than discrete optimization problems because the information about points in a neighborhood of one decision variable can be deduced more smoothly.

The constraints, i.e., equality constraints in Eq. (2.2) and inequality constraints in Eq. (2.3), are also functions of the decision variables. Their values decide which solutions are feasible. Some of the optimization problems do not have any constraints and they are therefore called unconstrained optimization problems. Sometimes, only simple constraints on the range of the input variables are given by means of intervals. These problems are usually referred to as box-constraints problems. Constrained optimization problems can be reformulated to unconstrained optimization problems in which the constraints are replaced by a penalty term in the objective function.

The optimization process is to find the values of decision variables that result in a maximum or minimum of the objective function $f(\mathbf{x})$, i.e., Eq. (2.1). Without loss of generality, the objective function is to be minimized in this work. In other words, the objective function is a measure to compare alternative solutions. The optimization problems having a single objective function are single-objective optimization problems. But in the real-world, the optimization problems with multiple objective functions, i.e.,

multi-objective optimization problems, are more common.

2.2 Multi-objective Optimization

A multi-objective optimization problem is an optimization problem with more than one objective function to be minimized. That is to say, instead of one single objective function $f(\mathbf{x})$, multiple objective functions, $f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})$, are optimized simultaneously. Here, m ($m \geq 2$) indicates the number of objectives.

In multi-objective optimization, the objectives are usually conflicting with each other. Therefore, there does not typically exist a feasible solution that minimizes all objective functions simultaneously; and the trade-off among different objectives gives rise to a set of potential compromise solutions. A minimal requirement for a compromise solution is that it should be a Pareto optimal solution. Pareto optimal solutions are solutions that cannot be improved in any of the objectives without deteriorating at least one of the other objectives.

2.2.1 Pareto Optimal and Non-dominated Solutions

The solutions are evaluated by the objective functions which represent a mapping from the *decision space* to the *objective space*. For an optimization problem, the decision space $\mathcal{X}$ comprises all candidate solutions. When the problem has m objectives, an m-dimensional Euclidean space forms its objective space in which objective function vectors coexist and where each coordinate axis corresponds to one objective. For each solution in the decision space, there is a point in the objective space. At the same time, multiple solutions in the decision space may be projected onto the same point in the objective space. A relative comparison between solutions can be achieved by the dominance relation.

Definition 2.1 (Dominance (Objective Space))**.** Given two solutions in the objective space, that is $\mathbf{y}^{(1)} \in \mathbb{R}^m$ and $\mathbf{y}^{(2)} \in \mathbb{R}^m$, solution $\mathbf{y}^{(1)}$ is said to dominate solution $\mathbf{y}^{(2)}$ if and only if $\forall\, i \in \{1, \ldots, m\} : y_i^{(1)} \leq y_i^{(2)}$ and $\exists\, j \in \{1, \ldots, m\} : y_j^{(1)} < y_j^{(2)}$, in symbols $\mathbf{y}^{(1)} \prec \mathbf{y}^{(2)}$.

Definition 2.2 (Dominance (Decision Space))**.** Given two solutions $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ in the decision space, then solution $\mathbf{x}^{(1)}$ is said to dominate $\mathbf{x}^{(2)}$ if and only if $\forall\, i \in \{1, \ldots, m\} : f_i(\mathbf{x}^{(1)}) \leq f_i(\mathbf{x}^{(2)})$ and $\exists\, j \in \{1, \ldots, m\} : f_j(\mathbf{x}^{(1)}) < f_j(\mathbf{x}^{(2)})$.

Definition 2.3 (Incomparability (objective space))**.** Solution $\mathbf{y}^{(1)}$ is said to be incomparable to solution $\mathbf{y}^{(2)}$ if and only if $\mathbf{y}^{(1)} \neq \mathbf{y}^{(2)}$, $\mathbf{y}^{(1)} \not\prec \mathbf{y}^{(2)}$ and $\mathbf{y}^{(2)} \not\prec \mathbf{y}^{(1)}$, in symbols $\mathbf{y}^{(1)} \| \mathbf{y}^{(2)}$.

Definition 2.4 (Indifference and incomparability (decision space))**.** Solution $\mathbf{x}^{(1)}$ is said to be indifferent to solution $\mathbf{x}^{(2)}$ ($\mathbf{x}^{(1)} \sim \mathbf{x}^{(2)}$) if and only if $\mathbf{f}(\mathbf{x}^{(1)}) = \mathbf{f}(\mathbf{x}^{(2)})$. Here $\mathbf{x}^{(1)} \sim \mathbf{x}^{(2)} \nRightarrow \mathbf{x}^{(1)} = \mathbf{x}^{(2)}$. Solution $\mathbf{x}^{(1)}$ is said to be incomparable to solution $\mathbf{x}^{(2)}$ ($\mathbf{x}^{(1)} \| \mathbf{x}^{(2)}$) if and only if $\mathbf{f}(\mathbf{x}^{(1)}) \| \mathbf{f}(\mathbf{x}^{(2)})$.

Definition 2.5 (Pareto Optimal and Non-dominated Solution)**.** In decision space, a decision vector x^* is a Pareto optimal solution if there does not exist a decision vector x ($x \neq x^*$) that dominates it, i.e., $\nexists\, x \in \mathcal{X} : \mathbf{f}(x) \prec \mathbf{f}(x^*)$. If x^* is Pareto optimal, $\mathbf{f}(x^*)$ is called a non-dominated point (solution).

The set of all Pareto optimal vectors in the decision space is referred to as the *Pareto optimal set* or efficient set; and the image of the Pareto optimal set in the objective space is referred to as the *Pareto Front*.

2.2.2 Pareto Front Geometry

Figure 2.1 shows several typical types of the Pareto fronts: convex, concave, neither convex nor concave and disconnected Pareto fronts. The Pareto front in the bottom left image consists of convex and concave parts.

The Pareto front can be represented by a function, $u : \mathbb{R}^{m-1} \to \mathbb{R}$ and m the number of objectives. A function is said to be convex if it satisfies the following equation [11]:

$$u(\theta x + (1-\theta)y) \leq \theta u(x) + (1-\theta)u(y) \tag{2.5}$$

with x, y in the domain of g and $\theta \in [0, 1]$. In words, it means that the line between $(x, u(x))$ and $(y, u(y))$ is above the graph between x to y. Accordingly, a function u is concave if $-u$ is convex.

Multi-objective optimization problems with more than three objectives are called many-objective optimization problems [41] and they form a special and important case of multi-objective optimization problems. An increase in the number of objectives causes a large portion of solutions to become non-dominated. This leads to the difficulty in searching for Pareto optimal solutions, meanwhile, a huge number of solutions may be needed to estimate the entire Pareto front. Many-objective optimization gives rise to a new set of challenges [3, 61]. The need for tackling many-objective

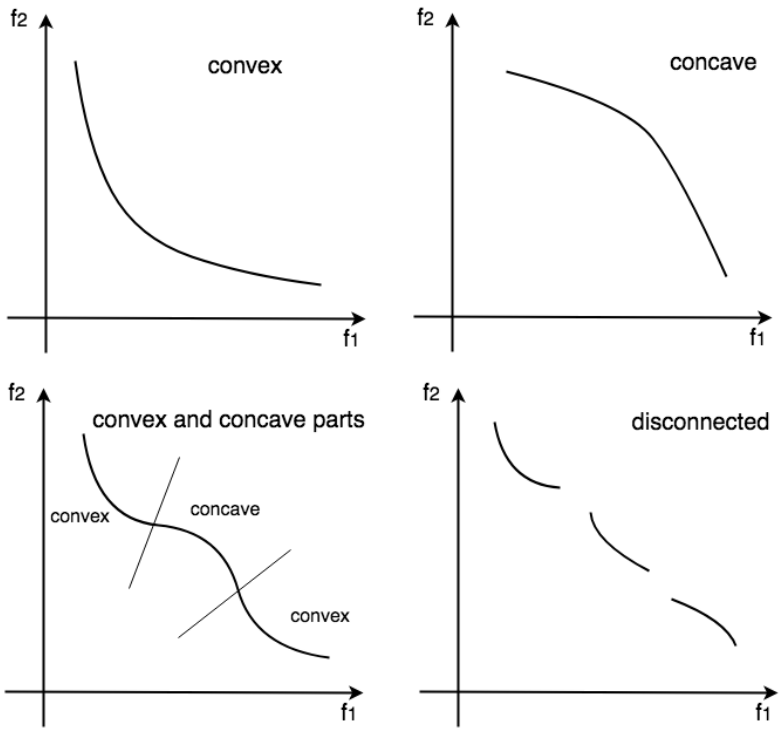


Figure 2.1: Examples of Pareto fronts.

problems became evident recently because it would allow us to solve more complex real world problems.

2.3 Multi-objective Evolutionary Algorithm

EA has been successfully adapted to dealing with multi-objective optimization and these specialised algorithms are called multi-objective evolutionary algorithm (MOEA) or, sometimes, also evolutionary multi-objective optimization algorithm (EMOA). The optimization mechanism of MOEA is very similar to EA, such as population-based search and information exchange among solutions (individuals). One special characteristic of MOEA is the use of the dominance relationship to assign the fitness to each solution in the population. In detail, at each iteration, the objective values are calculated for each individual and then used to determine the relationship of dominance in the population in order to choose a potentially better solution for the creation of the offspring population. At the same time, the ability to maintain diversity within a

population of individuals is another key component of MOEA.

2.3.1 Classification

Classical *Pareto dominance-based MOEAs*, such as NSGA-II [29], use Pareto dominance as a first ranking criterion and use a second ranking criterion to maintain and increase diversity. Pareto dominance-based MOEAs have been a mainstream class for a long time in the field of evolutionary multi-objective optimization (EMO). They are very efficient on multi-objective optimization problems with two or three objectives. However, their performance degrades significantly on many-objective optimization problems due to their ineffectiveness in distinguishing the quality of solutions when the number of objectives becomes large.

As the performance assessment of MOEAs reached a mature stage, performance measures (indicators) on the quality of Pareto front approximations were adopted to search for solutions. These indicators capture both convergence and diversity in a single value. Additionally for Pareto compliant indicators, it can be shown that they obtain their maximum in a diversified set of solutions on the Pareto front. In general, *indicator-based MOEAs* (IBEA) [139], such as SMS-EMOA [9] and R2-EMOA [104], have strong theoretical support. However, the commonly used performance indicators lead to a convergence in distribution with a high density on the boundary of the Pareto front, as well as on knee regions [9].

Decomposition is a search paradigm that was originally applied by EMO two decades ago [53] and recently regained prominence from the MOEA/D framework [135] and NSGA-III [26]. *Decomposition-based MOEAs* transform the original multi-objective problem into simpler, single-objective subproblems by means of scalarizations with different weights or reference vectors, therefore they can converge to a well defined, diverse set. However, the central issue in decomposition-based methods is how to select a set of weighting vectors that can provide a well distributed set of Pareto optimal points, given that the location and shape of the Pareto front are unknown a priori. Moreover, the number of weights required to sample a Pareto front with a sufficient resolution suffers an exponential growth from the objective space dimension [51].

2.3.2 Quality Measures

The goal of solving a multi-objective optimization problem is to approximate or compute all or a representative set of Pareto optimal solutions. The quality of the approx-

imation sets is evaluated to compare different algorithms when solving multi-objective optimization problems. The major characteristics for evaluating fronts include convergence and diversity, and the diversity consists of two aspects: distribution and spread. A good multi-objective optimization algorithm is required to generate solutions that are close to the Pareto front, well distributed and spread widely over the entire Pareto front at minimum computational cost.

Among numerous metrics, we choose the following ones to evaluate the quality of the obtained Pareto front approximation, which are also performance metrics commonly used by the evolutionary multi-objective evolutionary community.

Hypervolume Indicator

The hypervolume (HV) indicator [141], previously also known as S metric [137] or Lebesgue measure [72], is one of the most popular indicators for multi-objective optimization. It has been proven that the maximization of this performance measure is equivalent to finding the Pareto front [45] provided it is a finite set. In other cases, it leads to a well distributed approximation of the Pareto optimal set if the number of objectives is small (say ≤ 4). The HV indicator is an unary metric which evaluates one approximation set, and it measures the volume of the objective space jointly dominated by the Pareto front approximation, relative to a reference point $r \in \mathbb{R}^m$.

Definition 2.6 (Hypervolume Indicator)**.**

$$HV(Y, r) = \lambda_m(\bigcup_{y \in Y} [y, r]) \tag{2.6}$$

here λ_m denotes the Lebesgue measure on $\mathbb{R}^m$, with m being the number of objective functions.

The HV indicator considers both convergence and diversity. The HV indicator, and its variations, are the only known unary indicator to be strictly monotonic [138], i.e., if an Pareto front approximation A strictly dominates another Pareto front approximation B, $HV(A, r) > HV(B, r)$. Therefore, the HV indicator is said to be Pareto compliant. The major disadvantage of the HV indicator is calculating hypervolume exactly is NP-hard and exponential in the number of objectives [8]. For a small constant number of objectives, however, there exists fast computation algorithms.

To evaluate the fitness value of each solution in the Pareto front approximation, the hypervolume contribution can be used. The hypervolume contribution of a point $y \in Y$ is defined as the difference between the hypervolume indicator of Y and the

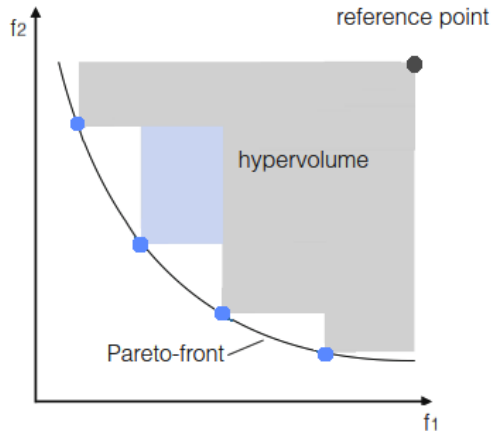


Figure 2.2: Illustration of the hypervolume indicator and hypervolume contribution for a bi-objective problem.

hypervolume indicator of $Y \setminus \{y\}$. Figure 2.2 shows the hypervolume indicator and hypervolume contribution for a bi-objective problem. The size of the blue part in the dominated region is the hypervolume contribution of one solution.

Inverted generational distance (IGD)

IGD [17] has been widely considered as a reliable performance indicator. It is complementary to generational distance (GD). Both IGD and GD use the true Pareto front as a reference set; if the true Pareto front is unknown, the reference set is usually a combination of the non-dominated points of several approximate fronts.

They are given by the following formulas:

Definition 2.7 (IGD Metric)**.**

$$IGD(Y,P) = \frac{1}{|P|}(\sum_{i=1}^{|P|} d(r_i,Y)^2)^{\frac{1}{2}} \tag{2.7}$$

where $|P|$ is the number of points in the reference front P and Y is the obtained Pareto front approximation; $d(r_i, Y)$ denotes the minimum Euclidean distance between a point in the reference front and the solutions in the Pareto front approximation Y.

Definition 2.8 (GD Metric)**.**

$$GD(Y,P) = \frac{1}{|Y|}(\sum_{i=1}^{|Y|} d(y_i,P)^2)^{\frac{1}{2}} \tag{2.8}$$

where $|Y|$ is the number of solutions in the Pareto set approximation Y and P is the reference front; $d(y_i,P)$ denotes the minimum Euclidean distance between solution y_i and the points in the reference front P.

It can be seen that $IGD(Y,P) = GD(P,Y)$, but there is significant difference between IGD and GD. IGD uses the reference front as reference and calculates the distance of each point from the reference front to the Pareto front approximation, which means no part of the reference front (or "true" Pareto front) can be missed. If sufficient members of the reference front are known, IGD could measure both the diversity and the convergence of a Pareto front approximation. The smaller the value of this metric, the closer the obtained front is to the true Pareto front. IGD is efficient to compute in low dimensions of the objective space, but it requires the knowledge of the Pareto front.

Beyond the metrics introduced here, the interested reader is referred to [89] and [4] for a general overview and introduction to performance metrics in multi-objective optimization.

2.3.3 Dominance Relations

The concept of Pareto dominance is of fundamental importance to multi-objective optimization. We use this section to discuss further the Pareto dominance and introduce other dominance relations. Let us first review the basic concept of binary relations and some general properties that binary relations can potentially have (see also [34]).

Definition 2.9 (Binary relation)**.** A binary relation R over a set X is defined as a set of pairs of elements of X, that is, a subset of $X \times X = \{(x,y)|\ x,y \in X\}$. The statement $(x,y) \in R$ reads "x is R-related to y" and is denoted by xRy.

Definition 2.10 (Properties of Binary Relations)**.** Given a set X, a binary relation R is said to be

- reflexive, if and only if $\forall x \in X : (x,x) \in R$.
- irreflexive, if and only if $\forall x \in X, (x,x) \notin R$.

- symmetric, if and only if $\forall x \in X, \forall y \in X : (x, y) \in R \Leftrightarrow (y, x) \in R$.
- asymmetric, if and only if $\forall x \in X, \forall y \in X : (x, y) \in R \Rightarrow (y, x) \notin R$.
- anti-symmetric, if and only if $\forall x \in X, \forall y \in X : (x, y) \in R \wedge (y, x) \in R \Rightarrow x = y$.
- transitive, if and only if $\forall x \in X, \forall y \in X, \forall z \in X : (x, y) \in R \wedge (y, z) \in R \Rightarrow (x, z) \in R$.

Based on these properties, different types of orders then can be defined.

Definition 2.11 (Pre-order, Partial Order, Strict Partial Order)**.** A binary relation R is

- pre-order (aka quasi-order), if and only if it is transitive and reflexive.
- partial order, if and only if it is an antisymmetric pre-order.
- strict partial order, if and only if it is irreflexive and transitive.

Note that a strict partial order is necessarily asymmetric (and therefore also anti-symmetric). Next, the definition of Pareto order is given in the objective space $\mathbb{R}^m$ and it can be viewed as a cone order from a geometrical perspective, as will be shown later on.

Recall Definition 2.1 introduced the concept of dominance in the objective space or Pareto dominance. In this section, to distinguish it from other orders, we denote it with $\prec_{Pareto}$. The Pareto order $\prec_{Pareto}$ is a strict partial order defined in the objective space, i.e., the m-dimensional Euclidean space $\mathbb{R}^m$ with the objective function values being the coordinate axes. It allows a comparison between (some) pairs of feasible solutions in the objective space. Moreover, it is a transitive relation and as in the more general case of a pre-order, minimal elements and maximal elements are defined. In comparison to the more general pre-order, a partial order relation also is constrained by the anti-symmetry axiom, which implicates that indifference falls together with incomparability. The concept of Pareto dominance implies that, for a solution to dominate another one, it should not be worse in any objective and must be strictly better in at least one objective. The Pareto order is a special case of a cone order, which are a family of partial orders defined on vector spaces.

Definition 2.12 (Non-trivial Cone)**.** A set $\mathcal{C} \subset \mathbb{R}^m$ with $\emptyset \neq \mathcal{C} \neq \mathbb{R}^m$ is called a non-trivial cone, if and only if $\forall \alpha \in \mathbb{R}, \alpha > 0, \forall c \in \mathcal{C} : \alpha c \in \mathcal{C}$.

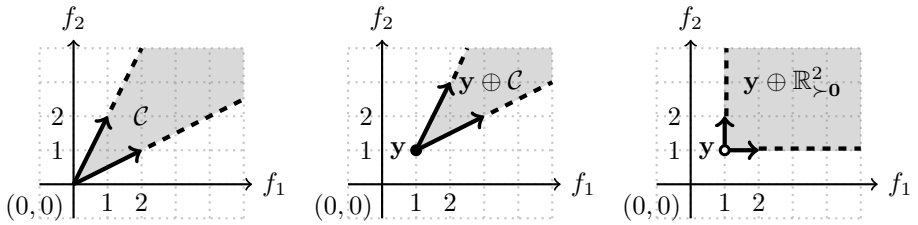


Figure 2.3: Example of a polyhedral cone $\mathcal{C}$ generated by $(2,1)$ and $(1,2)$ (left), Minkowski sum of a singleton $\{\mathbf{y}\} = \{(1,1)\}$ and $\mathcal{C}$ (middle), and Minkowski sum of $\{\mathbf{y}\}$ and the cone $\mathbb{R}^2_{\succ \mathbf{0}}$. The cone $\mathbb{R}^2_{\succ \mathbf{0}}$ is equal to the non-negative quadrant minus $\{(0,0)\}$. (cf. [36])

Definition 2.13 (Minkowski Sum)**.** The Minkowski sum (aka algebraic sum) of two sets $A \in \mathbb{R}^m$ and $B \in \mathbb{R}^m$ is defined as $A \oplus B := \{a + b \,|\, a \in A \wedge b \in B\}$. Moreover we define $\alpha A = \{\alpha a |\, a \in A\}$.

Figure 2.3 gives an illustration and examples of Minkowski sums, also refer to [36].

Definition 2.14 (Binary Relation Associated to Cone)**.** Given a cone $\mathcal{C}$, the binary relation associated to this cone, notation $R_{\mathcal{C}}$, is defined as follows: $\forall \mathbf{x} \in \mathbb{R}^m, \forall \mathbf{y} \in \mathbb{R}^m : (\mathbf{x}, \mathbf{y}) \in R_{\mathcal{C}}$ if and only if $\mathbf{y} \in \{\mathbf{x}\} \oplus \mathcal{C}$.

For any cone $\mathcal{C}$, the associated binary relation $R_{\mathcal{C}}$ is translation invariant (i.e., if $\forall \mathbf{u} \in \mathbb{R}^m : (\mathbf{x}, \mathbf{y}) \in R_{\mathcal{C}} \Rightarrow (\mathbf{x} + \mathbf{u}, \mathbf{y} + \mathbf{u}) \in R_{\mathcal{C}}$) and multiplication invariant by any positive real (i.e., $\forall \alpha > 0 : (\mathbf{x}, \mathbf{y}) \in R_{\mathcal{C}} \Rightarrow (\alpha \mathbf{x}, \alpha \mathbf{y}) \in R_{\mathcal{C}}$). At the same time, given a binary relation R which is translation invariant and multiplication invariant by any positive real, the set $\mathcal{C}_R := \{\mathbf{y} - \mathbf{x} \,|\, (\mathbf{x}, \mathbf{y}) \in R\}$ is a cone. The above two operations are inverses of each other, i.e., starting from a cone $\mathcal{C}$, a binary relation $R_{\mathcal{C}}$ which is translation invariant and multiplication invariant by any positive real can be associated to it; starting from a binary relation R which is translation invariant and multiplication invariant by any positive real, a cone $\mathcal{C}_R$ can be obtained. It can be seen there is a natural one to one correspondence between cones and binary relations on $\mathbb{R}^m$ which are translation invariant and multiplication invariant by positive reals (see also [80]).

We restrict our attention to relations which are translation invariant and positive multiplication invariant to get this bijection between cones and relations. Note if a translation invariant and positive multiplication invariant relation R is such that $\emptyset \neq R \neq \mathbb{R}^m \times \mathbb{R}^m$, the associated cone $\mathcal{C}_R$ is non-trivial. Relations associated to non-trivial cones are non-empty and not equal to $\mathbb{R}^m \times \mathbb{R}^m$ as well.

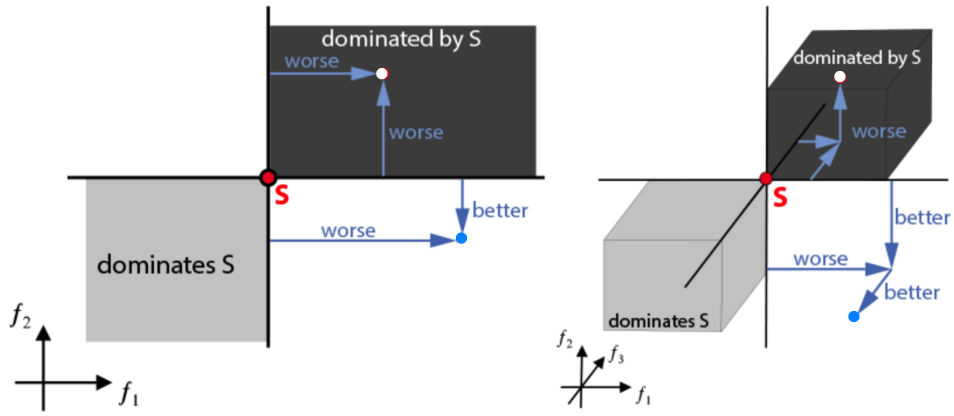


Figure 2.4: Pareto dominance relation and cone.

The reason to view the Pareto dominance relation or Pareto order as derived from a cone is that it gives the opportunity to study this order more geometrically. Figure 2.4 illustrates geometrically the Pareto order and Pareto cone in two and three dimensional spaces. In the objective space, each solution can be located in the vector space based on its objective values. One solution S dominates another solution if all objective values of S are better than the corresponding objective values of another solution, or the objective values of S are equal to but at least one better than the corresponding objective values of another solution. In other words, if a solution is located in the dark gray cone area of S (including the boundaries), it is dominated by S. Similarly, solutions in the light gray cone area of S (including the boundaries) dominate S; solutions in other areas, for examples, the blue points, are incomparable to S.

It can be seen that, in two dimensional space, the cone which associates the Pareto order is the positive quadrant and the angle between two edges of the cone is 90°. Similarly, the Pareto cone in three dimensional space is the positive octant. In the following, the detailed definitions are given.

Definition 2.15. Let m be a natural number bigger or equal to 1, the non-negative orthant of $\mathbb{R}^m$, denoted by $\mathbb{R}^m_{\geq 0}$, is the set of all elements in $\mathbb{R}^m$ whose coordinates are non-negative. Furthermore, the zero-dominated orthant, denoted by $\mathbb{R}^m_{\succ 0}$, is the set $\mathbb{R}^m_{\geq 0} \setminus \{0\}$ with 0 denoting the m dimensional vector $(0, \ldots, 0)$. Analogously, the non-positive orthant of $\mathbb{R}^m$, denoted by $\mathbb{R}_{\leq 0}$, is the set of elements in $\mathbb{R}^m$ whose coordinates are non-positive. Furthermore, the set of elements in $\mathbb{R}^m$ which dominate the zero vector 0, denoted by $\mathbb{R}^m_{\prec 0}$, is the set $\mathbb{R}^m_{\leq 0} \setminus \{0\}$. The set of positive reals is

denoted by $\mathbb{R}_{>0}$ and the set of non-negative reals is denoted by $\mathbb{R}_{\geq 0}$.

The Pareto order can be associated to cone $\mathbb{R}^m_{\succ 0}$, i.e., the Pareto order $\prec_{Pareto}$ on $\mathbb{R}^m$ is given by the cone order with cone $\mathbb{R}^m_{\succ 0}$. This cone is also referred to as the Pareto cone.

Definition 2.16 (Pointed cone and convex cone)**.** A cone $\mathcal{C}$ is pointed, if and only if $\mathcal{C} \cap -\mathcal{C} \subseteq \{\mathbf{0}\}$ where $-\mathcal{C} = \{-c \mid c \in \mathcal{C}\}$ and $\mathcal{C}$ is convex if and only if $\forall c_1 \in \mathcal{C}, c_2 \in \mathcal{C}, \forall \alpha$ such that $0 \leq \alpha \leq 1 : \alpha c_1 + (1-\alpha)c_2 \in \mathcal{C}$.

As $\mathbb{R}^m_{\succ 0}$ is a pointed [1] and convex cone and $0 \notin \mathbb{R}^m_{\succ \mathbf{0}}$, the associated binary relation is irreflexive, antisymmetric and transitive, therefore strict partial order.

The following concepts are useful in order to compare order relations.

Definition 2.17 (Order Extension)**.** An order relation R' on the set X is said to extend an order relation R on the set X if $R' \supseteq R$. In other words, for all $x, x' \in X$: xRx' implies $xR'x'$.

Definition 2.18 (Minimal Element)**.** A minimal element $x \in X$ in a (strictly) partially ordered set (X, R) is an element for which there does not exist an $x' \in X$ with $x'Rx$ and $x' \neq x$. (In case, the order R is a strict partial order, $x'Rx$ implies $x' \neq x$).

Let (X, R) and (X, R') denote two strict partially ordered sets. If R' is an order extension of R, this implies:

1. The set of incomparable pairs in (X, R') is a subset of the set of incomparable pairs in (X, R).

2. The set of minimal elements of (X, R) is a superset of minimal elements of (X, R').

The first statement is true because if a pair of elements is incomparable in R', such a pair will also be incomparable in the smaller relation R. The second statement is also clear: as $R \subseteq R'$ a minimal element with respect to R' is also a minimal element with respect to R. In other words, if an element is non-dominated in R' it cannot be dominated in a smaller relation.

In the context of many objective optimization, extensions of the Pareto dominance order play an important role, since they on the one hand preserve the important and

[1] Different definitions of pointed cone are given in literature, here we use the definition by Matthias Ehrgott [34].

somewhat essential solutions, because one would not like to consider a Pareto dominated solution to be optimal; on the other hand, by extending Pareto dominance relation, the number of minimal (non-dominated) elements and the number of incomparable solutions can be reduced. Thereby, if wisely chosen, they can provide a (partial) remedy to the curse of dimensionality that occurs if the number of objective functions (m) increases.

The study of cone-based dominance goes back to the early work of Yu [130] and Miettinen relates it in her book to proper Pareto dominance and bounded trade-off [77]. It has also been related to equity preferences in the work of Shukla [96]. Next, we introduce several recently proposed alternative dominance relations which utilizing the extensions of the Pareto dominance order. They are also called relaxed forms or loose versions of Pareto dominance. It can be shown that they are also special cases of cone orders.

α-dominance

Ikeda et al. proposed α-dominance [60] to deal with dominance resistant solutions, i.e., solutions that are extremely inferior to other solutions in at least one objective, but hardly dominated in the other objectives. The idea behind α-dominance is that a small detriment in one or perhaps several of the objectives is permitted if an attractive improvement in the other objective(s) is achieved.

The α-dominance uses linear trade-off functions to define the tolerance of dominance. The approach is to define the following m functions on $\mathbb{R}^m \times \mathbb{R}^m$ with codomain $\mathbb{R}$ using m^2 a-priori given real numbers α_{ij} $(i, j \in \{1, \cdots, m\})$ as follows:

$$g_i(\mathbf{y}, \mathbf{y}') := \sum_{j=1}^{m} \alpha_{ij}(\mathbf{y}'_j - \mathbf{y}_j).$$

In [60], $\alpha_{ij} \geq 0$ and $\alpha_{ii} = 1$. For each such m-tuple of such functions, a strict partial order on $\mathbb{R}^m$ can be defined, denoted by $\overset{\alpha}{\prec}$ as follows:

$$\mathbf{y} \overset{\alpha}{\prec} \mathbf{y}' :\Leftrightarrow \forall i \in \{1, \cdots, m\} : g_i(\mathbf{y}, \mathbf{y}') \geq 0 \text{ and } \exists k \in \{1, \cdots, m\} : g_k(\mathbf{y}, \mathbf{y}') > 0.$$

ϵ-dominance

Laumanns et al. proposed the concept of ϵ-dominance [71].

Definition 2.19 (ϵ-dominance)**.** Let $\mathbf{y}$, $\mathbf{y}' \in \mathbb{R}^m$ and $\epsilon \in \mathbb{R}$, with $\epsilon > 0$, then $\mathbf{y}$ is said to ϵ-dominate $\mathbf{y}'$ (denoted by $\mathbf{y} \prec_\epsilon \mathbf{y}'$) if and only if $\forall i \in \{1, ..., m\} : y_i - \epsilon \leq y'_i$.

Optimizing based on ϵ-dominance creates, for objective functions with bounded ranges, a finite set of non-dominated solutions that will be placed distant from each other. If the decision maker wants to maintain a set of maximally T non-dominated solutions and let K denote a constant such that $0 \leq f_i \leq K, \forall i \in \{1, \ldots, m\}$, then ϵ can be adjusted to $\epsilon = (K/T)^{1/(m-1)}$. An alleged disadvantage of ϵ-dominance is that certain regions of the Pareto front with steep trade-off are underrepresented.

Batista et al. [6] proposed cone ϵ-dominance to improve the way solutions distribute as compared to ϵ-dominance. For this they propose to use a polyhedral cone given by a $\mathrm{R}^{m \times m}$ generator matrix and exemplify their approach in two dimensions, whereas some concepts were introduced also in higher dimensions.

Control Dominance Area of Solutions (CDAS)

Sato et al. proposed an approach to control the dominance area of solutions (CDAS) [90]. In CDAS, the objective values are modified and the i-th objective value of x after modification is defined as: $\hat{f}_i(x) = \frac{r \cdot \sin(w_i + S_i \cdot \pi)}{\sin(S_i \cdot \pi)}$, where r is the norm of $f(x)$, w_i is the declination angle between $f(x)$ and the coordinate axis. The degree of expansion or contraction of the dominance area of solutions can be controlled by the user-defined parameter S, i.e., $\hat{f}_i(x) > f_i(x)$ when $S_i < 0.5$; in case of $S_i = 0.5$, $f_i(x)$ does not change; and when $S_i > 0.5$, $\hat{f}_i(x) < f_i(x)$. Depending on increasing or decreasing the parameter S, the dominance area of solutions expands or increases. Only in case all the $S_i \leq 1$, CDAS is an extension of the Pareto relation.

Angle dominance

Liu et al. defined angle dominance [75]. For each point $\mathbf{y}$, a point dependent "cone" is constructed as follows. For each i ($i = 1, \cdots, m$), a point $P^{(i)} := (0, \cdots, 0, p_i, 0, \cdots 0)^\top \in \mathbb{R}^m$ is introduced, all coordinates of $P^{(i)}$ are zero, except the i-th coordinate. The i-th coordinate is derived from the worst point and a parameter $k > 0$. The worst point w is the point for which its i-th coordinate is equal to $w_i := \sup\{f_i(x) \,|\, x \in \mathcal{X}\}$, where f_i is the i-th objective ($i = 1, \cdots, m$), and $\mathcal{X}$ is the search space. Using the parameter k one defines $P^{(i)} := (0, \cdots, 0, p_i = kw_i, 0, \cdots, 0)^\top$. The second ingredient used is the ideal point (or if needed the utopian point). Denote the ideal point by $\mathbf{z}^{\mathrm{ideal}}$. Then to a point $\mathbf{y}$, it associates m angles: $(\alpha_1, \cdots, \alpha_m)$. The cosine of α_i is equal to

$$\cos(\,\alpha_i\,) \;=\; \frac{(P^{(i)} - \mathbf{y}) \cdot (P^{(i)} - \mathbf{z}^{\mathrm{ideal}})}{|P^{(i)} - \mathbf{y}|\,|P^{(i)} - \mathbf{z}^{\mathrm{ideal}}|},$$

where in the numerator the inner product is used. The angle dominance relation is defined as follows.

Definition 2.20 (Angle Dominance)**.** $\mathbf{y} \prec_{\text{angle}} \mathbf{y}' :\Longleftrightarrow \forall i \in \{1, \cdots, m\} : \alpha_i \leq \alpha'_i$ and $\exists i \in \{1, \cdots, m\} : \alpha_i < \alpha'_i$, where α_i are the m angles associated to $\mathbf{y}$ and α'_i are m angles associated to $\mathbf{y}'$.

The authors show that given the premise that the parameter k is greater than 1, the angle dominance is irreflexive, asymmetric and transitive. Therefore, the angle dominance defines a strict partial order.

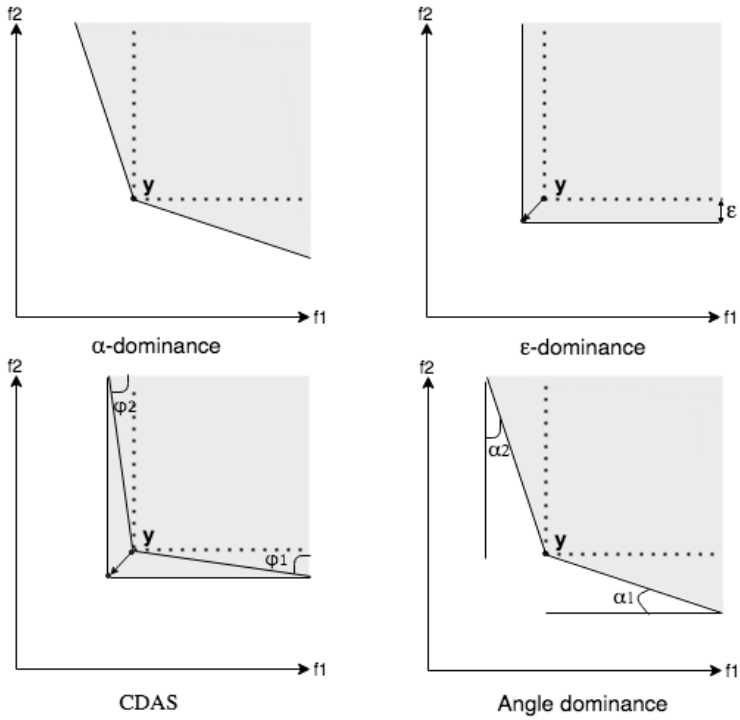


Figure 2.5: Dominance relations in a two-dimensional objective space.

Figure 2.5 illustrates these dominance relations in a two-dimensional objective space. The dotted lines indicate the space where solutions are Pareto dominated by the point $\mathbf{y}$. The gray areas show the dominated spaces by $\mathbf{y}$ based on α-dominance, ϵ-dominance, CDAS and angel dominance respectively. For the α-dominance (at the top left corner), the α-dominated area by $\mathbf{y}$ can be seen as expanding the angle of

Pareto cone and the degree of expansion is controlled by α. For ϵ-dominance (at the top right corner), the ϵ-dominated area by $\mathbf{y}$ can be seen as shifting the point $\mathbf{y}$ towards a position which Pareto dominates $\mathbf{y}$, therefore, the dominated area by $\mathbf{y}$ is expanded and the degree of expansion is controlled by ϵ. The dominated space by CDAS (at the bottom left corner) can also be seen as expanding the dominance area of $\mathbf{y}$ by translating Pareto cone. The degree of translation is decided by the parameter S because S controls φ ($\varphi_i = S_i \cdot \pi$). Angle dominance area is also obtained by expanding the angle of Pareto cone; the degree of expansion is decided by the parameter k and the worst point which controls α.

These dominance relations extend the Pareto dominance. Despite the difference in algorithm and controlling mechanisms, their eventual implementation is to allow a solution to dominate a larger space. At the same time, shifting the point to its dominating space which dominates it or opening the edges of its dominated space which is dominated by it, they are convertible. For instance, the dominated space of a point by CDAS can also be seen as opening the edges of Pareto order cone with a angle of $\pi/2 - S \cdot \pi$ because the opening angles are the same for each point. More details are available in [33]. In the next chapter, we use the geometrical interpretations directly to extend the Pareto dominance relation.

Chapter 3

Diversity-based and Cone-based Multi-objective Evolutionary Algorithms

This chapter proposes an algorithm that will play an important role in solving multi-objective optimization problems of this thesis. The first part of this chapter is dedicated to answering RQ1, which is to develop an MOEA and compare it with state-of-the-art MOEAs. The proposed MOEA is called diversity indicator-based MOEA (DI-MOEA). DI-MOEA introduces a new principle to use non-dominated sorting combined with a set-based diversity indicator which can be efficiently computed, and it can achieve a uniformly distributed PF approximation regardless of the shape of the PF.

Followed by the introduction of several alternatives to Pareto dominance relationship in the previous chapter, the second part of this chapter aims to improve the performance of Pareto dominance by making use of its geometrical property, furthermore, propose an approach to promote the behavior of MOEAs in general (RQ2). The proposed cone order increases solutions' dominance area and the convergence speed of MOEAs adopting it. Special emphasis is given to many-objective optimization due to the degraded ability of Pareto dominance to establish a ranking when handling many-objective problems.

3.1 Diversity Indicator-based MOEAs

As mentioned before, indicator-based optimization has been a successful principle for MOEA design. The idea is to guide the search for approximating the Pareto front by a performance indicator. Ideally, the indicator captures both convergence to the Pareto front and a high diversity, and it does not require a priori knowledge of the Pareto front shape and location. It is, however, so far difficult to define indicators that scale well in computation time for high dimensional objective spaces, and that distribute points evenly on the Pareto front. Moreover, the behavior of commonly applied indicators depends on additional information, such as reference points or sets. For example, when the hypervolume indicator is used for performance comparison in indicator-based MOEAs, it has been show that the distribution of points is biased towards the knee point and the boundary if the reference point is not properly set [62]. Some multi-indicator-based MOEAs have been created to overcome these issues by using multiple quality indicators, such as [39], [40]. In this work, a diversity-indicator based multi-objective evolutionary algorithm is proposed. It combines principles from Pareto dominance-based approach and from indicator-based algorithms. Instead of requiring the indicator to take into account diversity and Pareto dominance, it is proposed to

- use dominance rank as a primary selection indicator, in order to ensure convergence to the Pareto front;
- use performance indicators that measure the diversity *of a set* of mutually non-dominated solutions.

However, as opposed to Pareto dominance-based approaches such as SPEA2 and NSGA-II that also maintain diversity, in DI-MOEA, the diversity of a set is measured by a scalar value, such that convergence to a maximum diverse set can be achieved and theoretically assessed.

Based on these principles, DI-MOEA therefore takes advantage of Pareto dominance-based approaches, and excludes the complex structure and parameters in decomposition-based and contemporary indicator-based approaches. Most importantly, experimental results show that it can find well converged and evenly spaced Pareto front approximations without the involvement of any reference points and assumptions about the location and shape of the Pareto front.

From here on, the adopted diversity indicator, i.e., the Euclidean distance based geometric mean gap indicator is introduced in detail. The proposed algorithm is

described and experimental results on benchmark problems are shown. Lastly, a conclusion of the work is given and some possible future work is indicated.

3.1.1 Diversity Indicators and Gap Contribution

There exist many indicators that assess the diversity of a distribution of points in $\mathbb{R}^m$. Among these, the Weitzman indicator and discrepancy measures have excellent theoretical properties, but their computation is expensive. The Hausdorff distance and related measures are indicators that would require the knowledge of the set on which points should be distributed, which is typically not available in Pareto optimization. The Solow-Polasky indicator has been suggested in the context of diversity assessment due to its moderate computational effort and good theoretical properties [105]. However, it is sensitive to the choice of the correlation strength parameter of an exponential kernel function and it requires matrix inversion which might cause numerical instability. The gap indicators (or the averages of distances to nearest neighbours) have been suggested in [37]. They are very fast to compute and easy to implement diversity indicators. In addition, they have certain favorable theoretical properties and empirical results show that their maximization results in diversified, evenly spread approximation sets. These results were obtained for multimodal optimization [124] and evolutionary level set approximation [74] for a wide range of test problems.

Let A define a set of points in $\mathbb{R}^m$, $D(x, A \setminus \{x\}) = \min_{a \in A \setminus \{x\}}\{d(x, a)\}$ and d denote the Euclidean distance, then the gap indicators (GI) are defined as follows:

$$
\begin{aligned}
GI_{\min}(A) &= \min_{x \in A}\{D(x, A \setminus \{x\})\} && \text{Minimal gap} \\
GI_{\Sigma}(A) &= \frac{1}{|A|} \textstyle\sum_{x \in A} D(x, A \setminus \{x\}) && \text{Arithmetic mean gap} \\
GI_{\Pi}(A) &= \left(\textstyle\prod_{x \in A} D(x, A \setminus \{x\})\right)^{\frac{1}{|A|}} && \text{Geometric mean gap.}
\end{aligned}
$$

Note, that $GI_{\min}$ is the well known diversity indicator used in the max-min diversity problem [50]. One can leave out the exponent in GI_{Π} and this yields the product distance to the nearest neighbour (PDNN) indicator, considered by Wessing [124] in the context of multimodal optimization. Wessing [124] pointed out that GI_{Π} obtains the value of zero in case of duplicates in the set, a property that also holds for $GI_{\min}$. Besides, it can only be used for comparing sets of equal size. Since we are using the indicator contribution as a relative measure of performance of points, these two properties do not cause problems.

In indicator-based steady state selection [9] is to optimize a quality indicator QI for a solution set. W.l.o.g. we assume the quality indicator is to be maximized. The selection strategy is to add a non-dominated solution x to an approximation set A of size μ and then retain the best subset $S \subset P$ with $|S| = \mu$ of the new set $P = A \cup \{x\}$. This can be achieved by removing the point that contributes the least to the quality indicator. The indicator contribution of a point $p \in P$ is defined as:

$$\Delta_{QI}(p, P) \leftarrow QI(P) - QI(P \setminus \{p\}).$$

In DI-MOEA, the set-indicator contribution of the individual $p \in P$ is defined as the difference of the geometric mean gap indicator value of the set with the individual p minus the indicator-value of the set without it. The computation of the minimal contributor in case of the gap indicators can be solved by computing the solution to the all point nearest neighbour problem (APNN). The straightforward implementation, i. e. measuring distance between all pairs, requires a running time of $O(n^2)$. The APNN problem can be solved by Vaidya's algorithm [107] in optimal time $O(n \log n)$ for a fixed dimensional space and any Minkowski metric, including the Euclidean metric. The Euclidean distance is chosen as distance measure due to its rotational invariance.

3.1.2 Algorithm

A hybrid selection scheme: the $(\mu + \mu)$ generational selection operator and the $(\mu + 1)$ steady state selection operator, is utilized in DI-MOEA. The algorithm consists of two components:

- The $(\mu + \mu)$ generational selection operator: When the population is layered to multiple (more than one) dominance ranks, it indicates that the population has not yet converged to the true Pareto front. In this case, the $(\mu + \mu)$ generational selection operator is used to explore the decision space for dominating solutions. In this stage, a strict consideration of the diversity indicator is not yet the key determinant factor. Rather the first priority should be to push the population quickly to the Pareto front. Still, diversity is considered as a secondary ranking criterion in order to bring the points in a good starting position for searching for a uniformly distributed population. Overall, the selection operator is using non-dominated sorting as a primary ranking criterion, then if more than μ solutions are obtained by adding a layer, two alternative strategies are proposed

Algorithm 1 DI-MOEA

```
1: P_0 ← init();      //Initialize random population
2: popsize ← |P_0|;
3: (R_1, ..., R_{ℓ_0}) ← Partition P_0 into subsets of increasing dominance rank; //Non-dominated sorting
4: for each i ∈ {1, ..., ℓ_0} do
5:     calculate diversity indicator for all solutions based on the current front;
6: end for
7: t ← 0;
8: while Stop criterion not satisfied() do
9:     if ℓ_t > 1 || t == 0 then
10:        // (μ + μ) selection operator
11:        Q_t ← Gen(P_t); // Generate offspring with the size of popsize by variation
12:        Evaluate Q_t;
13:        P_t = P_t ∪ Q_t    // Combine offspring and parents
14:        (R_1, ..., R_{ℓ_t}) ← Partition P_0 into subsets of increasing dominance rank; //Non-dominated sorting
15:        i ← 0;    P_{t+1} ← ∅;
16:        while |P_{t+1}| < popsize do
17:            P_{t+1} ← all solutions on i-th front R_i;
18:            i ← i + 1;
19:        end while
20:        if |P_{t+1}| > popsize then
21:            n ← |P_{t+1}| − popsize
22:            while n > 0 do
23:                calculate diversity indicator for all solutions on the last front;
24:                remove the least contributor solution based on rank and diversity;
25:                n ← n − 1;
26:            end while
27:        end if
28:    else
29:        // (μ + 1) selection operator
30:        q ← Gen(P_t);        // Generate only an offspring by variation
31:        P_t ← P_t ∪ {q};
32:        Rank P_t based on Pareto dominance rule; //Non-dominated sorting
33:        for each front do
34:            calculate set-indicator contribution for all solutions on the least ranked front |R_{ℓ_t}|, if |R_{ℓ_t}| > 1;
35:        end for
36:        remove the least contributor to diversity-indicator on the least ranked front;
37:    end if
38: end while
```

to truncate: the crowding distance (*variant 1*) as in NSGA-II, and the diversity indicator contribution (*variant 2*), where points are successively removed in a greedy manner and the contributions are recomputed after each removal. Under the condition that the μ selected solutions are mutually non-dominated after an iteration, the algorithm switches to the $(\mu + 1)$ steady state selection operator.

- The $(\mu+1)$ steady state selection operator: When the parent population consists of only one non-dominated set, it is likely that the population has already reached a region near the Pareto front. In this case, the indicator-based $(\mu + 1)$ steady state selection operator is applied, as described in Section 3.1.1. It discards the least contributor to the quality indicator, here, the diversity indicator. The intent is to achieve a uniformly distributed set on the Pareto front, that is to converge to a maximum of the diversity indicator. If there are more than one dominance ranks in the resulting population, the algorithm switches back to a $(\mu + \mu)$ generational selection operator.

Besides the hybrid selection scheme, another important design choice is the quality indicator, to be specific, the Euclidean distance based geometric mean gap indicator is used to guide the search towards the uniformly distributed Pareto front approximations regardless of the shape of the Pareto front. The proposed algorithm is presented as pseudo-code in Algorithm 1.

3.1.3 Experimental Results and Discussion

In this section, simulations are conducted to demonstrate the performance of the proposed algorithm. Because two different diversity measures are employed in the (μ + μ) generational selection operator, two variants of DI-MOEA are involved in the experiments: the crowding distance and the set-indicator contribution are chosen as the second measure in the generational (μ + μ) selection operator in algorithm DI-1 and algorithm DI-2 respectively.

In the simulations[1], the SBX operator with an index of 15 (30 in NSGA-III and a differential evolution operator is used in MOEA/D.) and polynomial mutation with an index 20 are used. The crossover and mutation probabilities are set to 1 and $1/L$ respectively and L is the number of variables. In NSGA-III, the number of

[1] All MOEAs in the thesis are implemented and tested based on the MOEA Framework (version 2.1, available from `http://www.moeaframework.org`). The MOEA Framework is Java-based framework for multi-objective optimization and it supports a number of MOEAs, test problems and search operators. It is also easy to be extended to introduce new problems and algorithms.

subdivisions is 99 for bi-objective problems, and 12 for tri-objective problems. The number of evaluation (NE) is chosen to be dependent on the complexity of the test problem. 20000 NE is used for ZDT problems and 100000 NE for DTLZ problems. The population size is 100 for all problems. Here we set the population size to be 100 because such number can be sufficient to represent the Pareto front of the adopted benchmark problems and it is an intuitive number usually given by the decision maker. However, for decomposition-based MOEAs, such as MOEA/D and NSGA-III, it is better to consider the number of reference vectors when specifying the population size. For example, the number of weight vectors in MOEA/D for three-objective problems is $1+2+3+4+...+H$ (where H is an integer), which is the same as the population size in MOEA/D. Thus, the population size of MOEA/D on three-objective problems can be 91 or 105 (instead of 100).

Experiments on bi-objective problems

For bi-objective problems, algorithms are tested on ZDT1, ZDT2 and ZDT3 with 30 variables. Two new algorithms, DI-1 and DI-2, are compared with NSGA-II, SMS-EMOA, NSGA-III and MOEA/D. Table 3.1 and Table 3.2 show the aggregate hypervolume and aggregate IGD across 30 independent runs (with a different seed for each run but same seeds for all algorithms). The aggregate value is the value obtained when the Pareto solutions from all runs are combined into one. For each problem in the two tables, the upper row denotes the aggregate hypervolume/IGD. (The best value is highlighted in bold.) The lower row is the standard deviation (Std) of results from 30 runs. The Mann-Whitney U test is used to determine if the medians of different algorithms for the same problem are significantly indifferent. In the tables, algorithms whose median performance is indifferent to the algorithm with the best aggregate performance are also highlighted. It can be observed that SMS-EMOA or NSGA-III can achieve the best hypervolume and the best IGD on all three problems, and the proposed DI-MOEA can obtain better hypervolume and IGD than NSGA-II and MOEA/D. In some instances, DI-MOEA can even get better hypervolume and IGD than NSGA-III or SMS-EMOA.

Experiments on tri-objective problems

For tri-objective problems, DTLZ1 with 7 variables, DTLZ2 with 12 variables and DTLZ7 with 22 variables are tested. Both DI-1 and DI-2 behave very well, and they are indifferent on the statistical significance of median performance of aggregate

Table 3.1: The aggregate hypervolume (HV) on bi-objective problems.

HV / Std ↘	NSGA-II	SMS-EMOA	NSGA-III	MOEA/D	DI-1	DI-2
ZDT1	0.66399	**0.66602**	0.66428	0.66029	0.66473	0.66491
	4.8379e-04	7.2331e-05	3.9507e-04	0.0028	3.5973e-04	2.8447e-04
ZDT2	0.33002	0.33265	**0.33266**	0.32849	0.33073	0.33141
	4.7756e-04	8.7207e-05	0.0086	0.0030	4.9232e-04	5.8483e-04
ZDT3	0.51600	0.51718	**0.51720**	0.51582	0.51623	0.51634
	3.9954e-04	0.0013	0.0010	0.0011	4.1969e-04	2.7955e-04

Table 3.2: The aggregate IGD on bi-objective problems.

