EVOLUTIONARY APPROACHES TO MANY-OBJECTIVE COMBINATORIAL OPTIMIZATION PROBLEMS

by

Hayrullah Mert Şahinkoç

B.S., Industrial Engineering, Boğaziçi University, 2012

M.S., Industrial Engineering, Boğaziçi University, 2014

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Graduate Program in Industrial Engineering

Boğaziçi University

2020

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to my adviser Prof. Dr. Ümit Bilge for her motivation, enthusiasm and immense knowledge. The way she addresses the obstacles is incredibly inspiring. This work could not have been accomplished without her continuous support throughout my entire graduate study.

Besides my adviser, I am grateful for having the opportunity to work with Prof. Dr. Necati Aras and Prof. Dr Serpil Sayın for their insightful comments and encouragement. I would like to thank Prof. Dr. Kuban Altınel and Prof. Dr. Bülent Çatay for being part of my thesis committee and for their valuable comments.

My appreciation also extends to my dear friends and fellow colleagues. Thanks to Gökalp, Gökçe, Kübra, Bahadır, Ekin and Çiğdem for the great times we spent together. I also take this opportunity to record my deepest thanks to all the precious faculty in our department for providing the foundation for my future academic career.

I am eternally grateful to my dear family for their love, unlimited encouragement and giving me the perspective whenever I need it. My life would have been much harder without their unconditional support. I am indebted to them for making me the person who I am, and they deserve to take credit for all my accomplishments.

Last but definitely not the least, to my caring, loving and supportive wife Elvan, my deepest gratitude. I am very much thankful for your love and the endless motivation I've been getting all this while. My life would be incomplete without you. Thank you for loving me more than everything.

This work is supported by Boğaziçi University Research Fund Grant Number 13843. I also thank TÜBİTAK for financially supporting me during my doctoral studies under BIDEB-2211 program.

ABSTRACT

EVOLUTIONARY APPROACHES TO MANY-OBJECTIVE COMBINATORIAL OPTIMIZATION PROBLEMS

Many-objective evolutionary approaches try to characterize and overcome the challenges posed by the large number of objectives and have been shown to be very effective for achieving good Pareto approximations. Despite the growing interest, most of the existing studies work on well-defined continuous objective functions with designed features, and studies on combinatorial problems are still rare. The proposed many-objective evolutionary algorithm is characterized by elitist nondominated sorting and reference set based sorting where the reference points are mapped onto a fixed hyperplane obtained at the beginning of the algorithm by solving single-objective problems. All evolutionary mechanisms such as reference point guided path relinking, repair and local improvement procedures are designed to complement the reference set based sorting. Moreover, the reference set co-evolves simultaneously with the solution set, using both cooperative and competitive interactions to balance diversity and convergence, and adapts to the topology of the Pareto front in a self-adaptive parametric way. The proposed algorithm works successfully both under binary and permutation encoding, as well as for correlated objectives or objectives with different scales. Near optimal solutions can be used to construct the hyperplane without any significant deterioration in the quality of the Pareto approximation. Moreover, when an optimization problem under scenario-based uncertainty is modeled as a many-objective problem, the proposed algorithm can provide good solutions simultaneously for several robust measures. Numerical experiments demonstrate the success of the proposed algorithm compared to state-of-art approaches and confirm that it can be applied sustainably to a variety of many-objective combinatorial problems.

ÖZET

PEK-ÇOK AMAÇLI TÜMLEŞİK PROBLEMLERE EVRİMSEL YAKLAŞIMLAR

Pek-çok amaçlı evrimsel eniyileme adı verilen araştırma alanında, çok sayıda amacın getirdiği zorluklar aşılmaya çalışılmaktadır ve bu yordamların başarılı Pareto yaklaşımlarına ulaşmak için çok etkili olduğu gösterilmiştir. Artan ilgiye rağmen, mevcut çalışmaların çoğu önceden tasarlanmış özelliklere sahip, iyi tanımlanmış sürekli işlevler üzerinde geliştirilip denenmektedirler ve tümleşik problemlerle ilgili çalışmalar oldukça nadirdir. Önerilen pek-çok amaçlı evrimsel yordamın farklı sürümleri seçkinci baskılanmamış sıralama ve referans seti temelli sıralama kullanır, ancak referans noktaları yordamın başında tek amaçlı problemler çözülerek elde edilen sabit bir aşırı düzlem üzerinde konumlandırılır. Tüm evrimsel mekanizmalar, referans seti temelli sıralamayı tamamlayıcı şekilde tasarlanmıştır. Bu nedenle çaprazlama şeması olarak referans noktası güdümlü yol birleştirme önerilmektedir. Onarım ve yerel iyileştirme izlekleri de referans noktaları tarafından yönlendirilir. Ayrıca, referans seti, çeşitliliği ve yakınsamayı dengelemek için hem işbirlikçi hem de rekabetçi etkileşimler kullanarak çözüm seti ile eş zamanlı olarak birlikte-evrimleşir ve Pareto cephesinin topolojisine kendi kendini uyarlayan bir parametrik yolla uyum sağlar. Önerilen algoritma, hem ikili ve hem permütasyon kodlaması ile ilişkili amaç fonksiyonları ve farklı ölçeklere sahip amaç fonksiyonları altında başarıyla çalışmaktadır. Hiper düzlemi inşa etmek için en uyguna yakın çözümler de Pareto yaklaşımının kalitesinde önemli bir bozulma olmadan kullanılabilir. Ayrıca, senaryoya dayalı belirsizlik altındaki eniyileme problemi pek-çok amaçlı bir problem olarak modellendiğinde, önerilen algoritma birçok gürbüz ölçüt için aynı anda iyi çözümler sağlayabilmektedir. Sayısal deneyler, önerilen algoritmanın başka güncel yordamlara kıyasla başarısını göstermekte ve pek-çok amaçlı tümleşik problemlere sürdürülebilir şekilde uygulanabileceğini doğrulamaktadır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$age(\vec{x})$	Age of a particular solution $\vec{x}$
$C1_R$	Ratio of the Reference Points Found
$C(\mathcal{S}_1, \mathcal{S}_2)$	Coverage of Two Sets performance metric
C_k	Capacity parameter for constraint k in the knapsack problem
c_{ijk}	Cost of traveling from city j to k for objective i in MaOTSP
c_t	Candidate in iteration t of path relinking
d	Division count
d_t	Distance between parent node and guiding reference ray in path relinking
d_{rs}	Distance between locations r and s in MaOQAP
f_i^{max}	The best value observed for objective i in current iteration in RPEA
f_i^{min}	The worst value observed for objective i in current iteration in RPEA
f_{ijk}	Flow between departments j and k for objective i in MaOQAP
H	Number of reference points
$I_{\epsilon+}$	Binary additive epsilon-indicator
l	Front level
$L(\vec{r}_x)$	Location of a particular reference point $\vec{r}_x$
m	Objective count
$maxGen$	Maximum iteration number of PICEA-w
M_3^*	Maximum Spread performance metric
n_{total}	Total difference in genotypes of two parent solutions
n	Variable count
N	Population size, threshold level for the truncation in nondominated sorting
n_t	End node in iteration t of path relinking
p	Constraint count
p_{ij}	Profit coefficient for objective i and item j in the knapsack problem

p_ω	Probability of scenario ω
$\mathcal{P}$	Population
$\mathcal{P}^*$	Pareto optimal set
$\mathcal{P}_{ext}$	External population
$\mathcal{P}_{off}$	Offspring solution set
$\mathcal{P}_{pairs}$	Parent pair set
$\mathcal{P}(A)$	Power set
$P(\vec{x})$	Penalty amount of a particular solution $\vec{x}$
$path$	Path in path relinking
$penaltyCoef$	Penalty coefficient
$\mathcal{PF}$	Pareto Front
Q	Subset of cities in MaOTSP
q_j	Calculated values for item j for sorting in repair/local improvement procedures in the knapsack problem
$\vec{r}$	Reference point or vector
$\mathcal{R}$	Reference point set
$r_{associate}$	Association of the most recent node in $path$
$\mathcal{R}_A$	Associated reference point set
$\mathcal{R}_{DEL}$	Empty reference points thought the recent cycle
$\mathcal{R}_E$	Empty reference point set
$\mathcal{R}_{img}$	Immigrated reference point set
$\mathcal{R}_{off}$	Offspring reference point set
$\mathcal{R}_{par}$	Parent reference point set
r_f	Feasibility ratio coefficient
r_g	Guiding reference point
$\mathcal{R}_G$	Guiding reference point set
$\mathcal{R}_N$	Neighbor reference point set
SC	Set of cities in MaOTSP
s_{off}	Offspring solution
$\mathcal{S}_t$	Solution set in iteration t
t	Iteration count during the evolution

t_{grp}	Parameter to determine reference set regeneration frequency in RPEA
$V_k(\vec{x})$	Violation amount of a particular solution $\vec{x}$ for constraint k
w	Weight function or vector
w_{jk}	Weight parameter for item k and constraint k in the knapsack problem
$\mathcal{X}_E$	Efficient set
x_j	Binary decision variable corresponding to whether item j is included in the knapsack collection or not
x_{jk}	Binary decision variable corresponding to whether the edge form city j to k in included in the tour or not
x_{jr}	Binary decision variable corresponding to whether the department j is assigned to location r or not
$\vec{y}^I$	Ideal point
$\vec{y}^N$	Nadir point
$\mathcal{Y}_{ND}$	Nondominated set
z^*	Reference point in MOEA/D
α	Randomly generated non-negative coefficient in the crossover of co-evolution and correlated objectives *or* coefficient used to select solutions in RPEA
δ	Move-value in path relinking *or* distance coefficient in RPEA
ϵ_i	Distance between generated reference point and corresponding solution for objective i in RPEA
θ	Angle between two vectors in PICEA-w
ρ_j	Niche count of reference point j
τ	Predefined number of generations used in A^2-NSGA-III
Ω	Set of all possible scenarios

LIST OF ACRONYMS/ABBREVIATIONS

ASF	Achievement Scalarizing Function
CE	Coverage Error
DM	Decision Maker
EMO	Evolutionary Multi-Objective Optimization
FHCo	Fixed Hyperplane Co-evolutionary Reference Points
FHFR	Fixed Hyperplane Fixed Reference Points
FHMR	Fixed Hyperplane Mobile Reference Points
GA	Genetic Algorithm
HR	Hypervolume Ratio
IGD	Inverse Generational Distance
MaOEA	Many-Objective Evolutionary Algorithm
MaOATSP	Many-Objective Asymmetric Traveling Salesman Problem
MaOKP	Many-Objective Knapsack Problem
MaOP	Many-Objective Optimization Problem
MaOQAP	Many-Objective Quadratic Assignment Problem
MaOTSP	Many-Objective Traveling Salesman Problem
MOCO	Multi-Objective Combinatorial Optimization
MOEA	Multi-Objective Evolutionary Algorithm
MOP	Multi-Objective Optimization Problem
NP	Nondeterministic Polynomial Time
NSGA	Nondominated Sorting Genetic Algorithm
OR	Operations Research
PDF	Probability Density Function
PRO	Pareto Robustly Optimal
QAP	Quadratic Assignment Problem
TSP	Traveling Salesman Problem
UL	Uniformity Level

1. INTRODUCTION

Multi-objective optimization problems (MOPs) include two or more conflicting objectives. Due to this conflicting nature, the optimal solutions to a MOP consist of a set of compromised solutions known as the Pareto optimal set. A *Pareto optimal* (or *efficient*) solution is characterized such that no single objective can be improved without degrading at least one of the others, thereby corresponds to a *nondominated* point in the objective space. Solving a MOP usually involves determination of the entire Pareto optimal set or at least as many Pareto optimum solutions as possible. Evolutionary algorithms, which can produce a set of efficient solutions in a single run due to their population-based structures, have proven to be very effective for achieving good *Pareto approximations*. Consequently, multi-objective evolutionary algorithms (MOEAs) are often referred as "the methods" to deal with MOPs (Deb, 2001).

On the other hand, when MOPs have more than three objectives, they are considered to deserve a new name as *many-objective optimization problems* (*MaOPs*). This is because, as the number of objectives increases, the number of nondominated solutions becomes huge and maintaining diversity as well as convergence for a good and well-spread approximation of the true Pareto front turns into a true challenge (Chand and Wagner, 2015). This often renders conventional MOEAs mostly ineffective and improved mechanisms are required. Several recent studies have attempted to characterize the challenges posed by MaOPs and propose approaches that adapt or extend existing MOEAs to overcome these challenges. As a result, the so-called many-objective evolutionary algorithms (MaOEAs) have become an emerging topic in the field of multi-objective optimization (Li *et al.*, 2015).

The reference set based approaches are one of the most promising branches of MaOEAs. These approaches introduce a set of references (reference points, reference vectors, or weight vectors) to assist the evolutionary process and the diversity of the resulting Pareto set approximation. A good quality reference set should be less sensitive to problem type and Pareto front geometry and should be scalable to keep algorithm

performance successful despite the increase in the number of objectives. Therefore, it is crucial to develop effective strategies for positioning and utilization of the reference set. One of the fundamental research topic questions in this thesis is to investigate several reference set positioning strategies in an attempt to develop an enhanced approach. Hence, the aim is to develop a successful MaOEA that can contribute to present methodologies to overcome existing challenges in MaOPs.

It has been observed that most methodological studies attempting to build multi-objective or many-objective evolutionary techniques use test suites consisting of well-defined continuous mathematical functions such as DTLZ (Deb *et al.*, 2005), ZDT (Zitzler *et al.*, 2000) and WFG (Huband *et al.*, 2006). These problems are repeatedly preferred because they are scalable to any number of objectives and decision variables, and the exact positions and shapes of the resulting Pareto optimal fronts are known in advance. Combinatorial optimization problems, on the other hand, still remain uncharted for the studies of many-objective approaches, although they have a vast amount of applications in real-life. However, these problems, which display specific characteristics due to their discrete nature, also exhibit special needs. Therefore, the problem domain of this thesis is combinatorial optimization problems.

Many-objective knapsack problem (MaOKP), many-objective traveling salesman problem (MaOTSP), and many-objective quadratic assignment problem (MaOQAP) are selected as benchmarks. Each of these benchmark problems serves different purposes during the analysis. For instance, MaOKP, which is an easy problem to generate, is a good starting point within the domain of combinatorial optimization. The constrained nature, and consequently infeasibility, is rarely a problem since infeasible solutions are very easy to repair. In fact, most of the recent pioneering studies in the field of combinatorial optimization assess the performance of MOEAs on MaOKP (Ishibuchi *et al.*, 2014a). Consequently, MaOKP is used as the benchmark during the development stages of the proposed MaOEA. It is then adapted to permutation encoding to validate the performance in different combinatorial problems such as MaOTSP and MaOQAP.