IGD / Std ↘	NSGA-II	SMS-EMOA	NSGA-III	MOEA/D	DI-1	DI-2
ZDT1	0.00163	**0.00039**	0.00168	0.00385	0.00116	0.00106
	2.6517e-04	1.9915e-05	8.2835e-04	0.0018	1.4110e-04	9.7026e-05
ZDT2	0.00202	0.00084	**0.00051**	0.00247	0.00159	0.00120
	2.1844e-04	1.0340e-04	0.0088	0.0014	2.1557e-04	2.4062e-04
ZDT3	0.00092	**0.00037**	0.00054	0.00190	0.00087	0.00092
	1.5809e-04	0.0100	0.0080	8.6720e-04	1.6713e-04	1.3157e-04

hypervolume and IGD. Statistical data averaging 10 runs per problem and algorithm are shown on Table 3.3 and Table 3.4. DI-1 beats all the algorithms on the aggregate hypervolume on all problems, and DI-2 also behaves better than other algorithms except for SMS-EMOA on DTLZ1. For IGD, the new algorithms perform the best on DTLZ1 and DTLZ2 problems. NSGA-II obtains the best IGD on DTLZ7, while IGD values of DI-1 and DI-2 are only slightly higher than NSGA-II on DTLZ7, but better than all other algorithms.

Table 3.3: The aggregate hypervolume (HV) on tri-objective problems.

HV / Std ↘	NSGA-II	SMS-EMOA	NSGA-III	MOEA/D	DI-1	DI-2
DTLZ1	0.80605	0.80732	0.78400	0.80198	**0.80806**	**0.80645**
	0.0062	1.8738e-04	0.0179	0.0015	0.0013	6.1716e-04
DTLZ2	0.44263	0.45269	0.41915	0.42907	**0.45511**	**0.45489**
	0.0070	5.8698e-05	5.1471e-04	0.0031	0.0033	0.0014
DTLZ7	0.31064	0.24694	0.30624	0.30164	**0.31227**	**0.31339**
	0.0034	0.0038	0.0328	0.0055	0.0051	0.0137

To easily observe the results of algorithms, the results on the tri-objective problems are visualized. Figure 3.1 shows the Pareto front approximations of a typical run on DTLZ1. It can be observed that the solutions of NSGA-II and MOEA/D are not uniformly distributed, and there are several overlaps in the result of NSGA-III. While, SMS-EMOA and DI-MOEA can obtain evenly spaced solutions on the linear Pareto front.

Table 3.4: The aggregate IGD on tri-objective problems.

IGD / Std ↘	NSGA-II	SMS-EMOA	NSGA-III	MOEA/D	DI-1	DI-2
DTLZ1	0.02149	0.02074	0.04266	0.02779	**0.01966**	**0.02381**
	0.0063	8.1450e-04	0.0159	0.0018	0.0017	0.0016
DTLZ2	0.02414	0.03415	0.05181	0.03902	**0.01799**	**0.01909**
	0.0047	0.0014	2.1056e-04	0.0026	0.0019	0.0030
DTLZ7	**0.01820**	0.09182	0.02381	0.041367	0.01826	0.02191
	0.0027	0.0020	0.2151	0.0867	0.0017	0.0944

Figure 3.2 shows the Pareto front approximations of a typical run on DTLZ2. For NSGA-III, we observed the same phenomenon: some solutions are overlapping or very close. The result of SMS-EMOA is distributed across the Pareto front with emphasis on the boundary and knee regions of the Pareto front. The results of the two DI-MOEA variants are uniformly distributed and evenly spaced on the Pareto front.

DI-MOEA also behaves well on the multimodal DTLZ7 problem, which has non-linear disconnected Pareto front regions. Figure 3.3 shows the results under 200 population size and 500000 NE.

When running the DI-MOEA, it can be observed that the population evolves towards the Pareto front at the initial stage (the first phase) using the generational selection operator. After a short period where the two selection operators alternate (the second phase), the steady state selection operator takes over and the population converges to a set with maximum diversity (the third phase). When the number of objectives becomes large, the third phase is more prominent than the previous two phases because it is more likely for solutions to be mutually non-dominated for a large objective number. In the runs conducted on tri-objective problems, the generational selection operator was applied around 100-200 iterations before it switched to the steady state selection operator for the first time. The intermittent alternating phase took about 20-50 iterations, and in most of the running time, the algorithm used the steady state selection operator and throughout this phase, only occasionally the algorithm switched back to generational selection operator for at most a single iteration. Overall, the first and the second phase took only a minor amount of the total running time.

It is worth noting that we observed dominance resistant solutions (DRSs) [54] occasionally on the linear Pareto front of DI-2 on DTLZ1 tri-objective problem; these are points that have a large contribution to diversity, but dominate only a very narrow region exclusively. It might be necessary to keep these "special solutions", but on the other side, they make the Pareto front approximation less evenly distributed. A

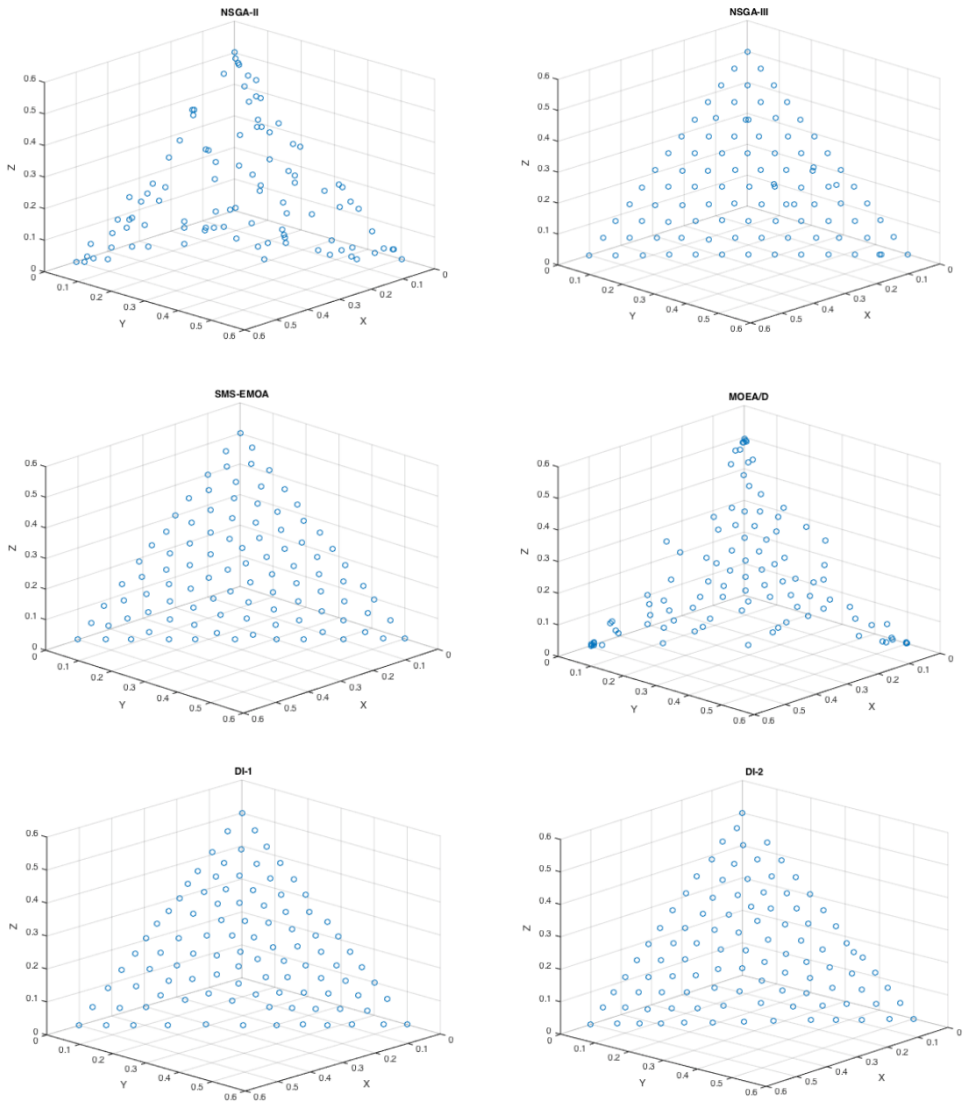


Figure 3.1: Representative Pareto front approximations of DTLZ1.

strategy has been tested to eliminate DRSs. Before the calculation of the set-indicator contribution for a front, each solution is checked by comparing with all other solutions: the distances between two solutions in all dimensions are calculated, if the result of the minimal distance divided by the maximal distance is too small, the current solution will be removed from the front. Therefore, a shrinked front is created and the diversity

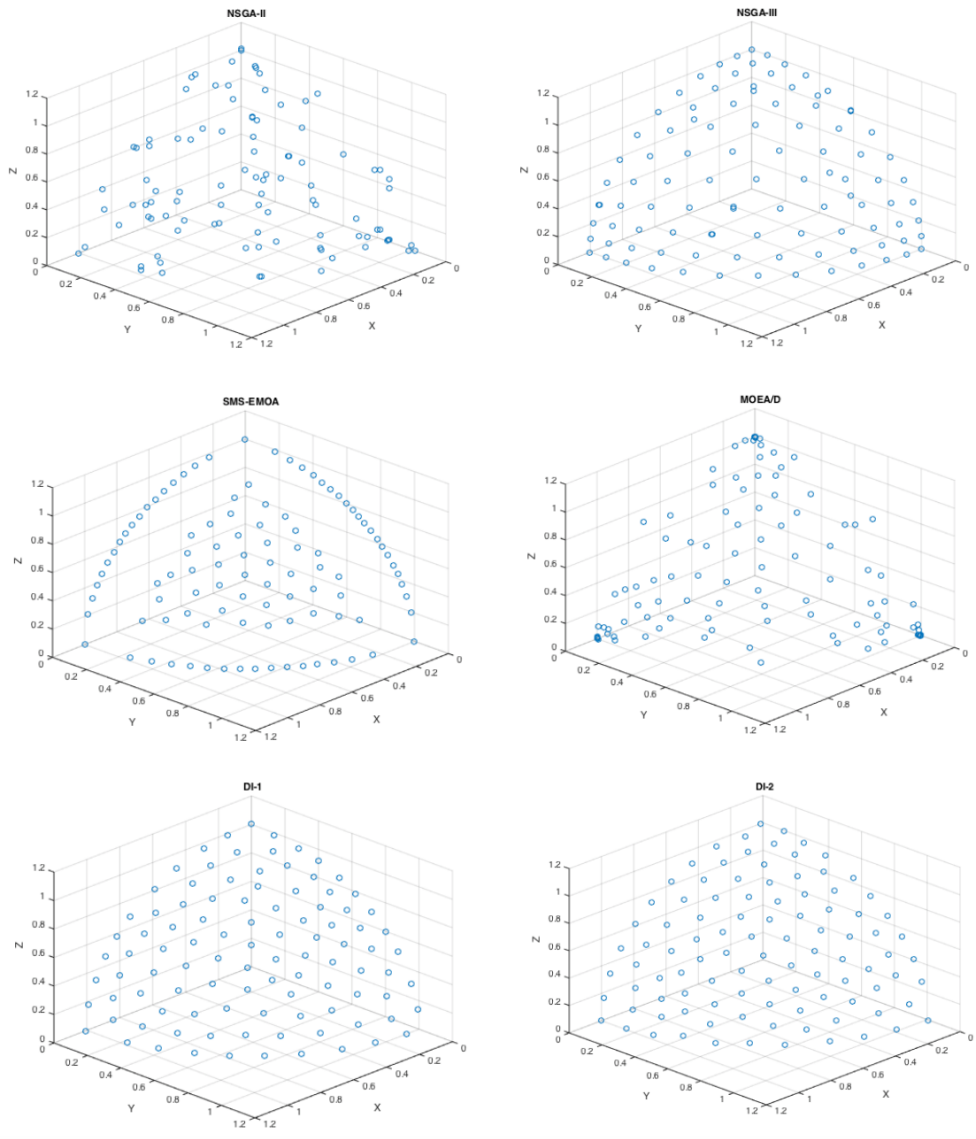


Figure 3.2: Representative Pareto front approximations of DTLZ2.

indicator can be calculated only in the new front. The underlying idea of this strategy is that for two solutions, if their distance is too close in one dimension and too large in another dimension, keeping both of them will result in an uneven distribution.

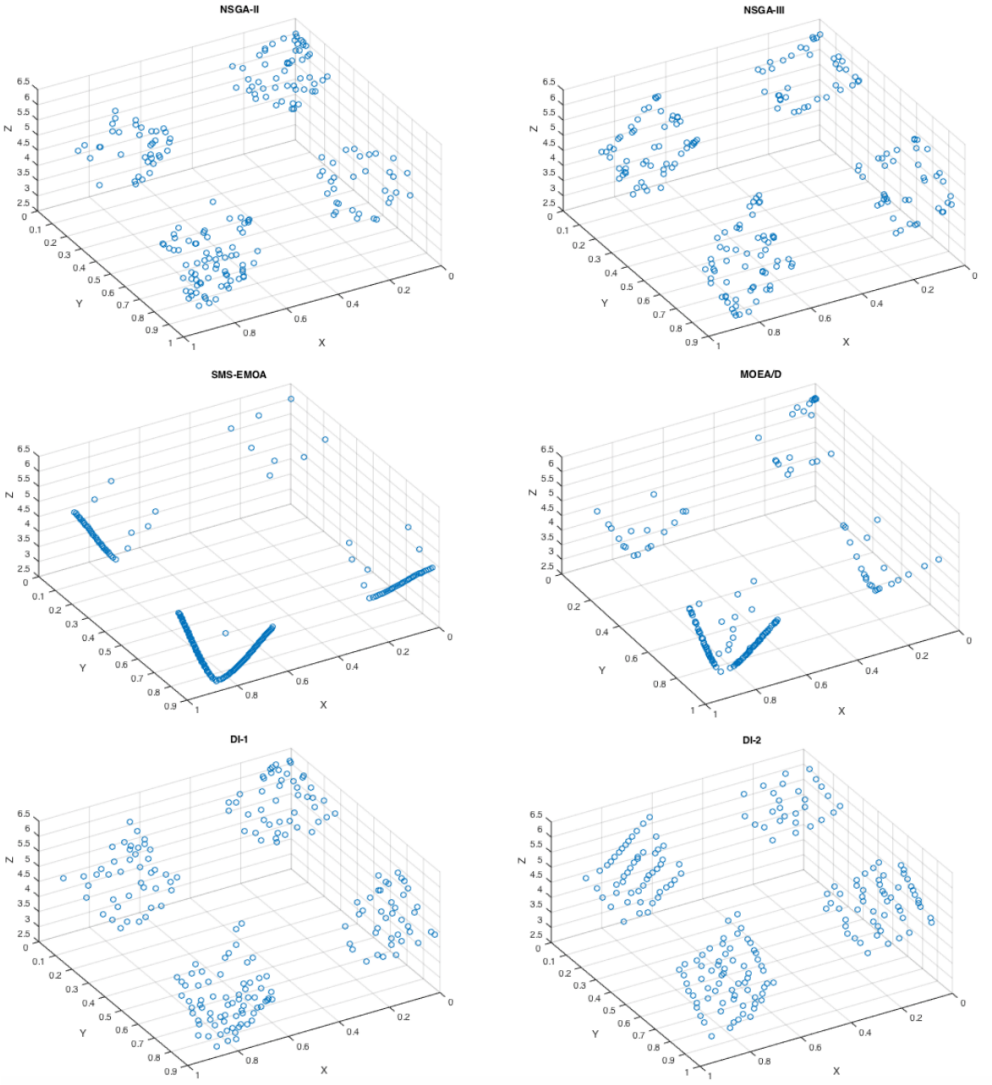


Figure 3.3: Representative Pareto front approximations of DTLZ7.

3.1.4 Conclusion and Further Work

The proposed DI-MOEA combines the advantage of Pareto dominance-based and indicator-based methods. Moreover, the achieved Pareto front approximations are excellent in both hypervolume indicator and IGD. In particular, the relative performance of DI-MOEA even gets better with an increasing number of objectives. The

set-indicator used in our algorithms is computationally simpler than the hypervolume indicator and only depends linearly on the number of objectives, making it possess a potential advantage on many-objective optimization problems. Most importantly, the uniformly distributed, evenly spaced solution set can be achieved without the use of decomposition sets and the estimation of the location and shape of the true Pareto front.

In the current implementation of DI-MOEA, only a naive way of calculating the Euclidean distance based geometric mean gap is implemented. Although the computational time of the implemented algorithm is shorter than SMS-EMOA, it should be further improved, e.g., by using Vaidya's algorithm [107] and incremental updates of contributions. Besides, DI-MOEA holds the promise of performing well in many-objective optimization. To study this, its performance should be tested on many-objective optimization benchmarks, paying special attention to effects that might occur in high dimensional objective spaces, such as distance concentration and the increasing number of non-dominated solutions.

3.2 Cone-based MOEAs

In this section, the edge-rotated cone order is first proposed for the purpose of building an ordering which can guide the search towards the Pareto front better than the Pareto order in MOEAs. Two different methods have been proposed to implement the edge-rotated cone order. Afterwards, the edge-rotated cone order is integrated in MOEAs by a proper approach which gives consideration to both convergence and diversity in the evolutionary searching process. The integrated MOEAs are then tested on multi-objective and many-objective optimization problems to compare with original MOEAs. Moreover, the ability of the edge-rotated cone order on expressing preferences in evolutionary multi-objective optimization is investigated.

3.2.1 Edge-rotated Cone Order

In an MOEA, if a solution can dominate more areas based on the adopted dominance relation, the algorithm is capable of exploring more solutions and hence accelerating convergence. To this end, the edge-rotated cone is devised by widening the angle of the Pareto order cone and it allows a solution to dominate a larger area. Given a linearly independent vector set $\{w_1, w_2, \ldots, w_m\}$, a cone can be generated in m-dimensional space.

Definition 3.1 (Generated m-dimensional cone)**.** The cone generated by the vectors $w_1, w_2, \ldots, w_m$ is the set $C = \{z : z = \lambda_1 w_1 + \lambda_2 w_2 + \cdots + \lambda_m w_m, \forall \lambda_1, \lambda_2, \ldots, \lambda_m \geq 0, \lambda \neq 0\}$; $w_1, \ldots, w_m$ are linearly independent.

Figure 3.4 presents a two dimensional scenario of generating the standard Pareto order cone and edge-rotated cones. The Pareto order cone is the cone constructed by two half-lines aligned with two vectors, $\vec{a}$ and $\vec{b}$, which support an angle of 90°. Similarly, vectors $\vec{L1}$ and $\vec{L2}$ determine a cone with an angle of 150°. Intuitively, this cone can be formed by rotating the edges of the Pareto order cone towards the opposite direction around the origin. Note that vectors $\vec{L3}$ and $\vec{L4}$ determine a cone with an angle of 180° and the cone with an angle of 180° is a line. Any two vectors between $\vec{L3}$ and $\vec{L4}$ (excluding at least one of $\vec{L3}$ and $\vec{L4}$) can construct a convex cone (Definition 2.16), i.e., the space to the right of the corresponding lines and the lines themselves.

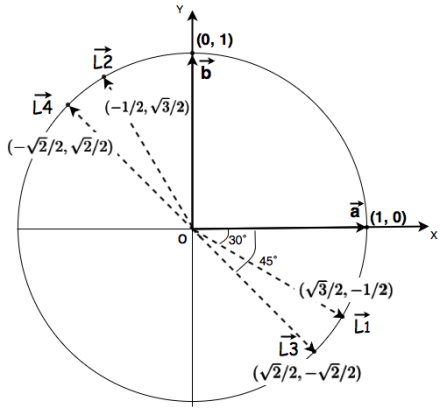


Figure 3.4: Cones with different angles.

When applying this cone order by means of the Minkowski sum (Definition 2.13), a solution can dominate more objective space. The left image of Figure 3.5 shows an example of applying the Pareto order cone to illustrate Pareto dominance relation, i.e., P dominates the points in $P \oplus \mathbb{R}^2_{\succ_o}$ and Q dominates the points in $Q \oplus \mathbb{R}^2_{\succ_o}$. Here, $\oplus$ is the Minkowski sum; $\mathbb{R}^2_{\succ_o}$ is equal to the cone constructed by $\vec{a}$ and $\vec{b}$ in Figure 3.4, the origin is excluded. In other words the non-negative quadrant with origin excluded.

It can be seen that P and Q are mutually non-dominated in terms of Pareto dominance relation because neither of them is in the dominating space of the other point. However, when an edge-rotated cone (e.g., the cone constructed by $\vec{L1}$ and $\vec{L2}$

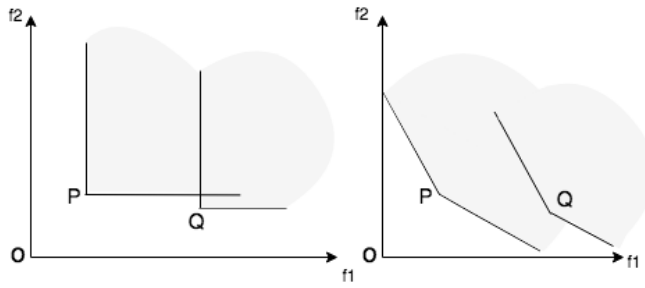


Figure 3.5: Pareto cone and edge-rotated cone orders.

in Figure 3.4) is adopted in the right image, the dominance relation between the point P and Q has changed and now Q is dominated by P.

The edge-rotated cone can be interpreted as a constraint on trade-offs. In Figure 3.5, two points $P = (p_1, p_2)$ and $Q = (q_1, q_2)$ are Pareto incomparable in $\mathbb{R}^2$. The trade-off, that is, the decrease in f_2 per unit of increase of f_1 is $\frac{p_2-q_2}{q_1-p_1}$. It is easily seen that in case $\frac{p_2-q_2}{q_1-p_1} \leq \tan(\alpha)$ (Here, α is the rotating angle on the edge of Pareto cone.), the points P and Q become comparable in the edge-rotated cone order and if $\frac{p_2-q_2}{q_1-p_1} > \tan(\alpha)$ the points P and Q stay incomparable in the edge-rotated cone order. Similarly, if for the decrease in f_1 per unit increase of f_2 it holds that $\frac{q_1-p_1}{p_2-q_2} \leq \tan(\alpha)$, then the points Q and P become comparable in the edge-rotated cone order and if $\frac{q_1-p_1}{p_2-q_2} > \tan(\alpha)$, then they are still incomparable in the edge-rotated cone order. In summary, if for two Pareto incomparable points one of the trade-offs is bounded by $\tan(\alpha)$, then the points are comparable in the edge-rotated cone order; in case, both trade-offs are bigger than $\tan(\alpha)$, the points are also incomparable with respect to the edge-rotated cone order.

When using the edge-rotated cone order in MOEAs, since the concave cones do not give rise to a strict partial order and the non-dominated points in the order generated by acute cones can be dominated in the Pareto order, the adopted edge-rotated cones are restricted to convex obtuse cones obtained by rotating each edge of the standard Pareto cone towards the outside with an angle of maximal 45°. For example, in the case of a bi-objective problem, one edge of the cone can exist between $\vec{a}$ and $\vec{L3}$ and another edge of the cone can exist between $\vec{b}$ and $\vec{L4}$. The approach of widening the standard Pareto cone in m-dimensional space ($m > 2$) is the same. Each edge of the standard Pareto order cone is rotated around the origin by an angle of maximal 45° towards the opposite direction of the identity line in the first cube's orthant on the plane consisting of the edge and the identity line. In m-dimensional space, the identity

line in the first cube's orthant is the line that passes through the origin and the point $(1, ..., 1)$.

It is worth noting that solutions that are dominating in the Pareto order are also dominating in the edge-rotated cone order. Therefore, it is guaranteed that a minimal element of the edge-rotated cone order is also a minimal element of the Pareto order, and thus algorithms that converge to globally efficient points under the edge-rotated cone order will also converge to globally Pareto efficient points.

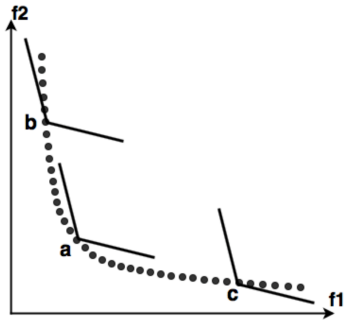


Figure 3.6: Trade-off on Pareto front.

The edge-rotated cone gives rise to an extended dominance relation and it establishes an ordering among Pareto incomparable solutions (i.e., being Pareto non-dominated to each other) in the sense that better Pareto incomparable solutions are preferred. By using the edge-rotated cone, a solution, especially the solution which is not in the knee region, has a higher chance to be dominated by other solutions. The knee region is the region where the maximum trade-off of objective functions takes place. For the Pareto front in Figure 3.6, the knee region is where the Pareto surface bulges the most, i.e., the region near solution a. When comparing the knee point a with another solution c, solution c has a better (i.e., lower) $f2$ value as compared to solution a. However, this small improvement leads to a large deterioration in the other objective $f1$. Due to the reason that in the absence of explicitly provided preferences, all objectives are considered equally important, solution a, thus, is more preferable than solution c. It has been argued in the literature that knee points are the most interesting solutions and preferred solutions [12, 13, 20, 24]. In this sense, although not all globally efficient points might be obtained by the edge-rotated cone orders, the edge-rotated cone orders naturally filter out non-preferred solutions. In Figure 3.6, when applying the edge-rotated cone, solutions in the knee region can survive, while

solutions like b and c are on the flat Pareto surface and are more easily to be dominated. The feature of the edge-rotated cone to eliminate solutions can be appreciated as an advantage especially in the realm of many-objective optimization considering the exponential increase in the number of non-dominated solutions necessary for approximating the entire Pareto front.

3.2.2 Implementation Methods

Two different methods have been proposed to implement the edge-rotated cone order. The first one emphasises on its geometrical property and the second one inclines towards its mathematical characteristic.

Method 1

Let us assume an edge-rotated cone constructed by $\vec{L1}$ and $\vec{L2}$ (Figure 3.4) is adopted in the right image of Figure 3.7. The area dominated by P can be determined by two lines: PA and PB. We can see that A is equal to $P + (\cos(-\alpha), \sin(-\alpha))$ and B is equal to $P + (\sin(-\alpha), \cos(-\alpha))$, where α is the rotated angle of the edge with respect to the standard Pareto order cone. In this example, α is $\pi/6$ (i.e., 30°); the points A and B are then $P + (\sqrt{3}/2, -1/2)$ and $P + (-1/2, \sqrt{3}/2)$ respectively.

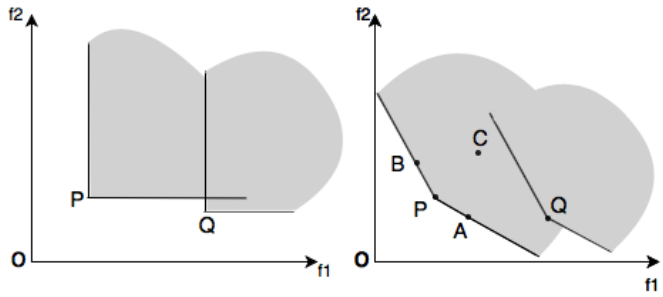


Figure 3.7: Pareto cone and edge-rotated cone orders.

In order to determine whether P dominates Q, we choose a point C on the identity line of the extension cone as a reference point which is known to be dominated by P. For instance, take C to be $P + (1, 1)$. To learn if another point Q is dominated by P or not, we only need to compare its position relative to the reference point, i.e., if Q and C are on the same side of line PA, and at the same time, both points are on the same side of line PB, Q is dominated by P.

Line PA can be defined by $ax + by + c = 0$, where the parameters, a, b and c, can be determined by two points on the line, i.e., P and A. To identify if two points, C and Q, are on the same side of this line, we only need to substitute these two points in $ax + by + c$, if the values of $ax + by + c$ are both negative or positive, it can be concluded that two points are on the same side of this line.

This comparison approach of identifying the dominance relationship between two points with the edge-rotated cone can also be easily implemented in larger dimensional space. When the number of objectives is m ($m > 2$), the space dominated by a point (e.g., P) is composed of m hyperplanes; each hyperplane is determined by m points and these m points include P and other $m - 1$ points picked from (P_1, ..., P_i, ..., P_m) successively. The point P_i ($\in \mathbb{R}^m$) is equal to $P + o_i$; and the point o_i locates the new position of the ith edge of the cone together with the origin, the value of o_i is the ith column of the following ($m \times m$) matrix.

$$\begin{bmatrix} \cos(-\alpha) & \frac{\sin(-\alpha)}{\sqrt{m-1}} & \cdots & \frac{\sin(-\alpha)}{\sqrt{m-1}} \\ \frac{\sin(-\alpha)}{\sqrt{m-1}} & \cos(-\alpha) & \cdots & \frac{\sin(-\alpha)}{\sqrt{m-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\sin(-\alpha)}{\sqrt{m-1}} & \frac{\sin(-\alpha)}{\sqrt{m-1}} & \cdots & \cos(-\alpha) \end{bmatrix} \tag{3.1}$$

The equation of a hyperplane in the m-dimensional space is $a_1x_1 + a_2x_2 + \cdots + a_ix_i + \cdots + a_mx_m + a_{m+1} = 0$, where x_i ($i \in (1, \ldots, m)$) is the ith objective value and a_i ($i \in (1, \ldots, m+1)$) is the parameter. All parameters of the hyperplane (i.e., from a_1 to a_{m+1}) can be calculated by the m points on the hyperplane. Again, the point C (i.e., $P + (1, \ldots, 1)$) can be used as the reference and if another point Q is dominated by P, the two points Q and C would be on the same side of all hyperplanes, meaning that if for each hyperplane, we put the two points in the equation $a_1x_1 + a_2x_2 + \cdots + a_ix_i + \cdots + a_mx_m + a_{m+1}$, both results would be negative or positive.

Method 2

In this method, a criterion is derived by which one can determine whether a point $Q \in \mathbb{R}^2$ is dominated by a point $P \in \mathbb{R}^2$ with respect to the edge-rotated cone order. Let $e_1 := \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $e_2 := \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ be the edges of the two-dimensional standard Pareto cone. Then the edges of the edge-rotated cone by a rotation angle α ($0 \leq \alpha \leq \frac{\pi}{4}$) are Ae_1 and Ae_2, where $A = \begin{bmatrix} \cos(-\alpha) & \frac{\sin(-\alpha)}{\sqrt{2-1}} \\ \frac{\sin(-\alpha)}{\sqrt{2-1}} & \cos(-\alpha) \end{bmatrix}$.

A point Q lies in the edge-rotated cone region of P if and only if for some λ, $Q = P + \lambda_1 Ae_1 + \lambda_2 Ae_2$ such that $\lambda_1, \lambda_2 \geq 0, \lambda \neq 0$. This is equivalent to: for some λ, $A^{-1}(Q - P) = \lambda_1 e_1 + \lambda e_2$ such that $\lambda_1, \lambda_2 \geq 0, \lambda \neq 0$. In short, *$P$ dominates Q with respect to the edge-rotated cone order if and only if the components of $A^{-1}(Q-P)$ are non-negative and at least one of them is strictly positive.* Thus, once the inverse matrix of A is computed ($A^{-1} = c \cdot \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}$, $c := \frac{1}{(\cos(\alpha))^2 - (\sin(\alpha))^2}$) , it can readily be determined whether Q is in the dominating region of P. Moreover, in case the components are non-zero and have opposite signs, then the points are incomparable. In case the components are non-positive and at least one of them negative, then Q dominates P.

The approach can easily be applied to three or many objective problems. When the number of objectives is m ($m > 2$) and the rotation angle for each edge of the cone is α, the $(m \times m)$ matrix (3.1) gives the coordinates of the unit point on rotated edges: for each unit point on the edge of the standard Pareto cone, each column of the matrix gives its new coordinates after rotation. For example, in three-dimensional space, (1, 0, 0) is the unit point on one edge of the standard Pareto cone, then $(\cos(-\alpha), \frac{\sin(-\alpha)}{\sqrt{2}}, \frac{\sin(-\alpha)}{\sqrt{2}})$ are its new coordinates after the edge is rotated by an angle of α ($0 \leq \alpha \leq \frac{\pi}{4}$).

When using the edge-rotated cone order in MOEAs by this method, the inverse matrix only needs to be calculated once and this leads to almost no extra computing time added in MOEAs.

3.2.3 Integration Algorithm

In a multi-objective optimization algorithm, by using the edge-rotated cone, a solution has a higher chance to be dominated by other solutions and thus the selection pressure toward the Pareto front is increased. However, an edge-rotated cone can degrade the diversity to some extent because more solutions will be dominated and therefore excluded from the result set. To circumvent this, Algorithm 2 is proposed to pick a proper cone order (the standard Pareto cone order or edge-rotated cone order) in each iteration of the algorithm in order to promote diversity in addition to convergence. Specifically speaking, at the beginning of each iteration, the population is ranked based on the current cone order; the edge-rotated cone order will be adopted only under the condition that all the solutions in the population are mutually non-dominated. In case the current population consists of multiple layers, the standard Pareto cone (i.e., the rotation angle is $0°$) is picked to select offspring. The underlying idea is when all

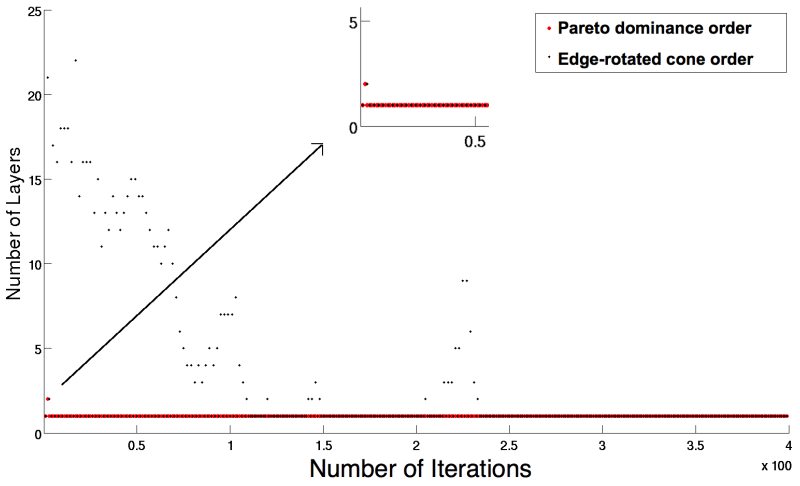


Figure 3.8: The dynamics of the number of layers.

the solutions are non-dominated with each other, the edge-rotated cone is adopted to enhance the selection pressure; otherwise, the standard Pareto cone is used to maintain the diversity of the population.

Algorithm 2 Applying a proper cone order in each iteration.

```
1: m ← the number of objectives;
2: Degree[m]; // the rotation angle for each edge of the standard Pareto order;
3: n_rank ← Pareto rank number of current population;
4: if n_rank = 0 then
5:     for each i ∈ {1, ..., m} do
6:         Degree[i] ← π/6; // rotation angle is 30°
7:     end for
8: else
9:     for each i ∈ {1, ..., m} do
10:        Degree[i] ← 0; // standard Pareto cone
11:    end for
12: end if
```

The ability of the edge-rotated cone to make Pareto incomparable solutions comparable can especially benefit many-objective optimization due to the reason that the likelihood of solution pairs of being comparable decreases exponentially with the increase of the dimension m. For many-objective optimization, a large portion of points in the objective space is non-dominated and the optimization process tends to produce

a large set of alternative solutions.

When Algorithm 2 is applied in NSGA-II on the DTLZ1 eight objective problem, Figure 3.8 compares the changes of the number of layers between running NSGA-II using only the Pareto dominance and involving the edge-rotated cone order with a rotation angle of 20° within the first 400 iterations (Population size is 100.). When running the original NSGA-II, except that one point lies at level 2 (i.e., the number of fronts is two) at the very beginning, the number of layers always remains one, meaning that all solutions in the current population are non-dominated with each other. As a result, the Pareto dominance relation has no effect on parent selection. That is, an individual with a larger crowding distance is always chosen as a parent in the binary tournament selection since all solutions have the same rank. In this manner, the selection pressure toward the Pareto front is severely weakened. However, when the edge-rotated cone is involved, the layering of the population is very noticeable. In this case, an ordering among the incomparable solutions is established and it can guide the search towards the Pareto front better.

3.2.4 Experimental Results

The proposed strategy can be integrated with any standard MOEA which works with a population in each iteration and uses the Pareto dominance relation to select solutions, such as NSGA-II [29], DI-MOEA [116], NSGA-III [26] and others. In this section, the edge-rotated cone order is applied by Algorithm 2 to observe its behavior on multi-objective and many-objective optimization problems respectively. To this end, different rotation angles have been tested, hypervolume and IGD have been adopted to compare the performance of the algorithms. The population size is 100 for all experiments.

Multi-objective Optimization

For starts, four tri-objective optimization problems have been chosen in the experiments, which are DTLZ1, DTLZ2, DTLZ7 and DTLZ2_convex. The first three problems are from the DTLZ problem test suite. The optimal Pareto front of DTLZ1 lies on a linear hyperplane. The optimal Pareto front of DTLZ2 is concave. DTLZ7 is a multi-modal problem. To measure the performance on the multi-objective optimization problem with a convex Pareto front, original DTLZ2 problem is transformed to DTLZ2_convex problem by simply decreasing all objective values by 3.5. When calculating the hypervolume of the solution set, the reference point is the point (0.6,

0.6, 0.6) for DTLZ1, the point (1.1, 1.1, 1.1) is the reference point for DTLZ2, the point (5, 5, 5) for DTLZ2_convex, and (1, 1, 6.5) for DTLZ7. Meanwhile, the origin is used as the ideal point. When calculating the IGD value, the reference sets of DTLZ1, DTLZ2 and DTLZ7 are from the MOEA framework, and the reference set of DTLZ2_convex is obtained by running DI-MOEA because DI-MOEA is good at achieving well-distributed solution sets. The merged non-dominated solution sets from 30 independent runs are used as the reference sets of DTLZ2_convex.

Firstly, the edge-rotated cone order is integrated by Algorithm 2 with NSGA-II which is one of the most popular Pareto dominance-based MOEAs. Regarding to the computing budget, when the number of evaluations is 30000, Table 3.5 shows the mean hypervolume and mean IGD from 30 independent runs when several different edge-rotated cone orders are adopted. The "P_cone" column provides the results obtained by the original MOEAs, i.e., the algorithms only adopt the standard Pareto order. The "30°" column gives the results of the algorithm involving the edge-rotated cone and each edge of the standard Pareto order cone has been rotated by 30°, and similar remarks are used for the remaining columns. The mean hypervolume and mean IGD values obtained by the original NSGA-II have been used as the reference values (printed in blue) to be compared with the results achieved by the algorithms involving the edge-rotated cone orders. For the algorithms combining the edge-rotated cones, the mean hypervolume and mean IGD values better than the values obtained by the original MOEAs have been highlighted in bold (i.e., a larger hypervolume value and lower IGD value); and the largest value for each algorithm among them is printed in red. At the same time, the standard deviation of each algorithm is also given under each mean hypervolume and mean IGD.

It can be observed that the performance of NSGA-II (for both the hypervolume and IGD values) can always be improved when the edge-rotated cone with a proper angle is involved, for example, when the rotation angle is 6° or 2°. However, the best performance takes place with different edge-rotated cones for different problems. When the rotation angle is 30°, the algorithm behaves the best on DTLZ1 problems and the rotation angle of 6° is the best on other problems. The standard deviations show the stable behavior of the algorithm involving the edge-rotated cone order, which is even better than the original NSGA-II.

When different edge-rotated cone orders are integrated in NSGA-III, Table 3.6 presents the mean hypervolume and mean IGD. NSGA-III is an extension of NSGA-II and it eliminates the drawbacks of NSGA-II such as the lack of a good diversity in a set of non-dominated solutions. Although NSGA-III is a decomposition-based MOEA,

Table 3.5: The mean hypervolume (M-HV) and mean IGD (M-IGD) when integrating edge-rotated cone orders with NSGA-II.

Problems	Metrics	P_cone	30°	12°	6°	2°
DTLZ1	M-HY	0.7064	**0.8621**	**0.8569**	**0.8490**	**0.8360**
	std	0.2731	0.0029	0.0143	0.0429	0.1111
	M-IGD	0.2420	**0.0676**	**0.0732**	**0.0807**	**0.0927**
	std	0.3102	0.0039	0.0168	0.0428	0.1194
DTLZ2 concave	M-HY	0.5276	0.2786	**0.5487**	**0.5399**	**0.5348**
	std	0.0044	0.0246	0.0029	0.0026	0.0031
	M-IGD	0.0709	0.4909	0.0801	**0.0674**	**0.0692**
	std	0.0034	0.0627	0.0040	0.0032	0.0028
DTLZ2 convex	M-HY	0.6862	0.4895	**0.6891**	**0.6921**	**0.6894**
	std	0.0030	0.0010	0.0018	0.0019	0.0025
	M-IGD	0.0707	0.3490	0.0835	**0.0706**	**0.0698**
	std	0.0030	0.0012	0.0026	0.0026	0.0032
DTLZ7	M-HY	0.2739	0.0151	0.2716	**0.2787**	**0.2767**
	std	0.0016	0.0000	0.0009	0.0019	0.0013
	M-IGD	0.0519	0.9587	0.1213	**0.0482**	**0.0492**
	std	0.0029	0.0001	0.0023	0.0029	0.0028

the basic framework of NSGA-III is similar to NSGA-II. It employs non-dominated sorting to partition the population into a number of fronts, but replaces the crowding distance operator with a clustering operator based on a set of reference points. NSGA-III is assumed to be powerful enough to handle these benchmark problems, however, according to the data in Table 3.6, it can be seen that its performance can still be improved by the edge-rotated cone order.

The same pattern can be observed from the two tables: the 30° rotation angle works best for DTLZ1 problem; a small rotation angle (i.e., 6° or 2°) works best for other problems, and a small rotation angle can almost always improve the behavior of the original algorithms. From the two tables, it can also be observed that the edge-rotated cone order can benefit NSGA-II more than NSGA-III. In some cases, the performance of NSGA-II with an edge-rotated cone can even reach the performance of the original NSGA-III.

A straightforward way to improve the results of MOEAs is to increase the computing budget. When the computing budget of the original MOEAs is increased to 300000, Table 3.7 gives the values of the mean hypervolume and mean IGD of NSGA-

Table 3.6: The mean hypervolume (M-HV) and mean IGD (M-IGD) when integrating edge-rotated cones with NSGA-III.

Problem	Metrics	P_cone	30°	12°	6°	2°
DTLZ1	M-HY	0.8709	**0.8771**	**0.8763**	**0.8756**	**0.8761**
	std	0.0108	0.0010	0.0015	0.0047	0.0013
	M-IGD	0.0616	**0.0538**	**0.0539**	**0.0552**	**0.0539**
	std	0.0224	0.0006	0.0006	0.0067	0.0004
DTLZ2 concave	M-HY	0.5593	0.2628	0.5526	**0.5595**	**0.5600**
	std	0.0006	0.0159	0.0010	0.0007	0.0003
	M-IGD	0.0554	0.5353	0.0804	**0.0534**	**0.0554**
	std	0.0005	0.0522	0.0030	0.0008	0.0004
DTLZ2 convex	M-HY	0.6941	0.4827	0.6823	0.6913	**0.6946**
	std	0.0024	0.0039	0.0025	0.0023	0.0023
	M-IGD	0.0635	0.3541	0.0898	0.0711	0.0641
	std	0.0028	0.0035	0.0041	0.0030	0.0027
DTLZ7	M-HY	0.2264	0.0697	0.2234	**0.2324**	**0.2288**
	std	0.0343	0.0496	0.0310	0.0352	0.0360
	M-IGD	0.3705	0.7702	0.4024	**0.3472**	**0.3614**
	std	0.2070	0.1296	0.1819	0.2085	0.2172

II and NSGA-III. It can be observed that the algorithms combining the edge-rotated cone order when only using a small computing budget can even behave better than the original NSGA-II and NSGA-III when using a large computing budget. Only in several cases (values in blue), the algorithm involving the edge-rotated cone order with the small budget cannot reach the performance of the original MOEAs with the large budget, but their behavior is already very close to the original MOEAs with the large budget.

Many-objective Optimization

In this section, four, six, eight objective DTLZ1, DTLZ2, DTLZ2_convex problems, UF11 and UF13 [136] have been chosen in the experiments. UF11 is a rotated instance of the 5D DTLZ2 test problem, and UF13 is the 5D WFG1 test problem. For each problem, the computing budget for running the algorithm (i.e., the number of evaluations) is determined by $\max\{100000, 10000 \times D\}$, where D is the number of decision variables. Likewise, hypervolume and IGD have been adopted to compare the performance of the algorithms. When calculating HV, the objective values of the

Table 3.7: The Mean hypervolume (M-HV) and mean IGD (M-IGD) of original NSGA-II and NSGA-III with larger computing budget.

Problem	DTLZ1	DTLZ2_concave	DTLZ2_convex	DTLZ7
M-HY	0.8630	0.5281	0.6884	0.2755
M-IGD	0.0657	0.0705	0.0698	0.0506
M-HY	0.8757	0.5603	0.6944	0.2284
M-IGD	0.0597	0.0552	0.0628	0.3599

reference point are 0.6 on DTLZ1, 1.1 on DTLZ2, 5 on DTLZ2_convex, 2.2 on UF11 and 11 on UF13. The origin is used as the ideal point. When calculating the IGD value, the merged non-dominated solution sets from all runs are used as the reference sets of the DTLZ2_convex problems and the reference sets of other problems are from the MOEA framework.

Tables 3.9 - 3.11 show the mean hypervolume and mean IGD from 15 independent runs when different edge-rotated cone orders are integrated in NSGA-II, DI-MOEA and NSGA-III. Similarly, for the algorithms combining the edge-rotated cone, the mean hypervolume and mean IGD values are better than the values obtained by the original MOEAs have been highlighted in bold; the largest respectively lowest value for each algorithm among them is printed in red. Tables for the DTLZ benchmark problems consist of four parts, namely four objective, six objective, eight objective with full budget, and eight objective with half budget. Both UF11 and UF13 are five objective problems and their behaviors with full budget and half budget are given in Table 3.11.

The following conclusions can be drawn from the data in these tables.

1. The algorithms do not work well when a large rotation angle is adopted (e.g., 30°).
2. The algorithms show similar performance to the original MOEAs when the rotation angle is very small (e.g., 3°).
3. When an intermediate rotation angle is adopted, the performance of the algorithms (both hypervolume and IGD values) shows a significant improvement except for a few cases which display values close to the original MOEAs.
4. Although it differs depending on the specific problems, the best performance is usually obtained when the rotation angle is 15°.
5. It can be seen that the edge-rotated cone can improve the performance of all

three adopted MOEAs (i.e., NSGA-II, DI-MOEA and NSGA-III) in most cases when an intermediate rotation angle is used. Even though NSGA-III is assumed to be powerful enough to handle these benchmark problems, its performance can still be improved by the edge-rotated cone approach.

6. The edge-rotated cone can benefit MOEAs even more with the increase of the number of objectives. For example, when a 15° rotation angle is applied on the DTLZ2 (concave) four objective problem, the hypervolume of NSGA-II is improved from 0.5953 to 0.6760; for the six objective problem, the hypervolume is improved from 0.1224 to 0.8156; and for the eight objective problem, the hypervolume is improved from 0.0168 to 0.8850.

7. The edge-rotated cone can benefit the algorithm with a small computing budget more than the algorithm with a large budget. For example, when using half of the computing budget on UF13 five objective problem and the rotation angle is set to 20°, the hypervolume values of the Pareto fronts from NSGA-II, DI-MOEA and NSGA-III can be improved to 0.7259, 0.7254, 0.7073, which are already larger than the hypervolume values obtained by the original MOEAs with full budget, namely 0.6937, 0.6611 and 0.6497.

8. Even though the median values of the hypervolume and IGD values have not been presented in the tables, they show similar values as the mean values. At the same time, the standard deviations show a stable behavior of the edge-rotated cone order when it is integrated in MOEAs.

Preference-based Multi-objective Optimization

In the previous experiments, the rotation angles on all edges of the Pareto cone are kept the same. However, the edge-rotated cone is not necessarily "symmetric". When rotating the different edges of the Pareto cone by different angles, the generated edge-rotated cone can lead the search towards different focuses on the Pareto front. To observe the effect of the "unsymmetrical" edge-rotated cone, it is integrated in DI-MOEA and the tri-objective DTLZ2 concave and convex problems are adopted to observe the experimental results due to the typical shape of their Pareto fronts. The setting of the other parameters, such as the number of evaluations, population size, is the same as in previous experiments for multi-objective optimization.

The two left images in Figure 3.9 show the results of only rotating one edge of the cone on the concave DTLZ2 problem. The black points give the entire Pareto front

Table 3.8: The mean hypervolume (M-HV) and mean IGD (M-IGD) on DTLZ1.

Four objective (NE = 100000)								
Algorithms	Metrics	P_cone	30°	20°	15°	10°	6°	3°
NSGA-II	M-HV	0.5811	**0.7735**	**0.9405**	**0.9403**	**0.9400**	**0.9393**	**0.9398**
	std	0.3347	0.1918	0.0024	0.0021	0.0026	0.0021	0.0018
DI-MOEA	M-HV	0.4842	0.0000	**0.9535**	**0.9537**	**0.9521**	**0.9538**	**0.9533**
	std	0.4250	0.0000	0.0010	0.0009	0.0056	0.0005	0.0009
NSGA-III	M-HV	0.9447	0.6399	**0.9448**	0.9444	**0.9458**	**0.9453**	**0.9452**
	std	0.0024	0.2561	0.0020	0.0024	0.0020	0.0018	0.0028
NSGA-II	M-IGD	0.9725	**0.3083**	**0.1537**	**0.1550**	**0.1553**	**0.1491**	**0.1511**
	std	0.0044	0.0444	0.0036	0.0026	0.0037	0.0046	0.0034
DI-MOEA	M-IGD	1.2772	763.0901	**0.1287**	**0.1287**	**0.1329**	**0.1303**	**0.1311**
	std	1.4694	9.6585	0.0021	0.0026	0.0128	0.0019	0.0027
NSGA-III	M-IGD	0.1300	0.4122	**0.1297**	**0.1295**	0.1315	**0.1298**	0.1313
	std	0.0024	0.2506	0.0038	0.0031	0.0029	0.0027	0.0032
Six objective (NE = 100000)								
NSGA-II	M-HV	0.0000	0.0000	**0.9857**	**0.9851**	**0.9844**	**0.9808**	**0.8922**
	std	0.0000	0.0000	0.0007	0.0007	0.0010	0.0023	0.2001
DI-MOEA	M-HV	0.0000	0.0000	**0.9911**	**0.9911**	**0.9906**	**0.9885**	**0.9728**
	std	0.0000	0.0000	0.0002	0.0003	0.0002	0.0019	0.0084
NSGA-III	M-HV	0.9880	0.0000	**0.9887**	**0.9885**	**0.9883**	**0.9881**	**0.9883**
	std	0.0009	0.0000	0.0005	0.0006	0.0005	0.0008	0.0006
NSGA-II	M-IGD	75.4078	744.6850	**0.3041**	**0.3026**	**0.3079**	**0.3256**	**0.4541**
	std	41.1790	49.7971	0.0169	0.0136	0.0183	0.0159	0.2359
DI-MOEA	M-IGD	349.0537	769.4755	**0.3086**	**0.3102**	**0.3151**	**0.3196**	**0.3791**
	std	76.0015	37.4396	0.0050	0.0043	0.0064	0.0104	0.0291
NSGA-III	M-IGD	0.2990	770.0300	**0.2935**	0.3007	0.3020	0.3015	0.3020
	std	0.0101	40.8585	0.0050	0.0085	0.0095	0.0085	0.0092
Eight objective (NE = 120000)								
NSGA-II	M-HV	0.0000	0.0000	**0.9957**	**0.9956**	**0.9937**	**0.9422**	**0.7397**
	std	0.0000	0.0000	0.0003	0.0004	0.0005	0.1638	0.3584
DI-MOEA	M-HV	0.0000	0.0000	**0.9976**	**0.9976**	**0.9965**	**0.8700**	**0.2850**
	std	0.0000	0.0000	0.0001	0.0002	0.0007	0.2892	0.3758
NSGA-III	M-HV	0.9877	0.0000	0.9855	0.9858	0.9853	0.9865	0.9854
	std	0.0025	0.0000	0.0027	0.0038	0.0032	0.0042	0.0025
NSGA-II	M-IGD	128.0384	721.0803	**0.4286**	**0.4272**	**0.4452**	**0.5575**	**0.8845**
	std	56.8022	57.7441	0.0199	0.0148	0.0232	0.2231	0.4798
DI-MOEA	M-IGD	517.2231	758.8918	**0.4843**	**0.4866**	**0.5043**	**0.8457**	**3.3619**
	std	108.7324	142.8642	0.0068	0.0056	0.0106	0.6234	3.5900
NSGA-III	M-IGD	0.3599	418.6033	**0.3461**	**0.3567**	**0.3565**	**0.3557**	**0.3594**
	std	0.0113	43.8714	0.0106	0.0106	0.0123	0.0192	0.0096
Eight objective - Half budget (NE = 60000)								
NSGA-II	M-HV	0.0000	0.0000	**0.9954**	**0.9944**	**0.7331**	**0.2971**	**0.2048**
	std	0.0000	0.0000	0.0006	0.0012	0.3764	0.3660	0.3120
DI-MOEA	M-HV	0.0000	0.0000	**0.9634**	**0.9972**	**0.9861**	**0.7335**	**0.0745**
	std	0.0000	0.0000	0.0863	0.0003	0.0341	0.3592	0.1555
NSGA-III	M-HV	0.9813	0.0000	**0.9855**	**0.9842**	**0.9849**	**0.9856**	**0.9863**
	std	0.0138	0.0000	0.0027	0.0033	0.0033	0.0034	0.0027
NSGA-II	M-IGD	170.7728	681.8762	**0.4248**	**0.4248**	**1.0092**	**2.3994**	**3.2542**
	std	92.0427	64.1913	0.0204	0.0151	0.8559	2.0576	3.1045
DI-MOEA	M-IGD	592.0768	747.5064	**0.5889**	**0.4881**	**0.5406**	**1.2579**	**4.0558**
	std	93.9853	94.6608	0.2782	0.0087	0.0983	1.1140	2.8857
NSGA-III	M-IGD	0.3777	405.6668	**0.3509**	**0.3576**	**0.3635**	**0.3663**	**0.3627**
	std	0.0588	48.9879	0.0192	0.0107	0.0124	0.0216	0.0169

Table 3.9: The mean hypervolume (M-HV) and mean IGD (M-IGD) on DTLZ2 (concave).