After exploring the prominent aspects of existing multi- and many-objective approaches, some of these features are combined with several effective evolutionary strategies in an innovative fashion to design a successful MaOEA. The proposed MaOEA uses elitist nondominated sorting and reference set based sorting, however the reference points are mapped onto a "*fixed hyperplane*" obtained at the beginning of the algorithm. The use of a fixed hyperplane is the first novel idea that provides a significant improvement in performance. The construction of the fixed hyperplane requires single-objective problems to be optimally/near optimally solved in advance at reasonable times, resulting in a hybrid structure for the proposed algorithm.

The way to locate the reference points on this fixed hyperplane is a critical issue investigated in detail. As opposed to the base policy where reference points are fixed on a well-spread mesh on the hyperplane, in another strategy, they are allowed to move form their original location on the mesh in an attempt to adapt to the progression of evolution. Finally, the reference point set co-evolves simultaneously with the population of candidate solutions. This "*co-evolutionary*" approach, which constitutes the key innovative part of the proposed MaOEA, provides the necessary adaptability to achieve both convergence and divergence, bringing also a conclusive success. Other successful features of the proposed MaOEA are path relinking recombination scheme integrated with complementing selection mechanisms and repair/local improvement procedures, as well as the use of an external population structure to store nondominated solutions.

The proposed approaches in this thesis are aimed to be designed as generic and adaptable to different problem characteristics as possible. In this way, the proposed MaOEA can be used for a wider spectrum of combinatorial optimization problems. In order to generalize the insights obtained from benchmark problems to other combinatorial optimization problems, extensive numerical experimentation is carried out in different application areas such as constrained problems, problems with correlated objectives and objectives of different scales. The performance of different versions of the proposed MaOEA is examined and validated in comparison to some existing state-of-the-art evolutionary algorithms.

One of the main propositions of this thesis is modeling a problem under scenario-based uncertainty as a MaOP where each uncertainty scenario results in a different objective function realization. Consequently, a new and natural application area is opened by developing the many-objective counterparts of the optimization problems that are subject to scenario-based uncertainty. In addition, both stochastic and robust optimization fields are contributed by proposing this alternative perspective. By achieving a successful Pareto approximation, satisfactory solutions in terms of many stochastic and robust metrics are found simultaneously in a single execution of MaOEA. In this way, the decision maker has the opportunity to choose efficient solutions that cover a wide range of different performance metrics according to his/her preferences.

Based on the above, targeted contributions of the proposed approach in this thesis are summarized as follows:

- A reference set based evolutionary approach using a fixed hyperplane obtained at the beginning of the algorithm
- Innovative complementary evolutionary strategies and genetic operators such as reference point guided path relinking
- Non-parametric and self-adaptive co-evolution of reference and solution sets including both cooperative and competitive aspects
- Yielding the best results so far in the literature for all the benchmark problems
- A new natural application area combining the optimization problems under scenario-based uncertainty and MaOEAs

The thesis consists of six chapters. In Chapter 2, related definitions, concepts and notation are introduced with a historical overview of related studies in the literature. Combinatorial optimization problems selected as benchmarks are described in Chapter 3. In Chapter 4, a detailed explanation for the proposed MaOEA is provided by going through its development stages and its key aspects. Numerical experimentation containing a variety of aspects such as the verification of the proposed MaOEA, correlated objectives, different objective scales, and searching for robust solutions is presented in Chapter 5. Finally, a conclusion is provided in Chapter 6.

2. DEFINITIONS, PRELIMINARIES AND LITERATURE REVIEW

This chapter provides an introduction to the basics in this field of study. Related definitions, notation and concepts are introduced, main challenges and suggested solutions are discussed along with an historical overview of related literature. In Section 2.1, definitions and studies related to multi-objective optimization and evolutionary approaches are presented. In Section 2.2, many-objective optimization problems are explained with particular emphasis on the challenges they bring, and a literature overview is provided together with a classification. Section 2.3 contains a discussion about co-evolutionary algorithms. Finally, the performance assessment of Pareto front approximations is reviewed in Section 2.4.

2.1. Multi-Objective Optimization Problems (MOPs)

Multi-objective optimization problems (*MOPs*) include two or more objective functions to be minimized (or maximized) at the same time. For each feasible solution, the objective function becomes a vector function representing values for each objective. Two spaces are considered in MOPs: n-dimensional space of decision variables and m-dimensional space of objective functions (Branke *et al.*, 2008). A point in the former space represents a solution, and in the second, it gives a specific point in terms of objective function values that display the quality of the corresponding solution.

The concept and purpose of optimality in MOPs is to find good compromises (or trade-offs) among the objectives rather than searching for a single global optimal point (Marler and Arora, 2004). Unlike a single-objective optimization problem, where the optimal solution is usually unique, it is almost impossible to have a unique solution with the best performance on all objectives simultaneously. There is no single optimal solution in a MOP, but there is a set of compromised solutions that reflect different

trade-offs throughout the objective functions of the problem. These solutions are called *efficient solutions* or *Pareto optimal solutions*. It is common to see conflicts between different objectives, and a solution that performs well on one objective may not perform well on other objectives. Even so, only one of the solutions in the Pareto optimal solution set should ultimately be chosen. This fact makes it difficult for a decision maker (DM) to evaluate and compare different solutions. As a result, MOPs involve two vital tasks: the optimization task of finding Pareto optimal solutions and the decision making task of choosing a single most preferred solution.

Despite the challenges and additional tasks, many real-life operations research (OR) problems involve multiple non-trivial objectives and it is beneficial to model the problems not only considering a single objective, but also optimizing multiple objectives each considering different goals simultaneously. As a consequence, multi-objective optimization is a relatively old, large and established field of research that dates back to the early days of OR. Nevertheless, studies in MOPs usually involve two or three objective functions, and currently available techniques operate appropriately for relatively few objectives.

In this section, Pareto terminology and related definitions adopted in multi-objective optimization are explained in Section 2.1.1. In Section 2.1.2, the classification and methods in multi-objective optimization are presented. Finally, multi-objective evolutionary algorithms (MOEAs) are introduced in Section 2.1.3.

2.1.1. Pareto Terminology and Related Definitions

The MOP, which assumes minimization objectives, is formulated in Equation 2.1.

$$\min \vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), ..., f_m(\vec{x})] \tag{2.1a}$$

$$\text{subject to} \quad \vec{x} \in X \tag{2.1b}$$

where $\vec{x} = [x_1, x_2, ..., x_n]$ is the vector of decision variables, $n \in \mathbb{Z}^+$ is the total number of decision variables, $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$, $i = 1, ..., m$ are the objective functions of the problem. Given two decision variable vectors $\vec{u}$, $\vec{v} \in \mathbb{R}^n$, it is said that $\vec{u}$ *Pareto dominates* $\vec{v}$ denoted by $\vec{u} \preceq \vec{v}$ if and only if $\vec{u}$ is partially less than $\vec{v}$ i.e. $\forall i \in \{1, ..., m\} : f_i(u_i) \leq f_i(v_i) \wedge \exists j \in \{1, ..., m\} : f_j(u_j) < f_j(v_j)$. Accordingly, $\vec{u}$ *Pareto strictly dominates* $\vec{v}$ denoted by $\vec{u} \prec \vec{v}$ if and only if $\vec{u}$ is less than $\vec{v}$ in all objectives i.e. $\forall i \in \{1, ..., m\} : f_i(u_i) < f_i(v_i)$. The former concept provides a basis for determining the superiority of a solution over other solutions. Within a solution set, *nondominated solution set* is a set of all solutions that are not dominated by any other member of the solution set.

In Figure 2.1, the Pareto dominance concept is illustrated in the two-dimensional objective space. As solution x_1 is no worse than solution x_2 in all objectives and it is strictly better in at least one objective, it is said that x_1 dominates x_2 (i.e. x_2 is dominated by x_1). Other conclusions that can be made from the figure are x_5 is dominated by all, x_2 and x_4 cannot dominate each other but both are dominated by x_3, and x_1 and x_3 cannot dominate each other.

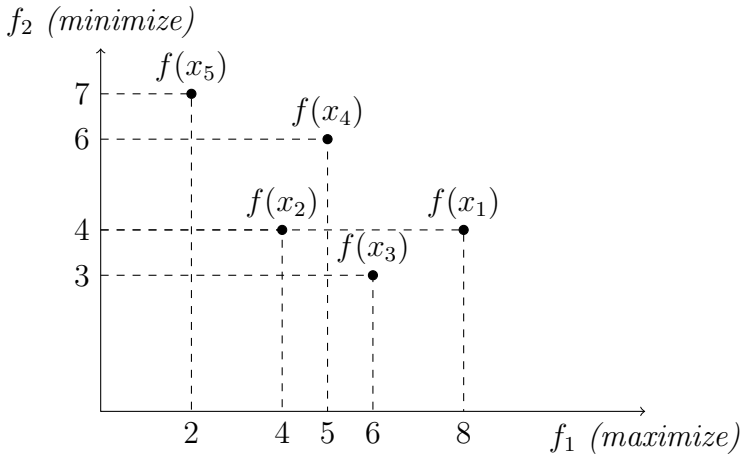

Figure 2.1. Illustration of the Pareto dominance concept.

It is said that a solution $x^* \in X$ is *weakly Pareto optimum (weakly efficient solution)* if there is no other $x \in X$ such that $x \prec x^*$. Similarly, a solution $x^* \in X$ is *[strictly] Pareto optimum ([strictly] efficient solution)* if there is no other $x \in X$ such

that $x \preceq x^*$. The vectors in the objective (criterion, outcome) space that correspond to efficient solutions are called *nondominated points*. In other words, efficiency and dominance are counterparts of each other in the decision and objective spaces, respectively. The relationship between decision and objective space is shown in Figure 2.2.

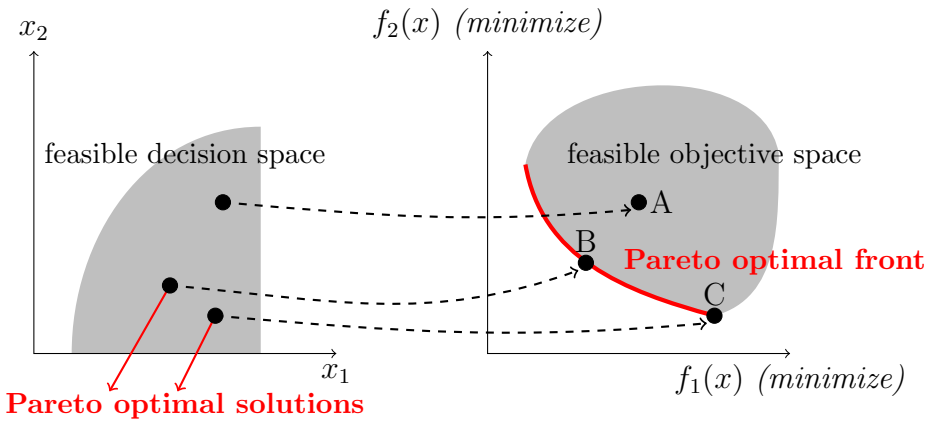

Figure 2.2. Relationship between the decision and objective space.

The set of all Pareto optimum points of the entire feasible decision space is called the *Pareto optimal set* $\mathcal{P}^*$, or the efficient set $\mathcal{X}_E$. The boundary defined by the set of all points mapped from the Pareto optimal set in the objective space forms the *[true] Pareto Front* ($\mathcal{PF}$) or the nondominated set $\mathcal{Y}_{ND}$. In other words, $\mathcal{P}^*$ and $\mathcal{X}_E$ are defined in the decision space whereas $\mathcal{PF}$ and $\mathcal{Y}_{ND}$ are their corresponding image in the objective space. It should be noted that multiple efficient solutions may correspond to the same nondominated point. In this case, it is sufficient to find only one of these efficient solutions.

The goal in MOPs is to determine the Pareto optimal set or at least as many Pareto optimum solutions as possible. Methods in multi-objective optimization target to find a set of solutions, called *Pareto approximation* that is as close as possible to the Pareto optimal front and as diverse as possible. These goals can be seen in the objective space in Figure 2.3.

Assuming the efficient set $\mathcal{X}_E$ is nonempty and bounded, the lower and upper bounds in the values of the nondominated set $\mathcal{Y}_{ND}$ is acknowledged by the ideal and

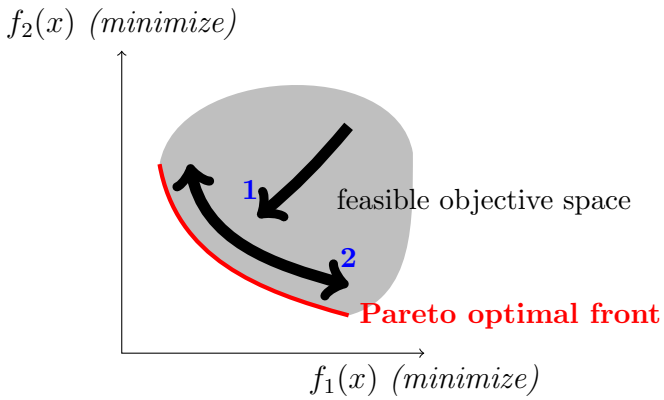

Figure 2.3. Goals of multi-objective optimization.

nadir point, respectively. The ideal (or utopian) point (or vector) $\vec{y}^I = (y_1^I, ..., y_m^I)$ is defined in the objective space as according to the individual minima of each objective. In other words, it is the best possible value for each criterion among the efficient set and thereby the feasible set. Ideal point can be constructed by optimizing each of the criteria separately and shown in Equation 2.2.

$$y_i^I := \min_{x \in \mathcal{X}_E} f_i(\vec{x}) \quad i = 1, ..., m \tag{2.2}$$

On the contrary, the nadir point (or vector) $\vec{y}^N = (y_1^N, ..., y_m^N)$ is defined in the objective space as according to the individual maxima of each objective. In other words, it is the worst possible value for each criterion among the efficient set and it is shown in Equation 2.3.

$$y_i^N := \max_{x \in \mathcal{X}_E} f_i(\vec{x}) \quad i = 1, ..., m \tag{2.3}$$

It must be noted that the nadir point should not be confused with the worst feasible solution. Ideal and nadir points are displayed in Figure 2.4.