Four objective (NE = 130000)								
Algorithms	Metrics	P_cone	30°	20°	15°	10°	6°	3°
NSGA-II	M-HV	0.5953	0.1971	0.5458	**0.6760**	**0.6525**	**0.6388**	**0.6333**
	std	0.0089	0.1182	0.0535	0.0041	0.0048	0.0080	0.0077
DI-MOEA	M-HV	0.6471	0.0913	0.5639	**0.6944**	**0.6897**	**0.6755**	**0.6688**
	std	0.0094	0.0012	0.0406	0.0038	0.0026	0.0066	0.0039
NSGA-III	M-HV	0.6597	0.2508	0.5749	**0.6863**	**0.6821**	**0.6652**	0.6592
	std	0.0054	0.1265	0.0362	0.0017	0.0040	0.0031	0.0066
NSGA-II	M-IGD	0.1634	0.8352	0.4037	0.1867	**0.1492**	**0.1536**	**0.1542**
	std	0.0045	0.2290	0.0794	0.0056	0.0040	0.0055	0.0041
DI-MOEA	M-IGD	0.1363	1.0405	0.3810	0.1731	**0.1264**	**0.1295**	**0.1279**
	std	0.0045	0.0183	0.0661	0.0049	0.0022	0.0061	0.0028
NSGA-III	M-IGD	0.1501	0.7553	0.3510	0.1749	**0.1361**	**0.1477**	**0.1490**
	std	0.0046	0.2196	0.0705	0.0039	0.0034	0.0054	0.0026
Six objective (NE = 150000)								
NSGA-II	M-HV	0.1224	0.0000	**0.4304**	**0.8156**	**0.7608**	**0.7284**	**0.6490**
	std	0.0701	0.0000	0.0254	0.0036	0.0067	0.0119	0.0221
DI-MOEA	M-HV	0.0000	0.0000	**0.4488**	**0.8397**	**0.8016**	**0.7479**	**0.6543**
	std	0.0000	0.0000	0.0126	0.0055	0.0055	0.0117	0.0347
NSGA-III	M-HV	0.8052	0.0000	0.4411	**0.8446**	**0.8185**	**0.8127**	**0.8111**
	std	0.0076	0.0000	0.0130	0.0048	0.0038	0.0056	0.0041
NSGA-II	M-IGD	0.7278	2.5612	0.7003	**0.3447**	**0.2856**	**0.2887**	**0.3137**
	std	0.0758	0.0090	0.0380	0.0119	0.0051	0.0046	0.0091
DI-MOEA	M-IGD	1.9390	2.5824	**0.6961**	**0.2913**	**0.2774**	**0.2898**	**0.3335**
	std	0.3246	0.0059	0.0285	0.0074	0.0026	0.0058	0.0172
NSGA-III	M-IGD	0.3125	2.5596	0.7260	**0.3073**	**0.3061**	**0.3092**	**0.3095**
	std	0.0105	0.0154	0.0283	0.0145	0.0071	0.0065	0.0080
Eight objective (NE = 170000)								
NSGA-II	M-HV	0.0168	0.0000	**0.4947**	**0.8850**	**0.8193**	**0.7068**	**0.4062**
	std	0.0355	0.0000	0.0576	0.0068	0.0068	0.0487	0.0754
DI-MOEA	M-HV	0.0000	0.0000	**0.4250**	**0.9002**	**0.8011**	**0.4619**	**0.0138**
	std	0.0000	0.0000	0.1260	0.0033	0.0196	0.1500	0.0516
NSGA-III	M-HV	0.8543	0.0000	0.3151	**0.9079**	**0.8727**	**0.8632**	0.8522
	std	0.0121	0.0000	0.0643	0.0044	0.0074	0.0078	0.0138
NSGA-II	M-IGD	1.2941	2.4798	**0.7887**	**0.5247**	**0.3955**	**0.4332**	**0.6433**
	std	0.1867	0.0422	0.0507	0.0210	0.0068	0.0201	0.0687
DI-MOEA	M-IGD	2.4722	2.5704	**0.8728**	**0.4483**	**0.4425**	**0.6013**	**2.3017**
	std	0.0430	0.0129	0.1118	0.0054	0.0088	0.0682	0.4257
NSGA-III	M-IGD	0.4594	1.9278	0.9662	0.4936	0.4659	0.4638	0.4680
	std	0.0105	0.1043	0.0491	0.0130	0.0099	0.0093	0.0175
Eight objective - Half budget (NE = 85000)								
NSGA-II	M-HV	0.0001	0.0000	**0.4674**	**0.8859**	**0.8161**	**0.7145**	**0.4251**
	std	0.0003	0.0000	0.0847	0.0047	0.0083	0.0334	0.0851
DI-MOEA	M-HV	0.0000	0.0000	**0.4196**	**0.9000**	**0.8061**	**0.5432**	**0.0213**
	std	0.0000	0.0000	0.1254	0.0050	0.0207	0.0931	0.0606
NSGA-III	M-HV	0.8526	0.0000	0.3223	**0.9063**	**0.8728**	**0.8616**	**0.8548**
	std	0.0084	0.0000	0.0553	0.0048	0.0054	0.0085	0.0116
NSGA-II	M-IGD	1.6856	2.4963	**0.8125**	**0.5167**	**0.3939**	**0.4295**	**0.6116**
	std	0.1949	0.0202	0.0763	0.0091	0.0060	0.0126	0.0869
DI-MOEA	M-IGD	2.4858	2.5688	**0.8765**	**0.4520**	**0.4391**	**0.5633**	**2.0740**
	std	0.0272	0.0276	0.1149	0.0073	0.0072	0.0403	0.5132
NSGA-III	M-IGD	0.4611	1.9307	0.9590	0.4923	0.4691	0.4630	**0.4597**
	std	0.0178	0.1646	0.0433	0.0127	0.0115	0.0101	0.0152

Table 3.10: The mean hypervolume (M-HV) and mean IGD (M-IGD) on DTLZ2_convex.

Algorithms	Metrics	P_cone	30°	20°	15°	10°	6°	3°
Four objective (NE = 130000)								
NSGA-II	M-HV	0.4433	0.2126	0.4302	**0.4613**	**0.4577**	**0.4514**	**0.4502**
	std	0.0046	0.0286	0.0025	0.0019	0.0027	0.0037	0.0036
DI-MOEA	M-HV	0.4643	0.0427	0.4308	**0.4673**	**0.4730**	**0.4688**	**0.4678**
	std	0.0071	0.0048	0.0039	0.0051	0.0017	0.0025	0.0019
NSGA-III	M-HV	0.4419	0.1978	0.4182	**0.4501**	**0.4552**	**0.4499**	**0.4470**
	std	0.0078	0.0329	0.0037	0.0036	0.0025	0.0053	0.0036
NSGA-II	M-IGD	0.1484	0.5018	0.2137	0.1512	**0.1454**	**0.1466**	**0.1458**
	std	0.0044	0.0444	0.0036	0.0026	0.0037	0.0046	0.0034
DI-MOEA	M-IGD	0.1284	0.7288	0.2108	0.1426	**0.1238**	**0.1252**	**0.1255**
	std	0.0093	0.0154	0.0055	0.0074	0.0017	0.0034	0.0026
NSGA-III	M-IGD	0.1471	0.5242	0.2295	0.1660	**0.1424**	**0.1439**	**0.1424**
	std	0.0094	0.0511	0.0052	0.0052	0.0031	0.0067	0.0043
Six objective (NE = 150000)								
NSGA-II	M-HV	0.1299	0.0223	**0.1304**	**0.1471**	**0.1376**	**0.1348**	**0.1325**
	std	0.0029	0.0042	0.0016	0.0017	0.0018	0.0027	0.0023
DI-MOEA	M-HV	0.1343	0.0133	0.1280	**0.1525**	**0.1408**	**0.1376**	**0.1365**
	std	0.0018	0.0009	0.0019	0.0011	0.0014	0.0017	0.0020
NSGA-III	M-HV	0.0993	0.0072	**0.1109**	**0.1386**	**0.1234**	**0.1116**	**0.1045**
	std	0.0078	0.0010	0.0026	0.0027	0.0045	0.0061	0.0072
NSGA-II	M-IGD	0.2713	0.5058	0.4106	0.2789	**0.2655**	**0.2686**	**0.2698**
	std	0.0047	0.0282	0.0043	0.0049	0.0046	0.0053	0.0054
DI-MOEA	M-IGD	0.2571	0.6012	0.4149	0.2657	**0.2513**	**0.2530**	**0.2557**
	std	0.0030	0.0044	0.0050	0.0054	0.0038	0.0029	0.0028
NSGA-III	M-IGD	0.2911	0.7106	0.4557	0.3039	**0.2677**	**0.2764**	**0.2869**
	std	0.0093	0.0245	0.0074	0.0110	0.0106	0.0070	0.0073
Eight objective (NE = 170000)								
NSGA-II	M-HV	0.0276	0.0155	0.0187	**0.0355**	**0.0298**	**0.0292**	**0.0283**
	std	0.0010	0.0013	0.0029	0.0005	0.0011	0.0007	0.0008
DI-MOEA	M-HV	0.0264	0.0213	0.0151	**0.0357**	**0.0280**	**0.0269**	**0.0267**
	std	0.0008	0.0007	0.0004	0.0005	0.0006	0.0007	0.0009
NSGA-III	M-HV	0.0210	0.0014	0.0127	**0.0256**	**0.0219**	**0.0211**	0.0206
	std	0.0010	0.0013	0.0005	0.0009	0.0015	0.0010	0.0014
NSGA-II	M-IGD	0.3649	0.4218	0.5285	**0.3607**	**0.3548**	**0.3573**	**0.3607**
	std	0.0087	0.0083	0.0236	0.0040	0.0061	0.0086	0.0067
DI-MOEA	M-IGD	0.3816	0.3946	0.5611	**0.3597**	**0.3736**	**0.3788**	**0.3803**
	std	0.0036	0.0047	0.0029	0.0048	0.0044	0.0057	0.0050
NSGA-III	M-IGD	0.4197	0.7074	0.5811	**0.4178**	**0.4176**	0.4198	0.4211
	std	0.0094	0.0272	0.0037	0.0073	0.0136	0.0095	0.0120
Eight objective - Half budget (NE = 85000)								
NSGA-II	M-HV	0.0282	0.0152	0.0187	**0.0356**	**0.0304**	**0.0293**	**0.0286**
	std	0.0007	0.0012	0.0025	0.0006	0.0006	0.0007	0.0010
DI-MOEA	M-HV	0.0263	0.0217	0.0150	**0.0359**	**0.0276**	**0.0268**	**0.0266**
	std	0.0010	0.0008	0.0005	0.0006	0.0006	0.0007	0.0008
NSGA-III	M-HV	0.0202	0.0012	0.0126	**0.0249**	**0.0213**	**0.0207**	0.0200
	std	0.0013	0.0009	0.0005	0.0009	0.0008	0.0010	0.0014
NSGA-II	M-IGD	0.3649	0.4218	0.5285	**0.3607**	**0.3548**	**0.3573**	**0.3607**
	std	0.0087	0.0083	0.0236	0.0040	0.0061	0.0086	0.0067
DI-MOEA	M-IGD	0.3801	0.3937	0.5616	**0.3588**	**0.3784**	**0.3796**	**0.3792**
	std	0.0071	0.0046	0.0037	0.0039	0.0065	0.0074	0.0035
NSGA-III	M-IGD	0.4263	0.7102	0.5815	**0.4210**	**0.4238**	**0.4234**	**0.4250**
	std	0.0113	0.0213	0.0042	0.0093	0.0086	0.0086	0.0110

Table 3.11: The mean hypervolume (M-HV) and mean IGD (M-IGD) on UF11 & UF13.

UF11 Five objective (NE = 300000)								
Algorithms	Metrics	P_cone	30°	20°	15°	10°	6°	3°
NSGA-II	M-HV	0.0000	0.0000	**0.0211**	**0.0291**	**0.0306**	**0.0218**	**0.0104**
	std	0.0000	0.0000	0.0024	0.0058	0.0012	0.0011	0.0014
DI-MOEA	M-HV	0.0029	0.0000	**0.0191**	**0.0336**	**0.0256**	**0.0188**	**0.0138**
	std	0.0018	0.0000	0.0035	0.0008	0.0012	0.0015	0.0024
NSGA-III	M-HV	0.0147	0.0000	**0.0266**	**0.0350**	**0.0278**	**0.0201**	**0.0171**
	std	0.0016	0.0000	0.0034	0.0017	0.0016	0.0014	0.0015
NSGA-II	M-IGD	1.5208	14.6626	**0.3890**	**0.2990**	**0.2685**	**0.3119**	**0.4531**
	std	0.2173	0.2878	0.0368	0.0374	0.0171	0.0241	0.0289
DI-MOEA	M-IGD	0.7304	15.1690	**0.6152**	**0.2807**	**0.3339**	**0.3946**	**0.4621**
	std	0.0944	0.2054	0.1997	0.0210	0.0228	0.0352	0.0545
NSGA-III	M-IGD	0.4517	15.0785	**0.4190**	**0.2795**	**0.3188**	**0.3848**	**0.4166**
	std	0.0388	0.2105	0.0697	0.0247	0.0235	0.0324	0.0183
UF11 Five objective - Half budget (NE = 150000)								
NSGA-II	M-HV	0.0000	0.0000	**0.0205**	**0.0269**	**0.0288**	**0.0201**	**0.0082**
	std	0.0000	0.0000	0.0025	0.0055	0.0014	0.0016	0.0017
DI-MOEA	M-HV	0.0012	0.0000	**0.0237**	**0.0316**	**0.0244**	**0.0185**	**0.0126**
	std	0.0011	0.0000	0.0030	0.0020	0.0010	0.0014	0.0017
NSGA-III	M-HV	0.0148	0.0000	**0.0268**	**0.0342**	**0.0270**	**0.0199**	**0.0170**
	std	0.0020	0.0000	0.0029	0.0013	0.0018	0.0016	0.0010
NSGA-II	M-IGD	1.7202	14.7243	**0.3951**	**0.3031**	**0.2731**	**0.3208**	**0.4846**
	std	0.2541	0.1769	0.0392	0.0343	0.0164	0.0289	0.0312
DI-MOEA	M-IGD	0.8730	15.1172	**0.4910**	**0.2939**	**0.3418**	**0.4061**	**0.4831**
	std	0.1485	0.2099	0.0619	0.0269	0.0244	0.0329	0.0439
NSGA-III	M-IGD	0.4606	15.0148	**0.3897**	**0.2752**	**0.3204**	**0.4009**	**0.4314**
	std	0.0433	0.1881	0.0615	0.0186	0.0265	0.0393	0.0335
UF13 Five objective (NE = 300000)								
NSGA-II	M-HV	0.6937	0.5041	**0.7410**	**0.7424**	**0.7177**	**0.7065**	**0.6994**
	std	0.0079	0.1742	0.0096	0.0070	0.0091	0.0084	0.0084
DI-MOEA	M-HV	0.6611	0.4625	**0.7343**	**0.7152**	0.6590	0.6567	0.6589
	std	0.0063	0.1580	0.0064	0.0119	0.0073	0.0067	0.0071
NSGA-III	M-HV	0.6498	0.4523	**0.7164**	**0.7226**	**0.7023**	**0.6703**	**0.6532**
	std	0.0130	0.1017	0.0048	0.0108	0.0085	0.0106	0.0077
NSGA-II	M-IGD	1.4761	**1.3108**	**1.4316**	**1.3805**	**1.4656**	**1.4391**	**1.4181**
	std	0.1315	0.2267	0.0565	0.0857	0.0664	0.1572	0.1029
DI-MOEA	M-IGD	1.5448	**1.5031**	**1.5151**	1.5481	1.7512	1.6351	1.5934
	std	0.0473	0.4180	0.0533	0.0646	0.0384	0.0667	0.0399
NSGA-III	M-IGD	1.8698	**1.6030**	**1.6324**	**1.5813**	**1.6675**	**1.7950**	**1.8527**
	std	0.1842	0.1835	0.0285	0.0658	0.0969	0.1457	0.1245
UF13 Five objective - Half budget (NE = 150000)								
NSGA-II	M-HV	0.6687	0.5016	**0.7259**	**0.7170**	**0.6915**	**0.6831**	**0.6738**
	std	0.0041	0.1749	0.0092	0.0058	0.0042	0.0047	0.0057
DI-MOEA	M-HV	0.6457	0.3427	**0.7254**	**0.7002**	**0.6513**	**0.6481**	**0.6497**
	std	0.0045	0.2041	0.0044	0.0133	0.0056	0.0053	0.0057
NSGA-III	M-HV	0.6432	0.4702	**0.7073**	**0.7045**	**0.6770**	**0.6579**	0.6417
	std	0.0086	0.0996	0.0074	0.0076	0.0103	0.0071	0.0056
NSGA-II	M-IGD	1.5720	1.3736	**1.5455**	**1.5074**	1.5968	1.5746	**1.5262**
	std	0.0946	0.1703	0.0638	0.0649	0.0786	0.1135	0.0860
DI-MOEA	M-IGD	1.6609	**1.5321**	**1.5939**	**1.6311**	1.8048	1.7286	1.6403
	std	0.0557	0.3781	0.0268	0.0781	0.0509	0.0794	0.0613
NSGA-III	M-IGD	1.8931	1.7553	**1.6824**	**1.6832**	**1.8163**	1.8976	1.9725
	std	0.1238	0.2361	0.0456	0.0376	0.0924	0.1200	0.0562

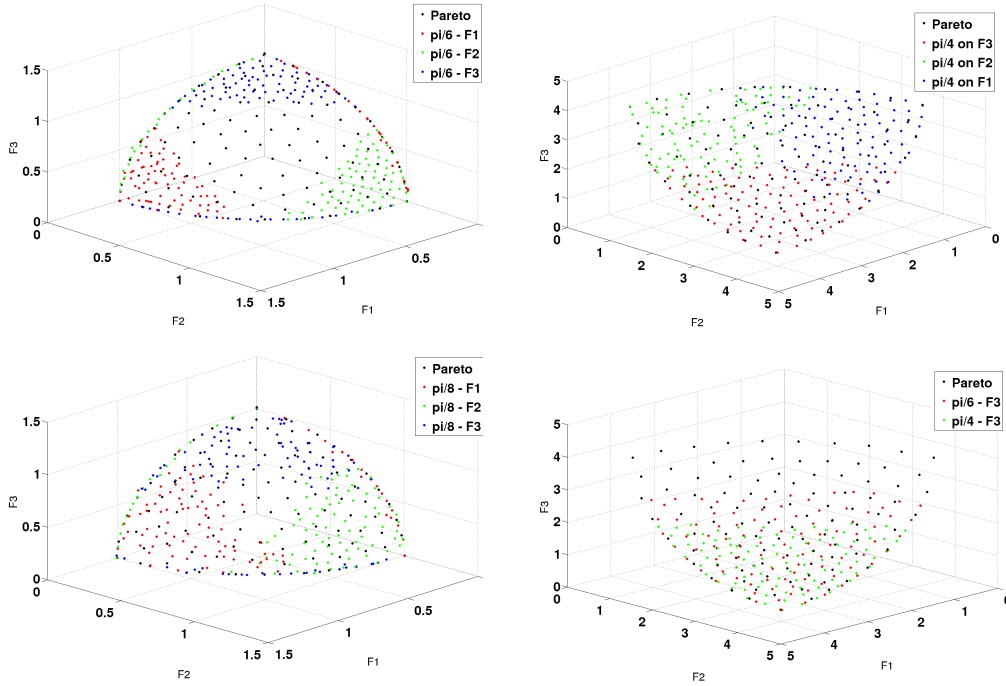


Figure 3.9: Pareto front approximations of (concave & convex) DTLZ2 obtained by the cone with different rotation angles on different edges.

from original DI-MOEA. The blue, green and red points in the top image show the results when only $F3$, $F2$, $F1$ of the standard Pareto cone is rotated outside by 30° ($\pi/6$) respectively and the other two edges of the cone remain. In the bottom image, the rotation angle is 22.5° ($\pi/8$) on the extended edge and the other edges are the same as those of the Pareto cone. Under the condition that the original Pareto front is concave, it can be observed that the achieved Pareto front focuses on a different corner and the side right against the corner of the entire Pareto front when only one edge of the cone is rotated. Moreover, the smaller the angle, the larger the covered Pareto front area. The right two images in Figure 3.9 present the results on the convex DTLZ2 problem. The points with different colors in the top image, again, show the Pareto fronts when only one edge of the cone is rotated and the rotation angle is 45° ($\pi/4$). It can be seen that the solutions focus on different corners. The bottom image shows again when the rotation angle is larger, the Pareto front can be narrowed down and concentrate more on the corner. The literature of the preference-based MOEAs mostly focus on the knee or central region of the Pareto front, however, the edge-

rotated cones make it possible to obtain preferred solutions if the "corner" regions are interested, i.e., the edges of the cone are rotated by different angles to express the preference on different objectives.

3.2.5 Conclusion and Future Work

The edge-rotated cone is generated by simply rotating the edges of the Pareto cone. The edge-rotated cone order, when used as the ranking criterion by MOEAs, can rank the Pareto incomparable solutions into different layers. Hence, the selection pressure toward the Pareto front can be strengthened and the convergence of the algorithm can be accelerated. To avoid neglecting the diversity, the edge-rotated cone order is designed to work together with the standard Pareto order in MOEAs. After testing the edge-rotated cones with various rotation angles on multi-objective and many-objective optimization problems and comparing their performance with the original MOEAs, it can be seen that many-objective optimization can really benefit from the edge-rotated cones. To be specific, a cone with a relatively small rotation angle ($> 3°$, e.g., $6°$) can almost always improve the performance of original algorithms. On many-objective optimization problems, the best behavior usually appears when an intermediate rotation angle (e.g., $15°$) is adopted. However, on multi-objective problems, the rotation angles which can achieve the best performance are usually smaller. In the experiment, it can also be observed that the performance of NSGA-II integrating the edge-rotated cone can reach the performance of NSGA-III in some cases. Moreover, when the algorithm uses a small computing budget and edge-rotated cones, it can achieve better behavior than when the algorithm uses a large budget but without edge-rotated cones.

From the experimental results, it can be seen that a smaller rotation angle is more suitable in low dimensions than in high dimensions. The reason is high dimensional problems need strong convergence. Otherwise, it is difficult to find a good Pareto front approximation. Therefore, a larger rotation angle is needed by many-objective problems. However, a larger rotation angle also leads to the focus of the search on a smaller region of the Pareto front (Please refer to Figure 3.9), MOEAs then need to find a good balance between the convergence and coverage to avoid that the obtained solution set covers only a part of region of the Pareto front. It has also been found that the properties of the problem determine the performance of a specific rotation angle more than the objective number. For example, the rotation angle which can achieve the best performance on DTLZ1 problems is always higher than on other problems, no matter the number of objectives is three, four or eight, and no matter whether the

edge-rotated cone is integrated in NSGA-II or NSGA-III. The reason behind this is that the Pareto front of DTLZ1 problems is linear. All points on the linear Pareto front can be found even when the rotation angle is large. Therefore, a large rotation angle can be used to improve the convergence ability of MOEAs without deteriorating the coverage of the Pareto front. However, when the Pareto front is non-linear, it is possible that only a part of the Pareto front can be obtained by a large rotation angle. In future, the mechanism that relates the properties of the problem with the rotation angle should be researched more. Furthermore, when the edges of the Pareto cone are rotated by different angles, the obtained Pareto front approximation can focus on the different region of the entire Pareto front, and these "unsymmetrical" cones are very promising to be used when exists different emphasis on the Pareto front. However, further research on its ability on articulating the preference on both multi-objective and many-objective optimization should be done. In general, the use of cone orders to formulate preferences based on trade-off rates and angles will be a topic that deserves also attention for problems with a larger number of objectives and tools to better guide users in choosing their preferences will be of crucial importance to improve applicability in practice.

Chapter 4

Preference-based Multi-objective Evolutionary Algorithms

With the MOEAs, the entire Pareto front of a multi-objective optimization problem can be approximated. However, finding a well-distributed set of solutions on the Pareto front requests a large population size and computational effort. At the same time, the final goal of EMO is to help the DM to find solutions which match his/her preferences most, and the DM may only pay attention to a smaller set of Pareto optimal solutions. Therefore, integrating preferences in solving MOPs has become the subject of intensive studies of EMO. In other words, instead of spreading a limited size of individuals across the entire Pareto front, the search for solutions will be only guided towards the preference region.

The existing preference-based optimization methods can be classified into three categories according to the time when preference information is incorporated, i.e., a priori, interactive, and a posteriori methods. In a priori methods, the DM articulates preference information before the optimization process. In a posteriori methods, a set of Pareto optimal solutions is obtained first, and then the DM selects the most preferred ones among them. In interactive methods, the DM participates in the optimization process and directs the search according to his/her preferences. With the increasing understanding of the problem as the optimization proceeds, DMs are able to fine tune their preferences according to the obtained solutions in the optimization

process.

The preference based multi-objective evolutionary algorithms drive the population towards the region(s) of interest (ROI). The definition of the ROI depends on the way how the DM articulates his/her preference information. The preference information can be represented as the reference point(s) (e.g., [30, 43]); preference region(s) (e.g., [68, 79]), reference direction(s) (e.g., [28]); light beams (e.g., [27]), ranking (e.g., [44]), trade-offs (e.g., [95]), etc. However, the essence of preference information is to imply the ROI which allows a more focused search, thus to save computational resources.

This chapter is related to answer RQ3. First, a method that incorporates target region(s) into the core of the optimization process is proposed in Section 4.1. Then, in Section 4.2 to reduce the burden of the DM, an automatic preference based algorithm is proposed and integrated in DI-MOEA. In this algorithm, the preference region is generated automatically and narrowed down step by step to benefit its accuracy.

4.1 Target Region Based MOEAs

A target region based multi-objective evolutionary algorithm framework is proposed to find a more fine-grained resolution of a target region without exploring the whole set of Pareto optimal solutions. The idea of the target region is to first present a rough approximation of the Pareto front, then let the DM decide which region of interest to zoom in. Therefore, the target region is assumed to be the preference region according to the DM. The algorithm framework has been combined with SMS-EMOA, R2-EMOA, NSGA-II to form three target region based multi-objective evolutionary algorithms: T-SMS-EMOA, T-R2-EMOA and T-NSGA-II (where T stands for target).

NSGA-II is a frequently-used Pareto dominance-based MOEA; SMS-EMOA [9] and R2-EMOA [104] are indicator-based approaches which use performance measures (indicators) on the quality of the PF approximations to guide the search. The hypervolume is used in SMS-EMOA and the R2 indicator [55] is used in R2-EMOA. These two indicators measure both convergence and diversity of a PF approximation.

The R2 indicator of a solution set A is defined as

$$R2(A, \Lambda, \mathbf{i}) = \frac{1}{|\Lambda|} \sum_{\lambda \in \Lambda} \min_{a \in A} \{ \max_{j \in \{1,\dots,m\}} \{\lambda_j |\mathbf{i}_j - a_j|\}\}. \tag{4.1}$$

Here $\mathbf{i}$ is the ideal point and m is the objective number, $\lambda = (\lambda_1, ..., \lambda_d) \in \Lambda$ is a given set of weight vectors. Usually, the weight vectors are chosen uniformly distributed over the weight space, for example, for m = 2 objectives, $\Lambda_k = (0, 1; \frac{1}{k-1}, 1 -$

$\frac{1}{k-1}; \frac{2}{k-1}, 1-\frac{2}{k-1}; ...; 1, 0)$ denotes k uniformly distributed weights in the space $[0, 1]^2$.

The three proposed target region based algorithms have been tested with rectangular and spherical target regions on some benchmark problems, including continuous problems and discrete problems. Moreover, the proposed algorithms have been enhanced to support multiple target regions and preference information based on a target point or multiple target points.

The remainder of this section is organized as follows. In Section 4.1.1, the proposed algorithms are described. The experimental results are reported in Section 4.1.2. The details and graphical results of enhanced algorithms are presented in Section 4.1.3. Section 4.1.4 concludes the work with the summary and outlook.

4.1.1 Basic Algorithms

In the proposed algorithms, i.e., T-SMS-EMOA, T-R2-EMOA and T-NSGA-II, three ranking criteria: 1. non-dominated sorting; 2. performance indicator (Hypervolume in T-SMS-EMOA, R2 in T-R2-EMOA) and crowding distance in T-NSGA-II; 3. the Chebyshev distance to the target region, work together to achieve a well-converged and well-distributed set of Pareto optimal solutions in the target region using preference information provided by the DM. The Chebyshev distance speeds up evolution toward the target region and is computed as the distance to the center of the target region.

The second level ranking criterion: hypervolume, R2 indicator or crowding distance, is used as a diversity mechanism and measured based on coordinate transformations using desirability functions (DFs). The concept of desirability is introduced by Harrington [56] in the context of multi-objective industrial quality control and the approach of expressing the preferences of the DM using DFs is suggested by Wagner and Trautmann [112]. DFs map the objective values to desirabilities which are normalized values in the interval $[0, 1]$, where the larger the value, the more satisfying the quality of the objective value. The Harrington DF [56] and Derringer-Suich DF [100] are two most common types of DFs. By mapping the objective values to desirabilities according to preference information, both of these two DFs can result in biased distributions of the solutions on the PF. In the proposed algorithms, a simple DF 4.2 is used and it classifies the domain of the objective function into only two classes, "unacceptable" and "acceptable".

$$D(x) = \begin{cases} 1 & x \text{ is in the target region,} \\ 0 & x \text{ is not in the target region.} \end{cases} \tag{4.2}$$

The desirability here is for a solution. Solutions out of the target region are considered as unacceptable solutions and their desirabilities are assigned to be 0; at the same time, all solutions inside the target region are assumed acceptable and of equal importance, their desirabilities are assigned to be 1. For solutions with desirability 0, their second level ranking criterion is assigned to be 0 and for solutions with desirability 1, their second level ranking criterion needs to be calculated further. Since only solutions in the target region are retained, an approach is derived to simplify the calculation of the indicator values and realize a reference point free version of indicators [38], which is coordinate transformation. To be specific, the target region is treated as a new coordinate space of which the origin being the lower bound. For the maximization problem in T-SMS-EMOA or the minimization problem in T-R2-EMOA, a coordinate transformation is performed for the i-th objective as:

$$Ct_i(x) = f_i(x) - LB(f_i). \tag{4.3}$$

For minimization problem in T-SMS-EMOA or the maximization problem in T-R2-EMOA, coordinate transformation is performed for the i-th objective as:

$$Ct_i(x) = UB(f_i) - (f_i(x) - LB(f_i)) \tag{4.4}$$

where $LB(f_i)$ and $UB(f_i)$ are the lower bound and upper bound of the i-th objective in the target region predefined by the DM.

The reason for distinguishing the maximization and minimization problem in coordinate transformation is that the origin of the new coordinate space (i.e., the lower bound of the target region) is adopted as the reference point when calculating the indicator values. In T-SMS-EMOA, the worst point in the target region is chosen as the reference point when calculating hypervolume. On the contrary, the ideal point is chosen as the reference point when calculating R2 indicator in T-R2-EMOA. After completing coordinate transformation, the calculation of the second ranking criterion is implemented only in the target region instead of the whole coordinate system. It does make sense because the target region is the desired space to the DM. No reference point is needed in the calculation of crowding distance, therefore, any of the two formulas of coordinate transformation can be used in T-NSGA-II. Figure 4.1 shows an example of obtaining solutions in the target region by the proposed approach.

The target region is used to express the preference information from the DM, the shape of the target region does not necessarily need to be rectangular, it could as

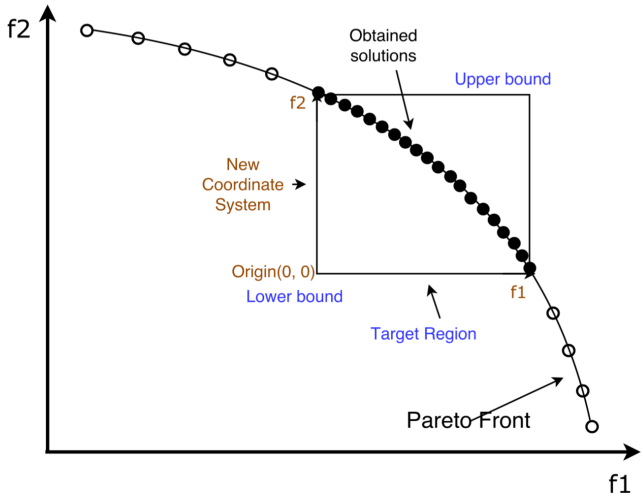


Figure 4.1: An example of obtaining solutions in the target region by proposed approach.

well be a circle, an ellipse or in other shapes as long as it can be confirmed efficiently whether or not a solution is in the target region. For instance, if the DM wants the solutions to be restricted into a sphere, s/he can specify the center point and radius of the sphere, the solution set can be achieved in the sphere.

T-SMS-EMOA

The details of T-SMS-EMOA are given in Algorithm 3.

The framework of T-SMS-EMOA is based on the framework of SMS-EMOA. However, after the step of non-dominated sorting, all solutions in the worst ranked front are partitioned into two parts (i.e., acceptable and unacceptable) by the DF. Solutions in the first part have desirability 0 and their hypervolume contributions are assigned to be 0. Solutions in the second part have desirability 1 and coordinate transformation is conducted on each objective of each solution in this part; afterwards, their hypervolume contributions are calculated in the new coordinate system and the origin in the new coordinate system is adopted as the reference point. The other difference between T-SMS-EMOA and SMS-EMOA is the involvement of the Chebyshev distance. In the early iterations, the existence of individuals in the target region is low, the Chebyshev distance works on attracting solutions towards the target region.

Algorithm 3 T-SMS-EMOA

```
1: P_0 ← init(); //Initialize random population
2: t ← 0;
3: while Stop criterion not satisfied() do
4:     q_{t+1} ← Gen(P_t); //Generate offspring by variation
5:     P_t ← P_t ∪ {q_{t+1}};
6:     {R_1, ··· , R_v} ← Nondominated-sort(P_t);
7:     ∀x ∈ R_v : compute D_Ch(x); //Chebyshev distance to center of target region
8:     R_v1 ∪ R_v2 ← R_v; //Solutions not in the target region → R_v1; others → R_v2
9:     ∀x ∈ R_v1 : HC(x) ← 0; //Hypervolume contribution
10:    R_v2 ← Coordinate Transformation(R_v2);
11:    ∀x ∈ R_v2 : HC(x) ← HV(R_v2) − HV(R_v2\x);
12:    if unique argmin{HC(x) : x ∈ R_v} exists then
13:        x* ← argmin{HC(x) : x ∈ R_v};
14:    else
15:        x* ← argmax{D_Ch(x) : x ∈ R_v}; //In case of tie, choose randomly
16:    end if
17:    P_{t+1} ← P\{x*};
18:    t ← t + 1;
19: end while
```

T-R2-EMOA

The details of T-R2-EMOA are given in Algorithm 4.

R2-EMOA is extended to T-R2-EMOA in the same way SMS-EMOA is extended to T-SMS-EMOA. The formula of coordinate transformation used in T-R2-EMOA, however, is opposite to the formula used in T-SMS-EMOA for the same problem since the origin of the new coordinate system is used as the reference point in the measure of both hypervolume indicator in T-SMS-EMOA and R2 indicator in T-R2-EMOA.

T-NSGA-II

The details of T-NSGA-II are given in Algorithm 5.

In T-NSGA-II, the size of the offspring population is the same as the size of the parent population, which is the specified population size. The next population is generated by choosing the best half solutions from the merged population: starting with points in the first non-domination front, continuing with points in the second non-domination front, and so on; if by adding all points in one front, the population size exceeds the specified population size, picking points in the descending order of crowding distance; if by adding all points with the same crowding distance, the population

Algorithm 4 T-R2-EMOA

$P_0 \leftarrow init()$; //Initialize random population
$t \leftarrow 0$;
while Stop criterion not satisfied() **do**
  $q_{t+1} \leftarrow Gen(P_t)$; //Generate offspring by variation
  $P_t \leftarrow P_t \cup \{q_{t+1}\}$;
  $\{R_1, \cdots, R_v\} \leftarrow$ Nondominated-sort(P_t);
  $\forall x \in R_v$: compute $D_{Ch}(x)$; //Chebyshev distance to center of target region
  $R_{v1} \cup R_{v2} \leftarrow R_v$; //Solutions not in the target region $\rightarrow R_{v1}$; others $\rightarrow R_{v2}$
  $\forall x \in R_{v1} : r(x) \leftarrow 0$; //R2 indicator contribution
  $R_{v2} \leftarrow$ Coordinate Transformation(R_{v2});
  $\forall x \in R_{v2} : r(x) \leftarrow R2(R_{v2} \backslash \{x\}; \Lambda; \mathbf{i})$; //$\mathbf{i}$ is ideal point
  if unique $argmin\{r(x) : x \in R_v\}$ exists **then**
    $x^* \leftarrow argmin\{r(x) : x \in R_v\}$;
  else
    $x^* \leftarrow argmax\{D_{Ch}(x) : x \in R_v\}$; //In case of tie, choose randomly
  end if
  $P_{t+1} \leftarrow P \backslash \{x^*\}$;
  $t \leftarrow t + 1$;
end while

Algorithm 5 T-NSGA-II

$P_0 \leftarrow init()$; //Initialize random population
$t \leftarrow 0$;
while Stop criterion not satisfied() **do**
  $Q_t \leftarrow Gen(P_t)$; //Generate offsprings by variation
  $P_t \leftarrow P_t \cup Q_t$;
  $\forall x \in P_t$: compute $D_{Ch}(x)$; //Chebyshev distance to center of target region
  $\{R_1, \cdots, R_v\} \leftarrow$ Nondominated-sort(P_t);
  for $i = 1, \cdots, v$ **do**
    $R_{i1} \cup R_{i2} \leftarrow R_i$; //Solutions not in the target region $\rightarrow R_{i1}$; others $\rightarrow R_{i2}$
    $\forall x \in R_{i1} : D_c(x) \leftarrow 0$; //Crowding distance
    $R_{i2} \leftarrow$ Coordinate Transformation(R_{i2});
    $\forall x \in R_{i2}$: compute $D_c(x)$;
  end for
  $P_{t+1} \leftarrow$ half the size of P_t based on rank, D_c and then D_{Ch};
  $t \leftarrow t + 1$;
end while

size still exceeds the specified population size, picking points in the ascending order of the Chebyshev distance. Unlike T-SMS-EMOA and T-R2-EMOA, no reference point is needed in T-NSGA-II.

4.1.2 Experimental Study

In this part, experiments are conducted on some benchmark problems, including ZDT, DTLZ and knapsack problems, to investigate performance of the proposed algorithms. In all experiments, the SBX operator with an index of 15 and polynomial mutation with an index 20 are used. The crossover and mutation probabilities are set to 1 and $1/L$, where L is the number of variables. The population size and the number of evaluations are chosen to be dependent on the complexity of the test problem. Table 4.1 shows the population size and the number of evaluations used on different test problems.

Table 4.1: Population Size and Number of Evaluation

Problems	Population Size	NE
ZDT1	100	10000
ZDT2-3	100	20000
DTLZ1-2	100	30000
knapsack-250-2 knapsack-500-2	200	200000
knapsack-250-3 knapsack-500-3	250	500000

Bi-objective ZDT Problems

Three bi-objective ZDT problems are considered. The 30-variable ZDT1 problem has a convex Pareto front which is a connected curve and can be determined by $f_2(x) = 1 - \sqrt{f_1(x)}$. The true PF spans continuously in $f_1 \in [0, 1]$. Four different target regions are chosen to observe the performance of T-SMS-EMOA, T-R2-EMOA and T-NSGA-II. The first target region covers the entire PF with the lower bound (0, 0) and the upper bound (1, 1). The second target region restricts preferred solutions to the central part of the PF and its lower bound is (0.1, 0.1), upper bound is (0.5, 0.5). The third and fourth target regions take two ends of the PF respectively and have their lower bounds to be (0, 0.6) and (0.6, 0), upper bounds to be (0.3, 1) and (1, 0.3).

Figure 4.2 $\sim$ Figure 4.4 show PF approximations obtained from the proposed algorithms on the four different target regions in a random single run. The target

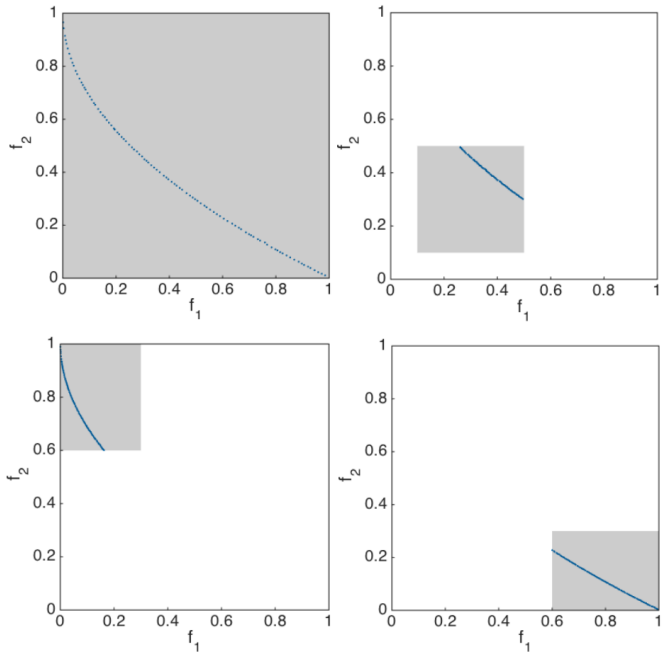


Figure 4.2: Representative PF approximations of T-SMS-EMOA on ZDT1.

regions are highlighted by gray boxes. It is observed that all three algorithms can find well-distributed and well-converged solutions on the PF in the target regions and no outliers exist. The solution set obtained by T-SMS-EMOA is more uniform than the solution sets obtained by the other two algorithms. It is also observable from the upper left graph in Figure 4.3 that the R2 indicator has a bias towards the center of the PF.

When examining the performance by the hypervolume metric, the hypervolume value of the obtained solution set is calculated within the target region by normalizing the values of each objective to the values between 0 and 1 and using the lower bound of the target region as the reference point for the maximization problem and the upper bound of target region as the reference point for the minimization problem. Table 4.2 shows the median and variance of hypervolume over 30 runs. The statistical results correspond to the observation that T-SMS-EMOA outperforms T-R2-EMOA and T-NSGA-II slightly. The original SMS-EMOA, R2-EMOA and NSGA-II are also involved in the comparison and the results of the original MOEAs are obtained by firstly presenting the target region as constraints in the description of the problem;

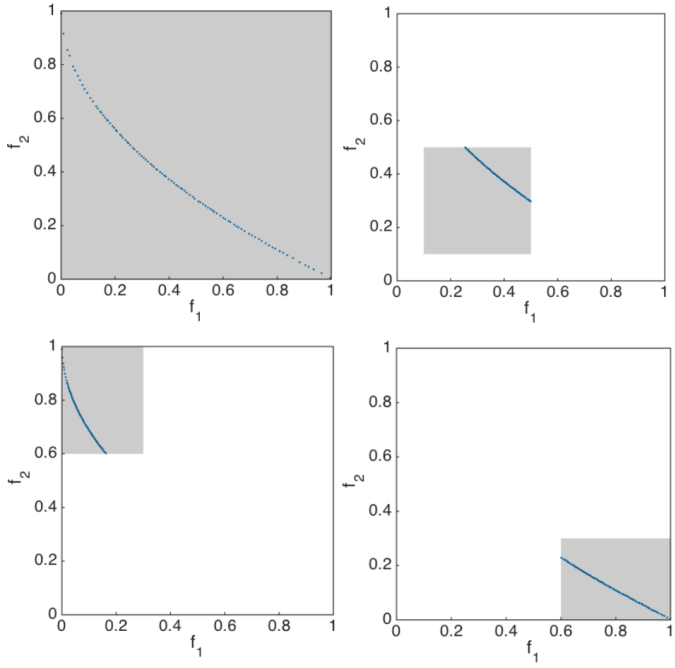


Figure 4.3: Representative PF approximations of T-R2-EMOA on ZDT1.

and secondly, making a change in the description of the problem. It is demonstrated that the new algorithms obtain higher hypervolume value than original MOEAs with no constraint descriptions in the problem. Although the results of the proposed algorithms are not better than original MOEAs with constraints on the range of objectives, experiments show that the proposed algorithms can reduce computation time dramatically on this problem.

In the table, the symbol of $*$ on the values for the same target region means the medians of these algorithms are significantly indifferent. The Mann-Whitney U test (also called the Mann-Whitney-Wilcoxon, Wilcoxon rank-sum test, or Wilcoxon-Mann-Whitney test) is used to determine if the medians of different algorithms for the same problem are significantly indifferent. The chances that the medians of T-SMS-EMOA and T-R2-EMOA are indifferent have been observed.

Next, circle target regions are adopted on the 30-variable ZDT2 and ZDT3 problems. ZDT2 has a concave Pareto front and ZDT3 has a disconnected set of Pareto front which consists of five non-contiguous convex parts. A circle with a center point (1, 0) and radius 0.5 intersects the whole PF of ZDT2 at its one end and a circle with

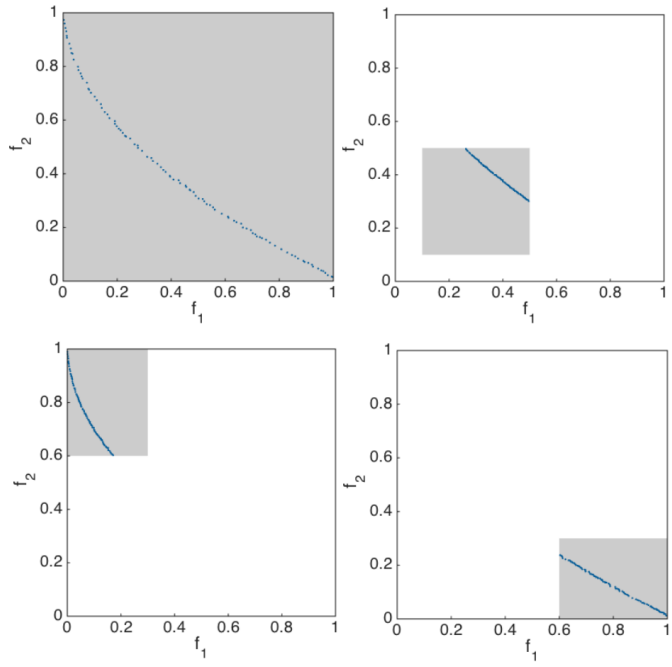


Figure 4.4: Representative PF approximations of T-NSGA-II on ZDT1.

a center point (0.6, 0.5) and radius 0.3 intersects the whole PF at its central part. The two different circles are chosen as examples for target regions on ZDT2 problem. Experiments for a circle with a center point (0.3, 0.1) and radius 0.3 as target region are conducted on ZDT3 problem.

Figure 4.5 shows PF approximations from T-SMS-EMOA in these target regions. Similar figures can also be achieved by T-R2-EMOA and T-NSGA-II. In the graph, the target regions are purple circles and center points are red points. Orange points denote the results obtained from T-SMS-EMOA with provided preference information. The blue points show the entire true PF of ZDT2 and ZDT3 problems. Statistical results of the median of hypervolume for three algorithms in 30 independent runs on each target region are shown in Table 4.3.

Tri-objective DTLZ Problems

Next, tri-objective DTLZ1 and DTLZ2 problems are involved in the experiments. The 7-variable DTLZ1 problem has a linear Pareto optimal front which is a three-

Table 4.2: The median, variance of hypervolume and average computation time (Sec.) on ZDT1 with respect to different target regions and different algorithms

New Algorithms		T-SMS-EMOA	T-R2-EMOA	T-NSGA-II
Target Region	Metric			
(0,0)(1,1)	HV(m)	0.6580	0.6566	0.6425
	Variance	6.4e-06	1.4e-06	1.0e-05
	Time	24.99	74.01	0.21
(0.1,0.1)(0.5,0.5)	HV(m)	0.1640*	0.1638*	0.1543
	Variance	1.4e-06	1.5e-06	8.9e-06
	Time	10.30	23.61	0.19
(0,0.6)(0.3,1)	HV(m)	0.8110	0.8103	0.7936
	Variance	5.9e-06	6.0e-06	4.4e-05
	Time	12.86	31.78	0.20
(0.6,0)(1,0.3)	HV(m)	0.6255*	0.6233*	0.6079
	Variance	8.9e-06	6.7e-06	4.5e-05
	Time	11.45	27.92	0.21
Original Algorithms (Constraints)		SMS-EMOA	R2-EMOA	NSGA-II
(0,0)(1,1)	HV(m)	0.6621	0.6610	0.6609
	Variance	8.9e-11	1.2e-08	5.3e-08
	Time	108.57	314.99	0.25
(0.1,0.1)(0.5,0.5)	HV(m)	0.1694	0.1693	0.1690
	Variance	1.6e-11	1.1e-11	6.2e-09
	Time	106.32	274.05	0.23
(0,0.6)(0.3,1)	HV(m)	0.8197	0.8185	0.8191
	Variance	1.6e-08	4.6e-08	2.9e-08
	Time	105.73	271.00	0.21
(0.6,0)(1,0.3)	HV(m)	0.6364	0.6348	0.6356
	Variance	3.2e-09	2.2e-08	3.8e-08
	Time	101.82	283.3	0.22
Original Algorithms		SMS-EMOA	R2-EMOA	NSGA-II
(0,0)(1,1)	HV(m)	0.6558	0.6566	0.6362
	Variance	1.6e-06	8.5e-07	3.5e-05
	Time	26.77	73.34	0.21
(0.1,0.1)(0.5,0.5)	HV(m)	0.1545	0.1585	0.1236
	Variance	4.7e-06	2.2e-06	4.4e-05
	Time	24.17	74.85	0.20
(0,0.6)(0.3,1)	HV(m)	0.8012	0.7972	0.7649
	Variance	6.3e-06	6.4e-06	0.00013
	Time	24.85	71.90	0.20
(0.6,0)(1,0.3)	HV(m)	0.6119*	0.6110*	0.5604
	Variance	2.3e-05	7.4e-06	0.00014
	Time	26.29	78.93	0.20

Table 4.3: The median hypervolume of ZDT2 and ZDT3 with a circular target region.

Algorithm	T-SMS-EMOA	T-R2-EMOA	T-NSGA-II
Target Region			
ZDT2 (1,0) 0.5	0.3168	0.3167	0.3159
ZDT2 (0.6,0.5) 0.3	0.3257	0.3256	0.3234
ZDT3 (0.3,0.1) 0.3	0.3377	0.3375	0.3365

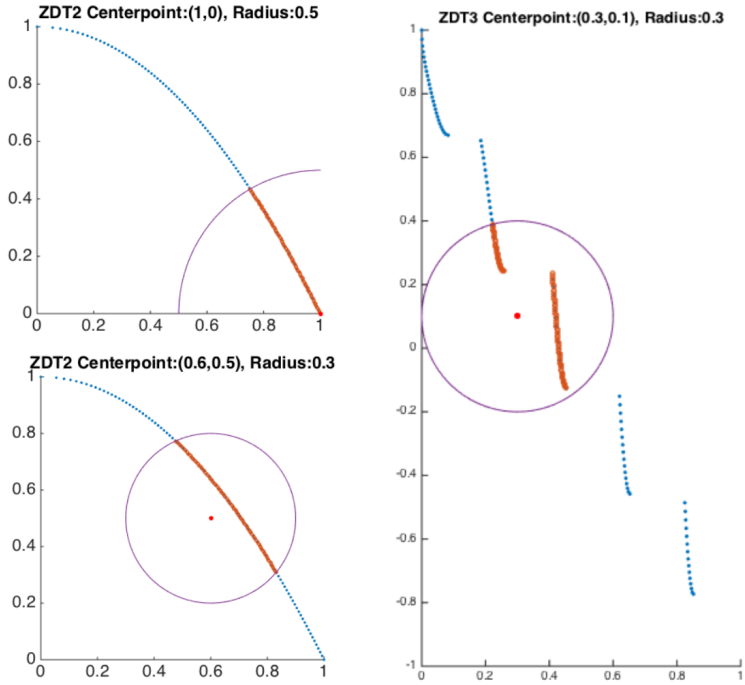


Figure 4.5: Representative PF approximations of T-SMS-EMOA on ZDT2 and ZDT3 with respect to different circular target regions.

dimensional, triangular hyperplane. A sphere with the center point (0.3, 0.3, 0.3) and radius 0.3 is defined as the target region for tri-objective DTLZ1 problem. The 11-variable DTLZ2 problem has a three-dimensional concave Pareto front. A box with the lower bound (0.4, 0.4, 0.2) and upper bound (0.8, 0.8, 0.8) is defined as the target region for DTLZ2 problem.