Both ideal and nadir points are imaginary points defined in the objective space and they do not necessarily correspond to a real solution in the decision space. Finding both points is critical as various approaches use these points to scale each objective of the Pareto front. However, it might be the case that finding these points, particularly

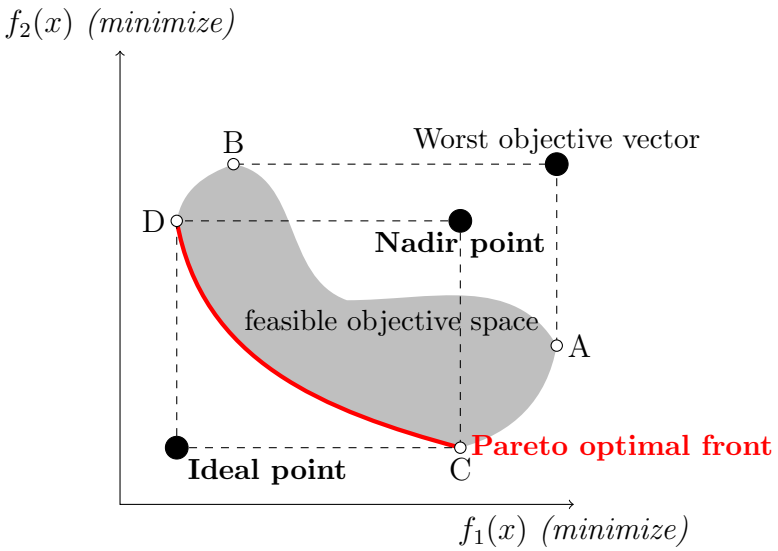

Figure 2.4. Ideal and nadir point in the objective space.

the nadir point is not straightforward, especially for problems with more than two criteria. Naturally, the convex hull of the nondominated set can be defined in the objective space. Nondominated points that cannot be represented as a convex combination of other nondominated points are called *extreme nondominated points*.

Another important concept regarding the nondominated set is supported and unsupported nondominated points. Nondominated points corresponding to an optimal solution for some single-objective problem obtained by any scalarization using weighted sums of the form defined in Equation 2.4 are called *supported nondominated points*, otherwise *unsupported nondominated points*.

$$\min_{\vec{x} \in X} \sum_{i=1}^{m} \lambda_i f_i(\vec{x}) \tag{2.4}$$

where $\sum_{i=1}^{m} \lambda_i = 1, \forall \lambda_i > 0$. It should be noted that extreme nondominated points are always supported. Consequently, definitions of extreme supported efficient, nonextreme supported efficient and unsupported efficient solutions can be made as well in the decision space.

The optimal solution for one of the criteria constitutes an *individual minimum* with respect to that criterion. Individual minima are weakly efficient solutions, but

they are not necessarily strictly efficient. On the other hand, a solution that is optimum for a lexicographic ordering of the criteria is called *lexicographic optimal solution*. These solutions are extreme supported [strictly] efficient solutions. These concepts and definitions are illustrated in Figure 2.5.

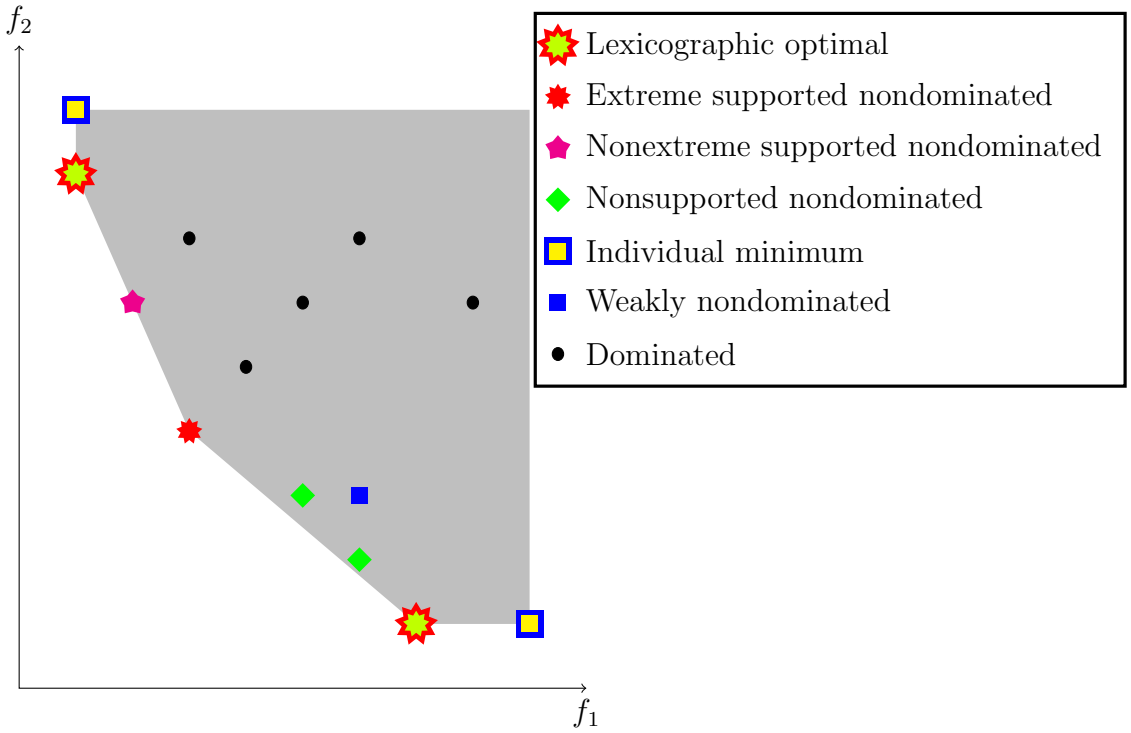

Figure 2.5. Types of solutions in the objective space.

Distances between points in the objective space are measured using L_q-metric. The L_q-distance between two vectors $\vec{x}$, $\vec{y} \in \mathbb{R}^m$ is measured in Equation 2.5.

$$\|\vec{x} - \vec{y}\|_q = \left[\sum_{i=1}^{m} |x_i - y_i|^q\right]^{1/q} \tag{2.5}$$

In the presence of a weight vector $\vec{w} = [w_1, ..., w_m]$, the weighted L_q-distance between two vectors $\vec{x}$, $\vec{y} \in \mathbb{R}^m$ is measured in Equation 2.6.

$$\|\vec{x} - \vec{y}\|_{\vec{w},q} = \left[\sum_{i=1}^{m} w_i|x_i - y_i|^q\right]^{1/q} \tag{2.6}$$

Different distance metrics can be obtained based on different values of the q parameter. The most commonly used distance metrics are rectilinear, Euclidean, and Tchebycheff. Their formulas are provided in Equation 2.7 to 2.9 in the respective order.

$$\text{Rectilinear:} \qquad L_1 = \sum_{i=1}^{m} |x_i - y_i| \tag{2.7}$$

$$\text{Euclidean:} \qquad L_2 = \sqrt{\sum_{i=1}^{m} |x_i - y_i|^2} \tag{2.8}$$

$$\text{Tchebycheff:} \qquad L_\infty = \max_{i=1,\ldots,m} |x_i - y_i| \tag{2.9}$$

2.1.2. Classification and Methods

The classification of the multi-objective optimization methodologies can be made based on the sequence of its phases (Branke *et al.*, 2008). Multi-objective optimization has three phases: model building, optimization, and decision making (or preference articulation). Once a multi-objective model has been built, the next step should be either an optimization phase or a decision making phase.

When the decision making process is put before optimization, this is called "*a priori*" method. The advantages of this method are that the problem can be reduced to a single-objective problem and optimization does not require much time and effort. However, it is not easy articulate preference in advance and the resulting solutions may not be the same as the solution the DM would choose from the Pareto optimal set. A priori methods include value function method (Keeney and Raiffa, 1976), lexicographic ordering (Fishburn, 1974), goal programming (Charnes and Cooper, 1957; Charnes *et al.*, 1955) and fuzzy multi-objective optimization methods (Rommelfanger and Słowiński, 1998).

When the decision making process and optimization interferes with each other, this is called "*an interactive*" method. In interactive methods, the decision maker plays an important role and preference information is gradually provided so that the

most preferred solution can be found. In interactive methods, three types of preference information are specified. These are trade-off information (Geoffrion *et al.*, 1972; Zionts and Wallenius, 1976), reference points (Wierzbicki, 1980), and classification of objective functions (Larichev, 1992).

When the optimization phase comes before the decision phase, this method is called "*a posteriori*". A posteriori methods are also called the methods for generating Pareto optimal solutions. In these methods, a satisfactory representation of Pareto optimal solutions is produced and presented to the decision maker who will choose one of these solutions as the most preferred final solution. The rationale behind a posteriori approaches is that it is much easier to choose the most preferred solution when a set of different Pareto optimal solutions is observed. The difficulties encountered in a posteriori methods are that the generation process of the Pareto front is often computationally expensive and difficult, and it can also be challenging to represent and display the Pareto front in a comprehensible way. A posteriori methods include weighted metrics (Zeleny, 1973), achievement scalarizing functions (Wierzbicki, 1980) and approximation methods (Payne and Carlson, 1975; Polak, 1976). Apart from these, weighting method (Gass and Saaty, 1955; Zadeh, 1963) and ϵ-constraint method (Haimes *et al.*, 1971) can also be used as a posteriori methods. More examples and research can be found in all these categories (Coello *et al.*, 2007).

When it comes to the methods of multi-objective approaches, it is worthwhile to mention exact methods and approximation algorithms, although the methodology used in this thesis follows an evolutionary approach. For instance, there exist multi-objective versions for enumeration techniques such as branch and bound or dynamic programming. In these methods, ideal and nadir points can be used as lower and upper bounds for the nondominated point set. Nevertheless, since these two points might be far away from the nondominated point set, they are inadequate to bound the search area. Instead, a set of points called bound sets can be used to reduce the search area. A detailed discussion about the bound sets can be found in Ehrgott and Gandibleux (2007). In approximation algorithms, a set of points, which is expected to represent the nondominated set and provide useful information, is generated. This set can consist

of verified nondominated points which are obtained by an exact method Ruzika and Wiecek (2005), or approximated nondominated points which are found by a heuristic approach Ehrgott and Gandibleux (2004).

On the other hand, it is impractical to calculate the entire Pareto optimal set in a posteriori methods, since it is very costly to enumerate the entire Pareto optimal set by obtaining a single solution in a single run with the above-mentioned techniques. This fact leads to the use of evolutionary methods for MOPs. Evolutionary algorithms are well suited for solving MOPs as they are population based approaches. They are very attractive solution techniques since they can simultaneously search different regions of the objective space and find a diverse set of efficient solutions for any types non-convex, discontinuous, and multi-modal problems (Coello *et al.*, 2007).

2.1.3. Multi-Objective Evolutionary Algorithms (MOEAs)

Extensive research has been conducted on evolutionary approaches to solve MOPs (Coello, 2000; Konak *et al.*, 2006; Zitzler *et al.*, 2004). Multi-objective evolutionary algorithms (MOEAs), based on natural evolution principles, are widely applied to solve MOPs. They are even referred as "*the*" method to explore the Pareto optimal front in MOPs (Deb, 2001). The main reason for using MOEAs in the field of evolutionary multi-objective optimization (EMO) is their capability to find multiple Pareto optimal solutions in a single run (Jones *et al.*, 2002). While working with a population of solutions, they can offer the decision maker a set of alternatives simultaneously. As a result, the field of EMO has grown rapidly in recent decades. Since MOEAs are a type of the metaheuristic approaches, the attempt is to find an acceptable Pareto approximation. Recently, MOEAs have also been used to find only a part or a specific region of the Pareto optimal set.

Similar to conventional evolutionary approaches, solutions in the MOEAs are represented by chromosome and genes to construct their genotypes in decision space and phenotypes in objective space. Throughout the evolutionary process, different sets of solutions establish populations (generations), parent and child (offspring) populations.

MOEAs also include genetic operators such as selection, recombination (crossover), mutation, immigration, local improvement, and repair.

The differences between a generalized evolutionary algorithm (EA) and a MOEA are shown in Figure 2.6. The main difference is observed in fitness function evaluation (Task 2). MOEA computes multiple fitness functions, and sometimes an extra procedure is required to transform fitness vectors into a quantitative evaluation measure (Task 2a). The rest of the MOEA is structurally the same as a generalized EA. The most characteristic differences between MOEAs are often the selection of solutions of the evolving population (Task 5).

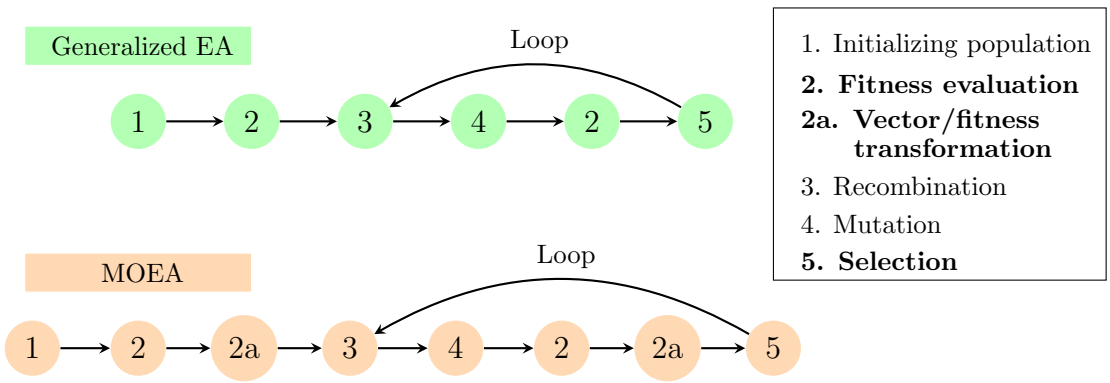

Figure 2.6. Task decomposition in generalized EA and MOEA (Adapted from Coello *et al.*, 2007).

A successful MOEA should be able to achieve a Pareto approximation as close as possible to the true Pareto front (Konak *et al.*, 2006). Preferably, members of the Pareto approximation should be a subset of the Pareto optimal set. In addition, members of the Pareto approximation need to be distributed homogeneously and diversely in order to provide a comprehensive representation of trade-offs to the decision maker. In other words, the Pareto approximation should represent the entire true Pareto front (Coello *et al.*, 2007). Numerous techniques have been proposed to ensure diversity in MOEAs such as fitness sharing (Deb and Goldberg, 1989), nearest neighbor approach (Zitzler *et al.*, 2001), crowding distance (Deb *et al.*, 2002), restricted mating (Lu and Yen, 2002), and weight vectors (Ishibuchi *et al.*, 2003). In Zitzler *et al.* (2000), the performances of MOEAs are evaluated in terms of the success of achieving a Pareto

approximation as close as possible to the true Pareto front, and diversity concerns not only avoiding local optima but also achieving the entire Pareto front.