Figure 4.6 shows PF approximation of tri-objective DTLZ1 problem. The graphs in the upper row are solutions from T-SMS-EMOA, graphs in the middle row are solutions from T-R2-EMOA and graphs in the lower row are solutions from T-NSGA-II. The blue points show the true entire PFs. The transparent spheres depict target regions and red points are solutions obtained by T-SMS-EMOA, T-R2-EMOA and T-NSGA-II. It can be observed that T-SMS-EMOA behaves the best in three algorithms. Statistical results of the median of hypervolume in Table 4.4 are coincident with our observation.

Figure 4.7 shows PF approximations of tri-objective DTLZ2 problem. The trans-

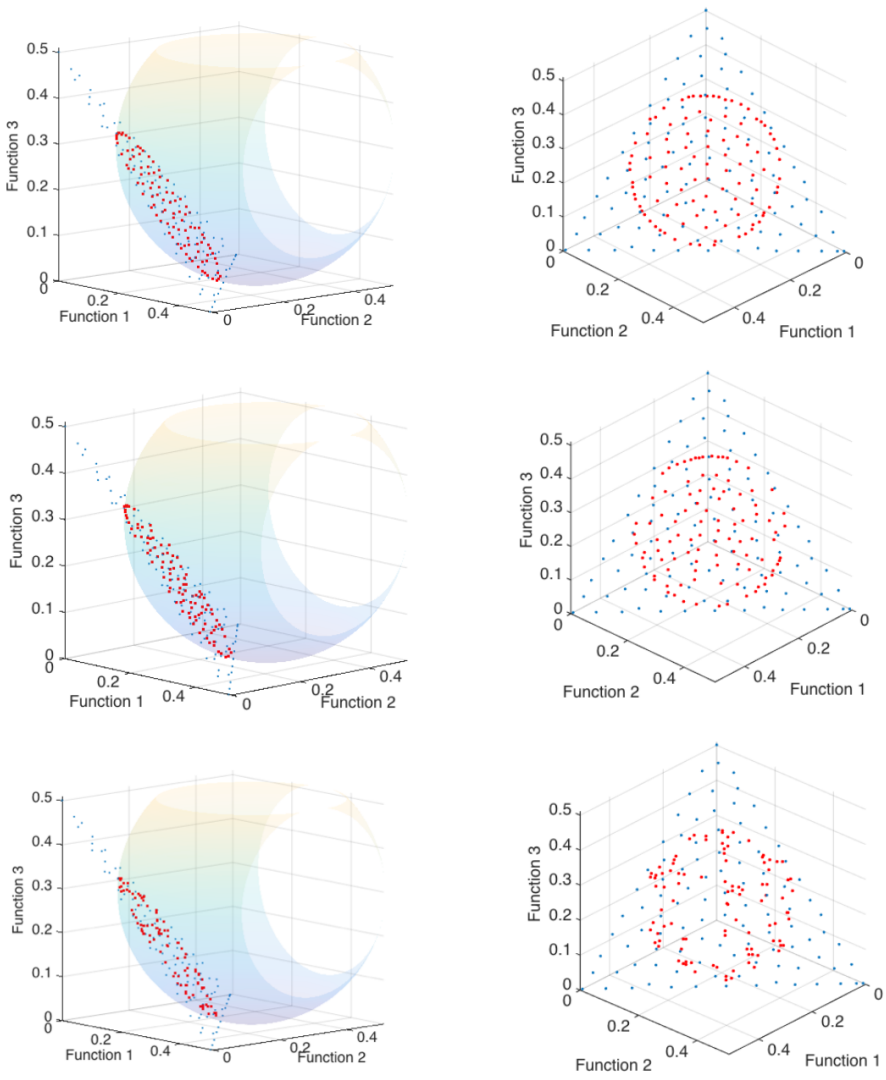


Figure 4.6: Representative PF approximations of T-SMS-EMOA(the upper row), T-R2-EMOA(the middle row) and T-NSGA-II(the lower row) with a spherical target region on tri-objective DTLZ1 problem.

Table 4.4: The median hypervolume of tri-objective DTLZ1 problem with a spherical target region.

Algorithm / Target Region	T-SMS-EMOA	T-R2-EMOA	T-NSGA-II
(0.3,0.3,0.3) 0.3	0.8028	0.7992	0.7823

parent boxes depict target regions and red points are solutions obtained by T-SMS-EMOA. Statistical results of the median of hypervolume for three algorithms and original MOEAs with constraints description in 30 independent runs on cubic target regions are shown in Table 4.5. It is observable that the best result is achieved by T-SMS-EMOA and all new algorithms outperform original MOEAs except for R2-EMOA.

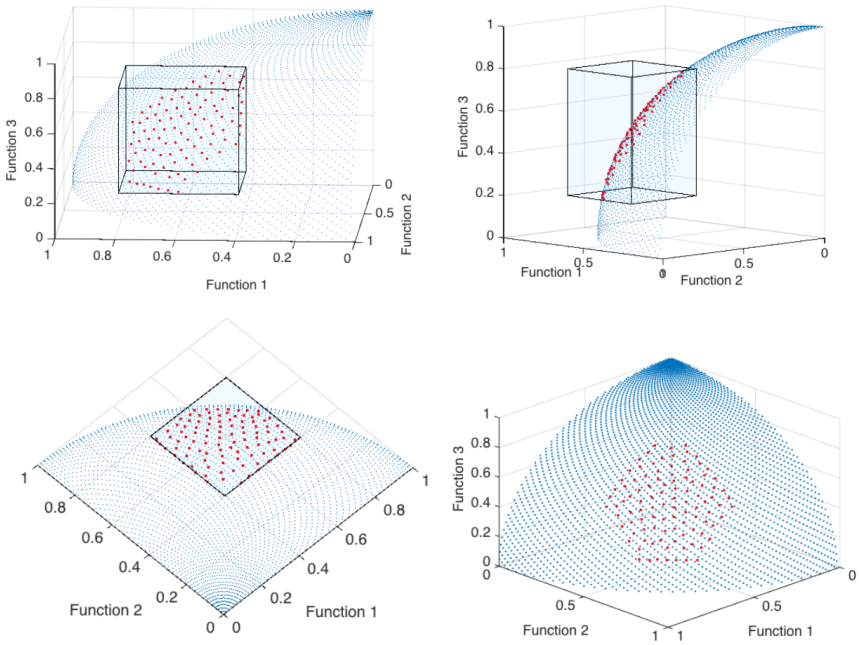


Figure 4.7: Representative PF approximations from T-SMS-EMOA in a cubic target region on tri-objective DTLZ2 problem.

Table 4.5: The median hypervolume on tri-objective DTLZ2 problem with a cubic target region.

MOEA / Target Region	T-SMS -EMOA	T-R2 -EMOA	T-NSGA -II	SMS-EMOA	R2-EMOA	NSGA -II
(0.4,0.4,0.2) (0.8,0.8,0.8)	0.4632	0.4303	0.4189	0.3369	0.4351	0.4185

Knapsack Problems

Knapsack Problems have been studied first by Dantzig in the late 1950′s [19]. The problem is a general, understandable, and one of the most representative discrete optimization problems. At the same time, it is difficult to solve (NP-hard). The Multi-objective 0/1 Knapsack Problems from Zitzler and Thiele [142] are used as discrete test problems in this part. Formally, the multi-objective 0/1 knapsack problem can be formulated as the following maximization problem:

$$\begin{aligned}
&\text{maximize} && f(x) = (f_1(x), f_2(x), \cdots, f_m(x)) \\
&\text{subject to} && \sum_{j=1}^{n} w_{ij} x_j \leq c_i, && i = 1, \cdots, m \\
& && x_j \in \{0, 1\}, && j = 1, \cdots, n \\
&\text{where} && f_i(x) = \sum_{j=1}^{n} p_{ij} x_j, && i = 1, \cdots, m.
\end{aligned}$$

Here n is the number of items and m is the number of knapsacks, w_{ij} is the weight of item j according to knapsack i, p_{ij} is the profit of item j according to knapsack i and c_i is the capacity of knapsack i. The Multi-objective 0/1 Knapsack Problem is to find Pareto optimal vectors x $= (x_1, x_2, \cdots, x_n)$ and $x_j = 1$ when item j is selected and $x_j = 0$ otherwise.

Figure 4.8 shows solutions obtained in a random single run when the number of knapsack is 2 and the number of items is 250. The results of SMS-EMOA, R2-EMOA and NSGA-II give the entire PF approximations. The target region for T-SMS-EMOA, T-R2-EMOA and T-NSGA-II is highlighted by the gray box. The lower bound is (9000, 9000), the upper bound is (9800, 9800). It can be observed that the proposed algorithms can find solutions in the target region except for several outliers from T-NSGA-II.

Statistical results of the median of hypervolume are presented in Table 4.6. No constraints of the target region are converted in the description of the problem for the results of original MOEAs in the table. In the experiments, two and three objectives are taken into consideration, in combination with 250 and 500 items. The test data sets are from [142]. The target region of knapsack-250-2 is from (9000, 9000) to (9800, 9800), of knapsack-250-3 is from (8500, 8500, 8500) to (10000, 10000, 10000), of knapsack-500-2 is from (18000, 18000) to (20000, 20000), of knapsack-500-3 is from

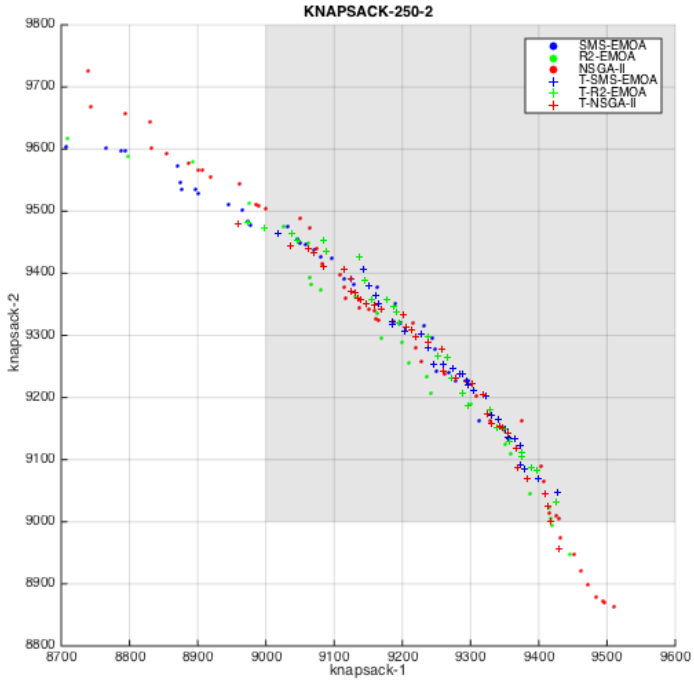


Figure 4.8: Representative PF approximations of knapsack-250-2 problem.

(17000, 17000, 17000) to (19000, 19000, 19000). The better results from the proposed algorithms than from the original algorithms can be seen.

4.1.3 Enhanced Algorithms and Experiments

As mentioned before, the second ranking criterion (Hypervolume, R2 indicator or crowding distance) in the proposed algorithms only works for solutions in the target region, which means if the intersection of the target region and true PF is empty, the second ranking criterion becomes useless. Under this condition, well-distributed solutions can not be obtained because solutions are guided only by non-dominated sorting and the Chebyshev distance. In addition, if multiple target regions are specified, sometimes the obtained solutions only converge to one target region. Inspired by some ideas from R-NSGA-II [30], two methods are adopted to overcome these limitations and extend the proposed algorithms.

Table 4.6: The median hypervolume of Knapsack problems with respect to different target regions.

Algorithms / Problems	T-SMS -EMOA	T-R2 -EMOA	T-NSG A-II	SMS- EMOA	R2- EMOA	NSGA -II
Knapsack -250-2	0.2117	0.2160	0.2230	0.2102	0.2099	0.1821
Knapsack -250-3	0.0295	0.0296	0.0311	0.0204	0.0293	0.0085
Knapsack -500-2	0.3273	0.3318	0.3230	0.3225	0.3272	0.2857
Knapsack -500-3	0.1936	0.1855	0.1713	0.1699	0.1747	0.0718

Separate Population to Different Targets

The first method can attract the population to different targets and it is used in the calculation of both the second ranking criterion (Hypervolume, R2 indicator or crowding distance) and the third ranking criterion (the Chebyshev distance). The aim of this method is to support multiple targets.

Taking R2 indicator as an example, after coordinate transformation, for all target regions, R2 indicator values of all solutions on the worst ranked front are calculated and the solutions are sorted in descending order of R2 indicator. Thereafter, R2 indicator values are replaced by *R2 indicator ranks*: solutions with the largest R2 indicator values for all target regions are assigned the same largest *R2 indicator rank*, solutions having next-to-largest R2 indicator values for all target regions are assigned the same next-to-largest *R2 indicator rank*, and so on, until the number of solutions which have been assigned the *R2 indicator rank* for each target region reaches its proportion in population. For the solutions assigned more than one ranks, the largest rank number is kept as its final *R2 indicator rank*. If the even distribution of solutions in all target regions is expected, the proportion of each target region should be divided equally between all target regions. For example, when the number of target regions is two, the proportion for each target region should be $1/2$ and the number of solutions being assigned a *R2 indicator rank* should reach half of the size of the worst ranked front. Lastly, for solutions that haven't been assigned a *R2 indicator rank*, their R2 indicator values should be mapped to values smaller than the smallest *R2 indicator rank*. One way to do this is to calculate their R2 indicator values in the combined region of all target regions and normalize them to values lower than all *R2 indicator ranks*. By this way, solutions with larger R2 indicator values in each target region are emphasized more.

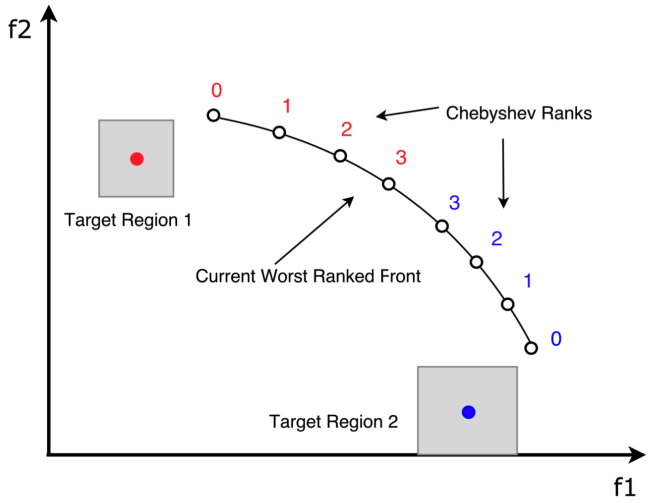


Figure 4.9: An example of assigning Chebyshev ranks to solutions on the current worst ranked front.

The *Chebyshev rank* is used in place of the Chebyshev distance when the method is added to work with the third ranking criterion. Firstly, the Chebyshev distances of each solution to all targets are calculated. The solutions are sorted in ascending order of distance. The *Chebyshev rank* of the solutions closest to all targets is assigned to be the same smallest rank of zero, the *Chebyshev rank* of the solutions having next-to-smallest Chebyshev distance is assigned to be the same next-to-smallest rank of one, and so on, until the number of solutions which have been assigned the *Chebyshev rank* for each target reaches its proportion in population. Similarly, the lowest rank of a solution is used as its Chebyshev distance if a solution is close enough to more than one target. Figure 4.9 shows an example of assigning *Chebyshev ranks* to solutions on the current worst ranked front. In the example, red Chebyshev ranks are assigned by sorting solutions in ascending order of the Chebyshev distance to Target Region 1 and blue Chebyshev ranks are assigned by sorting solutions in ascending order of the Chebyshev distance to Target Region 2, solutions with lower ranks are encouraged to remain in the population. The worst solution will be chosen from two solutions having the Chebyshev rank 3 randomly.

Improve Diversity

Under the circumstance that the second ranking criterion (Hypervolume, R2 indicator or crowding distance) doesn't work well, the diversity is lost and the solution set would converge to one point when the number of evaluations is high enough. To solve the problem, a parameter ϵ is introduced in the proposed algorithms and works with the third ranking criterion (the Chebyshev distance) to improve the diversity. Firstly, the solution with the shortest Chebyshev distance to the target is picked out. Thereafter, all other solutions having a Chebyshev distance less than the sum of the current shortest Chebyshev distance and ϵ are assigned a relatively large distance to discourage them to remain in the population. Then, another solution (not already considered earlier) is picked and the above procedure is performed again. This way, only one solution within a ϵ-neighborhood is emphasized and the diversity of the solution set is improved.

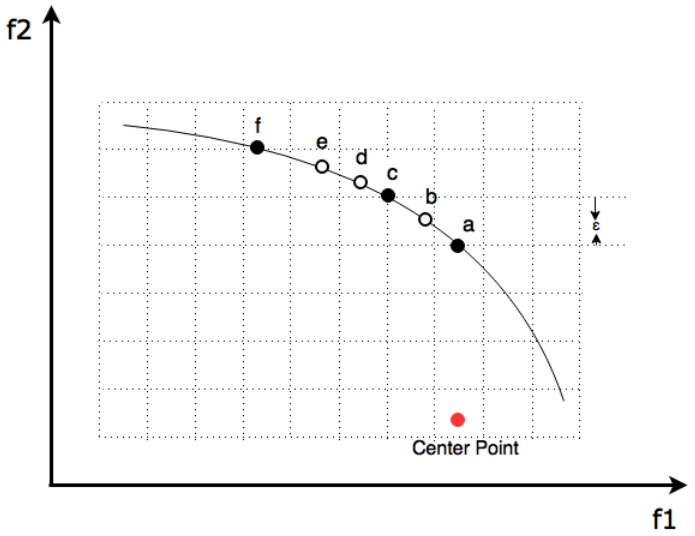


Figure 4.10: Illustration of ϵ parameter.

Figure 4.10 illustrates how the parameter ϵ takes effect when involved in the calculation of the Chebyshev distance. The Chebyshev distance between two vectors is the greatest of their differences along any coordinate dimension. Therefore, in the graph, the Chebyshev distances of solutions on the current font to the center point are distances along f_2. Apparently, point a is the point with the shortest Chebyshev distance to the center point, point b has a Chebyshev distance less than the sum of a's

Chebyshev distance and ϵ, thus, it will be assigned an relatively large distance value. So will point d and e. Point b, d and e are discouraged to remain in the population.

Algorithm Structure

When using above two methods in basic algorithms, Algorithm 6, 7, and 8 show the structures of enhanced algorithms.

Experiments on Multiple Target Regions

If multiple target regions of interest can be found simultaneously, the DM can make a more effective selection towards finding the ultimate preferred solution(s). The enhanced algorithms can guide the search toward multiple target regions. Three pair of spherical target regions are used on tri-objective DTLZ1 problem separately to demonstrate differences of results between T-SMS-EMOA (Figure 4.11), T-R2-EMOA (Figure 4.12) and T-NSGA-II (Figure 4.13). The center point and radius of two target regions are shown above each graph. The first pair of target regions have the same radius and both center points are on the PF which is a three-dimensional, triangular hyperplane. The second pair of target regions have the same radius, but one center point is on the PF, the other is not. The third pair of target regions have different radius, and both center points are on the PF. Experimental results over consecutive 30 runs show that all three algorithms can obtain uniform solutions in two target regions, no outliers exist. When the assigned population size is 100, each target region obtains 50 solutions for all 30 runs. While when the number of runs increases to 50, the case of 49 and 51 solutions in two regions also appears once.

In the above experiments, it has been specified that solutions are equally distributed in multiple target regions. For the population size of 100, this means that there are 50 solutions in each target region. It is also possible that we hope for a different proportion of solutions for each target region. For example, the real intersection areas of the third pair of target regions and the PF are obviously different. Therefore, $1/4th$ of population size is specified as the number of obtained solutions in the small intersection area and $3/4th$ of population size is specified as the number of obtained solutions in the larger intersection area. Figure 4.14 shows the difference between equally distributed solutions (left graph) and solutions with newly-specified proportion for each target region (right graph) of T-SMS-EMOA.

Furthermore, when there is no intersection between the target region and the PF, the enhanced algorithms can still obtain solutions close to the target region. Fig-

Algorithm 6 T-SMS-EMOA Enhanced version

$P_0 \leftarrow init()$; //Initialize random population
$t \leftarrow 0$;
while Stop criterion not satisfied() **do**
  $q_{t+1} \leftarrow Gen(P_t)$; //Generate offspring by variation
  $P_t \leftarrow P_t \cup \{q_{t+1}\}$;
  $\{R_1, \cdots, R_v\} \leftarrow$ Nondominated-sort(P_t);
  for all targets **do**
    $\forall x \in R_v$: compute $D_{Ch}(x)$; //Chebyshev distance to current target
    $R'_v \leftarrow R_v$; //Sorted in ascending order of D_{Ch}
    for unlabeled $x \in R'_v$ **do** //Start from the first solution in R'_v
      $Label(x)$;
      for $x' \in R'_v \setminus X$ **do** //X is the set of labeled points
        if $D_{Ch}(x') < D_{Ch}(x) + \epsilon$ **then**
          $Label(x')$; $D_{Ch}(x') \leftarrow$ relatively large value;
        end if
      end for
    end for
    $R_v \leftarrow R'_v$; //Sorted in ascending order of D_{Ch}
    for pre-defined number of $x \in R_v$ **do**
      $D_{Ch}(x) \leftarrow D_{Ch}\_rank(x)$; //Start from the smallest; keep smaller rank
    end for
  end for
  for unranked $x \in R_v$ **do**
    $D_{Ch}(x) \leftarrow Normalized\_D_{Ch}(x)$; //$Normalized\_D_{Ch}$ >largest D_{Ch}_rank
  end for
  for all targets **do**
    $R_{v1} \cup R_{v2} \leftarrow R_v$; //solutions not in the target region $\rightarrow R_{v1}$; others $\rightarrow R_{v2}$
    $\forall x \in R_{v1} : HC(x) \leftarrow 0$;
    $R_{v2} \leftarrow$ Coordinate Transformation(R_{v2});
    $\forall x \in R_{v2} : HC(x) \leftarrow HV(R_{v2}) - HV(R_{v2} \backslash x)$;
    $R'_v \leftarrow R_v$; //Sorted in descending order of HC values
    for pre-defined number of $x \in R'_v$ **do**
      $HC(x) \leftarrow HC\_rank(x)$; //Start from the largest; keep larger rank
    end for
  end for
  for unranked $x \in R_v$ **do**
    $HC(x) \leftarrow Normalized\_HC(x)$; //$Normalized\_HC$ <smallest HC_rank
  end for
  if unique $argmin\{HC(x) : x \in R_v\}$ exists **then**
    $x^* \leftarrow argmin\{HC(x) : x \in R_v\}$;
  else
    $x^* \leftarrow argmax\{D_{Ch}(x) : x \in R_v\}$; //In case of tie, choose randomly
  end if
  $P_{t+1} \leftarrow P \backslash \{x^*\}$; $t \leftarrow t + 1$;
end while

Algorithm 7 T-R2-EMOA Enhanced version

$P_0 \leftarrow init()$; //Initialize random population
$t \leftarrow 0$;
while Stop criterion not satisfied() **do**
  $q_{t+1} \leftarrow Gen(P_t)$; //Generate offspring by variation
  $P_t \leftarrow P_t \cup \{q_{t+1}\}$;
  $\{R_1, \cdots, R_v\} \leftarrow$ Nondominated-sort(P_t);
  for all targets **do**
    $\forall x \in R_v$: compute $D_{Ch}(x)$; //Chebyshev distance to current target
    $R'_v \leftarrow R_v$; //Sorted in ascending order of D_{Ch}
    for unlabeled $x \in R'_v$ **do** //Start from the first solution in R'_v
      $Label(x)$;
      for $x' \in R'_v \setminus X$ **do** //X is the set of labeled points
        if $D_{Ch}(x') < D_{Ch}(x) + \epsilon$ **then**
          $Label(x')$; $D_{Ch}(x') \leftarrow$ relatively large value;
        end if
      end for
    end for
    $R_v \leftarrow R'_v$; //Sorted in ascending order of D_{Ch}
    for pre-defined number of $x \in R_v$ **do**
      $D_{Ch}(x) \leftarrow D_{Ch}\_rank(x)$; //Start from the smallest; keep smaller rank
    end for
  end for
  for unranked $x \in R_v$ **do**
    $D_{Ch}(x) \leftarrow Normalized\_D_{Ch}(x)$; //$Normalized\_D_{Ch}$ >largest D_{Ch}_rank
  end for
  for all targets **do**
    $R_{v1} \cup R_{v2} \leftarrow R_v$; //solutions not in the target region $\rightarrow R_{v1}$; others $\rightarrow R_{v2}$
    $\forall x \in R_{v1} : r(x) \leftarrow 0$;
    $R_{v2} \leftarrow$ Coordinate Transformation(R_{v2});
    $\forall x \in R_{v2} : r(x) \leftarrow R2(P \setminus \{x\}; \Lambda; \mathbf{i})$; //$\mathbf{i}$ is ideal point
    $R'_v \leftarrow R_v$; //Sorted in descending order of $R2$ values
    for pre-defined number of $x \in R'_v$ **do**
      $r(x) \leftarrow r\_rank(x)$; //Start from the largest, keep larger rank
    end for
  end for
  for unranked $x \in R_v$ **do**
    $r(x) \leftarrow Normalized\_r(x)$; //$Normalized\_r(x)$ <smallest r_rank
  end for
  if unique $argmin\{r(x) : x \in R_v\}$ exists **then**
    $x^* = argmin\{r(x) : x \in R_v\}$;
  else
    $x^* = argmax\{D_{Ch}(x) : x \in R_v\}$; //In case of tie, choose randomly
  end if
  $P_{t+1} \leftarrow P \setminus \{x^*\}$; $t \leftarrow t + 1$;
end while

Algorithm 8 T-NSGA-II Enhanced version

$P_0 \leftarrow init()$; //Initialize random population
$t \leftarrow 0$;
while Stop criterion not satisfied() **do**
  $Q_t \leftarrow Gen(P_t)$; //Generate offsprings by variation
  $P_t = P_t \cup Q_t$;
  for all targets **do**
    $\forall x \in P_t$: compute $D_{Ch}(x)$; //Chebyshev distance to current target
    $P'_t \leftarrow P_t$; //Sorted in ascending order of D_{Ch}
    for unlabeled $x \in P'_t$ **do** //Start from the first in P'_t
      $Label(x)$;
      for $x' \in P'_t \setminus X$ **do** //X is the set of points have been labelled
        if $D_{Ch}(x') < D_{Ch}(x) + \epsilon$ **then**
          $Label(x')$;
          $D_{Ch}(x') \leftarrow$ relatively large value;
        end if
      end for
    end for
    $P_t \leftarrow P'_t$; //Sorted in ascending order of D_{Ch}
    for pre-defined number of $x \in P_t$ **do**
      $D_{Ch}(x) \leftarrow D_{Ch}\_rank(x)$; //Start from the smallest; keep smaller rank
    end for
  end for
  for unranked $x \in P_t$ **do**
    $D_{Ch}(x) \leftarrow Normalized\_D_{Ch}(x)$; //$Normalized\_D_{Ch}$ >largest D_{Ch}_rank
  end for
  $\{R_1, \cdots, R_v\} \leftarrow$ Nondominated-sort(P_t);
  for $i = 1, \cdots, v$ **do**
    for all targets **do**
      $R_{i1} \cup R_{i2} \leftarrow R_i$; //Solutions not in target region $\rightarrow R_{i1}$; others $\rightarrow R_{i2}$
      $\forall x \in R_{i1} : D_c(x) \leftarrow 0$; //Crowding distance
      $R_{i2} \leftarrow$ Coordinate Transformation(R_{i2});
      $\forall x \in R_{i2}$: compute $D_c(x)$;
      $R'_i \leftarrow R_i$; //Sorted in descending order of D_c
      for pre-defined number of $x \in R'_i$ **do**
        $D_c(x) \leftarrow D_c\_rank(x)$; //Start from the largest $D_c\_rank : |R'_i|$; keep larger rank if already assigned a rank.
      end for
    end for
    for unranked $x \in R'_i$ **do**
      $D_c(x) \leftarrow Normalized\_D_c(x)$; //$Normalized\_D_c(x)$ <smallest D_c_rank
    end for
  end for
  $P_{t+1} \leftarrow$ half the size of P_t based on rank, D_c and then D_{Ch};
  $t \leftarrow t + 1$;
end while

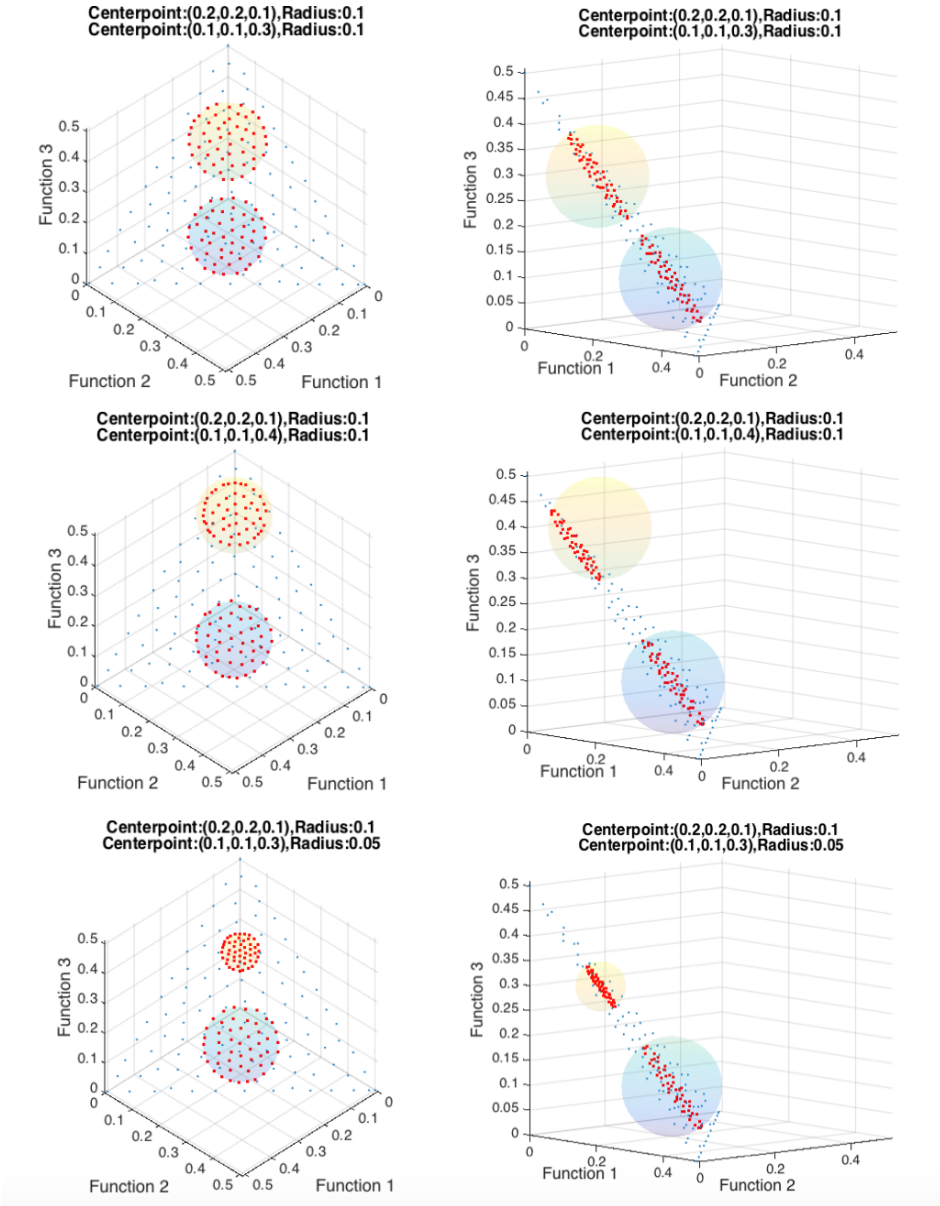


Figure 4.11: Representative PF approximations of T-SMS-EMOA with two spherical target regions on tri-objective DTLZ1.

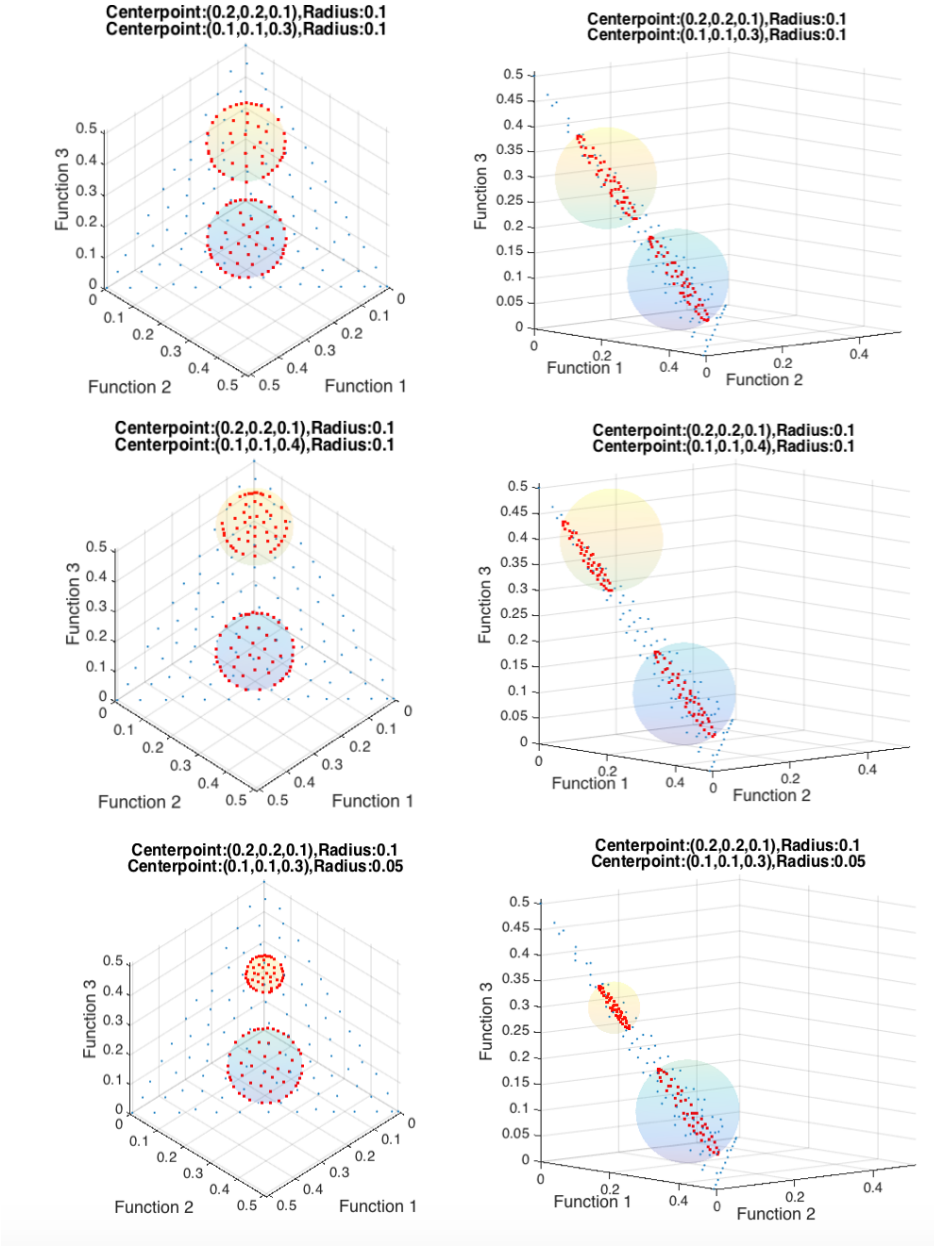


Figure 4.12: Representative PF approximations of T-R2-EMOA with two spherical target regions on tri-objective DTLZ1.

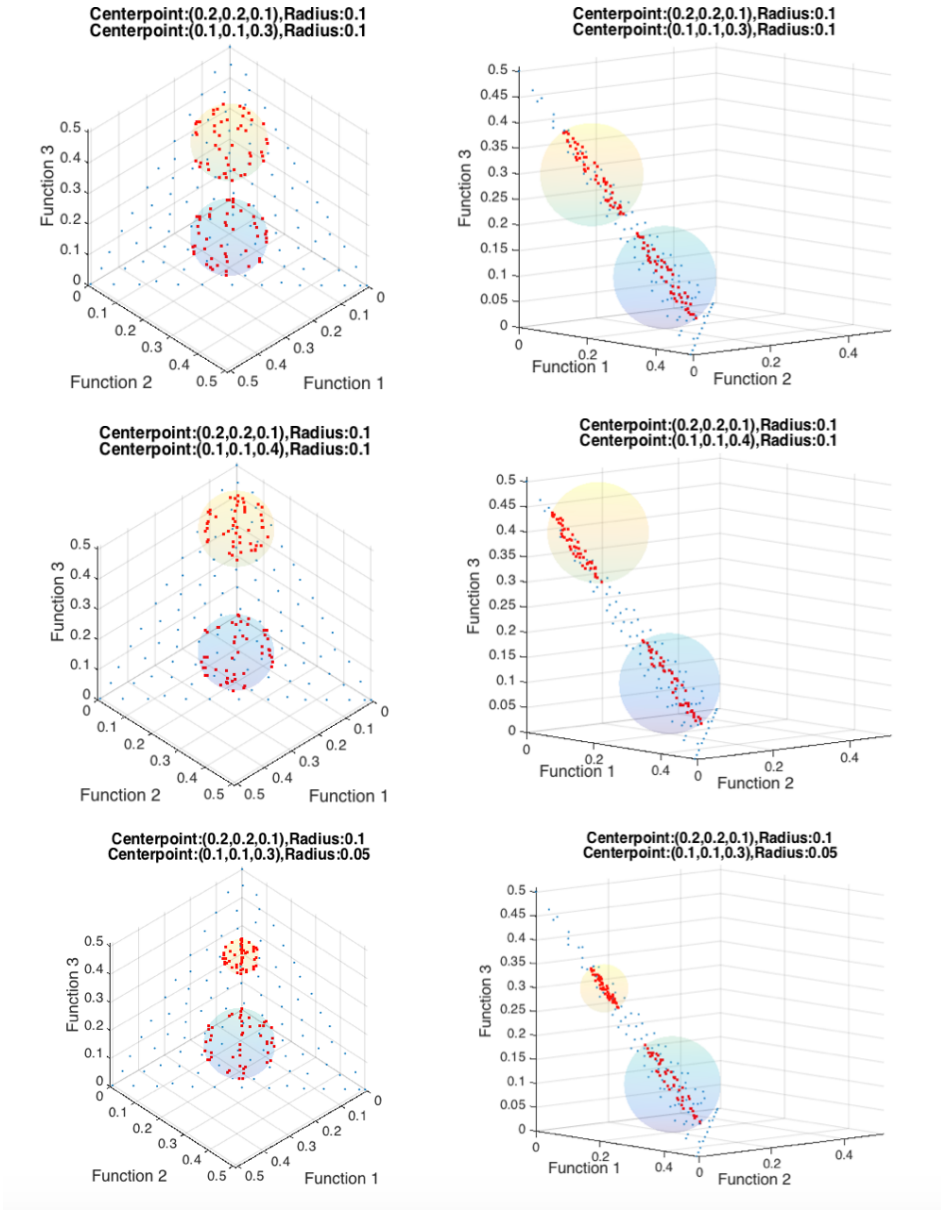


Figure 4.13: Representative PF approximations of T-NSGA-II with two spherical target regions on tri-objective DTLZ1.

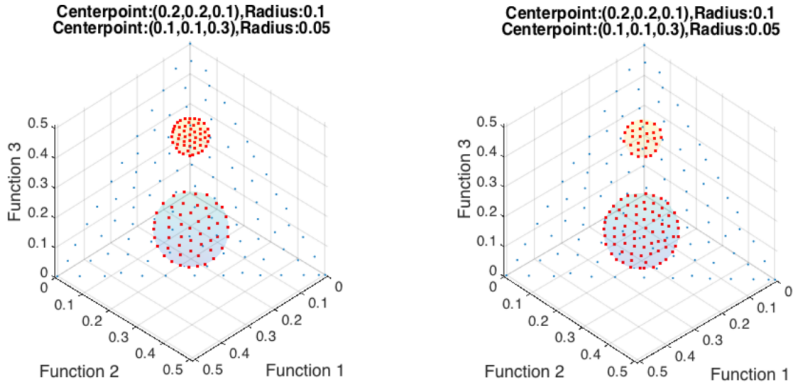


Figure 4.14: Representative PF approximations of T-SMS-EMOA with different solution distribution in two spherical target regions on tri-objective DTLZ1.

ure 4.15 shows PF approximations for different target regions which don't intersect with the PF.

Experiments on Single Target Point

The enhanced algorithms are not only capable of obtaining solutions in the target region, they also belong to reference point-based approaches. When the lower bound and the upper bound of the target region specified in the algorithms are the same, the target region shrinks to a target point. In this section, only results of T-SMS-EMOA are presented, T-R2-EMOA and T-NSGA-II can obtain similar results. Figure 4.16 shows PF approximations of T-SMS-EMOA for different single target points: the point around the PF, near the border, in the feasible area and in the infeasible area.

For three objective problem, the parameter ϵ plays an essential role in balancing convergence and diversity of the solutions near the target point. Figure 4.17 shows PF approximations of T-SMS-EMOA for one target point when the values of parameter ϵ are different. The black point is the target point and red points are obtained solutions; blue points indicate the entire PF. It is observed that when the parameter ϵ is smaller, obtained solutions are denser and more concentrated.

Experiments on Multiple Target Points

The enhanced algorithms can also work on multiple target points. Increasing the number of evaluations to 20000, Figure 4.18 shows PF approximations of T-SMS-

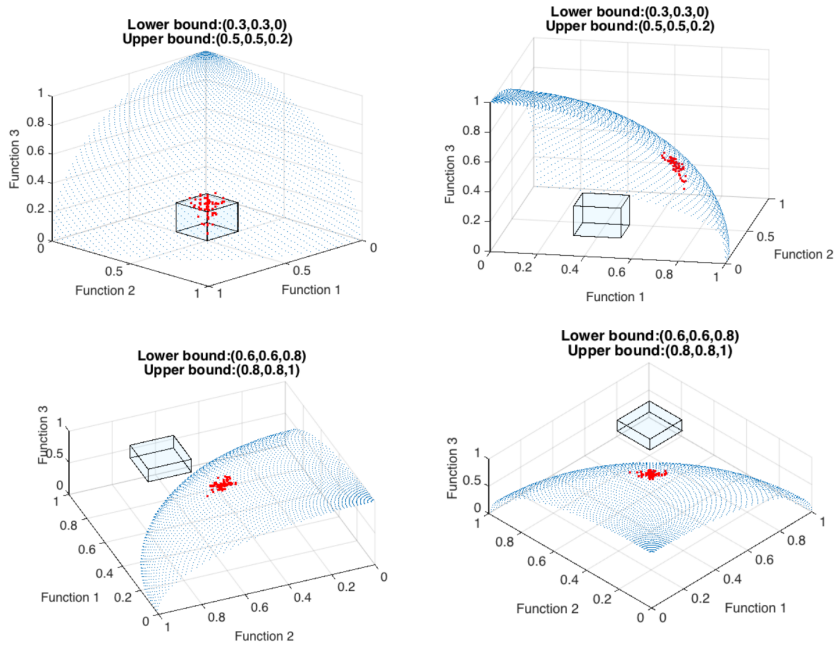


Figure 4.15: Representative PF approximations of T-SMS-EMOA on tri-objective DTLZ2; ϵ=0.001.

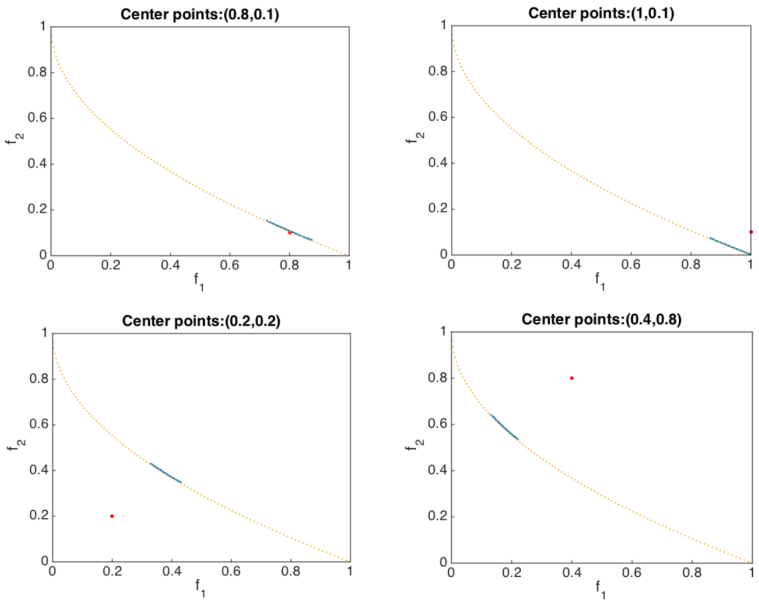


Figure 4.16: Representative PF approximations of T-SMS-EMOA on ZDT1; ϵ=0.0001.

EMOA for two target points on ZDT1 problem when the values of parameter ϵ are different. The red points are the target points and blue points are obtained solutions; purple points indicate the entire PF.

Increasing the number of evaluations to 50000, Figure 4.19 shows PF approximations of T-SMS-EMOA for two target points on tri-objective DTLZ1 problem when the values of parameter ϵ are different. The black points are the target points and red points are obtained solutions; blue points indicate the entire PF.

4.1.4 Conclusion

In this part, a target region based multi-objective evolutionary approach has been proposed. Three algorithms named T-SMS-EMOA, T-R2-EMOA and T-NSGA-II have been instantiated when combining the proposed algorithm framework with original SMS-EMOA, R2-EMOA and NSGA-II. These new algorithms have been applied to a number of continuous and combinational benchmark problems with two or three objectives. Experimental results show that the proposed algorithms can guide the search toward the preferred region on the Pareto optimal front. Almost no outliers appear

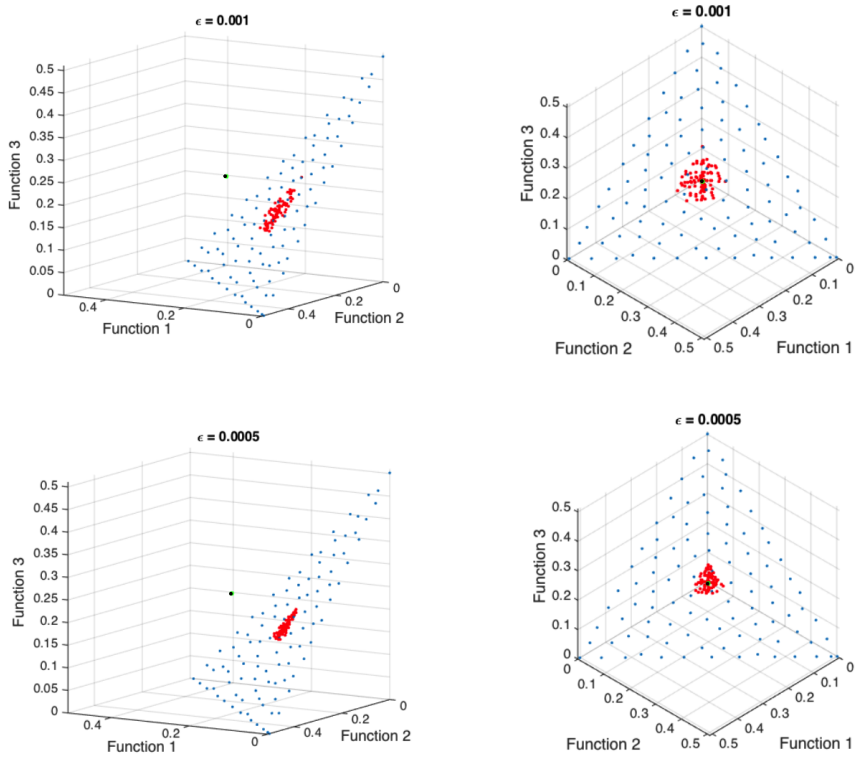


Figure 4.17: Representative PF approximations of T-SMS-EMOA on tri-objective DTLZ1 problem for one target point: (0.25, 0.25, 0.25).

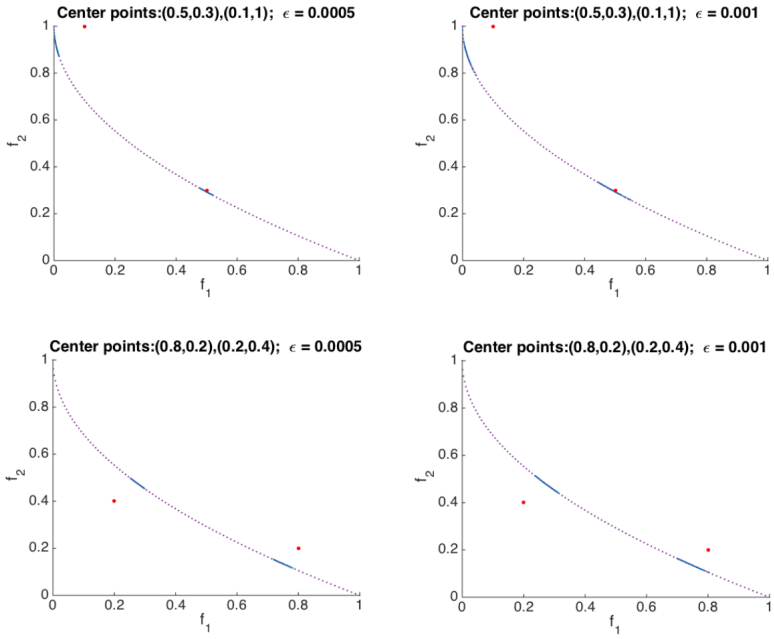


Figure 4.18: Representative PF approximations of T-SMS-EMOA on ZDT1 for two target points.

outside of the target region. In addition, these basic algorithms have been improved. The enhanced algorithms are more powerful and do not only support multiple target regions but also target point(s). It is worth noting that different numbers of solutions can be allocated to different targets by assigning the proportion of population size for each target.

On several instances, the proposed algorithms presented similar performance to the original MOEAs when converting the target region into constraints in the problem description. However, the proposed algorithms save a large amount of computational effort by guiding the search towards the preferred region without the calculation of the second ranking criterion in initial iterations. On the contrary, for original MOEAs, the increase in the number of constraints leads to the decrease of the search ability. Moreover, compared to the original MOEAs, the proposed algorithms exhibit the trend of behaving better with the increase in the number of objectives. More importantly, when there is no intersection between targets and the PF, the proposed algorithms can still find Pareto optimal solutions close to the targets. The future work would be

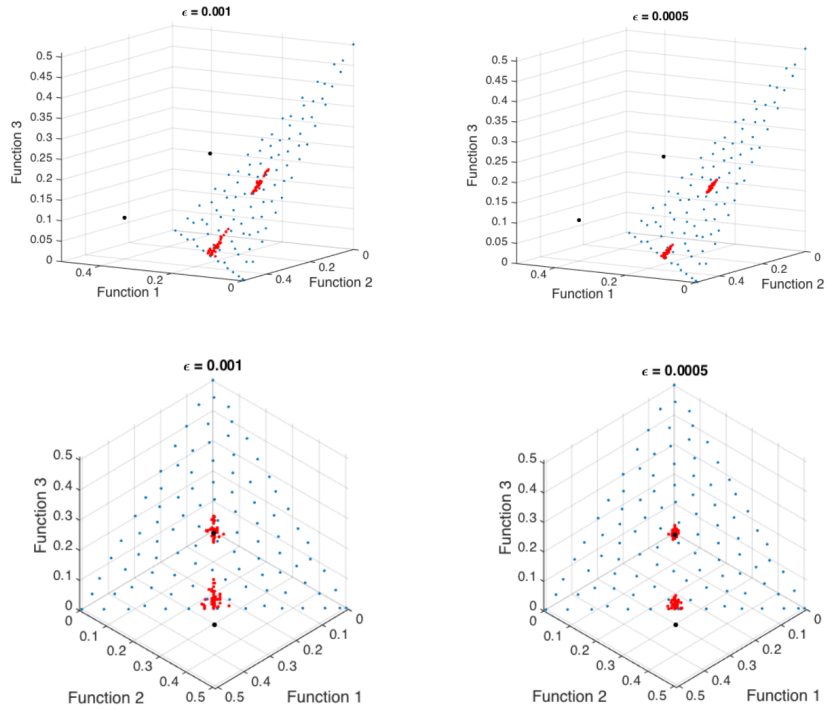


Figure 4.19: Representative PF approximations of T-SMS-EMOA on tri-objective DTLZ1 problem for two target points: (0.25, 0.25, 0.25) (0.4, 0.4, 0.1).

to compare the proposed algorithms with other preference-based MOEAs, especially multiple preferences based algorithms.