The studies in this thesis adopt a posteriori approach and aim to find as many solutions as possible on the Pareto front, leaving the remaining multi-criteria decision making process to the DM with the chance to choose a solution from a set that comprehensively reflects the trade-offs between different objectives. In this way, no prior preference information from the DM is required.

2.2. Many-Objective Evolutionary Algorithms (MaOEAs)

Multi-objective optimization problems (MOPs) with more than three objectives are referred as *many-objective optimization problems* (*MaOPs*). MaOPs appear widely in many real-world applications, engineering design problems in particular constitutes a natural domain for this field. *Many-objective evolutionary algorithms* (*MaOEAs*) constitute a new and key field of research for modern day evolutionary computation, and have received a sustained attention in the EMO community. While some pioneering studies can be encountered in the early 1990s, the majority of the research has been done in the last decade. One of the earliest algorithms applied to MaOPs is by Fonseca and Fleming (1998).

In this section, the sources of challenges in MaOPs are introduced next. Capabilities of the existing approaches to deal with these encountered challenges are discussed in Section 2.2.2. A brief literature survey is presented in Section 2.2.3. MaOEA categories are reviewed in Section 2.2.4. Reference set based MaOEAs are described in Section 2.2.5 and a subset of such approaches involving adaptation the reference set during the evolutionary process is specifically explored in Section 2.2.6.

2.2.1. Sources of Challenges

By definition, existing methods for MOPs can also be applied to MaOPs. Hence, evolutionary approaches can be seen as a natural solution method for MaOPs, too.

Unfortunately, during their implementation, especially for those based on Pareto dominance, researchers have faced a number of serious challenges that are presented throughout this section. Existing reviews on this topic present current challenges and the studies to address these challenges (Chand and Wagner, 2015; Ishibuchi *et al.*, 2008; Jaimes and Coello, 2015; Von Lücken *et al.*, 2014; Wagner *et al.*, 2007).

Increase in Nondominated Population. When there is a large number of objectives, more solutions become nondominated across the entire feasible set. When the population of a MaOEA consists largely of Pareto optimal solutions, the equivalence of the available solutions in the current population complicates the selection phase and obscures finding a good direction to investigate a successful Pareto approximation. This fact definitely undermines evolutionary approaches using selection procedures based on Pareto dominance. Thus, the overall search process slows down. To overcome this problem, Sato *et al.* (2010) propose Pareto partial dominance as an alternative to the Pareto dominance approach. Aguirre and Tanaka (2009) recommend partitioning the objective space into subspaces and searching in each subspace. The ϵ-domination principle used by Laumanns *et al.* (2002) also aims to address the problem of a large nondominated set of solutions. In this approach, a solution is said to dominate another solution if their objectives are within ϵ-distance. This definition allows for a large number domination relationship between the solutions and creates a lower, finite number of solutions in Pareto the optimal set.

Computational Efficiency in Assessment of Diversity Measures. As the number of objectives increases, not only the number of Pareto optimal solutions increases but also each solution is represented by a higher dimensional vector in the objective space. This results in MaOEAs to be computationally expensive. Various performance metrics require exponentially more computational effort when higher dimensional points are compared to each other. For example, it becomes computationally expensive to identify neighbors of a particular solution in the objective space, such as the case when calculating crowding distance (Deb *et al.*, 2002). To solve this problem, faster and more accurate algorithms have been proposed. Deb and Jain (2013) propose a reference point based approach where a number of well-spread reference points are used to

find a set of well-spread Pareto optimal solutions. The ϵ-domination principle can also be used to obtain a well-spread set of solutions, as computational complexity increases only linearly with the increase in the number of objectives.

Recombination. Recombination operators used to produce new offspring solutions are crucial for the success of evolutionary approaches, as they are considered key search operators. When only a finite number of solutions are found in a high-dimensional objective space, they tend to be far apart from each other in the objective space, that is, if conventional recombination schemes are used, they might generate an offspring far from the Pareto front (Jaimes and Coello, 2015). Conventional recombination operators developed for problems with fewer objectives become inefficient and insufficient, as the aim is to find the Pareto front that is likely to consist of extreme points in the solutions space. Even though the parent solutions are close to the Pareto optimal front, their offspring need not to be close to the front. To handle this issue, special recombination operators may be required and developed. For example, Deb and Jain (2013) propose a special recombination scheme called simulated binary crossover. In short, attention should be paid to the choice of the recombination operator and the selection of candidate parents to participate in the recombination scheme.

Visualization. Visualization is an important issue in the decision making stage. The DM should be able to grasp what is offered to them and choose according to their preferences. When the number of objectives is more than three, it naturally becomes difficult to visualize the objective space and therefore becomes difficult to decide on a final solution. Possible way to eliminate the visualization problem might be to visually switch between given objectives. In this way, the decision maker can shift between objectives and view different trade-offs, and thereby not get overwhelmed by the large number of objectives. Other approaches to assist the DM are to apply further graphical methods such as parallel coordinates plot, heat maps radial visualization, decision maps, and geodesic maps, etc. Reducing the number of objectives or mapping objective vectors to a lower dimensional space can be beneficial approaches.

2.2.2. Current Approaches

In this section, approaches to address the problems encountered in MaOPs are briefly outlined. Although different classification alternatives can be found to group these methodologies, they are divided into three categories in this section as methods that include scalability improvement, preference information, and objective reduction.

Scalability Improvement. Scalability improvement in many-objective optimization includes ideas that aim to improve existing evolutionary multi-objective approaches to achieve successful solution techniques for MaOPs. There are numerous improvement methods proposed in the literature which are principally based on existing MOEAs.

To improve the scalability of MOEAs, their capabilities for selection should be improved. Sato *et al.* (2007) modify the Pareto dominance concept to reduce the number of nondominated solutions in each generation by expansion or contraction of the dominance area. In Corne and Knowles (2007) and Sülflow *et al.* (2007), alternative rankings such as average ranking and ϵ-preferred are assigned to nondominated solutions namely, respectively.

Another idea is to use different fitness evaluation mechanisms instead of Pareto dominance. Indicator-based evolutionary algorithm in Wagner *et al.* (2007) use indicator functions such as hypervolume to evaluate each solution. Purshouse and Fleming (2003) use a number of different scalarizing functions for fitness evaluation. It should be noted that there are numerous examples of research employing scalability improvement since almost every evolutionary methodology proposed for MaOPs is based on an existing algorithm developed for MOPs.

Incorporation of Preference Information. The design goal of evolutionary algorithms is to achieve a set of nondominated solutions that successfully approximates the entire Pareto front. However, as mentioned earlier, the number of solutions required for a good approximation increases exponentially with the number of objectives. Therefore, some researchers suggest utilizing the decision maker's preference to focus on a

particular region of the Pareto front. The idea is simply to incorporate preference information interactively throughout the course of optimization process to focus on the region of the decision maker's interest, hence avoiding to evaluate an enormous number of solutions. In addition, preference information can be used to rank incomparable nondominated solutions and to deal with the large number of objectives.

Fonseca and Fleming (1998) used preference information to compare solutions and extend ranking mechanisms. In Fleming *et al.* (2005), preference information is used to cope with conflicting objectives; and in Deb and Sundar (2006), the reference points are extracted from the decision maker. Thiele *et al.* (2009) propose a variant of the indicator-based evolutionary algorithm, where preference information is incorporated through an achievement scalarization function. In López-Jaimes and Coello (2014), preference information based on Tchebycheff achievement function is used to analyze a real-world many-objective design problem.

Objective Reduction Approaches. The difficulties faced in MaOEAs may vanish if important objectives can be chosen and search is focused only on the dimensions specified on them. Deb and Saxena (2005) propose a method to reduce the number of objectives based on principal component analysis. The aim is to remove unnecessary objectives while maintaining the shape of the Pareto front in the reduced objective space. Brockhoff and Zitzler (2006, 2007) define two objective reduction algorithms based on the idea to remove an objective when it does not change the Pareto dominance relationship between solutions.

2.2.3. An Overview of Problem Domains in MaOEA Studies

In this section, a brief overview is provided for the related literature with respect to problem domains by grouping a non-exhaustive set of studies in two tables. In both tables the number of objectives, application area and main type of methodology are displayed. Design problems are a natural problem domain for many-objective approaches. As such, the first table, Table 2.1 is devoted to studies that try to solve a many-objective design problem.

Table 2.1. Literature review, design problems.

Reference	**Objective count**	**Test problem**	**Methodology**
Cvetković and Parmee (2002)	9 - 15	Conceptual airframe design	preference-based MOEA
Fleming *et al.* (2005)	8	Flight control system	MOGA-G, PDA
Deb and Sundar (2006)	2 - 10	ZDT, DTLZ, Welded Beam Design, Spring Design	R-NSGA-II
Justesen and Ursem (2010)	6 - 8	Centrifugal Pump Design	MODCODE
Wickramasinghe *et al.* (2010)	5	Designing Airfoils	Reference point based PSO
Sindhya *et al.* (2011)	5	Pollution monitoring station problem	PIE
Narukawa and Rodemann (2012)	6	Hybrid Car Controller Optimization	SMS-EMOA, D-PBI, D-TBC
Deb and Jain (2013)	3 - 20	Crash-Worthiness Design, Car Cab Design	NSGA-III
López-Jaimes *et al.* (2013)	3 - 6	Space Trajectory Design	NSGA-II and NSGA-Cheby
Sinha *et al.* (2013)	5 - 11	Storm drainage, car side-impact	NL-MVU-PCA
Asafuddoula *et al.* (2014)	3 - 15	DTLZ, WFG, car side impact, water resource management, aircraft design	I-DBEA
López-Jaimes and Coello (2014)	6	Airfoil Optimization	Reference point based NSGA-II
Mkaouer *et al.* (2014)	15	Automating Software Refactoring	NSGA-III
Bandyopadhyay and Mukherjee (2014)	6	Factory Shed Truss Optimization	α-DEMO

Table 2.2. Literature review, test problem suites.

Reference	**Objective count**	**Test problem**	**Methodology**
Deb and Saxena (2005)	5 - 30	DTLZ	NSGA-II with PCA
Deb and Saxena (2006)	3 - 50	DTLZ	NSGA-II with PCA
Praditwong and Yao (2006)	2 - 8	DTLZ	TAA
Brockhoff and Zitzler (2007)	5 - 9	DTLZ	δ-MOSS, k-EMOSS
Wagner *et al.* (2007)	3 - 6	DTLZ	ϵ-MOEA, MSOPS, IBEA, SMS-EMOA
Zhang and Li (2007)	2 - 4	DTLZ, WFG	MOEA/D
Thiele *et al.* (2009)	5	ZDT	PBEA
Bader and Zitzler (2011)	2 - 50	DTLZ, WFG	HypE
Deb and Jain (2013)	3 - 15	DTLZ	NSGA-III
Jain and Deb (2013)	3 - 8	DTLZ	A^2-NSGA-III
Wang *et al.* (2013)	2 - 10	WFG	PICEA-g
Jain and Deb (2014)	3 - 15	DTLZ	A-NSGA-III
Mohammadi *et al.* (2014)	4 - 10	DTLZ	R-MEAD2
Wang *et al.* (2014)	2 - 10	DTLZ	Two_Arc2
Wang *et al.* (2015)	2 - 7	WFG	PICEA-w
Cheng *et al.* (2016)	3 - 10	DTLZ	RVEA
Goulart and Campelo (2016)	5 - 20	DTLZ	R-NSGA-II, PBEA
Asafuddoula *et al.* (2017)	3 - 15	DTLZ	g-DBEA
Liu *et al.* (2017)	6 - 15	DTLZ	RPEA
Liu *et al.* (2020)	5 - 15	DTLZ, WFG	AnD

Most of the studies in the area, on the other hand, are methodological studies that try to develop improved solution methods for MaOEA. Nearly all of these use some scalable test problem suites constructed to perceive the performance of different evolutionary algorithms. These conventional test sets do not rely on real-world problems; however, they use well-defined continuous mathematical functions as objective functions. DTLZ (Deb *et al.*, 2005), ZDT (Zitzler *et al.*, 2000), WFG (Huband *et al.*, 2006) are examples in this class and are exercised extensively in many-objective optimization research. They are preferred because they are scalable to any number of objectives and decision variables, and the exact positions and shapes of the resulting

Pareto optimal fronts are known in advance. These studies are displayed in Table 2.2. An important remark is that there are very few applications for OR problems. The literature on combinatorial many objective problems will be surveyed in Chapter 3.

2.2.4. Categories of MaOEAs

MaOEAs can be broadly categorized into seven classes based on the main idea used in their algorithms (Li *et al.*, 2015). In the first class, the dominance rule is modified to distinguish the excessive amount of nondominated solutions and direct the selection pressure towards the Pareto front. Such approaches can be called as relaxed dominance-based approaches. The ϵ-dominance rule suggested by Laumanns *et al.* (2002) is an example studied and applied in this thesis. The second class, called diversity-based approaches, primarily addresses the maintenance of a diverse population by actively promoting density-based criteria to support the primary dominance-based criterion in the evolutionary selection. In the third class, called indicator-based approaches, performance indicator values used to evaluate the approximation sets are used to drive the evolutionary process. Hypervolume estimation algorithm (HypE) proposed by Bader and Zitzler (2011) uses the hypervolume indicator, the only metric known to be strictly monotonic to Pareto dominance. Fourth, it is the class of preference-based approaches, where the decision makers' preferences are incorporated in the search process to focus on the respective subset of the Pareto front. The fifth class, called dimensionality reduction approaches, aims to deal with MaOPs with redundant objectives. This is done by reducing the number of objectives and converting the original MaOPs into another problems with fewer objectives but with a similar Pareto set. The sixth class contains aggregation-based or decomposition-based approaches, where the objective functions of MaOPs are aggregated into a single-objective function and the Pareto front is decomposed into several sub-regions. Zhang and Li (2007), which uses alternative scalarizing functions such as weighted sum, weighted Tchebycheff, and boundary intersection methods, is an example of such an algorithm. Reference set-based approaches, the final class of MaOEAs in which the approach proposed in this thesis can be considered, are examined in detail in the next section.

2.2.5. Reference Set Based MaOEAs

The final class of MaOEAs is one of the most promising and it is called reference set based approach. In this approach, a set of reference points or solutions is used to guide the search process and measure the quality of the solutions. As a result, the quality of the reference set becomes an important issue as it is responsible for ensuring convergence and maintaining a diversified solution set. In other words, the balance between convergence and diversity is sustained by the reference set. A good quality reference set should be less sensitive to the problem type and Pareto front geometry and should be scalable to MaOPs to keep the algorithm's performance successful despite the increase in the number of objectives.

Pioneering approaches in the literature often adopt only a single reference point, representing the decision maker's preferred ideal solution. In Wierzbicki (1980), the aim is to achieve Pareto optimal solutions closest to a particular reference point, while in Deb and Sundar (2006) the goal is to obtain a set of solutions as close as possible to the reference point. Mohammadi *et al.* (2012) integrate decomposition methods with reference point based approaches to search preferred regions of the Pareto front. On the other hand, when the main interest is obtaining an approximation of the entire Pareto-front for *a posteriori* decision making, as is the case in this thesis, a set of well-distributed reference points can be employed.

In Figueira *et al.* (2010), after approximating the boundaries of the Pareto front, ideal and nadir points, a set of reference points is generated and solutions close to each reference point are found in parallel by the decomposition of the objective space using reference points. The fitness values of individuals are evaluated using achievement scalarizing functions.

In general, there are two issues with reference set based approaches. The reference set based methods differ from each other in how they construct the reference set and how they measure the quality of the solutions relative to the reference set during evolution. Regarding the first problem, some algorithms use real solutions in the refer-

ence set, where some members of the population are treated as reference points, while other algorithms use a virtual reference set that is used to create a set of virtual points in the objective space, i.e. an ideal front to lead the search process. Although more information about the population is included in a real reference set, virtual reference set approaches are more common as their locations and distributions can be better organized. The second issue is usually about how solutions in the population are associated with reference points. Different distance metrics and scalarizing functions such as Euclidean and Tchebycheff have been adopted by different algorithms.

In NSGA-III proposed by Deb and Jain (2013), the entire objective space is expected to be covered by a hyperplane that is built based on the current population. Reference points are evenly distributed over this hyperplane for diversity maintenance. After normalizing the objective function values of current population members, the reference points are placed on this normalized hyperplane. After using the conventional Pareto rule of domination, a niche-preservation strategy is used to measure the fitness of the individuals. In this algorithm, the hyperplane, which includes all reference points, evolves as the base population evolves whereas the reference points on the hyperplane are fixed.

In MOEA/D proposed by Zhang and Li (2007), a multi-objective problem is decomposed into a number of single-objective problems identified using the same scalarizing function with different weight vectors. While this algorithm is categorized as a decomposition based algorithm, there is a strong analogy between its weight vectors and the reference points. The solution corresponding to each weight vector is generated by recombining a pair of parent solutions randomly selected from the neighborhood. The new solution is compared with the existing solution of the corresponding weight vector and the existing solutions of its neighbors. It replaces those with a worse scalarizing function value in terms of the corresponding weight vector.

Two archive algorithm (TAA) and its modified version (Two_Arch2) are proposed by Praditwong and Yao (2006) and Wang *et al.* (2014), respectively. An important aspect of these algorithms is that they explicitly distinguish two goals of evolutionary

multi-objective optimization, convergence and diversity. In both algorithms, the solution set of each generation is divided into two groups, a convergence archive and a divergence archive. The convergence archive only includes nondominated solutions that once dominated existing archive members, and the diversity archive contains the solutions with the greatest distances to convergence archive members. In this approach, the solutions of the convergence archive can be seen as adaptive real reference points that are updated online.

2.2.6. Adaptation Methods in the Reference Set Based MaOEAs

The use of adaptive reference points is recommended to deal with irregular Pareto fronts and different scales of objectives. Irregular Pareto fronts include degenerate fronts, discontinuous fronts, inverted fronts and highly nonlinear fronts. The difficulties faced by the decomposition-based and reference set based methods on irregular Pareto fronts have been pointed out by several researchers (Hua *et al.*, 2018; Ishibuchi *et al.*, 2019; Liu *et al.*, 2019). It is claimed that sampling reference vectors from a uniform set of points inherently assumes that the Pareto front is bounded by the reference vectors, and it is nondegenerate, continuous, smooth, and without significant nonlinearity. On the other hand, methods with dynamic reference points can be challenged as their convergence speed may be degraded (Giagkiozis *et al.*, 2013) and added parameters may cause additional complexity. As a result, it is crucial to develop effective positioning and adaption strategies for the reference set.

In RPEA by Liu *et al.* (2017), reference points are updated only in some iterations and the frequency of the reference set regeneration is determined by a predefined parameter value. Reference points are formed as superior hypothetical solutions using nondominated solutions with the largest crowding distances. As a result, the reference points provide up-to-date information of the Pareto approximation.

In the reference vector guided evolutionary algorithm (RVEA) proposed by Cheng *et al.* (2016), two sets of reference vectors are used. One set maintains an even distribution over the objective space and the other set is adaptively adjusted to the normalized

values of the solutions. It is emphasized that an adaptive reference set strategy is required to deal with problems with different scales of objectives and irregular Pareto fronts such as degenerate, discontinuous, inverted and highly nonlinear fronts.

In generalized decomposition-based evolutionary algorithm (g-DBEA) proposed by Asafuddoula *et al.* (2017), reference vectors are periodically adapted based on the information collected during a learning period. Two sets of reference vectors are used: an active set consists of adaptive reference vectors and an inactive set contains discarded vectors that have the chance to return to the active set throughout evolution. Adaptive reference point mechanisms are applied to an indicator based evolutionary algorithm in Tian *et al.* (2017) and an entropy based evolutionary algorithm in Zhou *et al.* (2018).

It is noteworthy to explore two algorithms proposed by the NSGA-III authors using adaptive reference point sets. In the adaptive NSGA-III (A-NSGA-III) by Jain and Deb (2014), the reference point set structure of the NSGA-III is converted into an adaptive structure so that the algorithm is able to find a more suitable distribution of the reference points during the evolutionary process. The reference points are adjusted according to their association with the solutions in each iteration. The adaption method has two stages: adding new reference points close to crowded reference points with high niche counts and occasionally deleting the reference points with zero niche counts. In the addition phase, a simplex is created around a chosen crowded reference point and at most m number of new reference points are added which are believed to share the niche count with the current reference point. Adding new reference points continues until no reference point with niche count greater than one is left.

In a similar adaptive algorithm called A^2-NSGA-III proposed by Jain and Deb (2013), it is claimed that a more efficient approach is applied to create new reference points. In this approach, the followed simplex structure results in the addition of fewer new reference points. Creation of new reference points is only allowed when niche counts of crowded reference points remain constant for a predefined number of generations. There is also a limit on the maximum number of reference points and as a result, deletion is triggered more frequently.

2.3. Co-evolutionary Algorithms

Co-evolutionary algorithms, first developed in the 90s, provide a way to decompose and solve complex problems consisting of multiple interrelated subproblems, while respecting the interdependency between these subproblems. As an extension of classical evolutionary algorithms (Coello and Sierra, 2003), co-evolutionary algorithms imitate the process of symbiotic evolution in nature, where different species evolve in a way to adapt to each other. The problem is decomposed into subproblems and multiple populations are used to optimize different subproblems simultaneously. During evolution, populations of different species can interact with each other in different ways. The fitness value of a solution is calculated based on its interaction with other species.

Antonio and Coello (2017) provide a survey for the application of co-evolutionary techniques in MOEAs based on different ways of interaction among the species. In this regard, co-evolutionary approaches can be classified as cooperative or competitive. In the cooperative structure, good collaborators are encouraged and rewarded. Under the competitive scheme, the interaction is negative, as in the case where one population tries to develop increasingly difficult inputs for the other. It is also possible to have a hybrid structure that combines cooperative and competitive motives to promote diversity and avoid early convergence as the species work together to obtain good solutions to the problem.

Ma *et al.* (2018) provide a survey on cooperative co-evolutionary algorithms that is not exclusive to multi-objective problems. In limited applications involving multi-objective problems, the co-evolutionary structure is built almost exclusively among the cooperatively designed subpopulations of the solution set. In this context, co-evolution implies reciprocal or disproportional genetic changes in members of one subpopulation in response to another subpopulation.

In applications involving multi-objective problems, the cooperative structure is more common, and the decomposition is based on either the decision space or the objective space. In Iorio and Li (2004), the co-evolutionary structure is embedded in

NSGA-II for multi-objective continuous test problems. In the cooperative structure, good collaborators are encouraged and rewarded. In Tan *et al.* (2006) and in Zhao *et al.* (2014), co-evolutionary algorithms are developed for multi-objective optimization where decision variables are divided into smaller components. Similarly, Antonio and Coello (2016) apply co-evolutionary techniques to an indicator-based algorithm each subpopulation represents a particular component of the problem in the decision space, so members from all subpopulations are required to assemble a complete solution. Zhan *et al.* (2013) adopt co-evolutionary techniques to a particle swarm optimization algorithm in which each subpopulation is responsible for one objective. Nguyen *et al.* (2013) and Jiang *et al.* (2017) cooperative co-evolutionary structures are implemented across subpopulations for large-scale optimization problems, where each subpopulation is responsible for the subcomponents of the master problem.

There are also a few studies that involve many-objective cases. Zhang *et al.* (2017) implement a cooperative co-evolutionary structure in decomposition-based MaOEA. Each of the multiple cooperative subpopulations corresponds to a decomposition weight vector, while evolving in parallel, mating pools collect solutions from these subpopulations for recombination. Liu *et al.* (2018) introduce a co-evolutionary particle swarm optimization methodology to many-objective problems where multiple swarms are distributed to different areas of the Pareto front to maintain diversity and concentrate on these limited areas.

As a noteworthy approach, preference-inspired co-evolutionary algorithm using weight vectors (PICEA-w) proposed by Wang *et al.* (2015) presents an adaptive modification of decomposition weights (similar to reference points) by co-evolving them with candidate solutions. They advocate that the use of fixed reference points can be seen analogous to a priori methods and is not convenient for finding the entire Pareto front. In each iteration, the number of both solutions and decomposition weights are sequentially duplicated and truncated. The neighborhood structure between solutions and decomposition weights creates a ranking matrix used to sort and truncate both solutions and decomposition weights. In order for a candidate decomposition weight to be selected in the truncation procedure, it must be one of the weights that best

ranks a survived solution, or in the case of ties, it should be the furthest from the corresponding solution.

2.4. Performance Assessment of Pareto Front Approximations

The task of MOEAs is often to find a successful representation that will substitute the true Pareto front. When evaluating the success of an algorithm or comparing the performances of the Pareto approximations offered by different algorithms, the assessment need to be based on a group of vectors that constitute the Pareto front approximation. There are numerous solution set comparison metrics, each of which takes notice of different aspects.

Performance assessment indices are mainly classified into three subcategories depending on the aspects being investigated. These three categories are cardinality-based, accuracy-based, and distribution/spread-based performance indices. Accuracy-based performance indices can also be classified into two according to how accuracy is measured as distance-based or volume-based. There are various performance indices for each subcategory and several articles are available for a comprehensive review (Knowles and Corne, 2002a; Okabe *et al.*, 2003; Tan *et al.*, 2002; Zitzler *et al.*, 2002, 2003). As revealed in these reviews, there is no single "best" method to evaluate the quality of a solution set. Moreover, not all of the methods can be employed in every case.

For some performance metrics, the real Pareto must be known in advance. For some, there must be a bounded and discrete feasible space. Yet others can only be defined for two- or three-dimensional evaluations. Moreover, these techniques may lead to conflicting results and must be cross-checked to prevent misleading inferences.

This section contains a brief discussion of the different performance assessment indices. Representative metrics from each category are provided such that one example for each category requires the true Pareto information and makes performance evaluation selected, while the other can work in the absence of the true Pareto front information. For the sake of this discussion, some numerical analysis is also performed

with these selected indices using alpha versions of the proposed algorithm. This experimentation (see Appendix A) can be regarded as a preliminary study, both to choose the performance metric to be employed throughout the thesis and to assess the plausibility of some of the approaches to be used in the proposed algorithm.

Cardinality-based Performance Indices. Ratio of the Reference Points Found ($C1_R$) proposed by Hansen and Jaszkiewicz (1998) can be used when the true Pareto front is known. $C1_R$ is simply the ratio of the number of solutions in the set $\mathcal{S}$ which is also in the Pareto optimal solution set $\mathcal{P}^*$. The higher ratio indicates success in obtaining members of the true Pareto front.

In the absence of the true Pareto information, Coverage of Two Sets $C(\mathcal{S}_1, \mathcal{S}_2)$ proposed by Zitzler and Thiele (1999) can be used to compare two solutions sets $\mathcal{S}_1$ and $\mathcal{S}_2$ that are obtained from different algorithms as in Equation 2.10.

$$C(\mathcal{S}_1, \mathcal{S}_2) = |\{s_2 \in \mathcal{S}_2; \exists s_1 \in \mathcal{S}_1 : s_1 \succ s_2\}|/|\mathcal{S}_2| \tag{2.10}$$

$C(\mathcal{S}_1, \mathcal{S}_2)$ counts the number of solutions from the solution set $\mathcal{S}_2$ that are dominated by the members of $\mathcal{S}_1$ on the average. Thus, higher values are desired when it is computed for $\mathcal{S}_1$. It is important to remark that $C(\mathcal{S}_1, \mathcal{S}_2)$ is not equal to $1 - C(\mathcal{S}_2, \mathcal{S}_1)$. The $C(\mathcal{S}_1, \mathcal{S}_2)$ metric simply measures the average domination count for each of the two compared sets. While this metric is very useful when comparing two Pareto front approximations in the absence of the true Pareto front, the results can be difficult to interpret when there are multiple Pareto front approximations in evaluation.

Distance-based Performance Indices. Inverse Generational Distance (IGD) (Sato *et al.*, 2004) and Coverage Error (CE) (Sayın, 2000) are used when the true Pareto front is known. These metrics can provide combined information for both convergence and diversity of the solutions obtained, and widely used under the names such as Average Distance from Reference Set (DIST1) and Worst Distance from Reference Set (DIST2) (Czyżżak and Jaszkiewicz, 1998) or Maximum Pareto Front Error (MPFE) (Van Veldhuizen, 1999). They request that the Pareto optimal set or a reference set

be given or calculated since the distance is calculated according to these sets. IGD is defined as the average distance of the points in the true Pareto front to the closest members in the obtained solution set. Similarly, CE is the maximum distance from the points in the true Pareto front to the closest members in the obtained solution set. In other words, CE states the worst-represented element in the true Pareto front. Considering the Pareto optimal set consisting of a finite number of solutions, when these two metrics are calculated, lower values are desired for both. For distance calculations, any reasonable L_q metric, e.g. Euclidean distance ($q = 2$) or Tchebycheff distance ($q = \infty$) can be selected. It should be noted that, if the Tchebycheff distance is used, the maximum difference in each coordinate might be determined by the same objective due to the scale difference in objective parameters. When analyzing these metrics, the scale of the problem parameters should be taken into account.