4.2 Automatic Preference Based MOEAs

When the algorithm aims at converging to the preferred solutions of the DMs, the DMs are asked for preference information. However, inspecting and choosing solutions from a large amount of solutions of a multi-objective optimization problem is not a trivial task for the DMs. The visualization of high-dimensional space further aggravates the difficulty. Sometimes, the DMs have no domain knowledge about the problems, they might set unreasonable goals which may mislead the search process. Therefore, an automatic preference based MOEA is proposed to avoid these difficulties and generate solutions in an automatically detected knee region. It is developed based on the

framework of DI-MOEA (see Section 3.1) and named automatic preference based DI-MOEA (AP-DI-MOEA).

AP-DI-MOEA can generate the preference region automatically, narrow down the feasible objective space progressively, and eventually obtain the preferred solutions. The preference region in AP-DI-MOEA is determined by the knee point. The knee point is a point for which a small improvement in any objective would lead to a large deterioration in at least one other objective. Several features of AP-DI-MOEA include: (1) no prior knowledge is used in identifying the knee point and knee region; (2) the preference region is generated automatically and narrowed down step by step to benefit its accuracy; (3) the proposed strategy can handle both multi-objective and many-objective optimization problems; (4) although AP-DI-MOEA is proposed based on DI-MOEA, the proposed strategy can be integrated with any standard MOEAs to form automatic preference based MOEAs; (5) the proposed algorithm is capable of finding preferred solutions for multi-objective optimization problems with linear, convex, concave Pareto fronts and discrete problems.

The remainder of this section is organized as follows. A literature review on knee based optimization is provided in Section 4.2.1. In Section 4.2.2, the proposed algorithm is described in detail. The experimental results are reported in Section 4.2.3 and Section 4.2.4 concludes the work with the summary and outlook.

4.2.1 Literature Review

In AP-DI-MOEA, the search for solutions is only guided towards the preference region which is determined by the knee point. It has been argued in the literature that knee points are most interesting solutions, naturally preferred solutions and most likely the optimal choice of the decision maker [12, 20, 23, 76].

In the last decade, several methods have been presented to identify knee points or knee regions. Das [20] refers the point where the Pareto surface "bulges" the most as the knee point, and this point corresponds to the farthest solution from the convex hull of individual minima which is the minima of the single objective functions. Zitzler [140] defines ϵ-dominance: a solution a is said to ϵ-dominate a solution b if and only if $f_i(a) + \epsilon \geq f_i(b)\ \forall i = 1, ..., m$ where m is the number of objectives. A solution with a higher ϵ-dominance value with respect to the other solutions in the Pareto front approximation, is a solution having higher trade-offs and in this definition corresponds to a knee point. The authors of [128] propose to calculate the density of solutions projected onto the hyperplane constructed by the extreme points of the non-

dominated solutions, then identify the knee regions based on the solution density.

Different algorithms of applying knee points in MOEAs have also been proposed. Branke [12] modifies the second criterion in NSGA-II, and replaces the crowding distance by either an angle-based measure or a utility-based measure. The angle-based method calculates the angle between an individual and its two neighbors in the objective space. The smaller the angle, the more clearly the individual can be classified as a knee point. However, this method can only be used for two objective problems. In the utility-based method, a marginal utility function is suggested to approximate the angle-based measure in the case of more than two objectives. The larger the external angle between a solution and its neighbors, the larger the gain in terms of linear utility obtained from substituting the neighbors with the solution of interest. However, the utility-based measure is not suited for finding knees in concave regions of the Pareto front.

Rachmawati [85, 86] proposes a knee-based MOEA which computes a transformation of original objective values based on a weighted sum niching approach. The extent and density of coverage of the knee regions are controllable by the parameters for the niche strength and pool size. The strategy is susceptible to the loss of less pronounced knee regions.

Schütze [93] investigates two strategies for the approximation of knees of bi-objective optimization problems with stochastic search algorithms. Several new definitions for identifying knee points and knee regions for bi-objective optimization problems has been suggested in [25] and the possibility of applying them has also been discussed.

Besides the knee points, the reference points, which are normally provided by the DM, have also been used to find a set of solutions near reference points. Deb [30] proposes an MOEA, called R-NSGA-II, by which a set of Pareto optimal solutions near a supplied set of reference points can be found. The dominance relation together with a modified crowding distance operator is used in this methodology. For all solutions of the population, the distances to all reference points are calculated and ranked. The lowest rank (over all reference points) of a solution is used as its crowding distance. Besides, a parameter ϵ is used to control the spread of obtained solutions. Recently, R-NSGA-II was extended and the reference point based NSGA-III (R-NSGA-III) is proposed for solving higher objective problems [111]. Bechikh proposes KR-NSGA-II [7] by extending R-NSGA-II. Instead of obtaining the reference points from the DM, in KR-NSGA-II, the knee points are used as mobile reference points and the search of the algorithm was guided towards these points. The number of knee points of the optimization problem is needed as prior information in KR-NSGA-II.

Gaudrie [49] uses the projection (intersection in case of a continuous front) of the closest non-dominated point on the line connecting the estimated ideal and nadir points as default preference. Conditional Gaussian process simulations are performed to create possible Pareto fronts, each of which defines a sample for the ideal and the nadir point, and the estimated ideal and nadir are the medians of the samples.

Rachmawati and Srinivasan [87] evaluate the worthiness of each non-dominated solution in terms of compromise between the objectives. The local maxima is then identified as potential knee solutions and the linear weighted-sums of the original objective functions are optimized to guide solutions toward the knee regions.

Another idea of incorporating preference information into multi-objective optimization is proposed in [103]. They combine the fitness function and an achievement scalarizing function containing the reference point. In this approach, the preference information is given in the form of a reference point and an indicator-based evolutionary algorithm IBEA [139] is modified by embedding the preference information into the indicator. Various further preference based MOEAs have been suggested, e.g., [13, 88, 117].

In our proposed algorithm, i.e., AP-DI-MOEA, we adopt the method from [20] to identify the knee point, design the preference region based on the knee point, and guide the search towards the preference region.

4.2.2 Algorithms

Two variants of DI-MOEA, DI-1 and DI-2 (see Section 3.1), exist. Analogously, two variants of AP-DI-MOEA, i.e., AP-DI-1 and AP-DI-2, are derived from the two variants of DI-MOEA. The workings of AP-DI-MOEA are outlined in Algorithm 9. In the algorithm, the variable *evals_update* is used to record the condition of generating or updating the preference region. Its initial value is assigned to *divide_size* (line 4 in Algorithm 9). Exceedance of *divide_size* is a predefined condition to divide the algorithm into two phases: learning phase and decision phase. In the learning phase, the algorithm explores the possible area of Pareto optimal solutions and finds the rough approximations of the Pareto front. In the decision phase, the algorithm identifies the preference region and finds preferred solutions. When the algorithm starts running and the number of evaluations reaches or exceeds *divide_size* at some moment, the first preference region will be generated and *evals_update* will be updated for determining a new future moment when the preference region needs to be updated (line 12 - 16 in Algorithm 9). The process of updating *evals_update* repeats until the end

to narrow down the preference region step by step. The first value of *evals_update*, i.e., *divide_size*, is a boundary line. Before it is satisfied, AP-DI-MOEA runs exactly like DI-MOEA to approximate the whole Pareto front; while, after it is satisfied, the preference region is generated automatically and AP-DI-MOEA finds solutions focusing on the preference region. The subsequent values of *evals_update* define the later moments to update the preference region; eventually, a precise ROI with a proper size can be achieved.

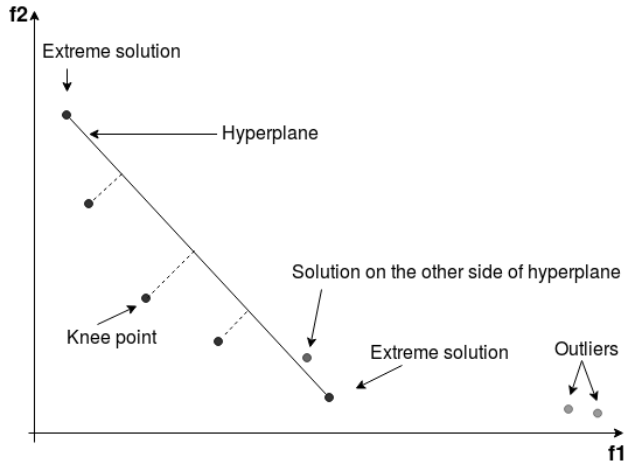


Figure 4.20: Finding the knee point in bi-dimensional space.

The first/new preference region is formed based on the population at the moment when the condition of *evals_update* is satisfied. To be specific, the preference region is determined by the knee point of the current Pareto front. Algorithm 10 gives the details of line 14 in Algorithm 9, it introduces the steps of finding the knee point of a non-dominated solution set and constituting a hypercube shaped preference region according to the knee point. Figure 4.20 also gives an illustration of finding the knee point in bi-dimensional space. Firstly, the upper quartile objective values (line 12 in Algorithm 10) in the solution set are used as a boundary to define outliers. To identify the knee point, solutions outside this boundary are removed from the solution set (line 15 - 19 in Algorithm 10). The extreme solutions (the solutions with the maximum value in one objective) are then found inside the boundary (line 22 in Algorithm 10) and a hyperplane is formed based on the extreme solutions (line 23 in Algorithm 10). In a bi-dimensional space (Figure 4.20), the hyperplane is only a line connecting two extreme solutions. According to the numbers of points below and above the hyperplane

Algorithm 9 AP-DI-MOEA-2

Inputs:
popsize; // population size
divide_size; // number of evaluations before generating 1st preference region
batch_size; // number of evaluations between preference region updates

$P_0 \leftarrow init(popsize)$; // initialize random population
existRegion $\leftarrow$ false; // indicates whether a preference region was already computed
evals $\leftarrow 0$; // number of evaluations to far
evals_update $\leftarrow$ *divide_size*; // number of evaluations when 1st preference region is computed
$(R_1, ..., R_{\ell_0}) \leftarrow$ non_dominated_sorting(P_0); // partition into fronts of increasing dominance ranks
for each $i \in \{1, \ldots, \ell_0\}$ **do**
  calculate diversity indicator for all solutions on R_i;
end for
$t \leftarrow 0$;
while Stop criterion not satisfied() **do**
  // update / computation of preference region
  if (*evals* > *evals_update* && $\ell_t == 1$) **then**
    existRegion $\leftarrow$ true;
    calculate *P_region*; //generate a (new) preference region, i.e., Algorithm 2
    evals_update $\leftarrow$ *evals_update* + *batch_size*;
  end if

  // offspring generation
  if ($\ell_t > 1$ || $t == 0$) **then**
    //($\mu + \mu$) generational scheme
    $Q_t \leftarrow Gen(P_t)$; // generate *popsize* offspring by recombination and mutation
    evaluate(Q_t);
    evals $\leftarrow$ *evals* + *popsize*;
    $M_{t+1} = P_t \cup Q_t$; // combine offspring and parent population
  else
    // ($\mu + 1$) steady state generational scheme
    $q \leftarrow Gen(P_t)$; // generate only one offspring by recombination and mutation
    evaluate(q_t);
    evals $\leftarrow$ *evals* + 1;
    $M_{t+1} = P_t \cup \{q_t\}$; // combine offspring and parent population
  end if

  // construction of new population based on non-dominated sorting
  $(R_1, ..., R_{\ell_{t+1}}) \leftarrow$ non_dominated_sorting(M_{t+1});
  $P_{t+1} \leftarrow \emptyset$;
  $i \leftarrow 0$;
  while $|P_{t+1}| <$ *popsize* **do**
    $i \leftarrow i + 1$;
    $P_{t+1} \leftarrow P_{t+1} \cup R_i$;
  end while

  // truncation of new population based on further ranking criterion(s)
  if ($|P_{t+1}| >$ *popsize*) **then**
    if (*existRegion* == false) **then**
      rank solutions in R_i by diversity indicator contribution;
    else
      rank solutions in R_i by diversity indicator contribution and Euclidean distance to knee point;
      assign the lowest possible rank to all solutions in R_i, which are outside *P_region*;
    end if
    $n \leftarrow |P_{t+1}| -$ *popsize*;
    remove the n solutions in R_i with lowest ranks from P_{t+1};
  end if

  $t \leftarrow t + 1$;
  $l_t \leftarrow$the number of fronts of P_t;
end while

Algorithm 10 Finding the knee point and defining the preference region.

Inputs:
$popsize$; // population size
n; // number of objectives
P_t; // current population
ϵ; // parameter (>0) for distinguishing convex/concave shape

declare($Q[n]$); //upper quartile objective values of P_t
declare($L[n]$); //worst objective values of P_t
declare($knee[n]$); //knee point of P_t
declare($P\_region[n]$); //preference region of P_t
declare($Epoints[n][n]$); //extreme points (single-objectives)
$foundknee \leftarrow false$; // indicates whether the knee point was already found
$P'_t \leftarrow P_t$; //copy the current population for finding the knee

// remove outliers with lowest 25% of objective values
for each $i \in \{1, \ldots, n\}$ **do**
  sort(P'_t) by the ith objective in ascending order;
  $Q[i] \leftarrow P'_t$.get_index($\frac{3}{4} \times popsize$).get_obj($i$); //upper quartile value of the ith objective
  $L[i] \leftarrow P'_t$.get_index($popsize$).get_obj(i); //the largest (worst) value of the ith objective
end for
for all solution $s \in P'_t$ **do**
  if s.get_obj($i = 1, ..., n$) $> Q[i]$ **then**
    remove s from P'_t; //remove outliers
  end if
end for

//find knee point by computing distance to hyperplane
$Epoints[\bullet][\bullet] \leftarrow$ extreme points in P'_t;
hyperplane($Epoints[\bullet][\bullet]$); //generate hyperplane by $Epoints[\bullet][\bullet]$
$num_a \leftarrow$ number of points in concave region of hyperplane;
$num_v \leftarrow |P'_t| - num_a$; // number of points in convex region
if $(num_v - num_a > \epsilon)$ **then**
  //roughly convex shape
  remove solutions in concave region from P'_t;
else if $(num_a - num_v > \epsilon)$ **then**
  //roughly concave shape
  remove solutions in convex region from P'_t;
else
  //roughly linear shape
  //find knee point by computing hypervolume
  for all solution $s \in P'_t$ **do**
    calculate hypervolume of s with reference point $L[\bullet]$;
  end for
  $knee[\bullet] \leftarrow$ solution with the largest hypervolume value;
  $foundknee \leftarrow true$;
end if
if $(foundknee == false)$ **then**
  for all solution $s \in P'_t$ **do**
    calculate distance between s and hyperplane;
  end for
  $knee[\bullet] \leftarrow$ solution with the largest distance;
end if

//determine current preference region by knee point.
for each $i \in \{1, \ldots, n\}$ **do**
  $P\_region[i] \leftarrow knee[i] + (L[i] - knee[i]) \times 85\%$
end for

(line 24 - 25 in Algorithm 10), the shape of the solution set can be roughly perceived. We will distinguish between "convex" and "concave" regions. Points in the *convex* (*concave*) *region* are dominating (dominated by) at least one point in the hyperplane spanned by the extreme points. However, when the number of the points in the convex region and the number of points in the concave region is close enough, it implies that the shape of the current solution set is almost linear. This occurs both when the true Pareto front is linear and when the solution set converges very well in a small area of the Pareto front. A parameter ϵ then is used to represent the closeness and it is a small number decided by the size of the solution set. In the case that the shape of the current solution set is (almost) linear, the solution with the largest hypervolume value with regards to the worst objective vector (i.e., $L[i]$ in line 13 in Algorithm 10) is adopted as the knee point (line 33 - 39 in Algorithm 10). While, under the condition that the shape of the current solution set is convex or concave, the knee point is identified by the method in [20]. The solution in the convex or concave region with the largest Euclidean distance to the hyperplane is chosen as the knee point (line 42 - 45 in Algorithm 10). After the knee point is found, the preference region can be determined based on the knee point by the following formula:

$$P\_region[i] = knee[i] + (L[i] - knee[i]) \times 85\%. \tag{4.5}$$

Let i denotes the ith objective, as in Algorithm 10, $L[i]$ is the worst value of the ith objective in the population, $knee[i]$ is the ith objective value of the knee point and $P\_region[i]$ is the upper bound of the preference region. W.l.o.g. We assume the objectives are to be minimized and the lower bound of the preference region is the origin point. According to the formula, we can see that the first preference region is relatively large (roughly 85% of the entire Pareto front). With the increase in the number of iterations, the preference region will be updated and become smaller and smaller because every preference region picks 85% of the current Pareto front. Eventually, we want the preference region to reach a proper range, say, 15% of the initial Pareto front. The process of narrowing down the preference region step by step can benefit the accuracy of the preference region.

Algorithm 9 only shows the workings of AP-DI-2. The difference between AP-DI-1 and AP-DI-2 is the same as the difference between DI-1 and DI-2, i.e., the diversity criterion in the $(\mu + \mu)$ generational selection operator in AP-DI-1 is the crowding distance, in AP-DI-2 it is the diversity indicator. In the algorithm, the initialized population is sorted based on non-domination and the diversity value of each solution

is calculated to be used in later parent selection (line 5 - 8 in Algorithm 9). When evolution of the population takes place, according to different phases of optimization, the $(\mu + \mu)$ generational selection operator generates multiple offspring in one iteration to explore more decision space and push the population quickly towards the Pareto front (line 20 - 24 in Algorithm 9); the $(\mu + 1)$ steady state selection operator generates only one offspring in order to achieve a uniformly distributed set (line 26 - 30 in Algorithm 9). To achieve the next generation population, the combination of parents and offspring is classified into different layers according to the non-dominance relation. The points from the first non-domination front are preserved, continuing with points in the second non-domination front, until the number of points reaches the population size (line 34 -40 in Algorithm 9). Under the case that the number of points surpasses the population size, a truncation selection is carried out (line 43 - 52 in Algorithm 9). When there is no preference region, the population will be truncated based on the diversity indicator. While, if a preference region already exists, the population will be truncated based on first the diversity indicator, then Euclidean distance to the knee point. In this process, the diversity indicator contribution and Euclidean distance to the knee point are calculated only for the solutions in the preference region. Solutions outside of the region are given relatively larger values and they will be eliminated in the following optimization process.

There are different strategies to set and update the value of *evals_update*. In our algorithm, we divide the whole computing budget into two parts, the first half is used to find an initial entire Pareto front approximation, and the second half is used to update the preference region and find solutions in the preference region. Assume the total computing budget is *budget_size* (the number of evaluations), then the first value of *evals_update* is $\frac{1}{2} \times budget\_size$. Due to the reason that we expect a final preference region with a size of around 15% of the initial entire Pareto front and each new preference region takes 85% of the current Pareto front, according to the formula: $0.85^{12} \approx 0.14$, the value of *evals_update* can be updated by the following formula:

$$evals\_update = evals\_update + batch\_size = evals\_update + (budget\_size/2)/12. \tag{4.6}$$

Another half of the budget can be divided into 12 partial-budgets and a new preference region is constituted after each partial-budget. In the end, the final preference region is achieved and solutions focusing on this preference region are obtained.

4.2.3 Experimental Results

Experimental Design

For the two variants of AP-DI-MOEA: AP-DI-1 and AP-DI-2, their performances have been compared with DI-MOEA: DI-1, DI-2 and NSGA-III. NSGA-III is involved in the comparison because NSGA-III is a representative state-of-the-art evolutionary multi-objective algorithm and it is very powerful to handle problems with non-linear characteristics. For bi-objective benchmark problems, algorithms are tested on ZDT1 and ZDT2 with 30 variables. For tri-objective benchmark problems, DTLZ1 with 7 variables and DTLZ2 with 12 variables are tested. On every problem, each algorithm runs for 30 times with different seeds, while the same 30 different seeds are used for all algorithms. All the experiments are performed with a population size of 100. For bi-objective problems, experiments are run with a budget of 22000 (objective function) evaluations; for DTLZ tri-objective problems, the budget is 120000 evaluations.

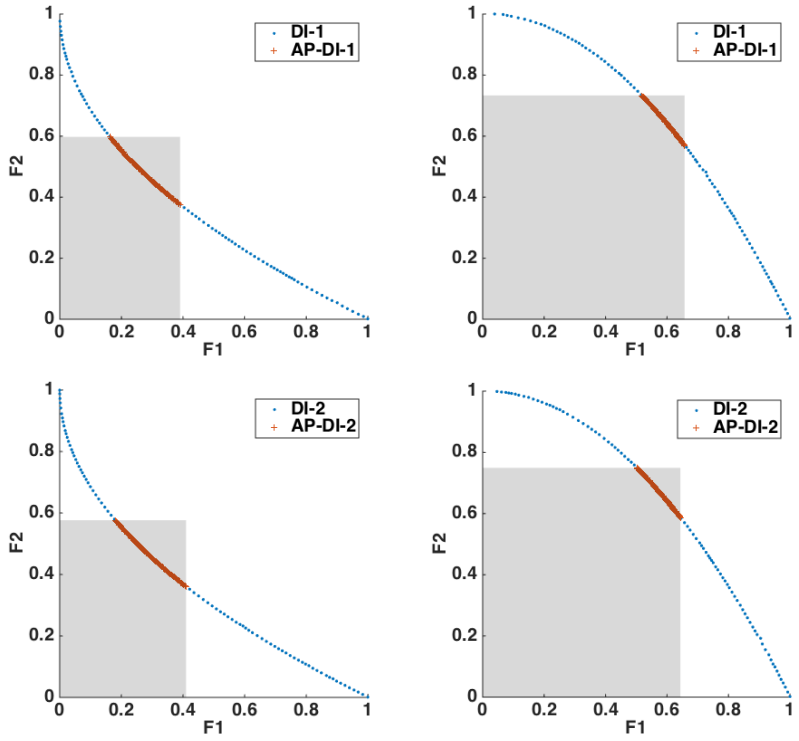
Experiments on Bi-objective Problems

Bi-objective problems are optimized with a total budget of 22000 evaluations, when the number of evaluations reaches 10000, the first preference region is generated, then after every 1200 evaluations, the preference region will be updated. Figure 4.21 shows Pareto front approximations from a typical run on ZDT1 (left column) and ZDT2 (right column). The graphs on the upper row are obtained from DI-1 and AP-DI-1, while the graphs on the lower row are from DI-2 and AP-DI-2. In each graph, the entire Pareto front approximations from DI-MOEA and the preferred solutions from AP-DI-MOEA (or *AP solutions*) are presented, at the same time, the final preference regions of AP-DI-MOEA are also shown by the gray areas.

Besides the visualization of the Pareto fronts, the knee point of the entire final Pareto front approximation from DI-MOEA is also computed via the strategy described in Algorithm 10. For each run of DI-MOEA and AP-DI-MOEA with the same seed, the following two issues have been checked:

- if the knee point from DI-MOEA is in the preference region achieved by its derived AP-DI-MOEA;
- if the knee point from DI-MOEA is dominated by or dominating AP solutions; or if it is a non-dominated solution (mutually non-dominated with all AP solutions).

Table 4.7 shows the results of 30 runs. For ZDT1 problem, all 30 knee points from DI-1 and DI-2 are in the preference regions from AP-DI-1 and AP-DI-2 respectively; in



(a) ZDT1. **(b)** ZDT2.

Figure 4.21: Pareto front approximations on ZDT1 and ZDT2.

all these knee points, 10 from DI-1 and 7 from DI-2 are dominated by AP solutions. For ZDT2 problem, most knee points are not in the corresponding preference regions, but for those in the preference regions, almost all of them are dominated by AP solutions. Please note that when a knee point from DI-MOEA is outside of the preference region from AP-DI-MOEA, it is not possible that it can dominate any AP solutions because all AP solutions are in the preference region and only solutions in the left side of the gray area can dominate AP solutions.

The same comparison is also performed between AP-DI-MOEA and NSGA-III, the results are shown in Table 4.8. For ZDT1 problem, all knee points from NSGA-III are in the preference regions from AP-DI-MOEA. Some of these knee points dominate AP solutions. For ZDT2 problem, most knee points from NSGA-III are not in the preference regions and these knee points are incomparable with AP solutions. For the knee points in the preference regions, all three dominating relations with AP solutions

Table 4.7: Space and dominance relation of knee point from DI-MOEA and AP solutions on ZDT problems.

Problem		ZDT1		ZDT2	
Algorithm		DI-1/ AP-DI-1	DI-2/ AP-DI-2	DI-1/ AP-DI-1	DI-2/ AP-DI-2
In preference region	Incomparable	20	23	1	1
	Dominated	10	7	9	9
	Dominating	0	0	0	0
Outside p-region	Incomparable	0	0	20	20
	Dominated	0	0	0	0

appear. For both problems, when the knee point from NSGA-III is dominating AP solutions, it only dominates one AP solution.

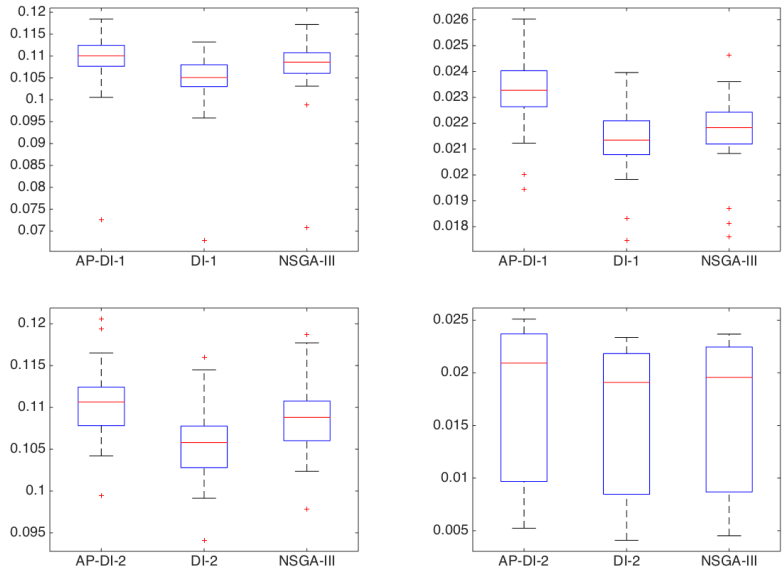
Table 4.8: Space and dominance relation of knee point from NSGA-III and AP solutions on ZDT problems.

Problem		ZDT1		ZDT2	
Algorithm		NSGA-III/ AP-DI-1	NSGA-III/ AP-DI-2	NSGA-III/ AP-DI-1	NSGA-III/ AP-DI-2
In preference region	Incomparable	14	19	3	1
	Dominated	0	0	2	3
	Dominating	16	11	4	6
Outside p-region	Incomparable	0	0	21	20
	Dominated	0	0	0	0

Instead of spreading the population across the entire Pareto front, the optimization only focuses on the preference region. To ensure that the algorithm can guide the search towards the preference region and the achieved solution set is distributed across the preference region, the performance of AP-DI-MOEA, DI-MOEA and NSGA-III is compared in the preference region. For each Pareto front approximation from DI-MOEA and NSGA-III, the solutions in the corresponding preference region from AP-DI-MOEA are picked, and these solutions are compared with AP solutions through the hypervolume indicator. The point formed by the largest objective values over all solutions in the preference region is adopted as the reference point when calculating the hypervolume indicator. It has been found that all hypervolume values of new solution sets from DI-MOEA and NSGA-III in the preference region are worse than the hypervolume values of the solution sets from AP-DI-MOEA, which proves that the mechanism indeed works in practice. Figure 4.22 shows box plots of the distribution of hypervolume indicators over 30 runs.

It can be seen that on both ZDT1 (with a convex Pareto front) and ZDT2 (with a concave Pareto front), AP-DI-MOEA can find solutions in the preference region which is determined by the knee points. Here for ZDT2 problem, its knee region is still a

compromise of two objectives. If the Pareto front is severely concave, adopting the knee region as the preference region is probably not a good decision because solutions in the knee region have poor values on both objectives, and it is not clear which region is preferred by the decision maker. But for the common cases, especially most real-world application problems, they don't have a severely concave Pareto front, the knee region is still recommended as the preference region.



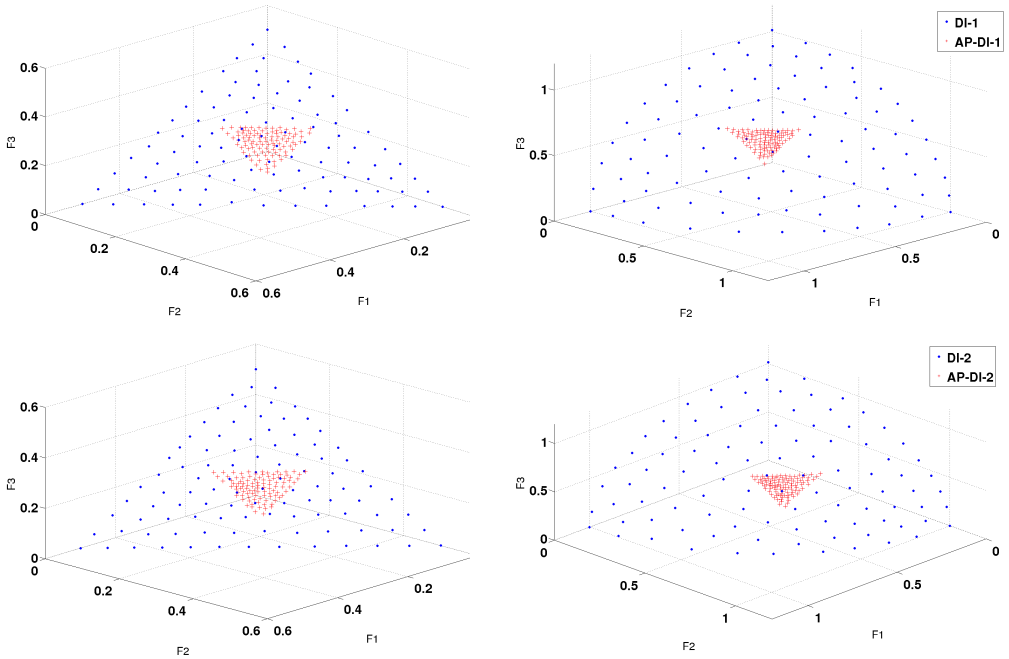
(a) ZDT1. **(b)** ZDT2.

Figure 4.22: Boxplots comparing the hypervolume values on ZDT1 and ZDT2.

Experiments on Tri-objective Problems

DTLZ1 and DTLZ2 are chosen as tri-objective benchmark problems to investigate the algorithms. They are performed with a total budget of 120000 fitness evaluations, when the number of evaluations reaches 60000, the first preference region is formed, then after every 5000 evaluations, the preference region is updated. Figure 4.23 shows the Pareto front approximations from a typical run on DTLZ1 (left column) and DTLZ2 (right column). The upper graphs are obtained from DI-1 and AP-DI-1, while the lower graphs are from DI-2 and AP-DI-2. In each graph, the Pareto front approximations from DI-MOEA and corresponding AP-DI-MOEA are given. Since the target region is actually an axis aligned box, the obtained knee region (i.e., the intersection of the

axis aligned box with the Pareto front) has an inverted triangle shape for these two benchmark problems.



(a) DTLZ1 tri-objective problem. **(b)** DTLZ2 tri-objective problem.

Figure 4.23: Pareto front approximations on DTLZ1 and DTLZ2.

Table 4.9 shows the space and dominance relation of the knee point from DI-MOEA and the solution set from AP-DI-MOEA over 30 runs. For DTLZ1 problem, most knee points from DI-MOEA are in their respective preference regions and all knee points are mutually non-dominated with AP solutions. For DTLZ2 problem, it has been observed that more knee points are not in the corresponding preference regions. This is because too few solutions from DI-MOEA are in the preference region. For DTLZ1 problem, six solutions from DI-MOEA are in the corresponding preference region on average for each run, while, for DTLZ2 problem, only less than two solutions are in the corresponding preference region on average. Therefore, it can seen that on the one side, it is normal that many knee points from the entire Pareto fronts are not in their corresponding preference regions; on the other side, the aim of finding more fine-grained resolution in the preference region has been well achieved because only few solutions can be obtained in the preference region if the population is spread across

the entire Pareto front. At the same time, one knee point from DI-1 on DTLZ2 is dominated by solutions from the corresponding AP-DI-1, which proves that AP-DI-MOEA can converge better than DI-MOEA because AP-DI-MOEA focuses on the preference region.

Table 4.9: Space and dominance relation of knee point from DI-MOEA and AP solutions on DTLZ problems.

Problem		DTLZ1		DTLZ2	
Algorithm		DI-1/ AP-DI-1	DI-2/ AP-DI-2	DI-1/ AP-DI-1	DI-2/ AP-DI-2
In preference region	Incomparable	29	27	10	13
	Dominated	0	0	1	0
	Dominating	0	0	0	0
Outside p-region	Incomparable	1	3	19	17
	Dominated	0	0	0	0

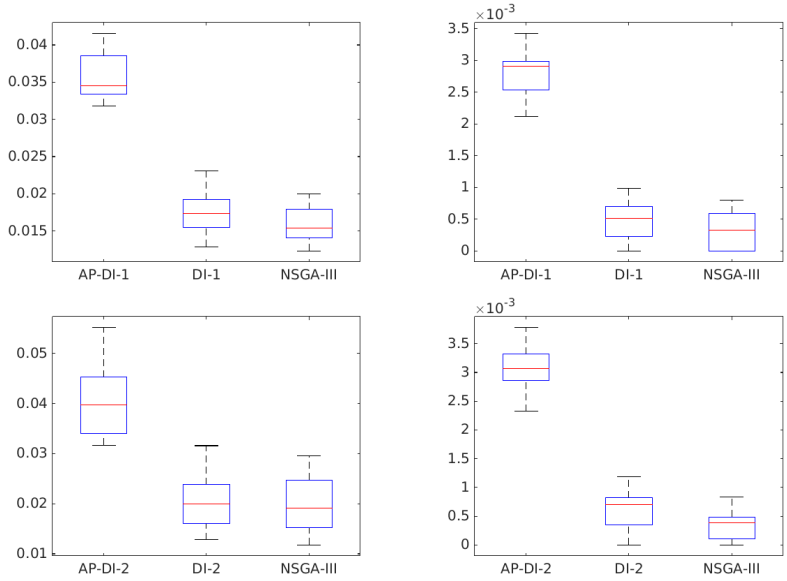
AP-DI-1 and AP-DI-2 have also been compared with NSGA-III in the same way. Table 4.10 shows the comparison result. For DTLZ1, the average number of solutions from NSGA-III in the corresponding preference regions from AP-DI-MOEA is six. Still, almost all knee solutions from NSGA-III are in the preference region. For DTLZ2, the average number of solutions from NSGA-III in the corresponding preference region from AP-DI-MOEA is less than one, while, in more than half of 30 runs, the knee points from NSGA-III are still in the preference region. To some extent, it can be concluded that the preference regions from AP-DI-MOEA are accurate. It can also be observed that AP-DI-1 behaves better than AP-DI-2 on DTLZ2, because two knee points from NSGA-III dominate the solutions from AP-DI-2.

Table 4.10: Space and dominance relation of knee point from NSGA-III and AP solutions on DTLZ problems.

Problem		DTLZ1		DTLZ2	
Algorithm		NSGA-III/ AP-DI-1	NSGA-III/ AP-DI-2	NSGA-III/ AP-DI-1	NSGA-III/ AP-DI-2
In preference region	Incomparable	30	29	14	17
	Dominated	0	0	1	1
	Dominating	0	0	0	2
Outside p-region	Incomparable	0	1	15	10
	Dominated	0	0	0	0

Similarly, solutions which are in the corresponding preference region of AP-DI-MOEA are picked from DI-MOEA and NSGA-III, and the hypervolume indicator value is compared between these solutions and AP solutions. It has been found that all hypervolume values of solutions from AP-DI-MOEA are better than those of solutions from DI-MOEA and NSGA-III. The left column of Figure 4.24 shows box plots of

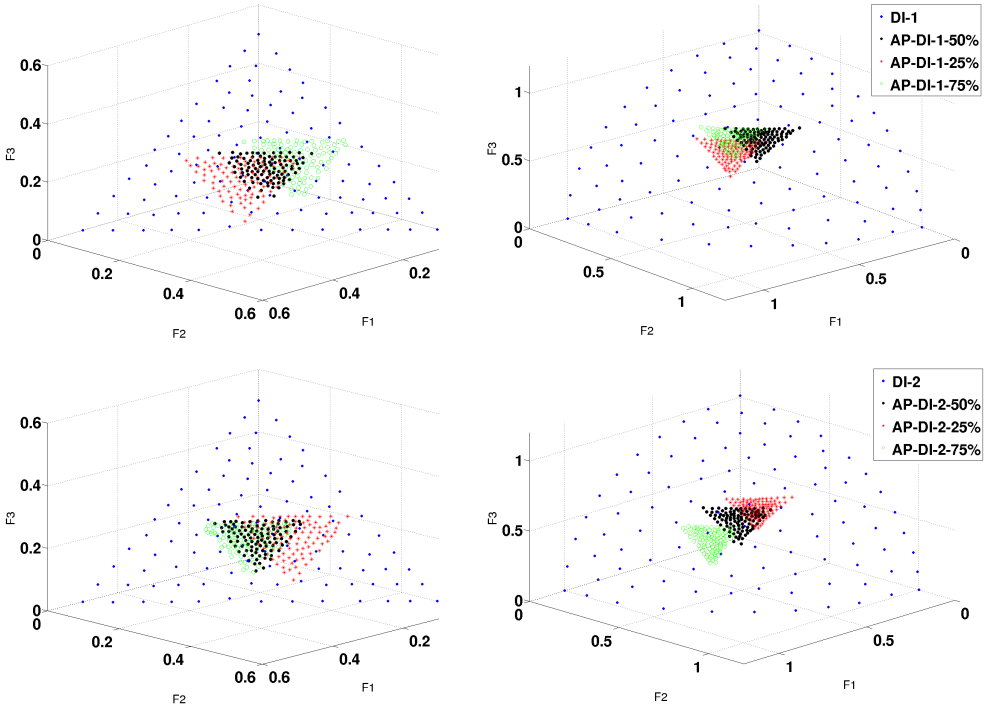
the distribution of hypervolume values over 30 runs on DTLZ1, and the right column shows the hypervolume comparison on DTLZ2.



(a) DTLZ1. **(b)** DTLZ2.

Figure 4.24: Boxplots comparing the hypervolume values on DTLZ1 and DTLZ2.

In the experiments, half of the total budget is used to find an initial Pareto front because it turned out to be a good compromise: half budget for the initial Pareto front and another half budget for the solutions focusing on the preference region. Experiments using 25% and 75% of the total budget for the initial Pareto front have also been conducted. Figure 4.25 presents the entire Pareto front from DI-MOEA and the Pareto front from AP-DI-MOEA with different budgets for the initial Pareto front. The left two images are on DTLZ1 and the right two images are on DTLZ2. The upper two images are from DI-1 and AP-DI-1; the lower two images are from DI-2 and AP-DI-2. In the legend labels, 50%, 25% and 75% indicate the budgets which are utilized to find the initial entire Pareto front. It can be observed that the preference region from AP-DI-MOEA with 50% of budget is located in a better position than with 25% and 75% of budget, and the position of the preference region from AP-DI-MOEA with 50% of budget is more stable. Therefore, in our algorithm, 50% of budget is used before the generation of the first preference region.



(a) DTLZ1 tri-objective problem. **(b)** DTLZ2 tri-objective problem.

Figure 4.25: Pareto front approximations by different budgets generating initial Pareto front.

4.2.4 Conclusion

A preference based multi-objective evolutionary algorithm, AP-DI-MOEA, is proposed. In the absence of explicitly provided preferences, the knee region is usually treated as the region of interest or the preference region. Given this, AP-DI-MOEA generates the knee region automatically and finds solutions with a more fine-grained resolution in the knee region. This has been demonstrated on the bi-objective ZDT1 and ZDT2 problems , and tri-objective DTLZ1 and DTLZ2 problems. In the benchmark, the proposed approach is also proven to perform better than NSGA-III which is included in the benchmark as a state-of-the-art reference algorithm.

It would be an interesting question how to adapt the algorithm to problems with multiple knee points and more irregular shapes. Besides, the proposed approach requires a definition of knee points. Future work will provide a more detailed comparison of different variants of methods to generate knee points.

Chapter 5

Multi-objective Scheduling Optimization

In previous chapters different MOEAs have been proposed for multi-objective optimization. However, the core issue of these algorithms is to solve the real-world application problems, as is required to answer RQ5. The exact real-world problem to be solved is the multi-objective vehicle fleet maintenance scheduling optimization (MOVFMSO) problem. As the preparatory work for solving this real-world scheduling optimization problem, in this chapter, first, study has been done on the benchmark multi-objective scheduling optimization problems, i.e., the flexible job shop scheduling problem (FJSP), and an MOEA has been proposed to solve the FJSP. Therefore, RQ4 is answered. Following this, the real-world MOVFMSO problem is formulated, its representation is defined and the problem specific genetic operators are developed. Based on these works, the previously proposed MOEAs can be applied to solve the MOVFMSO problem.

This chapter continues with Section 5.1, where the FJSP is introduced, the proposed algorithm for solving the FJSP is described and tested. Thereafter, in Section 5.2, the MOVFMSO application problem is established. To solve it, the problem representation, i.e., an encoding of the problem in decision variables, is designed for evolutionary computation. MOEAs have been devised, along with genetic operators. In this chapter, only the basic MOEAs are applied on the MOVFMSO problems, the preference-based MOEA and dynamic MOEA which is developed based on these static algorithms for dynamic environments on the MOVFMSO problem will be introduced

in next chapter.

5.1 Tailored NSGA-III Instantiation for Flexible Job Shop Scheduling

In this section, a customized multi-objective evolutionary algorithm is proposed for the multi-objective flexible job shop scheduling problem (MOFJSP) with three objectives (makespan, total workload, critical workload). Multiple initialization approaches have been adopted to produce the first-generation population based on the definition of the chromosome representation; at the same time, diverse genetic operators are applied to guide the search towards offspring with a wide diversity; especially, an algorithm configurator, i.e., the MIP-EGO configurator [110], is used to tune the parameter configuration; furthermore, two levels of local search are employed to explore the neighborhood for better solutions. In general, the proposed algorithm for the FJSP can be combined with any standard MOEAs to solve the MOFJSP. In this work, it has been combined with NSGA-III to solve some benchmark multi-objective FJSPs, whereas an off-the-shelf implementation of NSGA-III is not capable of solving them.

The remainder of this section is structured as follows. The multi-objective FJSP is first introduced in Section 5.1.1. Section 5.1.2 provides necessary background knowledge. In Section 5.1.3, the algorithm strategies for the FJSP are developed, and combined with NSGA-III, therefore, the tailored NSGA-III can solve the multi-objective FJSPs. After that, Section 5.1.4 reports the experimental results. Finally, Section 5.1.5 concludes the work and suggests future work directions.

5.1.1 Flexible Job Shop Scheduling

The job shop scheduling problem (JSP) is an important branch of production planning problems. The classical JSP consists of a set of independent jobs to be processed on multiple machines and each job contains a number of operations with a predetermined order. It is assumed that each operation must be processed on a specific machine with a specified processing time. The JSP is to determine a schedule of jobs, meaning to sequence operations on the machines. The FJSP is an important extension of the classical JSP due to the wide employment of multi-purpose machines in the real-world job shop. The FJSP extends the JSP by assuming that each operation is allowed to be processed on a machine out of a set of alternatives, rather than one specified machine. Therefore, the FJSP is not only to find the best sequence of operations on a machine,

but also to assign each operation to a machine out of a set of qualified machines. The JSP is well known to be strongly NP-hard [48]. The FJSP is an even more complex version of the JSP, so the FJSP is clearly also strongly NP-hard.

The MOFJSP addressed in this work is described as follows:

1. There are n jobs $J = \{J_1, J_2, \cdots, J_n\}$ and m machines $M = \{M_1, M_2, \cdots, M_m\}$[1].
2. Each job J_i comprises l_i operations for $i = 1, \cdots, n$, the jth operation of job J_i is represented by O_{ij}, and the operation sequence of job J_i is from O_{i1} to O_{il_i}.
3. For each operation O_{ij}, there is a set of machines capable of performing it, which is represented by M_{ij} and it is a subset of M.
4. The processing time of the operation O_{ij} on machine M_k is predefined and denoted by t_{ijk}.

At the same time, the following assumptions are made:

1. All machines are available at time 0 and assumed to be continuously available.
2. All jobs are released at time 0 and independent from each other.
3. Setting up times of machines and transportation times between operations are negligible.
4. Environmental changes (such as machine breakdowns) are neglected.
5. A machine can only work on one operation at a time.
6. There are no precedence constraints among the operations of different jobs, and the order of operations for each job cannot be modified.
7. An operation, once started, must run to completion.
8. No operation for a job can be started until the previous operation for that job is completed.

The makespan, total workload and critical workload, which are commonly considered in the literature on FJSP (e.g., [16], [131]), are minimized and used as the three objectives in our algorithm. Minimizing the makespan can facilitate the rapid response

[1] In this chapter and the next chapter, m is used to represent the number of machines or workshops to be consistent with prior literature. The objectives have been given for each specific problem.

to the market demand. The total workload represents the total working time of all machines and the critical workload is the maximum workload among all machines. Minimizing the total workload can reduce the use of machines; minimizing the critical workload can balance the workload between machines. Let C_i denote the completion time of job J_i, W_k the sum of processing time of all operations that are processed on machine M_k. The three objectives can be defined as follows:

$$\text{Makespan}(C_{max}) : f_1 = \max\{C_i | i = 1, 2, \cdots, n\} \tag{5.1}$$

$$\text{Total workload}(W_t) : f_2 = \sum_{k=1}^{m} W_k \tag{5.2}$$

$$\text{Critical workload}(W_{max}) : f_3 = \max\{W_k | k = 1, 2, \cdots, m\}. \tag{5.3}$$

An example of MOFJSP is shown in Table 5.1 as an illustration, where rows correspond to operations and columns correspond to machines. In this example, there are three machines: M_1, M_2 and M_3. Each entry of the table denotes the processing time of that operation on the corresponding machine, and the tag " $-$ " means that a machine cannot execute the corresponding operation.

Table 5.1: Processing time of a FJSP instance.

Job	Operation	M_1	M_2	M_3
J_1	O_{11}	3	-	2
	O_{12}	5	7	6
	O_{13}	-	-	2
J_2	O_{21}	2	4	3
	O_{22}	2	-	1
J_3	O_{31}	4	2	2
	O_{32}	3	5	-

5.1.2 Background and Related Work

Algorithms for MOFJSP

The FJSP has been investigated extensively in the last three decades. According to [15], EA is the most popular non-hybrid technique to solve the FJSP. Among all EAs for FJSP, some are developed for the more challenging FJSP: the MOFJSP which we formulated in Section 5.1.1, and [16], [113], [131] are very successful MOFJSP algorithms and have obtained high-quality solutions. [113] proposed a multi-objective genetic algorithm (MOGA) based on the immune and entropy principle. In this MOGA,

the fitness was determined by the Pareto dominance relation and the diversity was kept by the immune and entropy principle. In [16], a simple EA (SEA) was proposed, which used domain heuristics to generate the initial population and balanced the exploration and exploitation by refining duplicate individuals with mutation operators. A memetic algorithm (MA) was proposed in [131] and it incorporated a local search into NSGA-II [29]. A hierarchical strategy was adopted in the local search to handle objectives. In Section 5.1.4, these algorithms have been compared with the proposed algorithm on the benchmark MOFJSPs.

Parameter Tuning

EA involves using multiple parameters, such as the crossover probability, mutation probability, computational budget, as so on. The preset values of these parameters affect the performance of the algorithm in different situations. The parameters are usually set to values which are assumed to be good. For example, the mutation probability normally is kept very low, otherwise the convergence is supposed to be delayed unnecessarily. But the best way to identify the probability would be to do a sensitivity analysis: carrying out multiple runs of the algorithms with different mutation probabilities and comparing the outcomes. Although there are some self-tuning techniques for adjusting these parameters on the go, the hyper-parameters in EA can be optimized using the technique from machine learning.

The optimization of hyper-parameters and neural network architectures is a very important topic in the field of machine learning due to the large number of design choices for a network architecture and its parameters. Recently, algorithms have been developed to accomplish this automatically since it is intractable to do it by hand. The MIP-EGO [110] is one of these configurators that can automatically configure convolutional neural network architectures and the resulting optimized neural networks have been proven to be competitive with the state-of-the-art manually designed ones on some popular classification tasks. Moreover, MIP-EGO allows for multiple candidate points to be selected and evaluated in parallel, which can speed up the automatic tuning procedure. In this work, several parameters are tuned with MIP-EGO to find the best parameter setting for them.

NSGA-III

NSGA-III is a decomposition-based MOEA, it is an extension of the well-known NSGA-II and eliminates the drawbacks of NSGA-II such as the lack of uniform di-

versity among a set of non-dominated solutions. The basic framework of NSGA-III is similar to the original NSGA-II, while it replaces the crowding distance operator with a clustering operator based on a set of reference points. A widely-distributed set of reference points can efficiently promote the population diversity during the search and NSGA-III defines a set of reference points by Das and Dennis′s method [21].

In each iteration t, an offspring population Q_t of size N_{pop} is created from the parent population P_t of size N_{pop} using usual selection, crossover and mutation. Then a combined population $R_t = P_t \cup Q_t$ is formed and classified into different layers (F_1, F_2, and so on), each layer consists of mutually non-dominated solutions. Thereafter, starting from the first layer, points are put into a new population S_t. A whole population is obtained until the first time the size of S_t is equal to or larger than N_{pop}. Suppose the last layer included in S_t is the l-th layer, so far, members in $S_t \setminus F_l$ are points that have been chosen for P_{t+1} and the next step is to choose the remaining points from F_l to make a complete P_{t+1}. In general (when the size of S_t doesn't equal to N_{pop}), $N_{pop} - |S_t \setminus F_l|$ solutions from F_l needs to be selected for P_{t+1}.

When selecting individuals from F_l, first, each member in S_t is associated with a reference point by searching the shortest perpendicular distance from the member to all reference lines created by joining the ideal point with reference points. Next, a niching strategy is employed to choose points associated with the least reference points in P_{t+1} from F_l. The niche count for each reference point, defined as the number of members in $S_t \setminus F_l$ that are associated with the reference point, is computed. The member in F_l associated with the reference point having the minimum niche count is included in P_{t+1}. The niche count of that reference point is then increased by one and the procedure is repeated to fill the remaining population slots of P_{t+1}.

NSGA-III is powerful to handle problems with non-linear characteristics as well as having many objectives. Therefore, we decided to enhance NSGA-III in our algorithm for the MOFJSP.

5.1.3 Proposed Algorithm

The proposed algorithm, *Flexible Job shop Scheduling Problem Multi-Objective Evolutionary Algorithm* (FJSP-MOEA) can in principal be combined with any MOEA and help MOEAs solve the MOFJSP, whereas the standard MOEAs cannot solve MOFJSP solely. The algorithm follows the flow of a typical EA and generates improved solutions by using local search. Details of the following components are given in the next subsections.

- Initialization: encode the individual and generate the initial population.
- Genetic operators: generate offspring by crossover and mutation operators.
- Local search: decode the individual and improve the solution with local search.

Initialization

1. Chromosome Encoding

The MOFJSP is a combination of assigning each operation to a machine and ordering operations on the machines. In the algorithm, each chromosome (individual) represents a solution in the search space and the chromosome consists of two parts: the operation sequence vector and the machine assignment vector. Let N denote the number of all operations of all jobs. The length of both vectors is equal to N. The operation sequence vector decides the sequence of operations assigned to each machine. For any two operations which are processed by the same machine, the one located in front is processed earlier than the other one. The machine assignment vector assigns the operations to machines, in other words, it determines which operation is processed by which machine and the machine should be the one capable of processing the operation.

The format of representing an individual not only influences the implementation of crossover and mutation operators, a proper representation can also avoid the production of infeasible schedules and reduces the computational time. In the algorithm, the chromosomal representation proposed by Zhang et al. in [134] is adopted and an example is given in Table 5.2.

Table 5.2: An example of a chromosome representation.

Operation sequence	**1**	**2**	**3**	**2**	**1**	**1**	**3**
	O_{11}	O_{21}	O_{31}	O_{22}	O_{12}	O_{13}	O_{32}
Machine assignment	**2**	**1**	**1**	**3**	**2**	**2**	**1**
	O_{11}	O_{12}	O_{13}	O_{21}	O_{22}	O_{31}	O_{32}
	M_3	M_1	M_3	M_3	M_3	M_2	M_1

In Table 5.2, the first row shows the operation sequence vector which consists of only job indexes. For each job, the first appearance of its index represents the first operation of that job and the second appearance of the same index represents the second operation of that job, and so on. The occurrence number of an index is equal to the number of operations of the corresponding job. The second row explains the

first row by giving the real operations. The third row is the machine assignment vector which presents the selected machines for all operations. The operation sequence of the machine assignment vector is fixed, which is from the first job to the last job and from the first operation to the last operation for each job. The fourth row indicates the fixed operation sequence of the machine assignment vector and the fifth row shows the real machines of the operations. Each integer value in the machine assignment vector is the index of the machine in the set of alternative machines of that operation. In this example, O_{13} is assigned to M_3 because M_3 is the first (and only) machine in the alternative machine set of O_{13} (Table 5.1). The alternative machine set of O_{22} is $\{M_1, M_3\}$, the second machine in this set is M_3, therefore, O_{22} is assigned to M_3.

2. Initial Population

The algorithm starts by creating the initial population. The machine assignment and operation sequence vectors are generated separately for each individual. In the literature, a few approaches have been proposed for producing individuals, such as global minimal workload in [67]; AssignmentRule1 and AssignmentRule2 in [82]. In the proposed algorithm, several new methods are proposed, namely the *Processing Time Roulette Wheel* (PRW) and *Workload Roulette Wheel* (WRW) for initialising the machine assignment and the *Most Remaining Machine Operations* (MRMO) and *Most Remaining Machine Workload* (MRMW) for initialising the operation sequence. These new approaches have been used together with some commonly used dispatching rules in initializing individuals for the purpose of enriching the initial population. When generating a new individual, two initialization methods are randomly picked from the following two lists; one for the machine assignment vector and one for the operation sequence vector.

Initialization Methods for Machine Assignment

1. Random assignment (Random): an operation is assigned to an eligible machine randomly.
2. Processing time Roulette Wheel (PRW): for each operation, the roulette wheel selection is adopted to select a machine from its machine set based on the processing times of these capable machines. The machine with the shorter processing time is more likely to be selected.
3. Workload Roulette Wheel (WRW): for each operation, the roulette wheel selection is used to select a machine from its machine set based on the current workloads

plus the processing times of these capable machines. The machine with lower sum of the workload and processing time is more likely to be selected.

PRW and WRW are proposed to assign the operation to the machine with less processing time or accumulated workload, at the same time, maintaining the freedom of exploring the entire search space.

Initialization Methods for Operation Sequence

1. Random permutation (Random): starting from a fixed sequence: all job indexes of J_1 (the number of J_1 job indexes is the number of operations of J_1), followed by all job indexes of J_2, and so on. Then the array with the fixed sequence is permuted and a random order is generated.
2. Most Work Remaining (MWR): operations are placed one by one into the operation sequence vector. Before selecting an operation, the remaining processing times of all jobs are calculated respectively, the first optional operation of the job with the longest remaining processing time is placed into the chromosome.
3. Most number of Operations Remaining (MOR): operations are placed one by one into the operation sequence vector. Before selecting an operation, the number of succeeding operations of all jobs is counted respectively, the first optional operation of the job with the most remaining operations is placed into the chromosome.
4. Long Processing Time (LPT)[127]: operations are placed one by one into the operation sequence vector, each time, the operation with maximal processing time is selected without breaking the order of jobs.
5. Most Remaining Machine Operations (MRMO): operations are placed into the operation sequence vector according to both the number of subsequent operations on machines and the number of subsequent operations of jobs. MRMO is a hierarchical method and takes the machine assignment into consideration. First, the machine with the most subsequent operations is selected. After that, the optional operations in the subsequent operations on that machine are found based on the already placed operations. For example, if $O_{11} \rightarrow O_{12} \rightarrow O_{21}$ are placed operations, the current optional operation can only be chosen from O_{13}, O_{22}, and O_{31}. In these optional operations, those which are assigned to the selected machine are picked and the one that belongs to the job with the most subsequent operations is placed into the chromosome. In this example, O_{31} will be chosen if it is assigned to the selected machine because there are two subsequent operations for J_3 and only

one subsequent operation for J_1 and J_2. Note that it is possible that no operation is available on that machine, in that case, the machine with the second biggest number of subsequent operations will be selected, and so forth.

6. Most Remaining Machine Workload (MRMW): operations are placed into the operation sequence vector according to both the remaining processing times of machines and the remaining processing times of jobs. MRMW is a hierarchical method similar to MRMO. After finding the machine with the longest remaining process time and the optional operations on that machine, the operation which belongs to the job with the longest remaining process time is placed into the chromosome. Again, if no operation is available on that machine, the machine with the second longest remaining processing time will be selected, and so forth.

MRMO and MRMW are proposed to give priority to both the machine and the job with the most number of remaining operations (MRMO) and the longest remaining processing time (MRMW).

Crossover

Crossover is a matter of replacing some of the genes in one parent with the corresponding genes of the other. Since the representation of chromosomes has two parts, crossover operators applied to these two parts of chromosomes are implemented separately as well. Two new crossover operators, *Precedence Preserving Two Points Crossover* (PPTP) and *Uniform Preservative crossover* (UPX), are proposed and used together with several commonly adopted crossover operators. When executing the crossover operation in the proposed algorithm, one crossover operator for machine assignment and one operator for the operation sequence, are randomly chosen from the following two lists to generate the offspring.

Crossover Operators for Machine Assignment

1. No crossover
2. One point crossover: a cutting point is picked randomly and genes after the cutting point are swapped between two parents.
3. Two points crossover: two cutting points are picked randomly and genes between the two points are swapped between two parents.
4. Job-based crossover (JX): it generates two children from two parents by the following procedure:

   a A vector with the size of the jobs is generated, which consists of random values 0 and 1.
   b For the job corresponding to value 0, the assigned machines of its operations are preserved.
   c For the job corresponding to value 1, the machines of its operations are swapped between two parents.

5. Multi-point preservative crossover (MPX)[133]: MPX generates two children from two parents by the following procedure:

   a A vector with the size of all operations is generated, which consists of random values 0 and 1.
   b For the operations corresponding to value 0, their machines (genes) are preserved.
   c For the operations corresponding to value 1, their machines (genes) are swapped between the two parents.

Crossover Operators for Operation Sequence

1. No crossover
2. Precedence preserving one point crossover (PPOP) [102]: PPOP generates two children from two parents by the following procedure:

   a A cutting point is picked randomly, genes to the left are preserved and copied from parent1 to child1 and from parent2 to child2.
   b The remaining operations in parent1 are reallocated in the order they appear in parent2.
   c The remaining operations in parent2 are reallocated in the order they appear in parent1.

   An example of PPOP is shown in Figure 5.1 and the cutting point is between the third and fourth operation. Red numbers in parent2 are the genes on the right side of the cutting point in parent1 and they are copied to child1 with their own sequence following the genes on the left side of the cutting point in parent1, and vice versa.
3. Precedence Preserving Two Points Crossover (PPTP): PPTP generates two children from two parents by the following procedure:

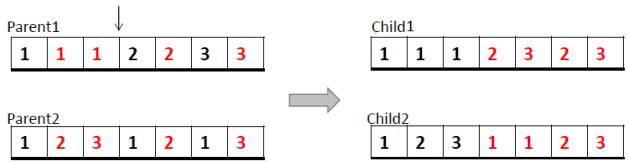


Figure 5.1: The process of PPOP.

a Two cutting points are picked randomly, genes except for those between the two points are preserved and copied from parent1 to child1 and from parent2 to child2.

b Operations between the two cutting points in parent1 are reallocated in the order they appear in parent2.

c Operations between the two cutting points in parent2 are reallocated in the order they appear in parent1.

4. Improved precedence operation crossover (IPOX)[132]: IPOX divides the job set into two complementary and non-empty subsets randomly. The operations of one job subset are preserved, while the operations of another job subset are copied from another parent.

5. Uniform Preservative crossover (UPX): UPX generates two children from two parents by the following procedure:

   a A vector with the size of all operations is generated, which consists of random values 0 and 1.

   b For the operations corresponding to value 0, the genes are preserved and copied from parent1 to child1 and from parent2 to child2.

   c For the operations corresponding to value 1, the genes in parent1 are found in parent2 and copied from parent2 with the sequence in parent2, and vice versa.

Mutation

The mutation operator flips the gene values at selected locations. By forcing the algorithm to search areas other than the current area, the mutation operator is used to maintain genetic diversity from one generation of a population to the next. In this algorithm, insertion mutation and swap mutation (including one point swap and two points swap) are proposed and used.

Insertion Mutation Operator generates a new individual by the following procedure:

- Two random numbers i and j ($1 \leq i \leq N$, $1 \leq j \leq N$) are selected.
- For the operation sequence vector, the operation on position j is inserted in front of the operation on position i.
- For the machine assignment vector, a machine is randomly selected for both the operations on i and on j respectively. If the processing time on the newly selected machine is lower than that on the current machine, the current machine is replaced by the new machine. If the processing time on the new machine is longer than that on the old machine, there is only a 20% probability that the new machine replaces the old machine.

Swap Mutation Operator generates a new individual by the following procedure:

- One random number i ($1 \leq i \leq N$) is selected or two random numbers i and j ($1 \leq i \leq N$, $1 \leq j \leq N$) are selected.
- For the operation sequence vector, with only one swap point i, the operation on the swap point is swapped with its neighbour; with two swap points, the operations on position i and j are swapped.
- For the machine assignment vector, the machine on position i (and j) is replaced with a new machine by the same rule used in the insertion mutation operator.

Decoding and Local Search

Decoding a chromosome is to convert an individual into a feasible schedule to calculate the objective values which represents the relative superiority of a solution. In this process, the operations are picked one by one from the operation sequence vector and placed on the machines from the machine assignment vector to form the schedule. When placing each operation to its machine, local search (in the sense of heuristic rules to improve solution) is involved to refine an individual in order to obtain an improved schedule in the proposed algorithm. Two levels of local search are applied to allocate each operation to a time slot on its machine. We know that idle times may exist between operations on each machine due to precedence constraints among operations of each job, and two levels of local search utilize idle times in different degrees.

The First Level Local Search

Let S_{ij} be the starting time of O_{ij} and C_{ij} the completion time of O_{ij}, an example of the first level local search is shown in Figure 5.2. Because O_{mn} needs to be processed after the completion of O_{mn-1}, an idle time interval between the completion of O_{ab} and the starting of O_{mn} appeared on machine M_k. O_{ij} is assigned to M_k and we assume that O_{mn} is the last operation on M_k before handling O_{ij}, therefore the starting time of O_{ij} is $\max\{C_{mn}, C_{ij-1}\}$, which in this example is C_{mn} and it is later than C_{ij-1}, thus, there is an opportunity that O_{ij} can be processed earlier. When checking the idle time on M_k, the idle time interval $[C_{ab}, S_{mn}]$ is found available for O_{ij} because the idle time span $[C_{ij-1}, S_{mn}]$, which is part of $[C_{ab}, S_{mn}]$, is enough to process O_{ij} or longer than t_{ijk}.

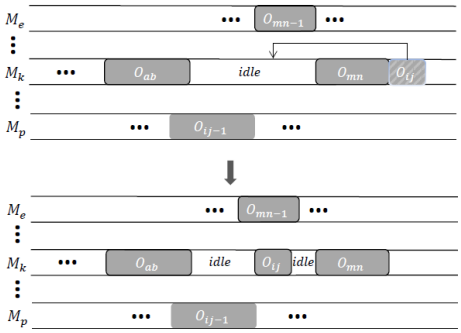


Figure 5.2: First level local search

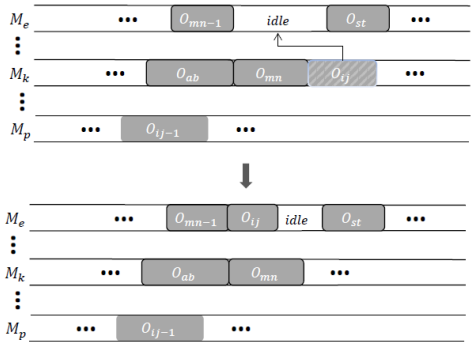


Figure 5.3: Second level local search

Let S_k^d be the starting time of the dth idle time interval on M_k and C_k^d be the completion time. O_{ij} can be transferred to an earliest possible idle time interval of its machine which satisfies the following equation:

$$\max\{S_k^d, C_{ij-1}\} + t_{ijk} \leq C_k^d, (C_{ij} = 0, \text{if } j = 1). \tag{5.4}$$

After using the idle time interval, the starting time of O_{ij} is $\max\{S_k^d, C_{ij-1}\}$ and the idle interval is updated based on the starting and completion time of O_{ij}: (1) the idle time interval is removed; (2) the starting or completion time of the idle time interval is modified; (3) the idle time interval is replaced by two new shorter idle time intervals, like in the example of Figure 5.2.

After decoding a chromosome, the operation sequence vector of the chromosome is updated according to new starting times of operations, and three objective values are

calculated. The first level local search only finds for each operation the available idle time interval on its assigned machine. After generating the corresponding schedule with the first level search method, it is possible that there are still operations that can be allocated to available idle time intervals to benefit the fitness value. To achieve this, decoding the chromosome which has been updated with the first level local search is performed with the second level local search, and again operations are moved to available idle time intervals.

The Second Level Local Search

The second level local search not only checks the idle time intervals on the assigned machine, but also the idle time intervals on alternative machines. An example of making use of the idle time interval on another machine is shown in Figure 5.3. Let S_{ijk} be the starting time and C_{ijk} be the completion time of O_{ij} on M_k. In this example, O_{ij} is assigned to M_k in the initial chromosome, we assume that O_{ij} can also be performed by M_e. Under the condition that the starting time of O_{ij} on M_k is later than the completion time of O_{ij-1}, the idle time intervals on all alternative machines which can process O_{ij} are checked. An idle time interval on M_e could be a choice and O_{ij} can be reallocated to M_e. In this example, the processing time of O_{ij} on M_e is even shorter then the processing time on M_k, therefore, this reallocation can at least benefit the total workload.

In the second level local search, all available idle time intervals of an operation are checked one by one until the first "really" available idle time interval is found and then the operation is moved to that idle time interval. Any idle time interval on an alternative machine which can satisfy Equation 5.4 is an available idle time interval, while it must meet at least one of the following conditions to become a "really" available idle time interval.

1. The processing time of the operation on the new machine is shorter than on the initially assigned machine if the available idle time interval is on a different machine;
2. The operation can be moved from the machine with the maximal makespan to another machine.
3. The operation can be moved from the machine with the maximal workload to another machine.

The total workload can be improved directly by the first condition; the motive of the

second condition is to decrease the maximal makespan and the third condition can benefit the critical workload.

After the reallocation of the operations with the second level local search, the corresponding schedule is obtained and objective values are calculated. While, instead of updating the chromosome immediately, the new objective values are compared with the old objective values first, the chromosome can be updated only when at least one objective is better than its old value. This is to make sure that the new schedule is at least not worse than the old schedule (The new solution is not dominated by the old solution). Another difference between the first and second level local search is that the first level local search is performed on every evaluation, while the second level local search is only performed with a 30% probability for each chromosome to avoid local optima. Although these two local searches can be applied repeatedly to improve the solution, to avoid that the algorithm is stuck in a local optima, they are employed only once for each evaluation.

5.1.4 Experimental Results

The algorithms are tested on two sets of well-known FJSP benchmark instances: 4 Kacem instances (ka4x5, ka10x7, ka10x10, ka15x10) and 10 BRdata instances (Mk01-Mk10). Table 5.3 gives the scale of these instances. The first column is the name of each instance; the second column shows the size of the instance, in which n stands for the number of jobs and m the number of machines; the third column represents the number of operations; the fourth column lists the flexibility of each instance, which means the average number of alternative machines for each operation in the problem.

All the experiments are performed with a population size of 100, each run of the algorithm will stop based on a predefined number of evaluation, which is 10000 for Kacem instances and 150000 for BRdata instances. For each problem instance, the proposed algorithm is independently run 30 times. The resulting set is formed by all non-dominated solutions from the union of 30 runs.

The crossover probability is set to 1 and two random crossover operators can be chosen each time (one for operation sequence and one for machine assignment). For Kacem instances, the mutation probabilities are set to 0.6. For BRdata instances, which include larger-scale and more complex problems, the MIP-EGO configurator [110] is adopted to tune both insertion and swap mutation probabilities (one point swap mutation and two points swap mutation) to find the best parameter values for each problem. The hypervolume of the solution set has been used in MIP-EGO as the

Table 5.3: The scale of benchmark instances.

Instance	n → m	#Opr	Flex.
ka4x5	4 → 5	12	5
ka10x7	10 → 7	29	7
ka10x10	10 → 10	30	10
ka15x10	15 → 10	56	10
Mk01	10 → 6	55	2
Mk02	10 → 6	58	3.5
Mk03	15 → 8	150	3
Mk04	15 → 8	90	2
Mk05	15 → 4	106	1.5
Mk06	10 → 15	150	3
Mk07	20 → 5	100	3
Mk08	20 → 10	225	1.5
Mk09	20 → 10	240	3
Mk10	20 → 15	240	3

objective value to tune three mutation probabilities. Although the true PFs for test instances are unknown, [131] provides the reference set for Kacem and BRdata FJSP instances, which is formed by gathering all non-dominated solutions found by all the implemented algorithms in [131] and also non-dominated solutions from other state-of-the-art MOFJSP algorithms. The reference point for calculating the hypervolume value is determined by the largest value in this reference set. To be specific, each objective function value of the reference point is: 1.1 × largest objective function value of the respective dimension in the reference set. The origin point is used as the ideal point. Other basic parameter settings of MIP-EGO are listed in Table 5.4. For each mutation probability, we only consider a discretized number with only one digit after the decimal point, therefore, the search space is ordinal or integer space, which in MIP-EGO are handled in the same way.

Table 5.4: Settings for MIP-EGO.

Parameter	value
maximal number of evaluations	200
surrogate model	random forest
optimizer for infill criterion	MIES
search space	ordinal space

Table 5.5 shows the percentage of the evaluations which can achieve the largest hypervolume value (or the best PF) by MIP-EGO (200 Evaluations). In each evaluation, MIP-EGO assigns a specific parameter setting for our optimization algorithm. It

can be observed for Mk05 and Mk08 that all the evaluations have obtained the largest hypervolume value; it means that all parameter values of mutation probabilities in MIP-EGO can achieve the best PF for these two problems. It can also be seen in Table 5.3 that both problems have a low flexibility value. On the contrary, for Mk06, Mk09 and Mk10, these problems have a large operation number and high flexibility. It seems that they can be difficult to solve because only one best parameter setting for mutation probabilities has been found among all evaluations. This also means that it is highly likely better solution sets can be found with a higher budget.

Table 5.5: Probability of finding best configuration.

Mk01	Mk02	Mk03	Mk04	Mk05	Mk06	Mk07	Mk08	Mk09	Mk10
73%	60%	95%	1%	100%	0.5%	4.5%	100%	0.5%	0.5%

With the best parameter setting of the mutation probabilities for BRdata instances, our experimental results are compared with the reference set in [131]. The proposed algorithm can achieve the same Pareto optimal solutions as in the reference set for all BRdata instances except for Mk06, Mk09 and Mk10. At the same time, for Mk06 and Mk10, our algorithm can find new non-dominated solutions. Table 5.6 is the list of new non-dominated solutions obtained by our algorithm, each row of an instance is a solution with three objectives: makespan, total workload, and critical workload.

Table 5.6: Newly achieved non-dominated solutions.

Mk06			Mk10		
61	427	53	218	1973	195
63	428	52	218	1991	194
63	435	51	219	1965	195
65	453	49	220	1984	191
66	451	49	225	1979	194
66	457	48	226	1954	196
			226	1974	194
			226	1979	192
			228	1973	194
			235	1938	199
			236	1978	193

Another comparison is between our algorithm (FJSP-MOEA) and MOGA [113], SEA [16] and MA1, MA2 [131]. In [131], there are several variants of the proposed algorithm with different strategies in the local search. MA1 and MA2 are chosen as compared algorithms because they perform equally good or superior to other variants

on almost all problems. Table 5.7 displays the hypervolume value of the PF approximation from all algorithms and the new reference set which is formed by combining all solutions from the PF by all algorithms. The highest hypervolume value on each problem in all algorithms has been highlighted in bold. It can be observed that FJSP-MOEA and MA1, MA2 show the best and similar performance, and MOGA behaves the best for three of the BRdata instances. The good performance of MOGA on three problems is interesting. MOGA has an entropy-based mechanism to maintain decision space diversity which might be beneficial for solving these problem instances. When using one best parameter setting, the average hypervolume and standard deviation from 30 runs on each problem are given in Table 5.8, the standard deviation of each problem shows the stable behaviour of each run.

Table 5.7: Hypervolume from MOGA, SEA, MA1, MA2, FJSP-MOEA and the reference set.

Problem	MOGA	SEA	MA1	MA2	FJSP-MOEA	Ref
Mk01	0.00426	0.00508	**0.00512**	**0.00512**	**0.00512**	0.00512
Mk02	0.01261	0.01206	**0.01294**	**0.01294**	**0.01294**	0.01294
Mk03	**0.02460**	0.02165	0.02165	0.02165	0.02165	0.02809
Mk04	**0.06906**	0.06820	0.06901	0.06901	0.06901	0.07274
Mk05	0.00626	0.00635	**0.00655**	**0.00655**	**0.00655**	0.00655
Mk06	0.05841	0.06173	0.06585	0.06692	**0.06709**	0.07065
Mk07	0.02244	0.02132	**0.02269**	**0.02269**	**0.02269**	0.02288
Mk08	**0.00418**	0.00356	0.00361	0.00361	0.00361	0.00428
Mk09	0.01547	0.01755	0.01788	**0.01789**	0.01785	0.01789
Mk10	0.01637	0.01778	0.02145	**0.02196**	0.02081	0.02249

Table 5.8: Average hypervolume and std with the best parameter setting.

Problem	Mk01	Mk02	Mk03	Mk04	Mk05	Mk06	Mk07	Mk08	Mk09	Mk10
Ave-HV	0.0050	0.0122	0.0216	0.0672	0.0064	0.0598	0.0222	0.0036	0.0174	0.0186
Std	0	0.0003	0.0001	0.0004	0.0001	0.0019	0.0003	0	0.0002	0.0006

For Kacem instances and with fixed mutation probabilities, the obtained non-dominated solutions by the proposed algorithm are the same as the PF in the reference set. MA1 and MA2 also achieved the best PF for all Kacem instances, but the proposed FJSP-MOEA uses far less computational resources. It uses only a population size of 100 whereas the population size of MA algorithms is 300. FJSP-MOEA uses only 10000 objective function evaluations, whereas MA uses 150000 evaluations. In terms of computational resources the proposed FJSP-MOEA can therefore be used on

smaller computer systems, entailing broader applicability, and possibly also in real-time algorithm implementations such as dynamic optimization.

5.1.5 Conclusion

A novel multi-objective evolutionary algorithm for the multi-objective flexible job shop scheduling problem (MOFJSP) is proposed. It uses multiple initialization approaches to enrich the first generation population, and various crossover operators to create better diversity for offspring. Moreover, to determine the optimal mutation probabilities, the MIP-EGO configurator is adopted to automatically generate proper mutation probabilities. Besides, the straightforward local search is employed with different levels to aid more accurate convergence to the PF. The proposed customization approach in principle can be combined with almost all MOEAs. In this work, it is incorporated with one of the state-of-the-art MOEAs, namely NSGA-III, to solve the MOFJSP, and the new algorithm can find all Pareto optimal solutions in literature for most problems, and even new Pareto optimal solutions for the large scale instances.

The ability of the MIP-EGO configurator in finding the optimal mutation probabilities is shown in this work. However, there is more potential in the automated parameter configuration domain that can benefit EA. For example, to know the effects of different initialization approaches and crossover operators, we can optimize the initialization and crossover configuration. Furthermore, other parameters of the proposed algorithm, such as, population size, evaluation number, and so on, can also be tuned automatically. However, so far the efficiency of the existing tuning framework is limited when it comes to a larger number of parameters. It would therefore be a good topic of future research to find more efficient implementations of these.

5.2 MOEAs for Vehicle Fleet Maintenance Scheduling Optimization

Nowadays, companies, corporations, and organizations of all sorts rely on vehicle fleets to deliver products and services. Typical examples of such vehicle fleets are taxi cab fleets, public bus fleets, car rental fleets, delivery fleets, and so on. According to the statistical data from the European Automobile Manufacturers Association, the global vehicle fleet grows continuously, and the EU has a total fleet of 259.7 million

passenger cars and 39.1 million commercial vehicles in 2017[2]. The maintenance of fleet vehicles plays a critical role in their efficient use. Fleet vehicles should be maintained according to a schedule to ensure that they are safe for use; at the same time, a good maintenance schedule can reduce related expenses, increase the efficiency of the assets, ensure consistent service delivery, and even reduce its carbon footprint.

Due to various tasks the vehicles execute, damages to vehicles occur after varying duration. To be more precise, the critical components of each car need to be maintained based on their respective damages from wear and tear. To decide when each component should be maintained, the remaining useful lifetime (RUL) of each component, which is the time remaining until the component no longer meets operational requirements, can be predicted based on adequate predictive approaches or models [35]. Other than the predicted RUL of components, to maintain a vehicle fleet, different maintenance resources are needed. For example, a vehicle component may take several days to be repaired and the maintenance activity can be performed on one of several optional workshops.

The goal of optimizing the maintenance schedule for a vehicle fleet is to ensure that all maintenance tasks are performed on time, keeping the vehicle fleet in healthy operating condition and under business requirements. This section continues in Section 5.2.1 with the formulation of the vehicle fleet maintenance scheduling optimization (VFMSO) problem. A literature review is provided in Section 5.2.2. The customized multi-objective evolutionary algorithm for the VFMSO problem is described in Section 5.2.3, Section 5.2.4 presents and discusses experiments and their results. Lastly, Section 5.2.5 concludes the work and outlines directions for future work.

5.2.1 Problem Formulation

In the real world, the size of a vehicle fleet can be large and its distribution wide, which makes it necessary to distribute the maintenance of a vehicle fleet in multiple separate workshops. At the same time, each workshop has its own capacity and ability, meaning that on the one hand, each workshop has its own team and each team can work on only one car simultaneously; on the other hand, each workshop is limited to the maintenance of the specific component(s) due to restrictions in the equipment or skill level of the staff. These conditions form the primary constraints faced by the vehicle fleet maintenance scheduling optimization. Moreover, the cost and time which are needed to repair car components by different workshops are required. It

[2]https://www.acea.be/statistics/tag/category/size-distribution-of-vehicle-fleet retrieved on 17th of December 2018.

is possible that the maintenance of the same component produces different costs and workloads when the activity is performed in different workshops because, for example, the distances between vehicles and the workshops are different. The set-up cost and set-up time are fixed for each visit of a car to a workshop, which correspond to the cost and time required for the preparation of the maintenance. In this work, the predicted RULs of components are converted to the due dates to determine the maintenance time of each component which is the estimate of the date when the component fails in case no maintenance takes place before its due date.

The vehicle fleet maintenance scheduling optimization problem addressed in this work is defined as follows:

1. There are n cars $C = \{C_1, C_2, \cdots, C_n\}$ and m workshops $W = \{W_1, W_2, \cdots, W_m\}$.
2. Each car C_i comprises l_i components to be maintained for $i = 1, \cdots, n$.
3. For each component O_{ij}, i.e., the jth operation of car C_i, there is a set of workshops capable of repairing it. The set of workshops is represented by W_{ij} which is a subset of W.
4. The processing time of the maintenance of the component O_{ij} in workshop W_k is predefined and denoted by p_{ijk}.
5. The maintenance cost of the maintenance of the component O_{ij} in workshop W_k is predefined and denoted by q_{ijk}.
6. The set-up time of car C_i in workshop W_k is predefined and denoted by X_{ik}.
7. The set-up cost of car C_i in workshop W_k is predefined and denoted by Y_{ik}.
8. The number of teams in workshop W_k is predefined and denoted by Z_k.
9. The due date and previous repair time of each component O_{ij} are predefined and denoted by D_{ij} and R_{ij} respectively.
10. All business requirements or vehicle demands are predefined. There are r vehicle demands and the format of one demand is: N_i cars are required from day d_{i1} to day d_{i2} for $i = 1, \cdots, r$.

At the same time, the following assumptions are made:

1. All workshops and cars are available at time 0 and assumed to be continuously available.
2. All the components are independent from each other.

3. Times required for transport of cars from/to workshops are included in the maintenance time and cost of cars.

4. Environmental changes (such as car accidents) are not considered here.

5. There are no precedence constraints among the components of different cars. Cars are maintained on a first-come-first-served basis.

6. An operation, once started, must run to completion.

7. No operation can start before completion of the previous operation.

A multi-objective scheduling optimization problem is considered in this work and three objectives are assumed to be relevant for the vehicle fleet operator, which are the total workload, total cost and demand satisfaction. The reason why demand satisfaction is defined as objective is that it is treated as flexible and violable. Let T_k denote the sum of the maintenance times spent on all operations that are processed in workshop W_k; M_i the sum of all costs of all operations of car C_i; N^t_{avail} the number of cars which are not in workshops on day t, N^t_{demand} the number of cars required on day t. Three objectives can be defined as:

$$\text{Minimize the total workload} : f_1 = \sum_{k=1}^{m} T_k \tag{5.5}$$

$$\text{Minimize the total cost} : f_2 = \sum_{i=1}^{n} M_i \tag{5.6}$$

$$\text{Maximize the demand satisfaction} : f_3 = \sum_{t} \min\{N^t_{avail} - N^t_{demand}, 0\}. \tag{5.7}$$

For illustration purposes, an example of parameters for a car is shown in Table 5.9, where rows correspond to components of the car; columns correspond to the cost, processing time, set-up time and set-up cost of the car in alternative workshops, also due date and previous repair date. In this example, there are two workshops: W_1 and W_2. The tag " $-$ " means that the workshop cannot repair the corresponding component, therefore, workshop W_1 cannot maintain O_{14} and workshop W_2 cannot maintain O_{11}. Here, the predefined costs of components are the same at both workshops: €200, so are the processing times, set-up costs and set-up times. Of course, times and costs in different workshops can be different. The due date and previous repair time are relative to day 0.

Table 5.9: Parameters of car C_1.

Comp	cost		time		date		set-up cost		set-up time	
	W_1	W_2	W_1	W_2	Due	Pre	W_1	W_2	W_1	W_2
O_{11}	200	-	2	-	74	-61	100	1	100	1
O_{12}	200	200	2	2	25	-6				
O_{13}	200	200	2	2	15	-50				
O_{14}	-	200	-	2	60	-1				

5.2.2 Literature Review

In the early studies of predictive maintenance, also called condition-based maintenance (CBM) [63], the condition monitoring, failure diagnostics, and prognostics always attracted more attention than planning the maintenance schedule based on the information obtained from the condition monitoring, failure diagnostics and prognostics. In [83], an onboard locomotive diagnostic system was invented to monitor the transmitted onboard vehicle data, determine whether any of the monitored data is out of a predetermined range, compare monitored data with historical data and calculate trends, predict if any vehicle system(s) must be corrected to avoid vehicle failure and when such system(s) are likely to fail. It was mentioned in [83] that the onboard diagnostic systems are not helpful in optimizing locomotive maintenance scheduling because they do not communicate with a rail carrier's scheduling center.

Simultaneously, the prognostic information included in maintenance policies may be given in different formats such as RUL, efficiency decrease, the probability of failure. Prognostics and reliability information in [14] were the failure probabilities of the components. The failure probabilities were analyzed to schedule maintenance with minimum system risk using a genetic algorithm. The systematic risk was treated as part of the cost of performing the maintenance schedule. System conflicts and resources were also considered as constraints in this single-objective problem. [10] introduced deterioration models to take into account the component degradation information of a multi-component system. The optimal dates of maintenance operations can be computed based on the model, and additional costs can be evaluated if the operations are not executed at the optimal maintenance dates. Lastly, the optimal grouping of individual maintenance actions was found to reduce the maintenance cost, and the maintenance plan was updated dynamically at each inspection date.

Optimizing the maintenance schedule has been widely neglected in the study of predictive maintenance, while it can be crucial for effective maintenance planning and scheduling. As an essential branch of production planning problems and the basis

of maintenance scheduling optimization, the flexible job shop scheduling problem has been extensively studied in the literature (please refer to Section 5.1). According to [15], most research papers have addressed classical FJSP in the last 25 years, while only around 35% of papers considered different scenarios such as machine breakdown, uncertain processing times, overlapping operations and so on. [81] considered FJSP-PPF (process plan flexibility), where jobs can have alternative process plans. It was assumed that the process plans are known in advance and that they are represented by linear precedence relationships. Because only one of the alternative plans had to be adopted for each job, the FJSP-PPF dealt with not only routing and sequencing sub-problems, but also the process plan selection sub-problem. In this paper, a mixed-integer linear programming model was developed for the FJSP-PPF and makespan was adopted as the single performance measure.

In [32], a mathematical model and a genetic algorithm were proposed to handle the feature of overlapping in operations. It was assumed that a lot which contains a batch of identical items is transferred from one machine to the next only when all items in the lot have completed their processing, therefore, sublots are transferred from one machine to the next for processing without waiting for the entire lot to be processed at the predecessor machine, meaning that starting a successor operation of job is not necessary to finish of its predecessor completely.

Three features were considered in [129], which were (1) job priority; (2) parallel operations: some operations can be processed simultaneously; (3) sequence flexibility: the sequence of some operations can be exchanged. A mixed integer linear programming formulation (MILP) model was established to formulate the problem and an improved differential evolution algorithm was designed.

Because of unexpected events occurring in most of the real manufacturing systems, there is a new type of scheduling problem known as the dynamic scheduling problem. This type of problem considers random machine breakdowns, adding new machines, new job arrival, job cancellation, changing processing time, rush order, rework or quality problem, due date changing, etc. Corresponding works on the FJSP include [1], [2], [42], [94].

Compared with the FJSP introduced in Section 5.1.1, the VFMSO problem has some special properties: (1) flexible sequence: the sequence of the components is not predefined, and the starting time of each component is mainly determined by its due date. (2) multiple problem parameters: besides the processing time, other problem parameters like the maintenance cost, set-up time, set-up cost, repair teams, the demand for cars at a specific time, also have an impact on the maintenance schedule.

5.2.3 Customized Algorithm

A specialized evolutionary algorithm framework is defined and applied with MOEAs to solve the multi-objective vehicle fleet maintenance scheduling optimization (MOVFMSO) problem. In this part, the approach underlying the algorithm and the implementation of genetic search, including chromosome encoding, chromosome decoding and genetic operators, are described.

Components Grouping

When scheduling components, the idea of grouping several components of the same car for one visit is employed in the proposed algorithm. By grouping the maintenance of multiple components into one maintenance operation, the set-up cost and set-up time apply only once for the complete group of components. However, the maintenance cost could be indirectly penalized:

- with the reduction of the component useful life if the maintenance date is shifted backward;
- with the increase of the risk of breaking down on the road if the maintenance date is shifted forward.

Therefore, the maintenance of each component should not be shifted too far from its due date. In the proposed algorithm, an interval or execution window of the starting time is defined for each component, and the maintenance of the component can only start at a time spot inside the corresponding interval. The interval of each component consists of two parts and their respective lengths are defined as:

- Length of the interval before the due date: 0.3× (Due date - Previous repair date);
- Length of the interval after the due date: 0.1× (Due date - Previous repair date).

The interval is chosen relatively long so that maintenance before or after the interval hardly makes sense. With the interval, combining components can only be effective if their intervals overlap, and the starting time of the group maintenance must lie within the interval intersection.

Grouping components also means that not all the components can be maintained exactly at their due dates. For the component that is maintained before or after its due date, an extra cost is introduced to penalize either the reduction of the useful life when

the maintenance is performed before the due date or the increase of component failure probability when the maintenance is performed after the due date. Figure 5.4 gives an example of the execution window and penalty function of a component. Two linear penalty functions are proposed to calculate the penalty cost based on the following assumptions.

- If a component is maintained at the same time as the previous repair time, the penalty cost would be $c+s$: the sum of its maintenance cost and the set-up cost of the car;
- If a component is maintained at the end of the interval (the latest possible repair time), the penalty cost would be $100 * c$ (c: the maintenance cost of the component). This penalty cost is the combination of all losses: the expense needed if the defect occurs on the use (diagnostics, technical and logistics support, repair or replacement of the failed component), the loss of reputation, and so on.

As can be seen, if the component O_{ij} is maintained at t_{ij} (t_{ij} is in its interval), its penalty cost is:

$$\begin{cases} ((c+s)/(D_{ij}-R_{ij})) \times (D_{ij}-t_{ij}) & \text{if} \quad t_{ij} < D_{ij} \\ 0 & \text{if} \quad t_{ij} = D_{ij} \\ ((100 \times c)/((D_{ij}-R_{ij})/10)) \times (t_{ij}-D_{ij}) & \text{if} \quad t_{ij} > D_{ij}. \end{cases}$$

When components are grouped together, the maintenance time of the group is the sum of the processing times of all components in the group plus one set-up time; the cost of the group is the combination of one set-up cost, the maintenance costs and the penalty costs of all components in the group.

Chromosome Encoding

EAs typically start with a diverse set of feasible solutions (a population) and iteratively replace the current population by a new population. A suitable encoding is required for the problem. Based on the properties of the application problem, a three-vector chromosome (Figure 5.5) is proposed to represent an individual, which includes:

- group structure vector: the group structures of vehicles one by one;
- starting time vector: the starting times of group operations;
- workshop assignment vector: the workshops of group operations.

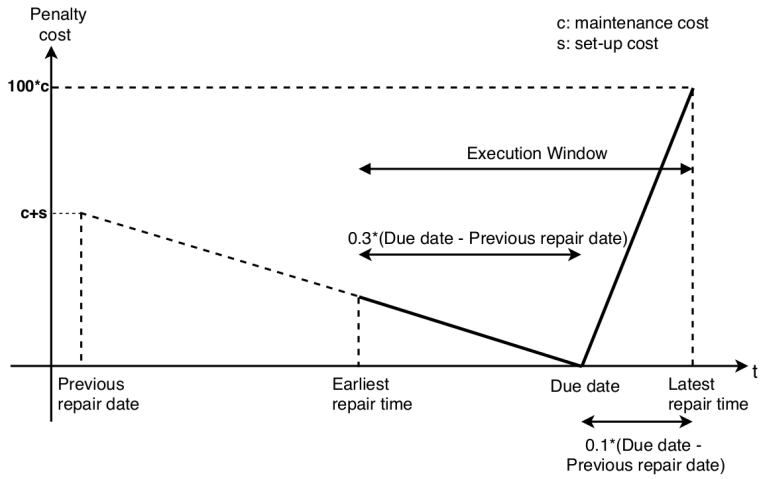


Figure 5.4: Execution window and penalty function of a component.

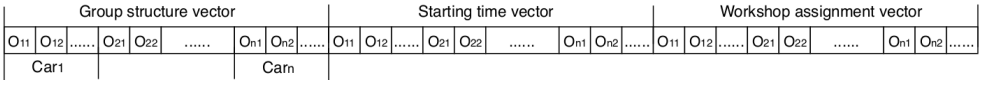


Figure 5.5: Three-vector chromosome.

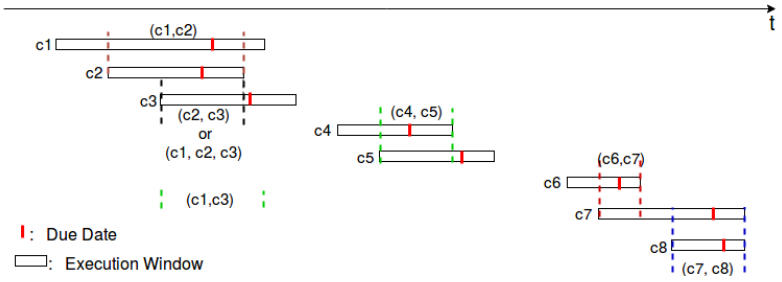


Figure 5.6: Possible groups of a car with eight components.

1. Group structure vector Figure 5.6 represents the intervals of eight components of a car. Component $c1$ can be grouped with $c2$ and/or $c3$ due to the overlap between their execution windows. Other possible group structures can be deduced in the same manner. However, the practical situation can be more complicated. For example, in Figure 5.7, $c1$ can be grouped with $c2$; $c1$ can be grouped with $c3$ and $c4$; $c1$ can also be grouped with $c3$ and $c5$. But $c1$ and $c2$ together cannot be grouped with other components; $c1$, $c3$ and $c4$ together cannot be grouped with $c5$. The overlaps and possible group structures of all components from one car are checked by the following

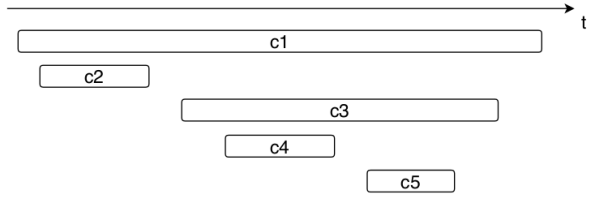


Figure 5.7: Five component intervals of a car.

sweep line algorithm:

1. All components of one car are sorted as a list according to the left end of the intervals.
2. Starting from the first component, all components are checked one by one. For each component, only this component and the subsequent components in the list are necessary in the check because the possible groups with the preceding components have been found when checking those components.
3. A stack is utilized to store the intervals and their corresponding components. The interval of the checked component is pushed onto the stack first. Look at the next component, if there is an overlap between them, push the overlap interval onto the stack; otherwise, pop off the topmost element from the stack, return the only group possibility and end the checking process.
4. Compare the topmost interval on the stack with the interval of the next component, if they overlap, push the overlap interval onto the stack; otherwise, pop off the topmost element from the stack. Repeat this step until the interval stack is empty or the last component has been checked.

All possible group numbers will be found after the above steps. Figure 5.8 shows the alternative groups of components on Figure 5.6 (left table) and Figure 5.7 (right table). Each component corresponds to a column and the numbers in its column are all the possible group numbers the component can choose. Components which have picked the same group number belong to the same group. Table 5.10 shows two randomly generated group structure vectors of the car in Figure 5.6. The first example "1 2 1 4 5 6 7 7" represents that $c1$ and $c3$ are in the same group; $c7$ and $c8$ are in the same group; $c2$, $c4$, $c5$, $c6$ are in four different groups. It does not matter what the group number is, the group number only tells us which components are in the same group. For example, "3 4 5 5 7" could be a group structure vector of the car in Figure 5.7.

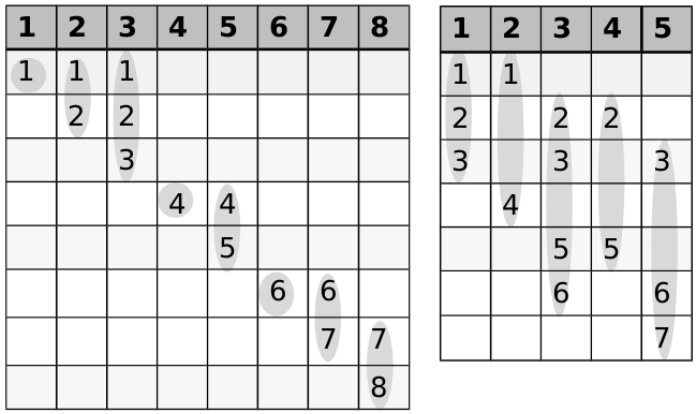


Figure 5.8: Alternative groups of components in Fig.5.6 and Fig.5.7.

Table 5.10: Examples of group structure.

Component	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Example 1	1	2	1	4	5	6	7	7
Example 2	1	1	1	4	4	6	7	8

2. Starting time vector A starting time vector can be randomly generated based on its group structure vector. Before picking the starting time for each group, the interval intersection of each group is calculated and the starting time is randomly picked from this intersection. Under the condition that a component is the sole member of a group, its starting time can be selected from its entire maintenance interval.

3. Workshop assignment vector A workshop is considered as "several workshops" based on its capacity (the number of teams). By this way, the schedule of each workshop team can be achieved from the solution. For example, consider that two workshops have three and four repairing teams respectively. Then, group operations can be randomly assigned to seven "workshops", the former three and the latter four represent corresponding teams in two workshops.

Genetic Operators

Based on the encoding, the corresponding genetic operators are designed, i.e., crossover and mutation operators. Crossover operators are applied separately to the three parts of the chromosomes.

For the group structure vector, one point or multi-point crossover can be used as crossover operator. Figure 5.9 is an example of one point crossover on the two group structure vectors in Table 5.10. Two new vectors are generated, and they are always valid because the group number of each component is from its own alternative group numbers. Because the whole group structure vector consists of all car components, one point crossover may not be efficient enough. Therefore, multi-point crossover can be applied.

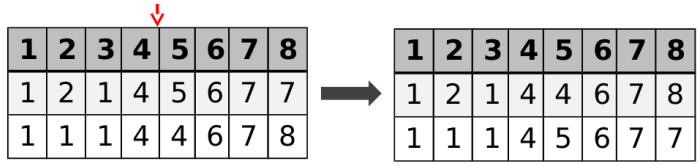


Figure 5.9: One point crossover.

The same cutting points are applied to the starting time vector when doing crossover. However, the change on the group structure vector caused by the crossover can result in the invalidity of the starting time vector because it is possible that the group members and intersection have changed due to the new group structure. Therefore, when performing the crossover on the starting time vector, the starting times of all group operations are checked and a new starting time is generated randomly from the new feasible intersection in the case that the starting time of a group is invalid.

A multi-point crossover can be applied to the workshop assignment vector as well. Every component is assigned a workshop team randomly. In the proposed algorithm, the workshop team of a group operation is decided by the first component in that group in case different workshop teams are assigned to components in the same group.

The mutation operators applied to three parts of the chromosome are also implemented separately. One or more gene values in the group structure vector can be exchanged by another alternative value from the same column in Figure 5.8 in order to generate a new individual. Again, the change applied to the group structure vector can result in the invalidity of the starting time vector. Hence, the starting time is checked for the corresponding groups after the mutation is done on the group structure vector; a valid starting time is generated randomly if it is invalid. Afterwards, some gene values can be altered in the starting time vector and workshop assignment vector to generate a new individual.

Chromosome Decoding

Decoding the chromosome is to convert an individual into a feasible schedule to calculate the objectives and constraints which represent the relative superiority of a chromosome. The group structure, the starting time and the workshop team of the operations can be obtained from each individual. Thereafter, the objective values can be calculated. When converting an individual to a schedule, it is possible that the processing times of two or more group operations assigned to the same workshop team are overlapping since the starting time of each group operation is fixed in the starting time vector. In this case, the principle of first-come-first-served is followed: the starting time and processing time of the earlier started group are not changed; the starting time of the later started group operation is not changed either; but the processing time of the later group operation is increased because the later group operation can be maintained only after the previous group operation is implemented. In the algorithm, the workshop abilities (components a workshop can maintain) are used as constraints to guarantee that a component will not be assigned to a workshop which cannot maintain it.

5.2.4 Experimental Results

The proposed evolutionary algorithm framework has been combined with four MOEAs: NSGA-III, SMS-EMOA, DI-MOEA (DI-1 and DI-2) to solve the real-world vehicle fleet maintenance schedule optimization problem, and their performance is presented in this section. Three application instances with different sizes are generated and their parameters are listed in Table 5.11. For example, the problem $P1$ includes 30 cars, each car consists of 4 components and 2 workshops are available; the two workshops have 3 and 4 repairing teams, and they can only maintain component o_1, o_2, o_3 and o_2, o_3, o_4 respectively. In order to make the results straightforward and better comparable, the processing time, cost as well as other variables are set to fixed values according to Table 5.9; the due dates and previous repair dates are generated randomly in 100 days (negative values are used for the previous repair dates); vehicle demands are randomly generated for each problem in a way that the demand never exceeds the total number of cars in the fleet.

All the experiments are performed with a population size of 100 and a budget of 500000 fitness evaluations. Both crossover and mutation probability are set to 1. For each problem, 30 optimization trials are performed with each algorithm. 10 cutting points are used in the multi-point crossover.

Table 5.11: Problem parameters.

Problem	#car	#comp	#ws	ws #car	ws #comp
$P1$	30	4	2	3	o_1, o_2, o_3
				4	o_2, o_3, o_4
$P2$	40	4	2	3	o_1, o_2, o_3
				4	o_2, o_3, o_4
$P3$	40	4	3	3	o_1, o_2, o_3
				4	o_2, o_3, o_4
				3	o_1, o_2, o_3, o_4

When analyzing the performance of different MOEAs, the empirical attainment function (EAF) is used to visualize the attained parts of the objective space. The 50% attainment surface shows that half of all Pareto optimal solutions will weakly dominate this surface and it is an estimator of what one would expect to achieve in 50% of runs [18]. Besides the average performance, the aggregate Pareto front approximation over 30 runs, i.e., the accumulated non-dominated solutions from 30 runs, has been used as another performance metric as well. Because extreme solutions are not preferable for the application problems and only one solution will be chosen to be deployed in workshops, the Pareto front approximation is zoomed in and the 50% attainment surface and the aggregate Pareto front approximation on the knee regions are plotted. Since relatively loose vehicle demands are considered currently, all the vehicle demands have been satisfied in all the solutions on the Pareto front approximations obtained from all algorithms. Therefore, the two-dimension plot can be shown to observe the results.

The performance of the algorithms has also been examined using the hypervolume indicator. Table 5.12 shows the aggregate and median hypervolume across 30 runs on three problems. For each instance, the upper row is the aggregate hypervolume, the middle row is the median hypervolume and the lower row is the standard deviation; the best hypervolume value has been highlighted in bold. When calculating the hypervolume indicator, the reference point is used by the maximum extent of the population plus an offset. It can be observed that SMS-EMOA, DI-2 and DI-1 performs best on $P1$, $P2$ and $P3$ respectively for the aggregate hypervolume; for the median hypervolume, SMS-EMOA performs best on $P1$, $P3$ and DI-1 performs best on $P2$.

For $P1$, if all the components are maintained exactly at their due dates, the total time and cost would be around 360 days and €36000, and vehicle demands cannot be guaranteed. Likewise, the total workload and cost would be around 480 days and €48000 for $P2$ and $P3$ when all the components are maintained exactly at their due

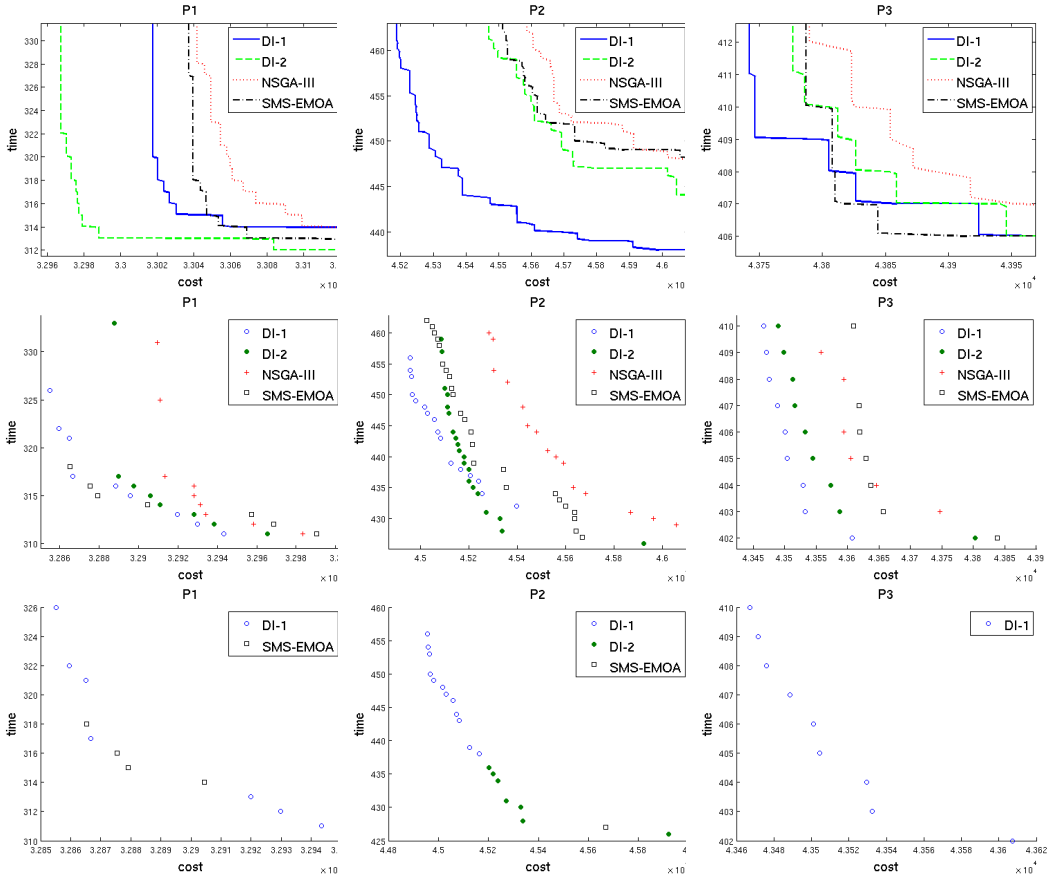


Figure 5.10: The 50% attainment surface (upper row), aggregate Pareto front approximation of each algorithm (middle row) and aggregate Pareto front approximation of all algorithms (lower row) of three instances on cost(days) & time(€).