Distribution and Spread-based Performance Indices. Performance indices selected for distribution and spread criteria are Uniformity Level (UL) (Sayın, 2000) and Maximum Spread (M_3^*) (Zitzler and Thiele, 1999), respectively. UL can be expressed as the minimum distance between two different solutions. Higher UL values indicate that the solution set is uniformly and well-distributed and does not contain any redundant solutions. It should be noted that this performance metric may conflict with cardinality-based performance metrics. As the number of solutions in the Pareto approximation increases, the distance between these solutions decreases. In M_3^*, after obtaining the hypercube by finding the ideal and nadir points, the length of the diagonal line of this hypercube is assigned as M_3^* metric. Distance measurement can be made by Euclidean metric.

Volume-based Performance Indices. Hypervolume (or Hyperarea) Ratio (HR) proposed by Zitzler and Thiele (1999) is used. For a given objective vector set, hypervolume is defined as the area dominated by this vector set. In Figure 2.7, an illustration is provided for two-dimensional objective space. In order to bound the measured space, it is usually required to define an ideal point and a nadir point from the true Pareto front. Nevertheless, it should be emphasized that the requirement of these two boundary points does not mean that HR values can only be computed when the true Pareto

front is known. A detailed discussion of finding satisfactory bounds is provided in Section 5.10.

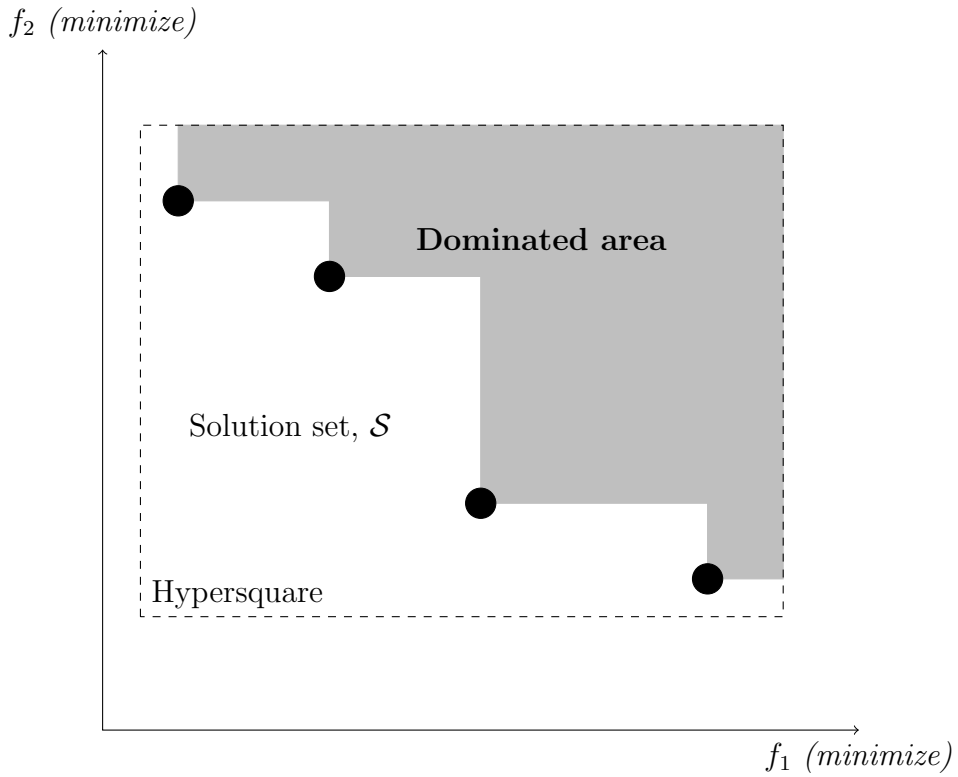

Figure 2.7. Hyperarea dominated by the solution set $\mathcal{S}$.

To calculate the dominated area by an objective vector set, the dominated area for each single solution should be measured and then their union should be calculated. Since this task has an excessive computational load for higher dimensional objective spaces, dominated areas can also be computed using Monte Carlo simulation by generating uniform random coordinate points within the hypercube (hypersquare) and by counting which of them belong to the dominated area.

HR provides information about the difference between the areas dominated by the objective vector set of a solution set $\mathcal{S}$ and by the Pareto optimal set $\mathcal{P}^*$. Higher ratios (or percentages) are preferred. HR metric is claimed to be the most beneficial metric as it uniquely quantifies the approximation set in a strictly monotonic manner with regard to Pareto dominance (Bader and Zitzler, 2011). This means that when comparing two Pareto sets, the HR value of the dominant set is always better than other. Higher values indicate that the corresponding Pareto approximation achieves to

dominate more volume in the objective space. Consequently, throughout this thesis, the HR value is calculated for each approximation obtained by different algorithms using Monte Carlo simulation by uniformly generating 10^5 random coordinate points within the hypercube. Although there are several other performance metrics each measuring different, desirable aspects of a Pareto approximation, HR is suggested to be a suitable metric even in the absence of true Pareto front information. Moreover, a higher HR result indicates not only a close approximation to the true Pareto front, but also a diverse and well-distributed approximation.

An alternative and similar analysis can be made by calculating the hypervolume of different region types when comparing two objective vector sets. After computing the ideal and nadir points and constructing the hypersquare where both sets appear, there are four possible regions for a randomly generated coordinate point in the objective space: a point that is not dominated by any of the sets ($R1$), dominated by both sets ($R2$), dominated by the set $\mathcal{S}_1$ but not by the set $\mathcal{S}_2$ ($R3$), and vice versa ($R4$). This is illustrated in Figure 2.8 in the two-dimensional objective space. When $R3$ and $R4$ are used to compare the performance of the two Pareto front approximations, the results show resemblance to the $C(\mathcal{S}_1, \mathcal{S}_2)$ metric. It must be noted that these areas can only be defined when two Pareto approximation pairs are compared to each other, whereas HR value can be used when multiple approaches are evaluated. Based on the findings obtained in both literature search and preliminary numerical analysis, HR is found as the most appropriate performance metric in terms of both interpretation accuracy and applicability. As a result, HR is the performance metric that will be used throughout this thesis.

Preliminary Numerical Experimentation. A preliminary numerical investigation with different performance metrics is performed using six-objective QAP instances. Problem instances are divided into two categories according to their sizes. For large size problem instances, 12 test problems are taken from QAPLIB (Burkard *et al.*, 1997). The selected instances cover a wide range of problem sizes and obey the triangular inequality in their distance matrices. The original flow matrix given in QAPLIB for each instance is taken as the first objective of the problem. Five more objectives are

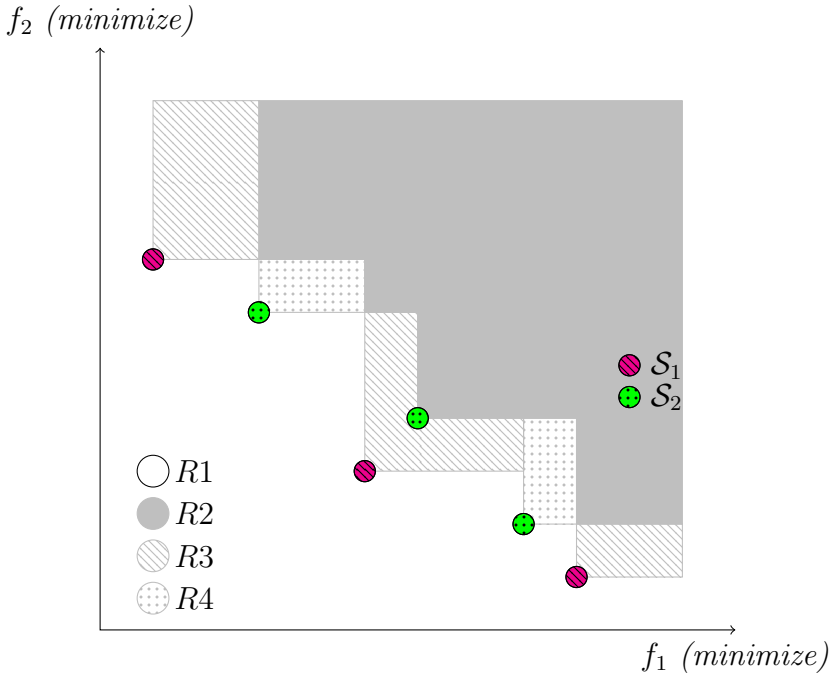

Figure 2.8. Hyperareas created by two solution sets, $\mathcal{S}_1$ & $\mathcal{S}_2$.

generated by randomly distorting the original flow matrix for each problem. For small size problem instances, nine more test problems are generated by reducing the sizes of three existing test problems *scr15*, *had16*, and *nug20* to eight, 10, and 12. The entire true Pareto front can be obtained by enumerating all permutations for these small problems. It should be recalled that the total number of feasible solutions in a n sized QAP is $n!$. Therefore, for large size problems, performance evaluation is made by comparing Pareto approximation pairs, each obtained by two algorithms.

The properties and behaviors of different performance metrics are analyzed through numerical experimentation using two evolutionary algorithms (Şahinkoç, 2014), a MOEA and a MaOEA. In order to verify the quality of the MOEA and MaOEA, the mean values of the selected performance indices are calculated for the final populations in 10 replications. In Appendix A, comparison results of Pareto front approximation pairs are given for large size problems in Table A.1. In Table A.2, comparison results with the true Pareto front are presented for small size problems.

From the results, a significant improvement is observed when using the MaOEA instead of the MOEA in all performance metrics. The only exceptions are the UL and

M_3^* metrics. This is because for the UL metric, the number of solutions obtained by the MaOEA is much higher than the MOEA. For large size problem instances, as the feasible region grows, the points in the Pareto approximation of MaOEA are distributed more uniformly leading to higher UL. Since the Pareto front approximations reported by both algorithms consists of populations of the same size, larger feasible regions result in scarce approximations in both algorithms. Nevertheless, solutions are distributed uniformly in the MaOEA whereas the formation of clusters is more likely to occur in the MOEA. For the M_3^* metric, while it is guaranteed that seeded single-objective optimal solutions exist in the final population of MOEA due to crowding distance, this is not the case for MaOEA. Since the comparison is based on the final populations and the hypercube is obtained with ideal and nadir points, the hypercube becomes narrow and the spread shrinks in the absence of single-objective optimal solutions. This is an important observation that will lead to the necessity of the use of external populations.

As an alternative method, different weighted linear aggregations of the objectives can be used to measure the performance of Pareto front approximations. Using each decomposition vector, the best solutions from both approximations are found by solving the corresponding single-objective problems and these solutions are evaluated. When a sufficient number of well-spread decomposition vectors are used in this evaluation, the results show which Pareto approximation includes a diverse set of trade-off solutions. The same principle will be used to create the reference point set in the proposed MaOEA.

Table A.3 in Appendix A presents the percentages of the number of vectors where the particular algorithm performs better than the other along with the improvement achieved by applying the MaOEA when the difference between solutions obtained from both algorithms for each decomposition vector is averaged.

Still another alternative to evaluate the performance of Pareto approximation is to use stochastic and robust performance measures to evaluate the performances of Pareto approximations. For instance, when the objectives correspond to different scenario realizations caused by uncertainty, as will be described in detail in Section 5.9, it

is an intriguing idea to evaluate the performance of the Pareto approximations using robust performance metrics. As stated in Iancu and Trichakis (2013), robust optimization problems can be modeled as MOPs with an (in)finite number of objectives (corresponding to uncertainty scenarios), and at least one of the robust optimal solutions must also be a Pareto optimal solution. The same fact applies to the weighted linear aggregations of scenario objectives. Some examples of robust measures are minimizing maximum cost, minimizing maximum absolute regret, and minimizing maximum relative regret. The best solutions obtained from both algorithms report the corresponding robust objective value in Table A.3 in Appendix A. The results indicate a significant improvement in finding more robust solutions when using the MaOEA instead of the MOEA.

Based on the preliminary analysis on several performance metrics, each exploring different aspects of the Pareto approximation, it is confidently concluded that it is worthwhile to pursue the approaches tested in MaOEA. As stated before, it is also concluded that HR can adequately assess the performance of different algorithms testes in this thesis.

3. MANY-OBJECTIVE COMBINATORIAL OPTIMIZATION PROBLEMS

This chapter describes the many-objective problems chosen as benchmark for in both developing and evaluating the proposed evolutionary algorithm. After providing the relevant definitions, notation and properties of combinatorial optimization problems, the benchmark problems namely, the many-objective knapsack problem (MaOKP), the many-objective traveling salesman problem (MaOTSP), and the many-objective quadratic assignment problem (MaOQAP) are introduced together with their mathematical formulations and literature overview. Finally, how to model optimization problems under scenario-based uncertainty as MaOPs is discussed.

3.1. Combinatorial Optimization Problems

In the field of mathematical programming, combinatorial optimization has been widely studied for decades and has numerous real-life applications including portfolio optimization, vehicle routing, layout planning, artificial intelligence etc. Wolsey and Nemhauser (1999) can be consulted for the theory of combinatorial optimization, and Korte *et al.* (2012) can be consulted to review theories and algorithms on combinatorial optimization problems. Knapsack, linear assignment, quadratic assignment, traveling salesman, minimum spanning tree, shortest path, and set covering problems are well-known examples of combinatorial optimization problems.

Combinatorial optimization problems, which are special cases of integer programming problems, have feasible sets consisting of a finite number of elements, although an exhaustive search is not tractable. Certain combinatorial structures such as paths, trees, tours and flows can be associated with combinatorial problems. In combinatorial optimization problems, The feasible set X is defined as a subset of the power set, $X \subseteq \mathcal{P}(A)$ where $A = \{a_1, ..., a_n\}$, $n \in \mathbb{Z}^+$ is a finite set and it is composed of 2^A elements. A combinatorial problem can also be formulated in terms of binary variables

by introducing a variable x_i for each element $a_i \in A$. As a result, a feasible solution S is represented by a binary vector $x \in \{0,1\}^n$. When this definition is used, a feasible solution becomes $S = \{a_i : x_i = 1\}$. Consequently, a feasible solution S can be described as a subset of A and binary vectors can be used to represent it: $S \subseteq \{0,1\}^n$.