Table 5.12: The aggregate hypervolume (Agg-HV) and median hypervolume (M-HV).

Algorithms / Problems		DI-1	DI-2	NSGA-III	SMS-EMOA
$P1$	Agg-HV	0.87983	0.91007	0.89932	**0.91633**
	M-HV	0.82728	0.83160	0.82506	**0.84137**
	std	0.0609	0.0506	0.0444	0.0563
$P2$	Agg-HV	0.89830	**0.98095**	0.93151	0.97677
	M-HV	**0.78950**	0.77907	0.72469	0.72647
	std	0.1649	0.1634	0.1259	0.1463
$P3$	Agg-HV	**0.88581**	0.88202	0.79875	0.87652
	M-HV	0.47185	0.51326	0.43115	**0.52816**
	std	0.2357	0.2244	0.2051	0.2424

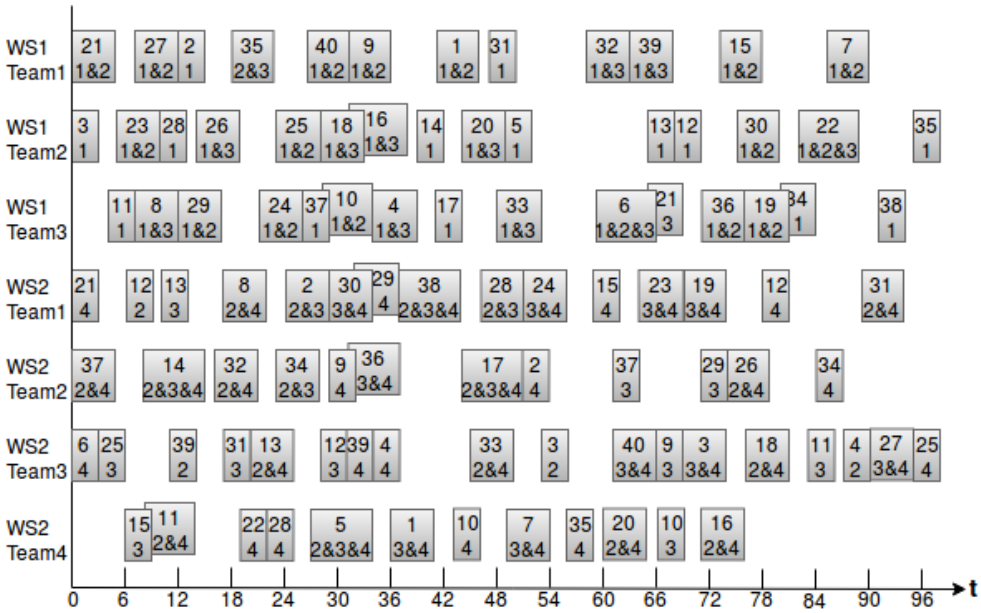


Figure 5.11: A schedule on the Pareto front approximation of DI-2 on $P2$.

dates. While, we can find many solutions with the workload less than 360 days and cost less than €36000 on the Pareto front approximations of all algorithms for $P1$. The same applies analogously for $P2$ and $P3$. Besides that the workload and cost of these solutions are already better than performing the operations on the due dates, all the vehicle demands have been satisfied at the same time.

Figure 5.11 presents a schedule on the Pareto front approximation of DI-2 on $P2$, the two objectives (cost and time) of this solution are €45338 and 428 days, the customer demands are totally satisfied. Each item is a group operation, the number above is the car number and the number below is the component number. We can see that some components are maintained individually and some components of a car are grouped together for one workshop visit. There is no maintenance of component $c4$ in workshop $ws1$ because this workshop has no ability to maintain component $c4$, and the same for component $c1$ on $ws2$. It can also be observed that some operations are overlapping to some extent. For example, the first item on $Team4$ of $ws2$ is an operation of $c3$ on car 15, its maintenance starts from day 6 and ends on day 9 because the total processing time plus set-up time is three days. However, before the completion of this task, car 11 (for the operation of $c2$ and $c4$) is sent to the same team and has to wait for one day for maintenance. The reason why car 11 waits in

the workshop instead of being sent to the workshop one day later is that the failure probability of $c2$ or $c4$ would increase if this car is in use for one more day. There is no penalty cost for the wait of the car in the workshop because it has been "punished" by the increased processing time.

5.2.5 Conclusion

The real-world problem of vehicle fleet maintenance scheduling optimization is formulated. The penalty function is defined in order to deal with uncertainties in the due dates and to prevent too early or too late maintenance. A problem-specific multi-objective evolutionary algorithm framework is designed based on the component grouping strategy. State-of-the-art MOEAs are incorporated with the algorithm framework to solve the application problem and their behavior is investigated. Although DI-MOEA is used for the first time for a real-world application problem, its performance is comparable with and for some instances even better than other popular MOEAs such as NSGA-III and SMS-EMOA. DI-MOEA, especially DI-1, is the best both for the average performance and the aggregate Pareto front approximation. For the hypervolume indicator, DI-MOEA and SMS-EMOA are the best on all three instances. According to the observation of solutions, it has been found that most components are maintained earlier than their due dates in the case that their maintenance times are shifted, the reason is that the penalty costs are too expensive if they are shifted forward. In order to increase the probability that the maintenance is shifted forward, the corresponding parameter can be adjusted.

The proposed algorithm framework can work for the generic application in various similar scenarios, for example, the aircraft maintenance, ship fleet maintenance, and so on. The problem formulation and the parameters can be flexibly adjusted based on the real application or by the decision maker. In the real-world applications, it is desirable to generate schedules that are robust within a reasonable range of disruptions and uncertainties such as machine breakdowns and processing time variability. Therefore, dynamic elements will also be taken into account in the next step.

Chapter 6

Preference-based and Dynamic Vehicle Fleet Maintenance Scheduling Optimization

The first version of the multi-objective vehicle fleet maintenance scheduling optimization problem has been formulated and solved by the proposed algorithms in the previous chapter. To make the problem more practical, rigorous and clear, after discussing with the DM from Honda Research Institute Europe GmbH, the problem is upgraded from the following aspects:

- There exists a lot of uncertainty when the predicted RUL of each component is used as its due date, because no matter how accurate the predictive model is, it is still possible that the component will break on other dates: before the due date or later. Therefore, instead of only using the predicted RUL, predicted RUL probability distribution should be used as the foundation to assign the maintenance time in scheduling optimization.
- The expected number of failures is adopted as an objective to reduce the chances that the vehicles are broken on the road.
- The teams in workshops don't need to be specified, each workshop can be treated as one team.

- The demand satisfaction is removed from objectives. It is not necessary to consider it for a more general problem.

To implement these changes, the MOVFMSO problem is reformulated in this section. Naturally, the corresponding MOEAs need to be modified to solve the newly formulated problem. These are all described in Section 6.1. Besides, AP-DI-MOEA (Automatic Preference based DI-MOEA) is also adopted in Section 6.1 to find solutions with a more fine-grained resolution in the automatically generated preference region.

To model the complete process of the vehicle fleet maintenance scheduling optimization, a VFMSO simulator is developed in Section 6.2. The VFMSO simulator starts from simulating driving tasks and available workshops for a vehicle fleet. The RULs of components are predicted when the vehicles execute the distributed driving tasks. Afterwards, the proposed MOEAs are applied to optimize the maintenance schedule, and the workshops can maintain the vehicles based on the optimal schedule. The process is running in a rolling-horizon fashion and a new maintenance schedule is generated periodically based on the newly predicted RULs. To do this, a fourth objective is added into the optimization, which is to minimize the changes between the new schedule and the previous schedule. Thus, the optimization algorithms are extended to dynamic MOEAs.

6.1 Preference-base MOEAs for MOVFMSO

This section starts with the new formulation of the multi-objective vehicle fleet maintenance scheduling optimization problem in Section 6.1.1. The tailored algorithm to solve the new optimization problem is described in Section 6.1.2. The performance of MOEAs and preference-based MOEAs on the problem are reported in Section 6.1.3. Lastly, Section 6.1.4 concludes the work and outlines directions for future work.

6.1.1 Problem Formulation

For a vehicle fleet running the driving tasks, the components of vehicles are getting damaged and should be maintained regularly. Some separate workshops are available for the maintenance of the car fleet, and the repair time and maintenance cost are known for each component in each workshop. Besides the time and cost for repairing the car component, a fixed set-up cost and set-up time are considered for each visit of a

car to a workshop, which correspond to the cost and time required for the preparation of the maintenance operation.

The updated VFMSO problem addressed in this section is defined as follows:

1. There are n cars $C = \{C_1, C_2, \cdots, C_n\}$ and m workshops $W = \{W_1, W_2, \cdots, W_m\}$.
2. Each car C_i comprises l_i components to be maintained for $i = 1, \cdots, n$.
3. For each component O_{ij} $(j = 1, \cdots, l_i)$, i.e., the jth component of car C_i, there is a set of workshops capable of repairing it. The set of workshops is represented by W_{ij} which is a subset of W.
4. The processing time for maintaining component O_{ij} in workshop W_k is predefined and denoted by p_{ijk}.
5. The cost for maintaining component O_{ij} in workshop W_k is predefined and denoted by q_{ijk}.
6. The set-up time of car C_i in workshop W_k is predefined and denoted by x_{ik}.
7. The set-up cost of car C_i in workshop W_k is predefined and denoted by y_{ik}.
8. The previous repair time of component O_{ij} is recorded and denoted by L_{ij}.

The constraint in this problem is that the maintenance periods of different operations for the same car should not overlap. It is obviously wrong if two overlapping maintenance operations of a car are assigned to different workshops because one car cannot be in two different workshops at the same time. If two overlapping maintenance operations of a car are assigned to the same workshop, it is not correct either because these two maintenance operations should be grouped together as one operation in this case.

Three objectives are taken into consideration, which are the total workload, total cost and expected number of failures. In a multi-objective optimization problem, the objectives typically are conflicting, i.e., achieving the optimal value for one objective requires some compromise on other objectives. In this problem, the fact that faster maintenance usually is more expensive leads to the conflict between the first two objectives. The expected number of failures counts the times when the vehicles are broken on the road. Here, the expected value is used because the actual value is unknown at the time of the optimization due to uncertainties in the predictions. When the expected number of failures is large, less maintenance tasks are performed, therefore, the workload and cost can drop.

Let T_k denote the sum of the times spent for all operations that are processed in workshop W_k; M_i the sum of all costs spent for all maintenance operations of car C_i; F_{ij} the number of failures of component O_{ij}. Three objectives can be defined as:

Minimize the total workload: $f_1 = \sum_{k=1}^{m} T_k$ (6.1)

Minimize the total cost: $f_2 = \sum_{i=1}^{n} M_i$ (6.2)

Minimize the expected number of failures:

$$f_3 = \sum_{i=1}^{n} \sum_{j=1}^{l_i} \mathbb{E}(F_{ij}). \tag{6.3}$$

6.1.2 Customized Algorithm

First the execution window is defined for each component based on its predicted RUL probability distribution which is assumed to be a normal distribution. The execution window suggests that the maintenance of the component can only start at a time spot inside the window. The mean (μ) and standard deviation (σ) of the predicted RUL probability distribution determine the interval of the execution window, which is defined as: $[\mu - 2 \times \sigma, \mu + 2 \times \sigma]$. The interval is chosen relatively long because 95% of the values are within two standard deviations of the mean, therefore, maintenance before or after the interval hardly makes sense.

After the determination of the execution window, the maintenance of several components can be combined to one visit if their execution windows overlap. Especially, by grouping the maintenance of multiple components into one maintenance operation, the set-up cost and set-up time are charged only once for the complete group of components. This part is the same as described in Section 5.2.3.

Within the execution window of a component, an arbitrary time can be chosen as the starting time for maintaining the component. However, the maintenance time of each component should be as close as possible to its real due date to save its useful time and avoid a car breakdown on the road. Therefore, Monte Carlo simulation is used to simulate the "real" due dates for each component. To be specific, 1000 samples of the due date are generated in the execution window of each component according to its predicted RUL probability distribution. Figure 6.1 shows an example of the execution window evolved from the predicted RUL probability distribution of a component. After 1000 sampled due dates are generated in the execution window, the

scheduled maintenance date of the component is compared with these samples one by one, and each comparison can lead to three situations. Let us use d_{ij}^v to denote the vth due date sample of component O_{ij}; and D_{ij} the scheduled maintenance date of component O_{ij}. Three possibilities after the comparison are:

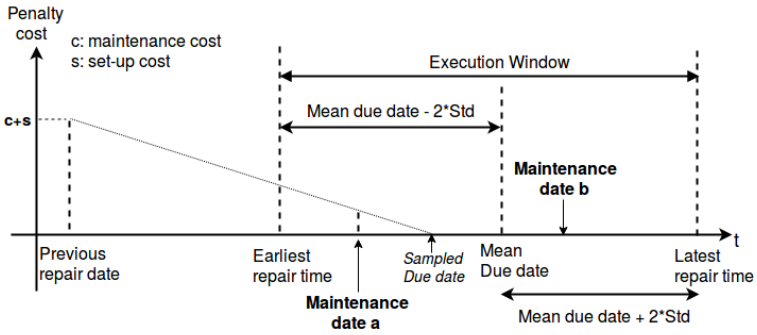


Figure 6.1: Execution window of a component.

Case 1 $D_{ij} < d_{ij}^v$

The scheduled maintenance date is earlier than the sample (or the "real" due date) means that the component will be maintained before it is broken. In this case, its useful life between the maintenance date and the due date will be wasted. Therefore, a corresponding penalty cost is imposed to reflect the waste. To calculate the penalty cost, a linear penalty function is suggested based on the following assumptions:

- if a component is maintained when it is new or the previous maintenance has just completed, the penalty cost would be the full cost of maintaining it, which is $c + s$: the maintenance cost of the component and the set-up cost of the car;
- if a component is maintained at exactly its due date, the penalty cost would be 0.

Assume d_{ij}^v is "Sampled Due date" in Figure 6.1, and D_{ij} is "Maintenance date a", in this case, D_{ij} is earlier than d_{ij}^v. The penalty cost of "Maintenance date a" for "Sampled Due date" would be the vertical dotted line above "Maintenance date a".

Case 2 $D_{ij} > d_{ij}^v$

The scheduled maintenance date is later than the sample means that the maintenance date is too late and the defect occurs on the use. Still, d_{ij}^v is "Sampled Due date"

in Figure 6.1, but the scheduled maintenance date D_{ij} is "Maintenance date b". In this case, D_{ij} is later than d_{ij}^v, and the vehicle will break down on the road. In the algorithm, the number of failures will be increased by one.

Case 3 $D_{ij} = d_{ij}^v$

The ideal situation is that the maintenance date is scheduled on the due date. The component can be maintained exactly at the date that the component is broken. In this case, there is no penalty or failure.

The averages of the penalty costs and the number of failures from 1000 due date samples will be used as the penalty cost and expected number of failures for the scheduled maintenance date of the component. For each operation (the single-component operation or group operation), its cost consists of three parts: the set-up cost of the car, the maintenance costs and the penalty costs of all components of the operation. The penalty cost of components is a part of the total cost, and the expected number of failures of components is the third objective to be minimized in the multi-objective optimization.

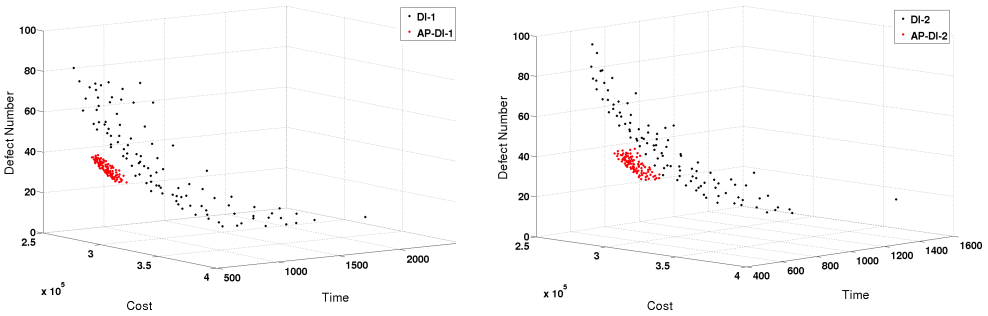
In Section 5.2.3, the implementation of tailored evolutionary algorithm for the first formulation of the MOVFMSO problem has been introduced, including how to represent an individual or solution in the population, how to take these chromosomes into a process of evolution, how to create variations of solutions in each iteration, etc. The algorithm can still be used on the updated problem. Next, AP-DI-MOEA (described in Section 4.2.2) is conducted on the updated MOVFMSO problems to demonstrate the performance.

6.1.3 Experimental Results

The performance of MOEAs and the preference-based MOEAs are compared on the VFMSO problems. The two variants of AP-DI-MOEA: AP-DI-1 and AP-DI-2, have been conducted on two instances with different sizes. On every problem, each algorithm runs 30 times with different seeds, while the same 30 different seeds are used for all algorithms. All the experiments are performed with a population size of 100. The budget of 1200000 evaluations has been used and 600000 of them are for the initial Pareto front; after that, the preference region is updated after every 50000 evaluations.

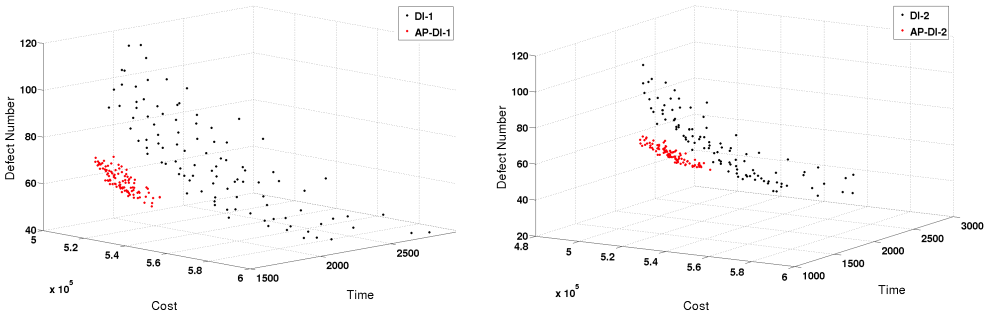
Figure 6.2 shows Pareto front approximations of a problem with 20 cars and 3 workshops (V1), and each car contains 13 components: one engine, four springs, four

brakes and four tires [109]. It can be observed that AP-DI-1 and AP-DI-2 can zoom in the entire Pareto front and find solutions in the preference region, at the same time, both AP-DI-1 and AP-DI-2 converge better than their corresponding DI-1 and DI-2. A similar conclusion can be drawn from Pareto fronts approximations of the problem with 30 cars and 5 workshops (V2) in Figure 6.3.



(a) DI-1 & AP-DI-1. **(b)** DI-2 & AP-DI-2.

Figure 6.2: Pareto front approximation on VFMSO problems with 20 cars and 3 workshops.

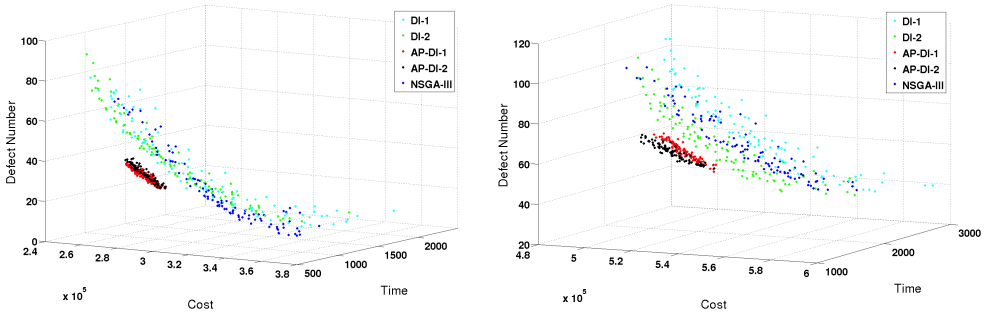


(a) DI-1 & AP-DI-1. **(b)** DI-2 & AP-DI-2.

Figure 6.3: Pareto front approximation on VFMSO problems with 30 cars and 5 workshops.

In Figure 6.4, the Pareto front approximations from DI-MOEA, AP-DI-MOEA and NSGA-III on V1 (left) and V2 (right) are put together. The behaviours of DI-1, DI-2 and NSGA-III are similar on V1, so are the behaviours of AP-DI-1 and AP-DI-2 on this problem. While DI-2 and AP-DI-2 converge better than DI-1 and AP-DI-1 on V2 problems. The behaviour of NSGA-III is between that of DI-1 and DI-2.

Table 6.1 gives the space and dominance relation of knee points from DI-MOEA



(a) V1. **(b)** V2.

Figure 6.4: Pareto front approximation on VFMSO problems by DI-MOEA, AP-DI-MOEA and NSGA-III.

and solutions from AP-DI-MOEA on these two VFMSO problems. For both problems, only few knee points from DI-MOEA are in the preference regions of AP-DI-MOEA, and the main reason is that the Pareto front of AP-DI-MOEA converges better than that of DI-MOEA, in some cases, the Pareto front of DI-MOEA cannot even reach the corresponding preference region. More importantly, it can be observed that most knee points from DI-MOEA, no matter whether in the preference region or outside of the preference region, are dominated by the solutions from AP-DI-MOEA. This phenomenon is even more obvious for the application problem with bigger size and run with the same budget as the smaller one: for V2, 90% of knee points from DI-MOEA are dominated by the solutions from AP-DI-MOEA.

Table 6.1: Space and dominance relation of knee point from DI-MOEA and AP solutions on V1 and V2.

Problem		V1		V2	
Algorithm		DI-1/ AP-DI-1	DI-2/ AP-DI-2	DI-1/ AP-DI-1	DI-2/ AP-DI-2
In preference region	Incomparable	0	0	0	0
	Dominated	9	7	9	6
	Dominating	0	0	0	0
Outside p-region	Incomparable	4	9	3	3
	Dominated	17	14	18	21

Table 6.2 gives the space and dominance relation of knee points from NSGA-III and AP solutions. For both problems, again, most knee points from NSGA-III are not in the preference regions of AP-DI-MOEA. Some knee points from NSGA-III are dominated by AP solutions and most of them are incomparable with AP solutions.

Table 6.2: Space and dominance relation of knee point from NSGA-III and AP solutions on V1 and V2.

Problem		V1		V2	
Algorithm		NSGA-III/ AP-DI-1	NSGA-III/ AP-DI-2	NSGA-III/ AP-DI-1	NSGA-III/ AP-DI-2
In preference region	Incomparable	0	0	0	1
	Dominated	0	1	3	2
	Dominating	0	0	1	1
Outside p-region	Incomparable	23	24	21	18
	Dominated	7	5	5	8

6.1.4 Conclusion

The multi-objective vehicle fleet maintenance scheduling optimization problems were updated after further discussion with the decision makers. In the new optimization problem, the maintenance time of each component was based on the predicted distribution of its remaining useful time. A new objective, i.e., the expected number of failures, was adopted to reduce the risk of car breakdown on the road.

The proposed MOEAs and preference-based MOEAs have been conducted on the updated MOVFMSO problems. The experimental results of AP-DI-MOEA on two application problem instances of different scales showed that AP-DI-MOEA can generate preference regions automatically and it (in both cases) found clearly better and more concentrated solution sets in the preference region than DI-MOEA. For completeness, it was also tested against NSGA-III and a better approximation in the preference region was observed by AP-DI-MOEA.

In the application of maintenance scheduling, it will also be important to integrate robustness and uncertainty in the problem definition. It is desirable to generate schedules that are robust within a reasonable range of disruptions and uncertainties such as machine breakdowns and processing time variability.

6.2 Dynamic MOEAs for MOVFMSO

Up to this point, the real-world application problem, i.e., the vehicle fleet maintenance scheduling optimization, has been formulated; the tailored multi-objective evolutionary algorithms have been developed; the basic MOEAs have been extended to the preference based MOEAs for the VFMSO problems. So far these proposed algorithms are used to solve the static problems. However, in the real-world scenario, after a maintenance schedule is released for execution, continuously updating the schedule is required due to the change of vehicle conditions and the ensuing changes in the

RUL predictions. The optimization of the maintenance schedule is an ongoing process running in a rolling-horizon fashion and it is therefore desirable to generate robust schedules.

According to the literature in robust scheduling methodologies, robustness is mainly grouped into quality robustness and solution robustness [57]. The quality robustness refers to the insensitivity of the scheduling performance such as makespan and total tardiness in the presence of uncertainty. The property that the start and the completion of each activity should be as close as possible to its previous schedule is known as the solution robustness and it is usually considered as a stability measurement of the schedule. When the proposed static algorithm is extended to a dynamic algorithm, a fourth objective is involved in the algorithm, which is the stability, i.e., the solution robustness.

To model the complete process of maintaining the vehicle fleet by way of scheduling optimization, a simulator is developed to observe the performance of dynamic MOEAs. The VFMSO simulator starts from simulating driving tasks and available workshops for a vehicle fleet, at the same time, in the simulator, the RUL of components can be predicted and used as the input information to optimize the maintenance schedule for the vehicle fleet. During the running of the simulator, the optimization process is running in a rolling-horizon fashion and the maintenance schedule is updated periodically. Accordingly, the optimization algorithm becomes a dynamic algorithm and a fourth objective is added into the dynamic MOEA, which is to minimize the changes between the new schedule and the previous schedule.

This section first introduces dynamic optimization for the VFMSO problems in Section 6.2.1. The RUL prediction is described in Section 6.2.2. Section 6.2.3 discusses the details of the simulator and Section 6.2.4 shows the experimental results. Finally, Section 6.2.5 briefly summarises the study, and proposes possible directions for future work.

6.2.1 Dynamic Optimization

By applying the proposed MOEA and preference-based MOEA, namely DI-MOEA and AP-DI-MOEA, after achieving a PF approximation, the knee point is picked as the final optimal schedule to be deployed in workshops. In the real-world application, the maintenance schedule needs to be updated periodically. To generate a new schedule for the next stage, the current schedule used for the vehicle fleet and workshops is also needed. Various disruptions may occur while running a maintenance schedule, for

example, the car is broken before its scheduled maintenance time, or new repairing tasks in workshops lead to the delay of the scheduled activities. In the face of various disruptions, adjustments in the schedule have to be made and this is also the reason that the maintenance schedule is updated periodically. A new schedule with the new arrangement of the maintenance activities is generated from the new condition of the vehicle fleet and workshops. However, the changes on the current schedule lead to additional costs such as the cost of reallocation of tools and equipment, the cost of reordering of raw materials, and etc. To reduce these costs, when updating the maintenance schedule, one important point is to maximize the similarity between the new schedule and the previous one to increase stability. For this purpose, the stability criterion is employed as one more objective in the dynamic algorithm. Let $d_{ij} = 1$ if the maintenance time or workshop of component O_{ij} in the previous optimal maintenance schedule is different with the assigned maintenance time or workshop of component O_{ij} in the currently optimized schedule, otherwise $d_{ij} = 0$. The stability of the schedule can be maximized by minimizing the difference in schedule.

$$\text{Minimize the schedule difference: } f_4 = \sum_{i=1}^{n} \sum_{j=1}^{l_i} d_{ij}. \tag{6.4}$$

The number of components which are assigned to different maintenance times or workshops from that in the current running schedule is minimized in the dynamic algorithm. Furthermore, since the stability of maintenance activities in the near future is more important than that of maintenance activities in the distant future, when calculating the stability objective, different weights are given to the components which are scheduled to be maintained within one week, within one month and beyond one month. The dynamic algorithm makes it possible that the maintenance schedule is optimized under different operational environments including dynamic and changing conditions. Most importantly, the dynamic algorithm updates the maintenance schedule based on the latest damage of components because the underlying predicted RUL of each component is based on the latest damage. In this way, the maintenance schedule becomes more accurate.

6.2.2 Remaining Useful Lifetime Prediction

Knowing the RUL is essential to establish an optimal maintenance schedule, and the RUL prediction provides the system residual life from its current condition and the past operation profile [106]. Commonly, approaches used in prognostics and predicting RUL

are classified into three types: physics-based approaches, data-driven approaches, and hybrid approaches [35]. In this work, the physics-based models are used to estimate the degradation and failures of four essential components of a vehicle, namely engine, brake pads, helical springs, and tires. The fatigue and wear mechanisms are established for these components.

Degradation of Helical Springs

Helical spring is the most common type of spring used in passenger cars. One of the the main mechanisms that reduces the lifetime of a helical spring is fatigue and it is often analyzed using the S-N curve which describes the relation between cyclic stress amplitude and number of cycles to failure. Figure 6.5 shows a typical S-N curve. The vertical axis shows the stress amplitude, whereas the horizontal axis indicates the corresponding number of cycles to failure at a given stress amplitude. A stress S is calculated from force F by the equation: $S = K\frac{8\times F\times D_{coil}}{\pi\times d_{wire}^3}$, where D_{coil} and d_{wire} are the diameter of the mean coil and the wire, respectively. $C = \frac{D_{coil}}{d_{wire}}$ is the spring index. $K = 1 + \frac{0.5}{C}$ is the so-called Wahl factor. According to the Paris-Erdogan's and Palmgren-Miner laws [97], the damage percentage of a spring can be formulated as: $d_s = \sum_{i=1}^{p} \frac{n_i}{N_i} \times 100\%$, where d_s is the total percentage of life consumed, p is the total number of the considered stress sources, n_i and N_i are the number of cycles with a stress amplitude and the corresponding number of cycles to failure at this stress with $i = 1, 2, ..., p$ from p sources. $\frac{n_i}{N_i}$ is the fractional damage received from the ith source. When $d_s \geq 100\%$, the spring's lifetime ends and a spring failure occurs.

Degradation of Brake Pads

A wear-out failure arises as a result of cumulative damage related to loads applied over an extended time. In the process of braking, due to friction between the surfaces of the friction couple, the zones of contacts are damaged after each braking event, resulting in worn-out material. The volume of the worn-out material of the ith braking event can be represented as: $\triangle V_{bi} = C_{brake} \times F_i \times \triangle d_i$, where C_{brake} is a constant and presents the brake pad quality, F_i and $\triangle d_i$ are the friction force and the relative displacement between the brake pad and the brake rotor of ith braking event, respectively. If V_{b0} is the maximum volume which the brake pad can reduce before a failure might occur, damage percentage of the brake pad (d_b) can be estimated by $d_b = \sum_{i=1}^{n} \frac{\triangle V_{bi}}{V_{b0}} \times 100\%$. The brake force is converted from the brake torque by dividing torque by the length of the level arm. For the values of parameters in the physical models, such as D_{coil},

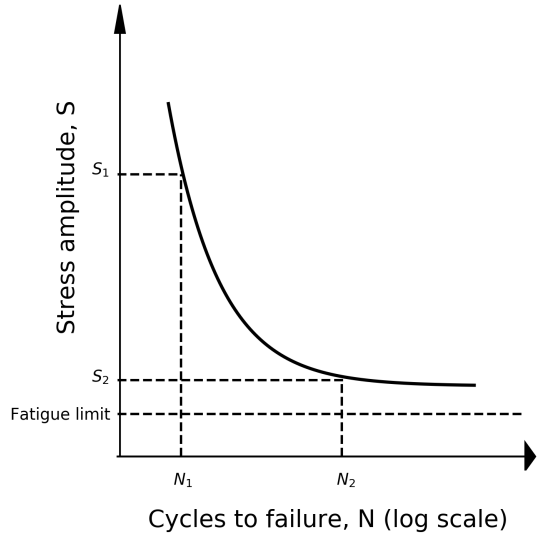


Figure 6.5: A typical S-N curve.

d_{wire}, C_{brake}, C_{tire}, C_{engine} (please refer to [109]).

Degradation of Tires

The wear mechanism is also applied to tires because tires' surfaces are in contact with the road surface and friction results in worn-out material of the tires. Two horizontal components of the force that cause the tire worn-out are F_x and F_y. The vertical force component F_z is only considered for pressure (overinflation, underinflation) damage of the tires. Similarly, a volume reduction of the tire due to worn-out material is formulated as: $\triangle V_{ti} = C_{tire} \times (|F_x| + |F_y|) \times \triangle d_i$, where C_{tire} is a constant and represents the tire quality. $\triangle d_i$ is the relative displacement between the tire surface and the road surface and it is simply the car travel distance. Again, the damage percentage of the tire (d_t) can be computed by: $d_t = \sum_{i=1}^{n} \frac{\triangle V_{ti}}{V_{t0}} \times 100\%$, where V_{t0} is the maximum volume which the tire can reduce before a failure might occur.

Degradation of Car Engine

A rough model is established to estimate the consumption lifetime of the car engine from the travel distance and the engine rotation speed. The equation is $d_{ei} = C_{engine} \times \triangle d_i \times R_i$, where C_{engine} is a constant and represents the engine quality. Here $\triangle d_i$

and R_i are the car travel interval, and the engine rotation speed corresponding to this travel interval, respectively. The consumed lifetime percentage of the engine is $d_e = \sum_{i=1}^{n} d_{ei} \times 100\%$. The engine needs to be maintained if the d_{ei} sum up to 1.

RUL Calculation

It is assumed that the physical models are accurate, therefore, the real damage on components up to now can be diagnosed. The RUL is predicted by extrapolating the future damage from the distribution of the damage so far. The RUL of a component can be calculated based on a damage percentage. If the RUL is estimated by a unit of week, the total damage percentage after the wth week is calculated by: $D = \sum_{i=1}^{w} D_i$, where D_i is the sum damage percentage of the ith week. Thus, the RUL after week w can be estimated by: $RUL = \frac{100\%-D}{D/w}$, here, 100% means that, at the beginning, the component is absolutely new. A Gaussian distribution is fitted to the distribution of the weekly damage percentage and the resulting standard deviation σ is used to calculate the lower and upper bound of the standard deviation confidence interval of RUL as following: $RUL_{-} = \frac{100\%-D}{D/w+\sigma}$ and $RUL_{+} = \frac{100\%-D}{D/w-\sigma}$.

6.2.3 VFMSO Simulator

A simulator has been developed to implement and evaluate the complete process of vehicle fleet maintenance scheduling optimization. In the VFMSO simulator, CarMaker[1] is adopted to simulate driving scenarios for a taxi fleet in New York City. The origin and destination coordinates from Green Taxi Company in January 2015 downloaded from NYC Open Data[2] are converted into taxi routes using Google API and are used as the driving tasks. In the CarMaker simulation, extra loads are added to all passenger seats of the car. For each passenger seat a load between 0 and 100 kg is randomly chosen with an equal probability. 4000 trips have been simulated with CarMaker. In the VFMSO simulator, each car is assigned to 40 random trips per day on average, and the maximum number of trips each car can execute per day is 50. These trips are randomly selected from the 4000 simulated trips. The sensor data of forces, brake torque and engine rotation speed yielded by CarMaker are used to estimate the damage percentage and the RUL of springs, tires, brake pads and engine

[1] CarMaker simulation is developed by IPG Automotive for testing driving scenarios of passenger cars and light-duty vehicles. It provides models for vehicles, roads, drivers and traffic for all simulation tasks in realistic driving scenarios. https://ipg-automotive.com/products-services/simulation-software/carmaker/#driver

[2] https://data.cityofnewyork.us/Transportation/2015-Green-Taxi-Trip-Data/gi8d-wdg5/data

by the physical models as described in Section 6.2.2.

Some parameters can be pre-defined to determine the vehicle fleet maintenance scheduling optimization problem before running the simulator: the size of the vehicle fleet, including the number of cars and the number of workshops; the costs and times of maintaining cars/components in workshops; the range of days of running the simulator; the frequency of generating a new maintenance schedule. After running the simulator for the defined period, the following items can be reported by the simulator.

- The number of defects: when a defect occurs, i.e., a component is broken before the scheduled maintenance date, the number of defects increases by one.
- The total cost: besides the set-up cost and maintenance cost, the waste of component lifetime has also been transferred to a cost, and has been included in the total cost by the simulator, this cost is called "too-early maintenance cost". Unlike the penalty cost in the optimization algorithm, the "too-early maintenance cost" in the simulator is the actual value because it is assumed that the physical models are 100% accurate and the due dates of the components calculated by them are used as the ground truth. When a component is maintained based on the maintenance schedule, the simulator can calculate its current damage percentage by the corresponding physical model and the remaining damage percentage is converted to a cost to reflect the waste of the useful lifetime. The "too-early maintenance cost" is calculated by the formula: remaining damage percentage $\times$ maintenance cost of the component. Obviously, no "too-early maintenance cost" arises for components which break before maintenance.
- The total maintenance time: the simulator records all the days that the vehicles cannot work, either the reason is a scheduled maintenance activity or a defect.
- The number of changed schedules: every time when the maintenance schedule is updated, the number of components which have a different maintenance date or workshop is recorded.
- The number of unsatisfied trips: when a car cannot execute its tasks, e.g., it is being maintained in a workshop, the assigned tasks for this car will be distributed to other available cars, but the maximum number of tasks a car can execute each day is 50. The tasks which cannot be satisfied are counted as unsatisfied trips.
- The number of scheduled maintenance activities: when a maintenance activity is executed based on the maintenance schedule, the number of scheduled maintenance activities increases by one.

These items are the final results after running the simulator for a pre-defined number of simulated days. It can be seen that, on the one hand, the simulator can show the results from different perspectives, which include not only the accumulated optimization objective values over the period of running it, but also the results that cannot be known by the optimization algorithm, such as the number of defects, the number of unsatisfied trips. On the other hand, these results are used for the final evaluation which is based on the "real" results and not on the raw optimization results. The raw optimization results cannot be used as actual results due to the reason that the optimizer does not have full knowledge of the future.

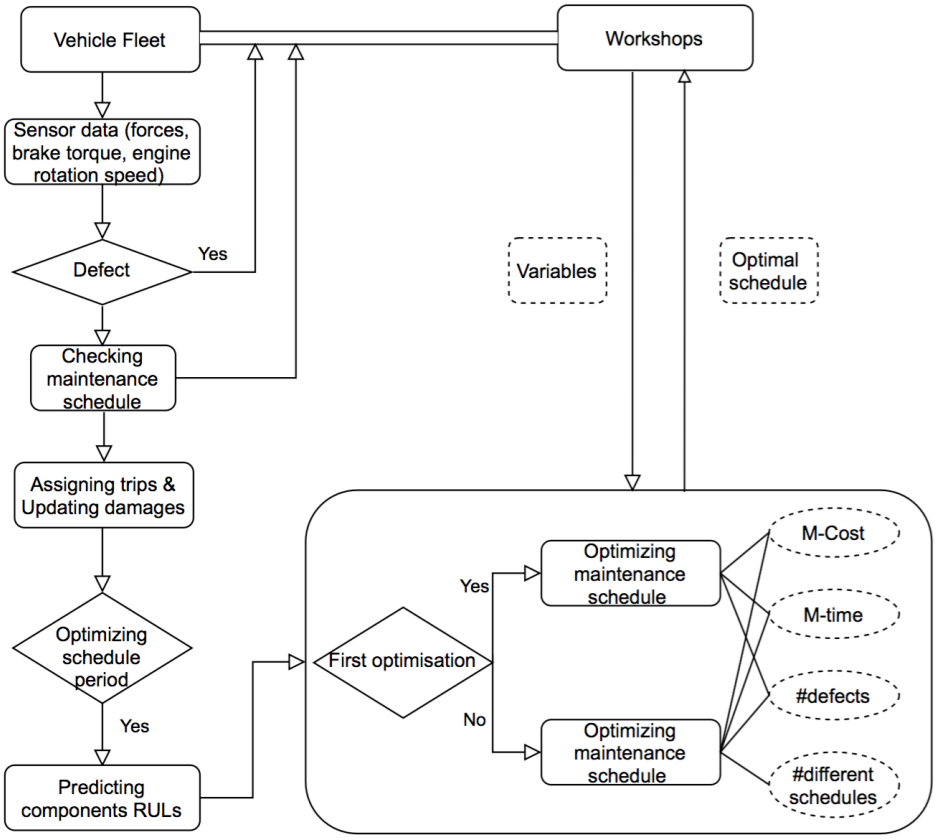


Figure 6.6: Daily workflow of the simulator.

Figure 6.6 shows the workflow of the simulator. The flow is executed on a daily basis. At the beginning of each day, vehicles in workshops are checked and sent back to work when their maintenance is done, meaning the damage of these components

is set to zero. Next, the damage of each component is investigated and vehicles are sent to workshops when defects occur, which means the damage percentage of a component reaches 100%. In the case of a defect, the car is sent to a random workshop. Afterwards, the maintenance schedule is checked and the vehicles are sent to the assigned workshops if they are assigned to be maintained on that day. Hereafter, the driving trips of that day are assigned to the available cars and the damages of components are updated. Lastly, when it is the day to generate a new maintenance schedule, the RUL distributions of components are predicted, and the maintenance schedule is optimized. In the case of generating the first maintenance schedule, only three objectives are employed. Later on, the stability of the schedule is involved in the optimization procedure as an extra objective. After obtaining the PF approximation from each optimization, the knee point on the PF is picked and deployed as the new schedule to replace the current schedule to maintain the vehicle fleet.

6.2.4 Experiments

To show and observe the impact of different maintenance strategies clearly, the simulator runs under the scenarios with the following combinations of parameters:

- the simulation time: 700 days,
- the size of the vehicle fleet: 20 cars with 2 workshops, 20 cars with 5 workshops,
- the frequency of updating schedule: weekly, monthly,
- the computing budget of optimization: 100000, 500000,
- the optimization algorithm: basic MOEA, preference based MOEA, dynamic basic MOEA, dynamic preference based MOEA.

The results of the prediction-based optimization algorithms are also compared with fixed-interval maintenance scheduling. To set the fixed-interval maintenance, firstly the simulator is run without the maintenance schedule. In this case, each component breaks until its due date or its damage reaches 100%, then it is maintained and sent back to perform the driving tasks again. The average mileages are obtained for 13 components to be maintained, which include engine, 4 brake pads (front left, front right, rail left and rail right respectively), 4 tires and 4 springs. They are used as the condition for the maintenance in the fixed-interval maintenance scheduling approach, i.e., if a component reaches its corresponding average mileage, it is sent for maintenance.

Tables 6.3 - 6.5 show the results from the simulator. In these tables, the first column shows which algorithm has been applied in the maintenance period (700 days in the experiments). To optimize the maintenance schedule, four different optimization algorithms have been applied and compared. Basic and preference-based algorithms only take into account three objectives: cost, time and the number of failures. Dynamic basic and dynamic preference-based algorithms involve the fourth objective (the stability of the schedule). It means that basic and preference based algorithms handle multi-objective optimization problems, and dynamic basic and dynamic preference based algorithms deal with many-objective optimization problems [84]. Many-objective optimization focuses on solving optimization problems with four or more objectives and it forms a special and important case of multi-objective optimization problems. Solving many-objective optimization problem is more challenging for MOEAs due to the high computational cost resulting from increased evaluation of the number of points required for the PF approximation.

The other columns in these tables give the final results according to the simulation. These results include the number of failures (#defects), the total cost (cost), the total maintenance time (time), the number of changed schedules (#ch-sch), the number of unsatisfied trips (#un-trips) and the number of scheduled maintenance activities (#sch-act). Since the maintenance schedule is based on the average mileage and is not updated for the fixed-interval maintenance, the number of changed schedules is not applicable in this case. The parameter setting for each scenario has also been given in the table, for example, "schedule-update: monthly; #evaluations: 100000" refers to the scenario when the maintenance schedule is updated monthly and the computing budget of the optimization algorithm is 100000. All experimental data are the average results from five runs, in each run a different seed for the simulation is used.

Table 6.3 shows the experimental results from two different scales of the problem: one is 20 cars and 2 workshops; another is 20 cars and 5 workshops. When there are more workshops, the maintenance time can get reduced because there is less chance for vehicles to wait for their maintenance. This results in a decrease of the number of unsatisfied trips because the waiting time in workshops is now used to execute trips. Accordingly, the number of maintenance jobs (both the number of scheduled maintenance activities and the number of defects) increases. So does the maintenance cost. When comparing the results from these two problems, it can be seen that the data match this logic.

When comparing the results of dynamic algorithms with four objectives and their corresponding algorithms without the fourth objective, it can be seen that dynamic

algorithms can always reduce the number of changed schedules, but this also means they have to sacrifice the other objectives to some extent. In some industrial scenarios, the stability objective plays a critical role. For example, in the case of aircraft maintenance, some maintenance activities are conducted during the intervals between takeoffs and landings, the change on the maintenance schedule may make an impact on the schedule of this flight and also might disrupt other flights, a rescheduling typically causes significant communication costs.

Next, with more computing budget for the optimization algorithms (i.e., the number of objective function evaluations is 500000.), it can be seen that the overall results after running the simulator get improved for three objective optimization (i.e., for basic and preference based algorithms.). The results here refer to the objectives that the algorithms optimize. However, for the dynamic algorithm, the results with more computing budget are sometimes mutually dominated with the results from using a smaller computing budget. For example, the number of defects can be reduced with the larger computing budget, but the total maintenance cost cannot get improved by more computing budget. This is led by the complexity of many-objective optimization. When determining the schedule to be deployed from the PF, the knee point is chosen. However, in four dimensional space, a small variation can lead to a big impact on the final result, especially on the accumulated results of multiple optimizations. With monthly schedule updates, the optimization algorithm is executed 22 times during one simulation run of 700 days.

When the schedule is updated more often, i.e., weekly, a reduction of the defect number is observed. Apparently, updating the maintenance schedule more frequently can promote the accuracy of it because the predicted RUL is more accurate. At the same time, an improvement of the total cost can be seen. The reason for the reduction of the total cost also comes from the accuracy of the schedule and the resulting decrease of the penalty cost which arises when the vehicle is maintained before it is broken, i.e., the cost for too-early maintenance. When updating the maintenance schedule more often, the maintenance time can not always get improved because the number of maintenance tasks does not always get decreased, the maintenance tasks may increase due to the accuracy of the schedule and the resulting increase on the number driving tasks which have been executed.

When comparing the preference based algorithm and basic algorithm, for both three objective and four objective optimization, it can be seen that the results of the preference based algorithm are usually better than its corresponding basic algorithm for the scenario of five workshops. However, if there are only two workshops, the

waiting of vehicles for their maintenance results in a performance degradation of the preference based algorithms. The similar working tasks of the vehicles lead to the phenomenon that the scheduled maintenance times for some vehicles are close, and this leads to that the workshops run out of capacity sometimes but become idle at other times. Therefore, a good solution would be to offer more workshops for the fleet, at the same time, these workshops can also work for other tasks besides for the fleet.

Lastly, when comparing with the fixed-interval maintenance, there are more defects, maintenance time and unsatisfied trips for the fixed-interval maintenance. Since most maintenance tasks are caused by defects, the too-early maintenance cost drops dramatically and this leads to the decrease of the total cost.

Table 6.3: Optimization results of different maintenance scenarios over 5 runs.

20 cars & 2 workshops						
Algorithm	#defects	cost	time	#ch-sch	#un-trips	#sch-act
Fixed-interval	226	474965	6269	NA	212450	52
	schedule-update: monthly; #evaluations: 100000;					
Basic	46	680666	4282	4509	121000	148
Preference	50	690871	4179	4630	112150	150
Dynamic basic	73	676149	5510	4023	175800	154
Dynamic preference	66	688934	4732	3729	137200	159
	schedule-update: monthly; #evaluations: 500000;					
Basic	39	675374	3936	4553	101750	150
Preference	40	677331	3903	4526	101950	145
Dynamic basic	68	717046	5262	3777	161300	157
Dynamic preference	42	690131	4669	3240	140750	150
	schedule-update: weekly; #evaluations: 100000;					
Basic	32	624078	4565	22884	126200	166
Preference	35	646117	4103	23016	109700	168
Dynamic basic	67	633854	5758	19660	185450	150
Dynamic preference	50	628228	4626	18049	140200	161
20 cars & 5 workshops						
Algorithm	#defects	cost	time	#ch-sch	#un-trips	#sch-act
Fixed-interval	330	747104	2996	NA	72750	92
	schedule-update: monthly; #evaluations: 100000;					
Basic	68	785777	1852	4877	27950	192
Preference	67	748044	1837	5012	25750	184
Dynamic basic	137	849203	2942	4466	62700	218
Dynamic preference	93	789592	2331	4247	42000	217
	schedule-update: monthly; #evaluations: 500000;					
Basic	55	756278	1725	4901	24850	182
Preference	50	718176	1649	4924	22550	177
Dynamic basic	125	831775	2754	4442	56950	223
Dynamic preference	91	797258	2130	4014	34750	210
	schedule-update: weekly; #evaluations: 100000;					
Basic	60	695181	1995	23973	35550	206
Preference	56	690720	1951	23982	31250	205
Dynamic basic	114	768697	3122	21715	77950	217
Dynamic preference	91	721193	2296	20397	44100	227

Besides the parameters which change the experimental environment, the variables in the optimization algorithm can also be adjusted to emphasize some aspects of the results. To reduce the number of failures of vehicles, the interval of the execution window is switched from $[\mu - 2 \times \sigma, \mu + 2 \times \sigma]$ to $[\mu - 3 \times \sigma, \mu + \sigma]$. Table 6.4 shows the results of 20 vehicles and 5 workshops. After shifting the execution window forward, the dramatic drop of the number of defects is achieved and the descent rate reaches 83.21% on average. Simultaneously, this activates the rise of the maintenance cost.

Table 6.4: Adjust execution window to reduce the number of defects.

20 cars & 5 workshops						
Algorithm	#defects	cost	time	#ch-sch	#un-trips	#sch-act
	schedule-update: monthly; #evaluations: 100000;					
Basic	7	886747	1872	4787	22050	227
Preference	10	840907	1834	4767	21450	223
Dynamic basic	28	1152867	3276	4143	63925	303
Dynamic preference	20	1032917	2399	4102	34350	285
	schedule-update: monthly; #evaluations: 500000;					
Basic	5	823662	1599	4826	15050	201
Preference	5	822518	1534	4804	12850	190
Dynamic basic	27	1133032	3618	3960	79100	292
Dynamic preference	22	979069	2339	3686	34950	258
	schedule-update: weekly; #evaluations: 100000;					
Basic	4	823683	1945	23496	25750	248
Preference	4	805841	1815	23472	22100	225
Dynamic basic	22	1070067	3092	20856	62450	287
Dynamic preference	15	864731	2313	19701	42350	231

It is worth noting that the problems with 20 vehicles and 13 components for each vehicle are already large scale scheduling optimization problems in terms of the domain of flexible job shop scheduling optimization. Moreover, the MOVFMSO problem is more complex than FJSS because the MOVFMSO problem needs to assign not only the workshops and maintenance times (sequences) for the maintenance activities, but also the combination of components for each activity. To investigate how scalable the proposed approach is, the questions asked are whether the algorithms can be applied to even larger fleet and whether consistent results can be achieved when the fleet becomes significantly larger. To this end, the fleet size has been increased to 50 vehicles and 15 workshops are available, the components to be maintained retain the same. Table 6.5 shows the simulator results and it can be observed that these results are consistent with the results presented earlier.

From the experimental results, some major insights on how to design schedules with respect to the objectives can be concluded as follows.

- Providing additional workshops can help reduce the overall maintenance time.

Table 6.5: Increase Problem Size.

	50 cars & 15 workshops					
Algorithm	#defects	cost	time	#ch-sch	#un-trips	#sch-act
Fixed-interval	864	1878632	7118	NA	162850	209
	schedule-update: monthly; #evaluations: 100000;					
Basic	45	2429887	5440	11933	63600	697
Preference	43	2395079	5104	12171	54600	669
Dynamic basic	109	2887019	7571	11598	127200	812
Dynamic preference	71	2751571	6481	11302	89300	792
	schedule-update: monthly; #evaluations: 500000;					
Basic	35	2320261	5016	12036	63600	635
Preference	32	2254907	4551	12176	42950	626
Dynamic basic	105	2990837	8454	11210	164700	853
Dynamic preference	70	2677516	6392	10971	87850	753
	schedule-update: weekly; #evaluations: 100000;					
Basic	24	2149010	5267	58736	70350	659
Preference	26	2141242	5047	58929	69100	643
Dynamic basic	102	2750433	6908	56272	107500	782
Dynamic preference	61	2418850	5819	55485	83400	706

- Moving the execution window to the left (earlier time) or updating the schedule more often can both be used to reduce the number of defects.

- Without introducing stability as an additional objective, schedule tends to be disrupted by dynamic updates.

- Both the use of the preference based algorithms and increasing the computing budget have a positive impact on the overall quality. However, the best way to improve the overall quality of the final results is to increase the number or capacity of workshops in combination with the preference based algorithms.

- Comparing the fixed-interval maintenance vs. prediction-based scheduling optimization, it can be concluded that fixed-interval maintenance leads to an unsatisfactory performance in terms of number of defects, whereas prediction-based scheduling optimization finds a balanced trade-off satisfying all objectives to high extents. Therefore, the extra computational effort required to make predictions and perform optimizations is well justified.

- The results on the large-scale benchmark problem with 50 vehicles indicate that the proposed algorithms can also handle larger problems and the main conclusions, as summarized in the previous points, remain the same.

6.2.5 Summary and Outlook

Since optimization algorithms are required to regularly update maintenance schedule in a dynamic environment, the proposed multi-objective evolutionary algorithms are extended to dynamic many-objective evolutionary algorithms that take stability as the fourth objective to aim for the robustness of maintenance schedule. The vehicle fleet maintenance scheduling optimization simulator has been developed, which can be used as a scalable benchmark for optimizing vehicle fleet maintenance schedules in an industrially relevant setting. The simulator and benchmark problems have been inspired by the instances faced by a taxi company with up to 50 cars. The proposed MOEAs can be compared and tested easily in the simulator in a rolling-horizon fashion. Parameters and algorithms can be adjusted to imitate various scenarios. Therefore, although the implementation of the approach is demonstrated in the example of taxi fleets, the proposed approach can be adapted to different industrial applications, for example, the maintenance of trucks, vessels, aircraft, etc.

The size of problems in the experiments is up to 50 vehicles and 13 components for each vehicle. Still, one might imagine the problems of even larger scale, and finding out the limit of the fleet size that the algorithm can handle would be an interesting future research. However, for this, high performance computing environments and parallel computing might be required, especially when it comes to statistical studies. In this work, to maintain clarity of presentation the dynamically changing element is so-far restricted, but in the future work additional dynamic elements and uncertainties should be considered. For example, the uncertainty on the maintenance duration could be modeled, as in [52], the presence of cost uncertainty in [31], etc.

Chapter 7

Conclusions

7.1 Summary

This thesis investigated several important aspects of multi-objective optimization and its application in the domain of scheduling optimization. In particular the thesis studied flexible job shop scheduling and dynamic prediction-based maintenance scheduling. Moreover, improvements in multi-objective optimization algorithms were proposed, such as using diversity indicators in selection; cone orders for faster convergence in many-objective optimization.

Next, the achievements and new insights discussed in each chapter will be briefly summarized.

Chapter 2 covered the basics of multi-objective optimization and provided an in-depth discussion of different ways to define dominance orders, including Pareto dominance, trade-off based dominance, angle dominance, etc. At the end of this chapter the formal relationships between these orders were highlighted and analysed. In particular it was clarified that orders defined by trade-off constraints and coefficients can be mapped to orders defined by cones via angles (or cone-generating vectors), and vice versa. This offers alternative ways for the decision maker or algorithm designer to define orders that are formally equivalent in his/her preferred way, which could be, for example, by means of trade-off bounds or by means of angles and/or cone-generating vectors.

Chapter 3 answered two research questions. The first question is:

RQ1 How can a multi-objective evolutionary algorithm (MOEA) be developed that generates uniformly distributed sets on the Pareto front regardless of the shape of the Pareto front?

To achieve a uniformly distributed Pareto front approximation regardless of the shape of the Pareto front and avoid complex algorithm structure and parameter setting, a diversity-indicator based multi-objective evolutionary algorithm (DI-MOEA) was proposed. DI-MOEA was tested on the state-of-the-art benchmark problems and showed excellent performance in terms of uniformity and convergence for Pareto fronts with convex, concave and disconnected shapes. The results also demonstrated that it is possible to use a diversity indicator (like the geometric mean gap in the proposed algorithm) that is not Pareto compliant, given that the selection gives priority to the non-dominated solutions and the diversity indicator is only used to select the most diverse subset among the non-dominated solutions. It is important to note, that a diversity indicator is here defined on a set as a whole and not on a single point, as it is the case for the crowding distance.

The complexities posed by many real-world optimization problems significantly worsen the performance of optimization algorithms, especially with the increase in the number of objectives. The following research question was therefore of our interest:

RQ2 How can the performance of MOEAs be improved generally?

Through analysis of Pareto dominance, it has been shown that its performance can be enhanced by utilizing geometric features of the Pareto order cone. Based on this technique, a new dominance relation, the edge-rotated cone order, was proposed to improve the performance of MOEAs. The edge-rotated cone order was implemented by rotating the edges of the standard Pareto order cone towards the outside. The edge-rotated cone order was integrated with several state-of-the-art MOEAs, and its ability to improve the performance of MOEAs has been tested on multi-objective optimization test problems. Especially on many-objective optimization problems the edge-rotated cone order led to even better performance.

Chapters 4 investigated the preference based multi-objective evolutionary algorithms in order to answer the third research question:

RQ3 Instead of the whole Pareto front, how can preferred solutions which are of real interest to the decision maker be obtained?

From the algorithm perspective, the majority of real-world multi-objective optimization problems suffer from having a large objective space and extent of the Pareto front, which result in a time consuming optimization process. Only focusing on the region of interest can reduce the objective space and speed up the convergence of the optimization algorithms.

Two preference based multi-objective evolutionary algorithms were presented in Chapter 4. The first algorithm is an a priori method. The DMs can provide a point, a region, multiple points or multiple regions as their preferences, the proposed algorithm can find solutions around the reference point(s) or in the reference region(s). Considering that it is difficult for the DMs to set preferences, the second algorithm is an automatic preference based algorithm which can generate the preference region automatically after identifying a knee point, narrow down the feasible objective space progressively, and eventually obtain the preferred solutions. The underlying idea is that in many cases the knee point provides a good compromised solution where all objectives achieve a good value. In particular this holds for connected and convex Pareto fronts, whereas for other shapes of Pareto fronts a default choice for a good solution on a Pareto front is more difficult to justify.

Chapter 5 discussed the design of novel algorithms for scheduling optimization problems. Our real-world application problem was a multi-objective vehicle fleet maintenance scheduling optimization problem (MOVFMSOP). Before solving this specific problem, one more common scheduling optimization problem in computer science and operations research, i.e., the multi-objective flexible job shop scheduling problem (MOFJSP) was investigated to answer the fourth research question:

RQ4 How to solve multi-objective flexible job shop scheduling problems?

A novel multi-objective evolutionary algorithm was developed for the MOFJSP. Multiple initialization approaches, various crossover operators and local searches were used in the algorithm. In particular, an algorithm configurator was adopted to tune the parameter setting in the algorithm. Based on the study of the MOFJSP, the MOVFMSOP was formulated mathematically and the algorithm was proposed in this chapter to answer the fifth research question:

RQ5 How to represent and solve a real-world scheduling optimization problem in the EA world?

To solve this problem, a three-vector chromosome was designed in the algorithm. The grouping strategy was proposed to determine the feasible grouping structure for

the grouping vector based on the predicted remaining useful time of components. At the same time, the starting time vector gave maintenance times for group activities, and the workshop vector decided which workshop performs each maintenance activity. Based on the encoding, the corresponding crossover and mutation operators were designed and applied separately to the three parts of the chromosome. The problem-specific multi-objective evolutionary algorithm was incorporated with several state-of-the-art MOEAs to investigate their performance on the MOVFMSOP.

Chapter 6 applied and compared the proposed MOEAs and preference based MOEAs on the vehicle fleet maintenance scheduling optimization problems to answer the sixth research question:

> *RQ6 How to apply and adapt the developed algorithms to the dynamic prediction-based maintenance scheduling optimization problem?*

In this chapter, the complete process of maintaining a vehicle fleet was simulated, including the simulation of the vehicle fleet and the prediction of remaining useful life for the fleet. The proposed MOEAs and preference based MOEAs were used in two ways: static and dynamic algorithms. Different updating frequency, computing budget, different scale of problems were adopted to show their performance.

In summary, following the discussion of issues on multi-objective optimization, this thesis proposed the multi-objective evolutionary algorithm, preference-based multi-objective algorithm, and the method to improve the general multi-objective optimization algorithms. Then the real-world vehicle fleet maintenance optimization problem was formulated as a scalable benchmark in an industrially relevant setting. The proposed algorithms were practically used on the application instances. To be able to optimize the maintenance schedule in a dynamic environment, the algorithms were further extended to dynamic algorithms and an additional objective, i.e., stability, was taken into consideration to retain stability of the schedule.

7.2 Future Work

Many potential lines of research are worthy of attention based on the achieved progress in the context of this thesis.

DI-MOEA uses the Euclidean distance based geometric mean gap as the diversity indicator, which is described in Chapter 3. The obvious feature of DI-MOEA is the uniformity of the achieved solution set. Besides the geometric mean gap, other gap

indicators, such as minimal gap, arithmetic mean gap [37] or the Riesz S-Energy [39], can also be used as the quality indicator. Furthermore, other distances, such as Chebyshev distance, crowding distance, mean ideal distance can replace the Euclidean distance to measure the distance between solutions in the search process. The features of the solution sets based on different distances and gaps can be compared. By doing so, the different strategies can be chosen when solving application problems with different features.

Regarding the edge-rotated cone, the mechanism that relates the properties of the problem with the rotation angle should be researched. Further research on its ability on articulating the preference on multi-objective optimization should be done. Especially on many-objective optimization, the angle setting of the cone order may provide a practical way to guide decision makers in choosing their preferences.

The knee point is defined as the point having the shortest distance to the line (the number of objectives is 2), plane (the number of objectives is 3) or hyperplane (the number of objectives is larger than 3) formed by the extreme points on the Pareto front in Chapter 4. However, it can also be defined in other ways, for example, the point having shortest distance to the Utopia point. Variants of methods to generate the knee point can be compared. By doing so, it is also possible to define multiple knee points to handle problems with irregular shapes of the Pareto front.

When optimizing for the flexible job shop scheduling problems in Chapter 5, we use an algorithm configurator to tune the mutation probabilities in MOEA. It would be a promising topic to utilize the automated parameter configuration more and the multi-objective evolutionary optimization can benefit more from the popular field of machine learning.

In Chapter 6, the optimization is performed periodically to update the maintenance schedule in the simulator. However, a better solution would be to detect the change in the dynamic environment and update the maintenance schedule when the change reaches a threshold. Moreover, since we try to maximize the similarity between the new schedule and the previous one, the arrangement of maintenance activities in the current running schedule can be used in the initialization of the population when updating the schedule. In this way information collected from previous searches can be utilized and the PF approximation can be found more effectively.

Bibliography

[1] Ehsan Ahmadi, Mostafa Zandieh, Mojtaba Farrokh, and Seyed Mohammad Emami. A multi objective optimization approach for flexible job shop scheduling problem under random machine breakdown by evolutionary algorithms. *Computers & Operations Research*, 73:56–66, 2016.

[2] Nasr Al-Hinai and Tarek Y ElMekkawy. Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm. *International Journal of Production Economics*, 132(2):279–291, 2011.

[3] Richard Allmendinger, Andrzej Jaszkiewicz, Arnaud Liefooghe, and Christiane Tammer. What if we increase the number of objectives? theoretical and empirical implications for many-objective optimization. *arXiv preprint arXiv:2106.03275*, 2021.

[4] Charles Audet, Jean Bigeon, Dominique Cartier, Sébastien Le Digabel, and Ludovic Salomon. Performance indicators in multiobjective optimization. *European Journal of Operational Research*, 2020.

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

[6] Lucas S Batista, Felipe Campelo, Frederico G Guimaraes, and Jaime A Ramírez. Pareto cone ε-dominance: improving convergence and diversity in multiobjective evolutionary algorithms. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 76–90. Springer, 2011.

[7] Slim Bechikh, Lamjed Ben Said, and Khaled Ghédira. Searching for knee regions in multi-objective optimization using mobile reference points. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 1118–1125, 2010.

[8] Nicola Beume, Carlos M Fonseca, Manuel Lopez-Ibanez, Luis Paquete, and Jan Vahrenhold. On the complexity of computing the hypervolume indicator. *IEEE Transactions on Evolutionary Computation*, 13(5):1075–1082, 2009.

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

[10] Keomany Bouvard, Samuel Artus, Christophe Bérenguer, and Vincent Cocquempot. Condition-based dynamic maintenance operations planning & grouping. application to commercial heavy vehicles. *Reliability Engineering & System Safety*, 96(6):601–610, 2011.

[11] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[12] Jürgen Branke, Kalyanmoy Deb, Henning Dierolf, and Matthias Osswald. Finding knees in multi-objective optimization. In *International Conference on Parallel Problem Solving from Nature*, pages 722–731. Springer, 2004.

[13] Marlon Alexander Braun, Pradyumn Kumar Shukla, and Hartmut Schmeck. Preference ranking schemes in multi-objective evolutionary algorithms. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 226–240. Springer, 2011.

[14] Fatih Camci. System maintenance scheduling with prognostics information using genetic algorithm. *IEEE Transactions on Reliability*, 58(3):539–552, 2009.

[15] Imran Ali Chaudhry and Abid Ali Khan. A research survey: Review of flexible job shop scheduling techniques. *International Transactions in Operational Research*, 23(3):551–591, 2016.

[16] Tsung-Che Chiang and Hsiao-Jou Lin. A simple and effective evolutionary algorithm for multiobjective flexible job shop scheduling. *International Journal of Production Economics*, 141(1):87–98, 2013.

[17] Carlos A Coello Coello, Gary B Lamont, David A Van Veldhuizen, et al. *Evolutionary Algorithms for Solving Multi-objective Problems*, volume 5. Springer, 2007.

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

[19] George B Dantzig. Discrete-variable extremum problems. *Operations Research*, 5(2):266–288, 1957.

[20] Indraneel Das. On characterizing the "knee" of the Pareto curve based on normal-boundary intersection. *Structural Optimization*, 18(2-3):107–115, 1999.

[21] Indraneel Das and John E Dennis. Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8(3):631–657, 1998.

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

[23] Kalyanmoy Deb. Multi-objective evolutionary algorithms: Introducing bias among Pareto-optimal solutions. In *Advances in Evolutionary Computing*, pages 263–292. Springer, 2003.

[24] Kalyanmoy Deb and Shivam Gupta. Towards a link between knee solutions and preferred solution methodologies. In *International Conference on Swarm, Evolutionary, and Memetic Computing*, pages 182–189. Springer, 2010.

[25] Kalyanmoy Deb and Shivam Gupta. Understanding knee points in bicriteria problems and their implications as preferred solution principles. *Engineering Optimization*, 43(11):1175–1204, 2011.

[26] Kalyanmoy Deb and Himanshu Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601, 2013.

[27] Kalyanmoy Deb and Abhay Kumar. Light beam search based multi-objective optimization using evolutionary algorithms. In *2007 IEEE Congress on Evolutionary Computation*, pages 2125–2132. IEEE, 2007.

[28] Kalyanmoy Deb and Abhishek Kumar. Interactive evolutionary multi-objective optimization and decision-making using reference direction method. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, pages 781–788, 2007.

[29] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[30] Kalyanmoy Deb and J Sundar. Reference point based multi-objective optimization using evolutionary algorithms. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pages 635–642. ACM, 2006.

[31] Aidin Delgoshaei, Ahad Ali, Mohd Khairol Anuar Ariffin, and Chandima Gomes. A multi-period scheduling of dynamic cellular manufacturing systems in the presence of cost uncertainty. *Computers & Industrial Engineering*, 100:110–132, 2016.

[32] Yunus Demir and Selçuk Kürşat İşleyen. An effective genetic algorithm for flexible job-shop scheduling with overlapping in operations. *International Journal of Production Research*, 52(13):3905–3921, 2014.

[33] André Deutz, Michael Emmerich, and Yali Wang. Many-criteria dominance relations. In Dimo Brockhoff, Michael Emmerich, Boris Naujoks, and Robin Purshouse, editors, *Many-Criteria Optimization and Decision Analysis*. Springer, Natural Computing Series, 2022.

[34] Matthias Ehrgott. *Multicriteria Optimization*, volume 491. Springer Science & Business Media, 2005.

[35] Hatem M Elattar, Hamdy K Elminir, and AM Riad. Prognostics: a literature review. *Complex & Intelligent Systems*, 2(2):125–154, 2016.

[36] Michael TM Emmerich and André H Deutz. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural Computing*, 17(3):585–609, 2018.

[37] Michael TM Emmerich, André H Deutz, and Johannes W Kruisselbrink. On quality indicators for black-box level set approximation. *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation*, pages 157–185, 2013.

[38] Michael TM Emmerich, André H Deutz, and Iryna Yevseyeva. On reference point free weighted hypervolume indicators based on desirability functions and their probabilistic interpretation. *Procedia Technology*, 16:532–541, 2014.

[39] Jesús Guillermo Falcón-Cardona, Carlos A Coello Coello, and Michael Emmerich. Cri-emoa: A pareto-front shape invariant evolutionary multi-objective algorithm. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 307–318. Springer, 2019.

[40] Jesús Guillermo Falcón-Cardona, Hisao Ishibuchi, Carlos A Coello Coello, and Michael Emmerich. On the effect of the cooperation of indicator-based multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 2021.

[41] Marco Farina and Paolo Amato. On the optimal solution definition for many-criteria optimization problems. In *2002 Annual Meeting of the North American Fuzzy Information Processing Society Proceedings. NAFIPS-FLINT 2002 (Cat. No. 02TH8622)*, pages 233–238. IEEE, 2002.

[42] Parviz Fattahi and Alireza Fallahi. Dynamic scheduling in flexible job shop systems by considering simultaneously efficiency and stability. *CIRP Journal of Manufacturing Science and Technology*, 2(2):114–123, 2010.

[43] Qiu Fei-yue, Wu Yu-shi, Wang Li-ping, and Jiang Bo. Bipolar preferences dominance based evolutionary algorithm for many-objective optimization. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

[44] Eduardo Fernandez, Edy Lopez, Fernando Lopez, and Carlos A Coello Coello. Increasing selective pressure towards the best compromise in evolutionary multiobjective optimization: The extended NOSGA method. *Information Sciences*, 181(1):44–56, 2011.

[45] Mark Fleischer. The measure of pareto optima applications to multi-objective metaheuristics. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 519–533. Springer, 2003.

[46] Carlos M Fonseca and Peter J Fleming. Multiobjective genetic algorithms. In *IEE Colloquium on Genetic Algorithms for Control Systems Engineering*, pages 6–1. IET, 1993.

[47] Alan D Fox, David W Corne, C Gabriela Mayorga Adame, Jeff A Polton, Lea-Anne Henry, and J Murray Roberts. An efficient multi-objective optimization method for use in the design of marine protected area networks. *Frontiers in Marine Science*, 6:17, 2019.

[48] Michael R Garey, David S Johnson, and Ravi Sethi. The complexity of flowshop and jobshop scheduling. *Mathematics of operations research*, 1(2):117–129, 1976.

[49] David Gaudrie, Rodolphe Le Riche, Victor Picheny, Benoit Enaux, and Vincent Herbert. Targeting solutions in bayesian multi-objective optimization: Sequential and batch versions. *Annals of Mathematics and Artificial Intelligence*, pages 1–26, 2019.

[50] Jay B Ghosh. Computational aspects of the maximum diversity problem. *Operations Research Letters*, 19(4):175–181, 1996.

[51] Ioannis Giagkiozis, Robin C Purshouse, and Peter J Fleming. Generalized decomposition and cross entropy methods for many-objective optimization. *Information Sciences*, 282:363–387, 2014.

[52] Hêriş Golpîra and Erfan Babaee Tirkolaee. Stable maintenance task scheduling: A bi-objective robust optimization model. *Computers & Industrial Engineering*, 137:106007, 2019.

[53] Prabhat Hajela and C-Y Lin. Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4(2):99–107, 1992.

[54] Thomas Hanne. On the convergence of multiobjective evolutionary algorithms. *European Journal of Operational Research*, 117(3):553–564, 1999.

[55] Michael Pilegaard Hansen and Andrzej Jaszkiewicz. *Evaluating the Quality of Approximations to the Non-Dominated Set*. IMM Technical Report, Department of Mathematical Modelling, Technical University of Denmark, 1998.

[56] Edwin C Harrington. The desirability function. *Industrial Quality Control*, 21(10):494–498, 1965.

[57] Willy Herroelen and Roel Leus. Project scheduling under uncertainty: Survey and research potentials. *European Journal of Operational Research*, 165(2):289–306, 2005.

[58] Christina J Hopfe, Michael TM Emmerich, Robert Marijt, and Jan Hensen. Robust multi-criteria design optimisation in building design. *Proceedings of Building Simulation and Optimization, Loughborough, UK*, pages 118–125, 2012.

[59] Jeffrey Horn, Nicholas Nafpliotis, and David E Goldberg. A niched Pareto genetic algorithm for multi-objective optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, pages 82–87. IEEE, 1994.

[60] Kokolo Ikeda, Hajime Kita, and Shigenobu Kobayashi. Failure of Pareto-based moeas: Does non-dominated really mean near to optimal? In *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546)*, volume 2, pages 957–962. IEEE, 2001.

[61] Hisao Ishibuchi, Naoya Akedo, and Yusuke Nojima. Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems. *IEEE Transactions on Evolutionary Computation*, 19(2):264–283, 2014.

[62] Hisao Ishibuchi, Ryo Imada, Yu Setoguchi, and Yusuke Nojima. How to specify a reference point in hypervolume calculation for fair performance comparison. *Evolutionary Computation*, 26(3):411–440, 2018.

[63] Andrew KS Jardine, Daming Lin, and Dragan Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20(7):1483–1510, 2006.

[64] Shouyong Jiang, Jinglei Guo, Bashar Alhnaity, and Qingyang Zhang. On analysis of irregular Pareto front shapes. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 15–25. Springer, 2021.

[65] Yaochu Jin and Bernhard Sendhoff. Pareto-based multiobjective machine learning: An overview and case studies. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(3):397–415, 2008.

[66] Nicolas Jozefowiez, Frédéric Semet, and El-Ghazali Talbi. Multi-objective vehicle routing problems. *European Journal of Operational Research*, 189(2):293–309, 2008.

[67] Imed Kacem, Slim Hammadi, and Pierre Borne. Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 32(1):1–13, 2002.

[68] Ibrahim Karahan and Murat Koksalan. A territory defining multiobjective evolutionary algorithms and preference incorporation. *IEEE Transactions on Evolutionary Computation*, 14(4):636–664, 2010.

[69] Ahmed Khalafallah and Khaled El-Rayes. Automated multi-objective optimization system for airport site layouts. *Automation in Construction*, 20(4):313–320, 2011.

[70] Frank Kursawe. A variant of evolution strategies for vector optimization. In *International Conference on Parallel Problem Solving from Nature*, pages 193–197. Springer, 1990.

[71] Marco Laumanns, Lothar Thiele, Kalyanmoy Deb, and Eckart Zitzler. Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation*, 10(3):263–282, 2002.

[72] Marco Laumanns, Eckart Zitzler, and Lothar Thiele. A unified model for multi-objective evolutionary algorithms with elitism. In *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No. 00TH8512)*, volume 1, pages 46–53. IEEE, 2000.

[73] Longmei Li, Yali Wang, Heike Trautmann, Ning Jing, and Michael Emmerich. Multiobjective evolutionary algorithms based on target region preferences. *Swarm and Evolutionary Computation*, 40:196–215, 2018.

[74] Lai-Yee Liu, Vitor Basto-Fernandes, Iryna Yevseyeva, Joost Kok, and Michael Emmerich. Indicator-based evolutionary level set approximation: Mixed mutation strategy and extended analysis. In *International Work-Conference on the Interplay Between Natural and Artificial Computation*, pages 146–159. Springer, 2017.

[75] Yuan Liu, Ningbo Zhu, Kenli Li, Miqing Li, Jinhua Zheng, and Keqin Li. An angle dominance criterion for evolutionary many-objective optimization. *Information Sciences*, 509:376–399, 2020.

[76] Christopher Mattson, Anoop Mullur, and Achille Messac. Minimal representation of multiobjective design space using a smart pareto filter. In *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, page 5458, 2002.

[77] Kaisa Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, 1999.

[78] SMJ Mirzapour Al-E-Hashem, Hooman Malekly, and Mir Bahador Aryanezhad. A multi-objective robust optimization model for multi-product multi-site aggregate production planning in a supply chain under uncertainty. *International Journal of Production Economics*, 134(1):28–42, 2011.

[79] Kaname Narukawa, Yu Setoguchi, Yuki Tanigaki, Markus Olhofer, Bernhard Sendhoff, and Hisao Ishibuchi. Preference representation using gaussian functions on a hyperplane in evolutionary multi-objective optimization. *Soft Computing*, 20(7):2733–2757, 2016.

[80] Vladimir D Noghin. Relative importance of criteria: a quantitative approach. *Journal of Multi-Criteria Decision Analysis*, 6(6):355–363, 1997.

[81] Cemal Özgüven, Lale Özbakır, and Yasemin Yavuz. Mathematical models for job-shop scheduling problems with routing and process plan flexibility. *Applied Mathematical Modelling*, 34(6):1539–1548, 2010.

[82] Ferdinando Pezzella, Gianluca Morganti, and Giampiero Ciaschetti. A genetic algorithm for the flexible job-shop scheduling problem. *Computers & Operations Research*, 35(10):3202–3212, 2008.

[83] Michael James Pierro and William Roy Schneider. Vehicle maintenance management system and method, October 2001. US Patent 6,301,531.

[84] Robin C Purshouse and Peter J Fleming. Evolutionary many-objective optimisation: An exploratory analysis. In *The 2003 Congress on Evolutionary Computation, 2003. CEC.*, volume 3, pages 2066–2073. IEEE, 2003.

[85] Lily Rachmawati and Dipti Srinivasan. A multi-objective evolutionary algorithm with weighted-sum niching for convergence on knee regions. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pages 749–750, 2006.

[86] Lily Rachmawati and Dipti Srinivasan. A multi-objective genetic algorithm with controllable convergence on knee regions. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1916–1923. IEEE, 2006.

[87] Lily Rachmawati and Dipti Srinivasan. Multiobjective evolutionary algorithm with controllable focus on the knees of the pareto front. *IEEE Transactions on Evolutionary Computation*, 13(4):810–824, 2009.

[88] Cristian Ramirez-Atencia, Sanaz Mostaghim, and David Camacho. A knee point based evolutionary multi-objective optimization for mission planning problems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1216–1223, 2017.

[89] Nery Riquelme, Christian Von Lücken, and Benjamin Baran. Performance metrics in multi-objective optimization. In *2015 Latin American Computing Conference (CLEI)*, pages 1–11. IEEE, 2015.

[90] Hiroyuki Sato, Hernán Aguirre, and Kiyoshi Tanaka. Controlling dominance area of solutions in multiobjective evolutionary algorithms and performance analysis on multiobjective 0/1 knapsack problems. *IPSJ Digital Courier*, 3:703–718, 2007.

[91] J David Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the First International Conference on Genetic Algorithms and their Applications*. Lawrence Erlbaum Associates. Inc., Publishers, 1985.

[92] J David Schaffer. Some experiments in machine learning using vector evaluated genetic algorithms. Technical report, Vanderbilt Univ., Nashville, TN (USA), 1985.

[93] Oliver Schütze, Marco Laumanns, and Carlos A Coello Coello. Approximating the knee of an mop with stochastic search algorithms. In *International Conference on Parallel Problem Solving from Nature*, pages 795–804. Springer, 2008.

[94] Xiao-Ning Shen and Xin Yao. Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems. *Information Sciences*, 298:198–224, 2015.

[95] Pradyumn Kumar Shukla, Christian Hirsch, and Hartmut Schmeck. A framework for incorporating trade-off information using multi-objective evolutionary algorithms. In *International Conference on Parallel Problem Solving from Nature*, pages 131–140. Springer, 2010.

[96] Pradyumn Kumar Shukla, Christian Hirsch, and Hartmut Schmeck. In search of equitable solutions using multi-objective evolutionary algorithms. In *International Conference on Parallel Problem Solving from Nature*, pages 687–696. Springer, 2010.

[97] Kazimierz Sobczyk and BF Spencer Jr. *Random Fatigue: From Data to Theory*. Academic Press, 2012.

[98] Zhiming Song, Xiaoyu Chen, Xin Luo, Maocai Wang, and Guangming Dai. Multi-objective optimization of agile satellite orbit design. *Advances in Space Research*, 62(11):3053–3064, 2018.

[99] Nidamarthi Srinivas and Kalyanmoy Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

[100] Ronald Suich and George C Derringer. Is the regression equation adequate?: One criterion. *Technometrics*, pages 213–216, 1977.

[101] Ma Guadalupe Castillo Tapia and Carlos A Coello Coello. Applications of multi-objective evolutionary algorithms in economics and finance: A survey. In *2007 IEEE Congress on Evolutionary Computation*, pages 532–539. IEEE, 2007.

[102] Wannaporn Teekeng and Arit Thammano. Modified genetic algorithm for flexible job-shop scheduling problems. *Procedia Computer Science*, 12:122–128, 2012.

[103] Lothar Thiele, Kaisa Miettinen, Pekka J Korhonen, and Julian Molina. A preference-based evolutionary algorithm for multi-objective optimization. *Evolutionary Computation*, 17(3):411–436, 2009.

[104] Heike Trautmann, Tobias Wagner, and Dimo Brockhoff. R2-emoa: Focused multiobjective search using r2-indicator-based selection. In *International Conference on Learning and Intelligent Optimization*, pages 70–74. Springer, 2013.

[105] Tamara Ulrich, Johannes Bader, and Lothar Thiele. Defining and optimizing indicator-based diversity measures in multiobjective search. In *International Conference on Parallel Problem Solving from Nature*, pages 707–717. Springer, 2010.

[106] George J Vachtsevanos. *Intelligent Fault Diagnosis and Prognosis for Engineering Systems*, volume 456. Wiley Hoboken, 2006.

[107] Pravin M Vaidya. An o(n logn) algorithm for the all-nearest-neighbors problem. *Discrete & Computational Geometry*, 4(2):101–115, 1989.

[108] Eelke van der Horst, Patricia Marqués-Gallego, Thea Mulder-Krieger, Jacobus van Veldhoven, Johannes Kruisselbrink, Alexander Aleman, Michael TM Emmerich, Johannes Brussee, Andreas Bender, and Adriaan P IJzerman. Multi-objective evolutionary design of adenosine receptor ligands. *Journal of Chemical Information and Modeling*, 52(7):1713–1721, 2012.

[109] Duc Van Nguyen, Steffen Limmer, Kaifeng Yang, Markus Olhofer, and Thomas Bäck. Modeling and prediction of remaining useful lifetime for maintenance scheduling optimization of a car fleet. *International Journal of Performability Engineering*, 15(9), 2019.

[110] Bas van Stein, Hao Wang, and Thomas Bäck. Automatic configuration of deep neural networks with ego. *arXiv preprint arXiv:1810.05526*, 2018.

[111] Yash Vesikar, Kalyanmoy Deb, and Julian Blank. Reference point based NSGA-III for preferred solutions. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1587–1594. IEEE, 2018.

[112] Tobias Wagner and Heike Trautmann. Integration of preferences in hypervolume-based multiobjective evolutionary algorithms by means of desirability functions. *IEEE Transactions on Evolutionary Computation*, 14(5):688–701, 2010.

[113] Xiaojuan Wang, Liang Gao, Chaoyong Zhang, and Xinyu Shao. A multi-objective genetic algorithm based on immune and entropy principle for flexible job-shop scheduling problem. *The International Journal of Advanced Manufacturing Technology*, 51(5-8):757–767, 2010.

[114] Yali Wang, André Deutz, Thomas Bäck, and Michael Emmerich. Improving many-objective evolutionary algorithms by means of edge-rotated cones. In *International Conference on Parallel Problem Solving from Nature*, pages 313–326. Springer, 2020.

[115] Yali Wang, André Deutz, Thomas Bäck, and Michael Emmerich. Edge-rotated cone orders in multi-objective evolutionary algorithms for improved convergence and preference articulation. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 165–172, 2020.

[116] Yali Wang, Michael Emmerich, André Deutz, and Thomas Bäck. Diversity-indicator based multi-objective evolutionary algorithm: DI-MOEA. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 346–358. Springer, 2019.

[117] Yali Wang, Longmei Li, Kaifeng Yang, and Michael TM Emmerich. A new approach to target region based multiobjective evolutionary algorithms. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 1757–1764. IEEE, 2017.

[118] Yali Wang, Steffen Limmer, Markus Olhofer, Michael Emmerich, and Thomas Bäck. Automatic preference based multi-objective evolutionary algorithm on vehicle fleet maintenance scheduling optimization. *Swarm and Evolutionary Computation*, page 100933, 2021.

[119] Yali Wang, Steffen Limmer, Markus Olhofer, Michael TM Emmerich, and Thomas Bäck. Vehicle fleet maintenance scheduling optimization by multi-objective evolutionary algorithms. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 442–449. IEEE, 2019.

[120] Yali Wang, Steffen Limmer, Duc Van Nguyen, Markus Olhofer, Thomas Bäck, and Michael Emmerich. Optimizing the maintenance schedule for a vehicle fleet: a simulation-based case study. *Engineering Optimization*, pages 1–14, 2021.

[121] Yali Wang, Bas van Stein, Thomas Bäck, and Michael Emmerich. Improving NSGA-III for flexible job shop scheduling using automatic configuration, smart initialization and local search. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, pages 181–182, 2020.

[122] Yali Wang, Bas van Stein, Thomas Bäck, and Michael Emmerich. A tailored NSGA-III for multi-objective flexible job shop scheduling. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 2746–2753. IEEE, 2020.

[123] Wei Wei and Lihong Qiao. Multi-objective optimization design of complex mechanical and electrical products based on improved evolutionary algorithm. In *Advanced Materials Research*, volume 311, pages 1384–1388. Trans Tech Publications, 2011.

[124] Simon Wessing. *Two-stage Methods for Multimodal Optimization*. PhD thesis, Universitätsbibliothek Dortmund, 2015.

[125] Robin Winter, Floriane Montanari, Andreas Steffen, Hans Briem, Frank Noé, and Djork-Arné Clevert. Efficient multi-objective molecular optimization in a continuous latent space. *Chemical Science*, 10(34):8016–8024, 2019.

[126] Emmanuel Xevi and Shahbaz Khan. A multi-objective optimization approach to water management. *Journal of Environmental Management*, 77(4):269–277, 2005.

[127] Li-Ning Xing, Ying-Wu Chen, and Ke-Wei Yang. An efficient search method for multi-objective flexible job shop scheduling problems. *Journal of Intelligent manufacturing*, 20(3):283–293, 2009.

[128] Guo Yu, Yaochu Jin, and Markus Olhofer. A method for a posteriori identification of knee points based on solution density. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2018.

[129] Lianfei Yu, Cheng Zhu, Jianmai Shi, and Weiming Zhang. An extended flexible job shop scheduling model for flight deck scheduling with priority, parallel operations, and sequence flexibility. *Scientific Programming*, 2017, 2017.

[130] PL Yu and George Leitmann. Nondominated decisions and cone convexity in dynamic multicriteria decision problems. In *Multicriteria Decision Making and Differential Games*, pages 61–72. Springer, 1976.

[131] Yuan Yuan and Hua Xu. Multiobjective flexible job shop scheduling using memetic algorithms. *IEEE Transactions on Automation Science and Engineering*, 12(1):336–353, 2013.

[132] Chaoyong Zhang, Peigen Li, Yunqing Rao, and Shuxia Li. A new hybrid GA/SA algorithm for the job shop scheduling problem. In *European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 246–259. Springer, 2005.

[133] Chaoyong Zhang, Yunqing Rao, Peigen Li, and Xinyu Shao. Bilevel genetic algorithm for the flexible job-shop scheduling problem. *Chinese Journal of Mechanical Engineering*, 43(4):119–124, 2007.

[134] Guohui Zhang, Liang Gao, and Yang Shi. An effective genetic algorithm for the flexible job-shop scheduling problem. *Expert Systems with Applications*, 38(4):3563–3573, 2011.

[135] Qingfu Zhang and Hui Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[136] Qingfu Zhang, Aimin Zhou, Shizheng Zhao, Ponnuthurai Nagaratnam Suganthan, Wudong Liu, and Santosh Tiwari. Multiobjective optimization test instances for the CEC 2009 special session and competition. 2008.

[137] Eckart Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*, volume 63. Citeseer, 1999.

[138] Eckart Zitzler, Joshua Knowles, and Lothar Thiele. Quality assessment of pareto set approximations. In *Multiobjective Optimization*, pages 373–404. Springer, 2008.

[139] Eckart Zitzler and Simon Künzli. Indicator-based selection in multiobjective search. In *International Conference on Parallel Problem Solving from Nature*, pages 832–842. Springer, 2004.

[140] Eckart Zitzler, Marco Laumanns, and Stefan Bleuler. A tutorial on evolutionary multiobjective optimization. In *Metaheuristics for Multiobjective Optimisation*, pages 3–37. Springer, 2004.

[141] Eckart Zitzler and Lothar Thiele. Multiobjective optimization using evolutionary algorithms — a comparative case study. In Agoston E. Eiben, Thomas Bäck, Marc Schoenauer, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature — PPSN V*, pages 292–301, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[142] Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.

Summary

Multi-objective optimization is an effective technique for finding optimal solutions that balance several conflicting objectives. It has been applied in many fields of our world, because practical problems usually have more than one desired goal. For example, developing a new vehicle component might involve minimizing weight while maximizing strength; choosing a portfolio might involve maximizing the expected return while minimizing the risk. The multi-objective optimization problems solved in the thesis originated from the CIMPLO (Cross-Industry Predictive Maintenance Optimization Platform) project. In the CIMPLO project, given a vehicle fleet, each vehicle comprises a set of components, and each component can be maintained in a workshop among given workshops with varying processing times and costs. The goal is to find the best maintenance order, location and time for each component, i.e., to optimize the maintenance schedule of the vehicle fleet. The maintenance schedule of the vehicle fleet is optimized to bring business advantages to industries, such as, to reduce maintenance time, increase safety, and decrease repair expenses. This problem is strongly NP-hard since the flexible job shop scheduling problem (FJSP) has been proven to be a strongly NP-hard problem and the vehicle fleet maintenance scheduling optimization (VFMSO) problem can be seen as an extension of the FJSP: the VFMSO provides a non-specific operation sequence and involves the processing costs of the operation on machines besides the processing times as in the FJSP. Therefore, evolutionary algorithms (EAs) have been chosen to solve our real-world application problem because they have proven to be a particularly suitable metaheuristic method to solve multi-objective optimization problems.

First of all, a multi-objective evolutionary algorithm (MOEA) is developed for general multi-objective optimization problems and it plays an important role in solving our application problems. The proposed MOEA is called diversity indicator-based MOEA (DI-MOEA). The main features or advantages of DI-MOEA include: it uses

a hybrid selection scheme (the $\mu + \mu$ generational selection operator and $\mu + 1$ steady state selection operator) for combining the advantages of Pareto dominance-based approaches to ensure fast convergence to the Pareto front (PF) with indicator-based approaches to ensure convergence to a diverse set; it avoids the use of complex structure and parameters; it can achieve a uniformly distributed PF approximation; it requires no priori knowledge of the PF shape and location; the adopted diversity indicator in DI-MOEA (i.e., Euclidean distance based geometric mean gap indicator) is computationally efficient.

Secondly, to improve the performance of MOEAs, the edge-rotated cone order is proposed for the purpose of building an ordering which can guide the search towards the PF better than the Pareto dominance in MOEAs. The edge-rotated cone is designed by rotating the edges of the standard Pareto order cone towards the outside; therefore, an ordering among Pareto incomparable solutions is established because a solution can dominate an enlarged region in the objective space. The edge-rotated cone order is integrated in MOEAs by a hybrid approach which gives consideration to both convergence and diversity in the evolutionary search process.

To only focus the search on the preferred solutions, the target based MOEAs have been proposed and implemented as three algorithms: T-NSGA-II, T-SMS-EMOA and T-R2-EMOA. In these algorithms, different types of target (e.g., point or region), different shapes of target region (e.g., sphere or square), different positions of target (e.g., on the PF or not), one or multiple targets can be decided by the decision maker (DM) and involved as preference information to find preferred solutions without exploring the whole set of Pareto optimal solutions. Furthermore, to avoid the difficulty for specifying the preference information by the DM, an automatic preference based DI-MOEA (AP-DI-MOEA) is proposed to generate the preference region by identifying the knee point because the knee points are assumed to be the most interesting solutions, naturally preferred solutions and most likely the choice of the DMs. AP-DI-MOEA divides the computing budget into different parts to first find a rough entire PF and the knee region, then narrow down the preference region step by step to benefit its accuracy, and eventually obtain the preferred solutions.

After the proposal of the above algorithms, our goal is to solve the VFMSO problem. Since our problem is a special extension of the FJSP and the FJSP is very difficult to solve and not much research has been devoted to solving this problem in the past. An MOEA to solve the multi-objective FJSP is proposed first, in which multiple initialization approaches, various domain-specific genetic operators, and different local search strategies are designed and used; the Mixed-Integer Programming

Efficient Global Optimization (MIP-EGO) configurator is adopted to automatically find the optimal mutation probabilities. Based on the study on the FJSP, the MOEA to solve our real-world multi-objective VFMSO problems can then be developed. To this end, the problem has been formulated, a three-vector chromosome is proposed to represent a solution and corresponding genetic operators are designed. At the same time, the special designs for the VFMSO are proposed, such as the combination of components as one maintenance task, the definition of the execution window for each component based on its predicted remaining useful lifetime, the simulation of the penalty costs and the failures for handling uncertainties in the predictions.

Lastly, the static MOEAs and preference based MOEAs are applied to the VFMSO problems. Since continuously updating the schedule is required in the real-world scenario, the static algorithms are extended to many-objective dynamic MOEAs in which an extra objective, i.e., the schedule stability, is involved in to minimize the changes between the new schedule and the previous schedule, and the underlying idea is to reduce additional costs such as the cost of reallocation of tools and equipment, the cost of reordering of raw materials and etc.

In summary, this thesis provides advancements to the design of general purpose multi-objective and many-objective optimization algorithms for finding uniform approximations to the (preferred region of the) Pareto front, and successfully applied and customized the algorithms in the domain of flexible job shop scheduling and the more specific domain of dynamic prediction-based maintenance scheduling.

Samenvatting

Multi-criteria optimalisatie is een effectieve techniek voor het vinden van optimale oplossingen die een afweging bieden tussen verschillende, tegenstrijdige criteria. Het heeft zijn toepassing gevonden in de wereld om ons heen omdat bij het oplossen van praktische, reële wereld problemen men gewoonlijk te maken heeft met meerdere na te streven doelen. Bij voorbeeld, bij het ontwikkelen van een onderdeel kan het gewenst zijn om het gewicht te minimaliseren en de sterkte te maximaliseren; bij het kiezen van een portfolio bij het beleggen kan men, bijvoorbeeld, eisen dat het verwachte rendement wordt gemaximaliseerd en het risico geminimaliseerd. Het multi-criteria optimalisatie probleem dat in dit proefschrift is opgelost is ontleend aan het CIMPLO ('Cross-Industry Predictive Maintenance Optimization Platform' - industrieonafhankelijk/overschrijdend voorspellend onderhoud optimalisatie platform) project. In het CIMPLO project is het doel om voor een vloot van voertuigen een optimaal onderhoudsschema op te stellen – elk voertuig bestaat uit een verzameling van componenten en elke component kan naar keuze in één van de werkplaatsen onderhouden worden waarbij de kosten en duur van het onderhoud van het onderdeel per werkplaats variëren. Het optimaliseren van het onderhoudsschema wordt gedaan om zakelijke voordelen te bieden aan de industrieën zoals het verlagen van de duur van het onderhoud, de veiligheid te verbeteren en het besparen van reparatiekosten. Dit probleem is een sterk NP-moeilijk ('strongly NP-hard') probleem, daar voor het flexibele-job-shop-planningsprobleem ('flexible job scheduling problem' FSJP) bewezen is dat het een sterk NP-moeilijk probleem is en ons voertuigen onderhoudsschema optimalisatie probleem ('Vehicle Fleet Maintenance Optimization' (VFMSO) probleem) een uitbreiding is van het FSJP: het VFMSO verschaft een niet-specifieke verwerkingsvolgorde en omvat naast de verwerkingskosten van de bewerking op machines ook de verwerkings-tijden zoals in de FJSP. Dit heeft tot gevolg dat evolutionaire algoritmen (EA's) gekozen zijn om dit praktische, reële wereld probleem op te lossen daar deze

meta-heuristische methode bijzonder geschikt is om multi-criteria problemen op te lossen.

Ten eerste is een multi-criteria evolutionair algoritme ('multi-objective evolutionary algorithm' MOEA) ontwikkeld voor generieke, multi-criteria optimalisatie problemen. Dit algoritme speelt een belangrijke rol in het oplossen van ons praktisch probleem. Het voorgestelde MOEA heet diversiteitsindicator gestoelde MOEA (DI-MOEA). De belangrijkste eigenschappen (voordelen) van DI-MOEA zijn: het gebruikt een hybride selectie schema (de $\mu + \mu$ generatie selectie operator en de $\mu + 1$ 'steady state' selectie operator) ten einde de voordelen van Pareto dominantie-gebaseerde benaderingen die een snelle convergentie naar het Pareto frontier (PF) garanderen te combineren met indicator gebaseerde aanpakken die convergentie naar een verzameling met goede diversiteit garanderen; het vermijdt het gebruik van complexe structuren en parameters; het biedt de mogelijkheid een uniform gedistribueerde benadering van de PF te genereren; het gebruikt geen a priori kennis van de vorm en locatie van de PF; de gebruikte diversiteitsindicator (i.e., op de Euclidische afstand gebaseerde meetkundig gemiddelde gap-indicator) is ook efficiënt te berekenen.

Ten tweede om de prestaties van MOEA's te verbeteren is een rand-geroteerde conus orde geïntroduceerd met als doel een ordeningsrelatie te verkrijgen waarmee de zoektocht naar de PF te verbeteren in vergelijking met de Pareto dominantie in MOEA's. De rand-geroteerde conus wordt verkregen door de randen van de standaard Pareto orde conus naar buiten te draaien; dit heeft als gevolg dat een orde wordt verkregen waarin meer oplossingen die in de Pareto orde niet vergelijkbaar zijn vergelijkbaar worden in de nieuwe orde daar een oplossing een groter deel van de criteria ruimte domineert. De rand-geroteerde conus orde is met zorg ingebed in de MOEA's opdat zowel convergentie en diversiteit in het evolutionaire zoekproces de nodige invloed krijgen.

Om alleen te focussen op voorkeursoplossingen, zijn doel ('target') gebaseerde MOEA's geïntroduceerd en geïmplementeerd als drie algoritmen: T-NSGA-II, T-SMS-EMOA en T-R2-EMOA. In deze algoritmen worden verschillende typen doelen ('targets') gebruikt (b.v., punt of gebied), verschillende vormen voor het doelgebied (b.v., cirkelschijf of vierkant), verschillende posities van de 'target' (b.v., op de PF of niet); een doel of meerdere doelen kunnen door de beslisser ('decision maker' DM) worden bepaald om gebruikt te worden als voorkeursinformatie waarmee niet naar alle Pareto optimale oplossingen gezocht hoeft te worden. Om de moeilijkheid voor het specificeren van de voorkeursinformatie door de DM te ontlopen, is er bovendien een op automatische voorkeur gebaseerde DIMOEA (AP-DI-MOEA) voorgesteld om het

voorkeursgebied dat door het kniepunt ('knee point') wordt geïndentificeerd te genereren – algemeen wordt aangenomen dat kniepunten de meest begeerde en van nature voorkeursoplossingen zijn die hoogst waarschijnlijk de keuze zijn van de DM's. De AP-DI-EMOA splitst het rekenbudget in twee delen op: het eerste deel wordt gebruikt om een ruwe benadering van de hele PF te vinden en daarmee het kniepunt gebied om vervolgens het tweede deel te gebruiken om het voorkeursgebied stap voor stap te verkleinen voor een grotere precisie om daarna de voorkeursoplossingen te verkrijgen.

Na de introductie van bovengenoemde, generieke algoritmen richten we onze aandacht op het oplossen van het VFMSO probleem. Aangezien ons probleem een uitbreiding vormt van het FJSP probleem en dit probleem al moeilijk op te lossen is en bovendien er niet veel onderzoek gewijd is aan dit probleem, gaan we eerst aan de slag met een MOEA om het multi-criteria FJSP probleem op te lossen. Hierbij ontwikkelen en gebruiken wij verscheidene initialisatie benaderingen, verscheidene domein-specifieke genetische operatoren en verschillende zoek strategieën. Bovendien is de Mixed-Integer Programming Efficient Global Optimization (MIP-EGO) configurator aangepast om automatisch de optimale mutatiekansen te vinden. Gewapend met de studie van het FJSP probleem ontwikkelen we het MOEA om onze reële wereld, multi-criteria VFMSO problemen op te lossen. Hiertoe is een specificatie van het probleem geformuleerd, een 3-vector chromosoom ontworpen om oplossingen te representeren en zijn ook de corresponderende genetische operatoren ontwikkeld. Tegelijker-tijd zijn speciale vernieuwingen voor het VFMSO probleem gemaakt zoals het combineren van componenten die dan één onderhoudstaak vormen, de definitie van de uitvoeringsvenster ('execution window') voor elke component dat gebaseerd is op de voorspelde bruikbare resterende levensduur, de simulatie van boetekosten en het niet omgaan met de onzekerheden in de voorspellingen.

Ten slotte zijn de statische MOEA's en op voorkeur gebaseerde MOEA's toegepast op de VFMSO problemen. Aangezien het noodzakelijk is om het planningsschema continu bij te werken in het reële wereld scenario, worden de statische algoritmen uitgebreid tot dynamische MOEA's waarin een extra criterium, namelijk de stabiliteit van het planningsschema wordt gebruikt om het aantal veranderingen tussen het nieuwe en oude schema zo min mogelijk te laten zijn. Het achterliggende idee voor dit criterium is het reduceren van extra kosten zoals de kosten vanwege hertoewijzing van gereedschap en apparatuur, de kosten van het nabestellen van grondstoffen etc.

Samengevat biedt dit proefschrift vooruitgang en verbeteringen in het ontwerpen van generieke multi-criteria en 'many'-criteria (dwz meer dan 3 criteria) algoritmen voor het vinden van uniforme benaderingen van de Pareto frontier (of een voorkeurs-

gebied van de Pareto frontier), en zijn deze algoritmen voor het domein van flexibele-job-shop-planning en het meer specifieke domein van dynamische, op voorspellingen gebaseerde onderhoudsplanning op maat gemaakt en met succes toegepast.

Curriculum Vitae

Yali Wang was born in Baiyin, China. She received her BSc of Computer Science at Wuhan University, China in 1997. After that, she worked first as a software engineer and then as a configuration management officer in H3C Technologies Co., Limited, Beijing, China. In 2015, she came to the Netherlands and completed her MSc at Leiden Institute of Advanced Computer Science (LIACS), Leiden University in 2017. Right after, she worked as a PhD at the same university under the supervision of Michael Emmerich and Thomas Bäck. Her research interests include multi-objective optimization, evolutionary algorithm, scheduling optimization, prediction-based optimization, preference based multi-objective optimization and dynamic optimization.

Acronyms

AP-DI-MOEA

Automatic Preference based Diversity Indicator-based Multi-objective Evolutionary Algorithm

DF

Desirability Function

DI-MOEA

Diversity Indicator-based Multi-objective Evolutionary Algorithm

DM

Decision Maker

DRS

Dominance Resistant Solution

EA

Evolutionary Algorithm

EAF

Empirical Attainment Function

EMO

Evolutionary Multi-objective Optimization

EMOA

Evolutionary Multi-objective Optimization Algorithm

EP

Evolutionary Programming

ES

Evolution Strategy

FJSP
Flexible Job shop Scheduling Problem

GA
Genetic Algorithm

GD
Generational Distance

GI
Gap Indicator

GP
Genetic Programming

HV
Hypervolume

IBEA
Indicator-based Evolutionary Algorithm

IGD
Inverted Generational Distance

JSP
Job shop Scheduling Problem

MIP-EGO
Mixed integer, Parallel - Efficient Global Optimization

MOEA
Multi-objective Evolutionary Algorithm

MOFJSP
Multi-objective Flexible Job shop Scheduling Problem

MOO
Multi-Objective Optimization

MOP
Multi-objective Optimization Problem

MOVFMSO
Multi-objective Vehicle Fleet Maintenance Scheduling Optimization

NE

Number of Evaluations

NSGA-II

Non-dominated Sorting Genetic Algorithm II

NSGA-III

Non-dominated Sorting Genetic Algorithm III

PF

Pareto Front

ROI

Region of Interest

RUL

Remaining Useful Lifetime

SMS-EMOA

$\mathcal{S}$-Metric Selection Evolutionary Multi-Objective Algorithm

VFMSO

Vehicle Fleet Maintenance Scheduling Optimization