Combinatorial problems typically have two types of objective functions: the sum objective and the bottleneck objective. Using a weight function $w : X \to \mathbb{Z}$, both can be formulated for a feasible solution S as in Equation 3.1 and 3.2.

$$\text{Sum objective:} \qquad f(S) = \sum_{a \in S} w(a) \tag{3.1}$$

$$\text{Bottleneck objective:} \qquad f(S) = \max_{a \in S} w(a) \tag{3.2}$$

Accordingly, the multi-objective combinatorial optimization (MOCO) problem, which is a subcategory of multi-objective integer programming problem, can be formulated as in Equation 3.3 to 3.8.

$$\min_x f(x) = Cx \tag{3.3}$$

subject to

$$Ax = b \tag{3.4}$$

$$x \in \{0,1\}^n \tag{3.5}$$

where

$$x \in \{0,1\}^n \leftarrow n \text{ variables, } j = 1, ..., n \tag{3.6}$$

$$C \in \mathbb{N}^{m \times n} \leftarrow m \text{ objectives, } i = 1, ..., m \tag{3.7}$$

$$A \in \mathbb{N}^{p \times n} \text{ and } b \in \mathbb{N}^{p \times 1} \leftarrow p \text{ constraints, } k = 1, ..., p \tag{3.8}$$

Dealing with MOCO problems is generally difficult for a variety of reasons. Finding the entire efficient solution set is generally intractable, since the number of efficient solutions increases exponentially depending on the problem size. MOCO problems are

generally NP-complete, even when their single-objective counterparts are not. Secondly, due to the discrete nature of MOCO problems, the solutions space is not convex and there exists quite a lot of unsupported efficient solutions. This fact persists even if the constraint matrix of the regarding problem is unimodular (Ulungu and Teghem, 1995). Since these solutions are not optimum for any weighted sum of the objectives, they are more difficult to find. As shown for the knapsack problem (Visée *et al.*, 1998), the number of unsupported efficient solutions is much higher than the supported efficient solutions, especially when the number of objectives is high. In addition, unlike a linear increase in the number of supported efficient solutions, the number of unsupported efficient solutions increases with an exponential function when there are multiple sum objectives. As a result, unsupported efficient solution set is essential to achieve a successful Pareto approximation.

Exact methods for MOCO problems include weighted sum scalarization method, compromise solution method, goal programming and ranking method. Other methods adapted from single-objective optimization include branch-and-bound and two-phase methods. To review MOCO theory, methodology and applications; the reader is referred to Ehrgott and Gandibleux (2000) and Ehrgott and Gandibleux (2003).

In this thesis, the problem domain is MOCO problems. These problems have important applications in real-life and still remain uncharted for the studies of many-objective approaches. One of the main goals of this thesis is to develop solution approaches that can be used for a wide spectrum of MOCO problems, therefore selected MOCO problems exhibit different features and applications. Nevertheless, they also exhibit some common features. The objective functions of the selected MOCO problems are sum objective type as they are more common for many-objective modeling and application. Also, the objective function coefficients of the benchmark formulations have the same type of distribution and scales, unless otherwise stated.

3.2. Many-Objective Knapsack Problem (MaOKP)

The 0-1 knapsack problem is one of the best known fundamental combinatorial optimization problems as it has real-life applications in many different areas, its solution methodologies include mathematical programming, dynamic programming, exact and heuristic algorithms, and metaheuristics. Briefly, in the knapsack problem, given a set of items each with a weight and a profit value, the objective is to decide whether each item will be included in a collection so that the total profit is as large as possible while the total weight is less than a given capacity.

The many-objective knapsack problem (MaOKP) is used in the development and performance evaluation of the proposed algorithm since this problem has been frequently used in the literature as a benchmark for several multi-objective and many-objective evolutionary approaches. Moreover, the effects of alternative operators and parameters can be easily observed in the knapsack problem, since the computation time is relatively small. In the provided formulation (Ulungu and Teghem, 1994), profit coefficient p_{ij} is defined for each objective i and for each item j in Equation 3.9. The binary decision variable x_j in Equation 3.11 corresponds to whether item j is included in the knapsack collection or not. In multi-constrained (or multi-dimensional) MaOKP, different weight parameters w_{jk} are generated for each item and constraint, and the capacity parameter C_k is defined for each constraint in Equation 3.10.

$$\max_x f_i(x) = \sum_j p_{ij} x_j \quad \forall i \tag{3.9}$$

subject to

$$\sum_j w_{jk} x_j \leq C_k \quad \forall k \tag{3.10}$$

$$x_j \in \{0, 1\} \quad \forall j \tag{3.11}$$

The chromosome structure of the MaOKP uses binary encoding, where the genotype of a solution is represented by a n-length binary string in the decision space and its phenotype corresponds to its location in the m-dimensional objective space.

Literature Survey of MaOKP. A detailed literature survey is provided for the multi- and many-objective knapsack problem since it is used in the development and performance evaluation of the proposed algorithm. Research using exact methods involve dynamic programming (Bazgan *et al.*, 2009b; Klamroth and Wiecek, 2000), two-phase branch and bound method (Gandibleux and Freville, 2000; Visée *et al.*, 1998), ϵ-constraint method (Kirlik and Sayın, 2014; Laumanns *et al.*, 2006) and other methods (Delort and Spanjaard, 2010; Florios *et al.*, 2010); whereas approximation algorithms can be exemplified by Erlebach *et al.* (2002) and Bazgan *et al.* (2009a).

Since single-objective and thereby multi-objective knapsack problem is NP-hard, various heuristic methods are also applied. Heuristic methods include simulated annealing (Czyżżak and Jaszkiewicz, 1998), tabu search (Abdelaziz *et al.*, 1999; Gandibleux and Freville, 2000), scatter search (da Silva *et al.*, 2006), labelling algorithm (Captivo *et al.*, 2003), and local search (Alsheddy and Tsang, 2010; Lust and Teghem, 2010b; Vianna and Arroyo, 2004). As seen in Table 3.1, most the exact methods, approximation algorithms, and heuristic approaches other than evolutionary methods are for multi-objective knapsack problem. It should be noted that these approaches generally have some practical and theoretical limitations, preventing them from scaling for a higher number of objectives.

When it comes to the field of evolutionary optimization, Zitzler and Thiele (1999) can be viewed as an influential pioneering study. In their study, they compared five different multi-objective genetic algorithms and showed that a Pareto evolutionary algorithm called SPEA was successful. Their method for creating create problem instances is used by many other researchers and as well as this study. Other evolutionary approaches include M-PAES (Knowles and Corne, 2000), MOGLS (Jaszkiewicz, 2002b), MOTGA (Alves and Almeida, 2007), NSGA-II (Sato *et al.*, 2007), multi-start search combined with path relinking (Beausoleil *et al.*, 2008), and MEMOTS (Lust and Teghem, 2008).

In recent years, in parallel with the rising popularity of MaOEAs, new algorithms have been proposed and tested on MaOKP, as well. Two algorithms, δ-MOSS and k-

Table 3.1. Literature review of non-evolutionary methods for MaOKP.

Name (Year)	**Method**	**Objective Count**
Czyzżak and Jaszkiewicz (1998)	Simulated annealing	2 - 4
Visée *et al.* (1998)	Two-phase B&B	2
Abdelaziz *et al.* (1999)	Tabu search	2 - 3
Gandibleux and Freville (2000)	Two-phase B&B, tabu search	2
Klamroth and Wiecek (2000)	Dynamic programming	2
Erlebach *et al.* (2002)	Approximation algorithm	no experimentation
Captivo *et al.* (2003)	Labeling algorithm	2
Vianna and Arroyo (2004)	Local search	2 - 4
da Silva *et al.* (2006)	Scatter search	2
Laumanns *et al.* (2006)	ϵ-constraint	2 - 3
Bazgan *et al.* (2009a)	Approximation algorithm	2 - 3
Bazgan *et al.* (2009b)	Dynamic programming	2 - 3
Alsheddy and Tsang (2010)	Local search	2
Delort and Spanjaard (2010)	Hybrid of two-phase and DP	2
Florios *et al.* (2010)	Multi-criteria B&B	3
Lust and Teghem (2010b)	Local search	2
Kirlik and Sayın (2014)	ϵ-constraint	3 - 4

EMOSS, with the goal of objective reduction are suggested in Brockhoff and Zitzler (2006). Other examples are a correlation-based weighted sum approach by Murata and Taki (2009), a hyperplane-based evolutionary algorithm, HypE by Bader and Zitzler (2011), and an artificial fish swarm optimization algorithm by Azad *et al.* (2014). Recent studies have identified difficulties encountered in MaOPs and improvements have been suggested to existing MOEAs to overcome these difficulties. Modifications are offered to NSGA-II in Sato *et al.* (2007), in Tanigaki *et al.* (2014), and in Ishibuchi *et al.* (2014b); whereas a search for correct parameterization for MOEA/D is carried out in Sato (2015).

In another line of research, the performance of existing MOEAs is evaluated as the number of objectives in MaOKP increases. While NSGA-II and I-IBEA are evaluated in Ishibuchi *et al.* (2008), the performances of different scalarizing functions of MOEA/D are verified in Ishibuchi *et al.* (2013). NSGA-II, SPEA2, IBEA, and MSOPS are

compared in Sato *et al.* (2013). Ishibuchi *et al.* (2014a) compare the performances of four evolutionary algorithms, namely NSGA-II, MOEA/D, SMS-EMOA, and HypE including the problems with different correlation levels.

Table 3.2 summarizes the literature for MOEAs and MaOEAs. As far as it is reported, reference point set based sorting algorithm proposed in NSGA-III (Deb and Jain, 2013) has not yet been adapted to MaOKP.

Table 3.2. Literature review of evolutionary methods for MaOKP.

Name (Year)	**Evolutionary Algorithm**	**Objective Count**
Zitzler and Thiele (1999)	SPEA	2 - 4
Knowles and Corne (2000)	M-PAES	2 - 4
Jaszkiewicz (2002b)	MOGLS	2 - 4
Brockhoff and Zitzler (2006)	δ-MOSS, k-EMOSS	5 - 25
Alves and Almeida (2007)	MOTGA	2 - 4
Sato *et al.* (2007)	NSGA-II	2 - 5
Beausoleil *et al.* (2008)	multi-start search, path relinking	2 - 4
Ishibuchi *et al.* (2008)	NSGA-II	2 - 6
Lust and Teghem (2008)	MEMOTS	2 - 3
Murata and Taki (2009)	Correlation-based weighed sum	10
Bader and Zitzler (2011)	HypE	2 - 50
Ishibuchi *et al.* (2013)	MOEA-D	2 - 10
Sato *et al.* (2013)	NSGA-II, SPEA2, IBEA, MSOPS	2 - 10
Azad *et al.* (2014)	Artificial Fish Swarm	2 - 30
Ishibuchi *et al.* (2014a)	NSGA-II, MOEA/D, SMS-EMOA, HypE	2 - 10
Ishibuchi *et al.* (2014b)	NSGA-II	2 - 10
Tanigaki *et al.* (2014)	NSGA-II	2 - 10
Sato (2015)	MOEA/D	2 - 8

3.3. Many-Objective Traveling Salesman Problem (MaOTSP)

Traveling salesman (salesperson) problem (TSP) is one of the well-known and intensively studied combinatorial optimization problems. Several other optimization problems such as sequential ordering problem, vehicle routing problem are derived from TSP. Because of wide variety of its applications, its multi- and many-objective

versions are also of interest. However, it is seldom studied in the field of multi-objective optimization (Jaszkiewicz, 2018; Jozefowiez *et al.*, 2008; Peng *et al.*, 2009; Ulungu and Teghem, 1994). Different applications of TSP in the literature include different definitions or treatments for the objectives of the problem. They can be defined as cost, distance, time, risk, energy or touristic attractiveness, each with their own distinctive characteristic and scale. Simultaneous optimization of these objective alternatives is within the domain of the many-objective optimization.

In TSP, given a set of cities SC, the aim is to find a minimal cost (or distance, time) tour that visits each city exactly once and returns to the initial city. In the provided formulation of the TSP, the binary decision variable x_{jk} in Equation 3.16 corresponds to whether the edge from city j to city k is included in the tour or not. In the many-objective version, the cost of traveling from city j to city k, c_{ijk}, is defined for each objective i as seen in Equation 3.12. Equation 3.13 and 3.14 ensure that each city is visited exactly once and 3.15 is the subtour elimination constraint defined for each subset of cities Q.

$$\min_{x} f_i(x) = \sum_{j,k} c_{ijk} x_{jk} \quad \forall i \tag{3.12}$$

subject to

$$\sum_{j \neq k} x_{jk} = 1 \quad \forall k \tag{3.13}$$

$$\sum_{k \neq j} x_{jk} = 1 \quad \forall j \tag{3.14}$$

$$\sum_{j,k \in Q} x_{jk} \leq |Q| - 1 \quad \forall Q \subseteq SC \tag{3.15}$$

$$x_{jk} \in \{0, 1\} \quad \forall j, k \tag{3.16}$$

Depending on the structure of the cost matrix, different types of problems can be obtained. When the cost from city j to city k is always equal to the cost from city k to city j, the problem is called *symmetric TSP*. These problems have undirected graph networks. Otherwise, it is called *asymmetric TSP (ATSP)* with directed graph networks. Secondly, it is important whether the cost matrix obeys the triangular

inequality. Finally, there might be cases where some c_{jk} in the cost matrix do not have a finite value, in other words, it is not possible to use the edge between city j and city k. When the graph is not complete and the problem becomes finding a feasible tour, it is called the hamiltonian cycle problem.

Since the permutation encoding is used for a TSP solution in the evolutionary methodology, this benchmark is different from the knapsack problem. All genetic operators need to be revised and aligned with this new encoding structure. In this way, the proposed algorithm can be claimed to be generic and adaptable to problems with different characteristics.

3.4. Many-Objective Quadratic Assignment Problem (MaOQAP)

Quadratic assignment problem (QAP), which also belongs to NP-hard class, is used to model many real-life problems such as facilities location and combinatorial data analysis. In Loiola *et al.* (2007), some of the most important QAP formulations are identified and classified, and a detailed discussion is made about exact and heuristic solution techniques, including metaheuristics. In addition, the main research trends are identified to guide future researches, providing a basis for the QAP research.

In the formulation of many-objective QAP (MaOQAP) similar to that introduced by Knowles and Corne (2002b), given a set of departments $j, k \in D$ and locations $r, s, \in L$, the problem is to assign all departments to different locations with minimal cost calculated as the product of flows f_{ijk} between departments with corresponding distances d_{rs} between locations. In MaOQAP, the flow parameter is defined for each objective i as seen in Equation 3.17. The binary decision variable x_{jr} in Equation 3.20 corresponds to whether the department j is assigned to location r. Equation 3.18 and 3.19 ensure that each department is assigned to exactly one location and vice versa, respectively.

$$\min_x f_i(x) = \sum_{j,k,r,s} f_{ijk} d_{rs} x_{jr} x_{ks} \quad \forall i \tag{3.17}$$

subject to

$$\sum_j x_{jr} = 1 \quad \forall r \tag{3.18}$$

$$\sum_r x_{jr} = 1 \quad \forall j \tag{3.19}$$

$$x_{jr} \in \{0, 1\} \quad \forall j, r \tag{3.20}$$

It is stated in Knowles and Corne (2002b) that multi-objective version of QAP is useful for some layout problems, such as hospital layout with different flow types; i.e. doctors, patients, and nurses lead to different objectives. They investigate landscape analysis issues to approximate the Pareto front with a hybrid local search algorithm, and in Knowles and Corne (2003), some instance generators and test suites are formulated. In Paquete and Stutzle (2006), a two-phased local search procedure is followed to solve the bi-objective QAP and ant colony optimization (ACO) is used for the bi-objective QAP in López-Ibánez *et al.* (2004). Among the studies that applied MOEA, Kleeman *et al.* (2004) and Day and Lamont (2005) use some variations of multi-objective messy genetic algorithms.

MaOQAP is chosen as a benchmark for two reasons. The first reason is the additional challenge it imposes due to the difficulty in obtaining optimal solutions even for the single-objective versions. Thus, it can be used to test the limits of the proposed algorithm.

Secondly, MaOQAP is also used to test the proposition regarding modeling optimization problems with scenario-based uncertainty as MaOPs. This proposition creates a novel research stream as well as a new and natural problem domain for MaOEAs. The foundations, main assumptions and contribution of this approach will be discussed in the next section. The relevant numerical experimentation is provided in Section 5.9.

3.5. Scenario-based Uncertainty

Optimization problems involving random parameters occur in almost all fields of science and engineering, from power generation to telecommunication and medicine. Most of the real-life applications of optimization models in the domain of operations research, such as facility location, aggregate production planning, or investment planning, involve decisions with a long time-span in which the environment they are modeling may change substantially. Thus, the parameters used as inputs in these models are often quite ambiguous. Uncertain parameters can be either continuous or discrete. Continuous parameters are generally restricted within some predetermined intervals, while in the discrete case, uncertainty is described by a set of alternative scenarios, each representing a particular realization. Furthermore, there may be cases where the DM has some information on the probability distributions that govern the values of random parameters, in other cases such information may not be available.

In the scenario-based approach, a probability p_ω can be associated with each scenario alternative ω, $\omega \in \Omega$ where Ω is the set of all possible scenarios, if such probability information is available. Only a finite number of sampled instances of uncertainty is considered and the stochastic problem reduces to a deterministic problem for a given scenario realization. In the literature, it is argued that the scenario approach generally results in more tractable models (Snyder, 2006) and it can successfully reflect the probabilistic characteristics of system uncertainties (Wu *et al.*, 2011). In addition, it has the advantage of allowing parameters to be statistically dependent, which is not the case when parameters are represented by probability distributions. Dependence is usually necessary to build realistic models, for instance, demands are usually correlated through time periods or geographical regions and costs are usually correlated through suppliers. In this thesis, the focus is on the discrete scenario-based approach.

Two main paradigms that address the scenario-based uncertainty in optimization problems are stochastic programming and robust optimization. Instead of using techniques derived from these methodologies, a novel approach employing many-objective optimization methodologies is proposed in this thesis. In this approach, the prob-

lem under analysis is modeled as a MaOP where the number of objectives is equal to the number of scenarios under consideration. Since the number of required scenarios may be quite high, such a problem definition becomes a natural domain for the many-objective optimization research area.

In this way, a new and natural field of application to many-objective optimization is introduced by developing many-objective counterparts of the OR problems that are subject to uncertainty. On the other hand, a contribution is also made to both stochastic programming and robust optimization by offering an alternative perspective. By obtaining a Pareto optimal set, optimal solutions for many metrics covered in stochastic and robust approaches are obtained simultaneously at a single step. In this way, after the optimization process, the decision maker will have the opportunity to choose efficient solutions that cover a wide range of different performance metrics and satisfy their preferences.

According to Snyder (2006), the scenario approach has two main drawbacks. First, identifying scenarios and assigning probabilities to them is a difficult task, and the other disadvantage is that people want to define relatively few number of scenarios for computational reasons, but this limits the range of future states in which decisions will be evaluated. Both issues cease to be disadvantages under the proposed approach.

3.5.1. Optimization Under Scenario-based Uncertainty

When optimizing under scenario-based uncertainty, the aim is to find solutions that perform well in all scenarios, and this is usually handled using either stochastic programming or robust optimization. In stochastic programming, uncertain parameters are managed with probability distributions previously known by the decision maker, and usually the goal is to optimize the expected behavior. In robust optimization, these probability distributions are not required or are not available in some cases, and the common attempt is to optimize the worst case performance of the system (Snyder, 2006). For a detailed theory and applications of robust discrete optimization, Kouvelis and Yu (2013) can be consulted.

One of the most common robustness measures is to minimize the cost of the scenario where the maximum cost occurs (*minimax cost*). Robustness measures involving the concept of regret are also very common. Regret for a scenario is the absolute or percentage deviation of the objective value of a solution from the objective value of the optimal solution for the corresponding scenario. Models try to minimize maximum regret in all scenarios (*minimax absolute regret* or *minimax relative regret*). The main appeal of such robustness measures is that the decision maker does not require to estimate scenario probabilities. Aissi *et al.* (2009) provide a survey on the discrete minimax cost and minimax regret versions of combinatorial optimizations problems. Other approaches include p-robustness Snyder and Daskin (2006), γ-robustness, α-reliability, and conditional value at risk.

Choosing an appropriate performance measure to deal with uncertainty in a particular problem is a critical issue, since the definition of good performance in all scenarios depends on the particular application and the DM. Techniques that are driven from both stochastic and robust approaches offer different ways to convert objective values given by a feasible solution under each scenario realization into a single performance measure. This approach is analogous to the a priori preference articulation paradigm in multi-objective optimization. As mentioned earlier, a priori approaches require the decision maker to define the importance and preference relationships of the objectives before searching for feasible solutions (Coello *et al.*, 2007). Any stochastic or robust measure is a particular predetermined function of scenario objectives and will lead to a compromising decision.

3.5.2. Relationship with Many-Objective Optimization

Stating that robust optimization is indifferent to the scenarios that are not the worst case scenario, Iancu and Trichakis (2013) argue that the classical robust optimization framework does not need to produce Pareto optimal solutions and therefore may lead to inefficiencies and suboptimal performance in practice. They point out that robust optimization can be seen as a MOP with an (in)finite number of objectives (corresponding to uncertainty scenarios) and introduce the concept of Pareto efficiency

in the context of robust optimization methodology. They provide a theoretical characterization of "*Pareto robustly optimal (PRO)*" solutions, stating that one of the robust optimal solutions should also be a Pareto optimal solution. By solving an additional model derived from the original robust optimization problem, they propose to generate a robust optimal solution that also exists in the Pareto optimal set.

A posteriori approach to analyze the performance of different decisions in an uncertain environment can be used instead of obtaining a Pareto robustly optimal solution that corresponds to a particular robust performance measure (i.e. a priori approach). A posteriori techniques can be applied to achieve the efficient set including all robust optimal solutions and optimal solutions of weighed linear aggregations (i.e. expected value). Note that an efficient solution in this context is such that there exists no other feasible solution to improve the objective for some scenarios, without worsening the objective in at least one other scenario. See Aissi *et al.* (2009) for the proof that efficient sets contain PRO solutions that correspond to minimax cost and minimax regret. To the best of our knowledge, there are no studies where optimization problems under discrete scenario-based uncertainty are treated using MaOEA techniques. Consequently, the study in this section contributes to both stochastic programming/robust optimization and many-objective evolutionary optimization research streams. Specifically:

- An a posteriori approach is implemented that achieves the efficient solution set for the problems modeled using scenario-based uncertainty.
- Once a satisfactory Pareto optimal set has been achieved, it provides better ways to help the decision maker comprehend what is offered to them and tailor their choices to their preferences.
- The basic shortcoming of missing the efficient solutions in robust optimization.
- The difficulties in identifying scenarios, assigning probabilities to them and limitation in the number of scenarios due to computational reasons are alleviated.

3.5.3. Characterization of Scenario Structures

The nature of scenario-based uncertainty that will allow a particular problem to be treated as a MaOP with respect to the scenarios can be discussed by analyzing the decision variables, coefficients of the objective function, parameters in the technology matrix or resource vector.

Decision Variables and Objective Function Coefficients. If a set of decision variables is defined over a set of scenarios, it can be interpreted as the DM has the chance to declare or update these decision variables after observing which scenario alternative has been realized. As a result, if all decision variables are defined over the scenario set, the uncertainty vanishes and the problem becomes deterministic. In such cases, all that needs to be done is to solve separate problems with distinct sets of parameters and decision variables. Therefore, although there is a chance to model some decision variables as scenario-based, it should not be the case to model all decision variables as scenario-based. Coincidentally, this requirement is also consistent with reality. In many real-life problems, decision variables must be determined often before realizing scenarios. In other words, some decisions must include a priori commitments. On the other hand, in order to have multiple objectives and lend the problem to multi-objective approaches, the objective function must contain some scenario-based coefficients or decision variables. For instance, the profit coefficients in the knapsack problem, flow parameters in QAP, and cost parameters in TSP may be scenario dependent.

Parameters in the Technology Matrix and Resource Vector. When the parameters in the technology matrix or resource coefficients are defined on the scenario set, the feasible space is affected. A feasible solution obtained through a set of parameters defined for a particular scenario alternative may be infeasible in other scenarios. The term "*model robustness*" (Mulvey *et al.*, 1995) reflects the idea that a solution is preferred when it is feasible in all scenarios. Also, in reality, it is extremely undesirable and risky to accept a solution that has no chance to be employed under some possible scenario realizations. In this way, the feasible space becomes the intersection of all scenario specific constraint sets. If scenario dependency does not exist in the parameters

in the technology matrix or resource coefficients, it can be said that a solution that is shown to be feasible in one scenario is feasible in other scenarios and the feasible solution sets of different scenarios overlap.

3.5.4. QAP Under Scenario-based Uncertainty

QAP under scenario-based uncertainty is used to demonstrate and validate these ideas. In this case, each objective in Equation 3.17 corresponds to a different flow scenario. As there is no option for alteration in the constraints, uncertainty in QAP does not arise infeasibility. Scenario-based QAP is suitable to be used for proof-of-concept since it is difficult to obtain stochastic and robust optimal solutions with exact methods and heuristic approaches are required.

4. PROPOSED MANY-OBJECTIVE EVOLUTIONARY ALGORITHM

This chapter provides a detailed explanation for the proposed many-objective evolutionary algorithm (MaOEA), reviewing its development stages and identifying its important aspects. Along with the parameters and genetic operator schemes, the pseudo-codes of the key parts are also elaborated. Additionally, variations of the proposed MaOEA regarding different procedures for adaptation of the reference point set are presented.

The proposed MaOEA adopts an elitist framework and requires a set of predefined reference points as input. In each generation, current population and offspring population are combined and sorted first by nondominated sorting and then the reference set based sorting, similar to NSGA-III (Deb and Jain, 2013). The main novel features of the proposed MaOEA are highlighted below:

- The algorithm starts by solving each objective separately and uses m individual optima (or near optima) solutions to construct a hyperplane that remains fixed over the course of the evolution. For this reason, the basic form of the algorithm is called fixed hyperplane fixed reference points nondominated sorting algorithm (*FHFR*).
- The recombination operator and selection mechanisms are designed in a way to complement and support the reference set based sorting. Reference point guided path relinking is proposed as the recombination scheme for this purpose. Additionally, repair and local improvement procedures are also guided by reference points.
- Reference points are mapped uniformly onto the fixed hyperplane and their allocation remains unchanged throughout the algorithm. Their locations on the fixed hyperplane can be changed when the algorithm has a reference point update feature (*FHMR*) or when the reference set is allowed to co-evolve with the

solution population ($FHCo$). These two alternatives are elaborated in Section 4.4 and Section 4.5, respectively.

- An archive structure called external population stores all nondominated solutions and reports at the end of the algorithm.

These features are presented within the main framework in Algorithm 1 in Figure 4.1 and elaborated in the subsequent subsections.

Algorithm 1: ***Proposed MaOEA***
Input: $parameterSetting$, $TerminationCriteria$
1. Find ***optimal (near optimal) solutions*** of single-objective problems
2. Set ***fixed_hyperplane*** and $referencePoints : \mathcal{R}$
3. Generate initial $population$: $\mathcal{P}$
4. Set ***external_population***: $\mathcal{P}_{ext}$
5. **While not** $TerminationCriteria$:
6. $parentPairs : \mathcal{P}_{pairs} :=$ ***Select_Pairs***$(\mathcal{P})$
7. $offspring : \mathcal{P}_{off} :=$***Reference_Point_Guided_Path_Relinking***$(\mathcal{P}_{pairs})$
8. ***Mutation***$(\mathcal{P}_{off})$, ***Repair***$(\mathcal{P}_{off})$ and ***Local_Improvement***$(\mathcal{P}_{off})$
9. ***Reference_Set_Based_Sorting***$(\mathcal{P} \cup \mathcal{P}_{off}, \mathcal{R})$
10. Trim $\mathcal{P} \cup \mathcal{P}_{off}$ and update $\mathcal{P}$
11 Update $\mathcal{P}_{ext}$ using eliminated, nondominated solutions in $\mathcal{P}_{off}$
12. ***Reference_Set_Update_Procedure*** (for $FHMR$)
13. ***Reference_Set_Co-evolution*** (for $FHCo$)
14. **End while**
Output: $\mathcal{P}_{ext}$

Figure 4.1. Pseudo-code of the main framework of the proposed algorithm.

4.1. Reference Set Based Sorting with a Fixed Hyperplane

In the Nondominated Sorting Algorithm-II (NSGA-II) originally implemented in Deb *et al.* (2002), members of the current population and offspring population are combined and ranked at the end of each iteration using dominance comparison. Among this combined population, the set of best individuals equal to a population size parameter is retained and declared as the new generation population. In order to keep the previously found good quality solutions in the population and use them in

genetic operators, elitism is achieved as all existing and new members are included in the combined population.

The preliminary results display the shortcomings of NSGA-II to approximate a Pareto front successfully when the problem under investigation has more than three objectives. Since the difficulties and challenges arising in MaOPs have been identified in numerous studies, a framework based on NSGA-II is proposed in Deb and Jain (2013), called NSGA-III. Although the overall evolutionary structure of NSGA-II has been preserved, it claimed that NSGA-III is equipped to handle MaOPs.

In NSGA-III, the Pareto front is attempted to be embraced by a hyperplane which is constructed repeatedly in each iteration of the algorithm based on the evolving population. The main difference between NSGA-III and NSGA-II is the way of imposing diversity among population members. NSGA-III ensures diversity by providing and updating a number of well-spread *reference points* instead of the crowding distance concept used in NSGA-II. Both algorithms use the conventional Pareto dominance rule, but then a niche-preservation strategy is used to measure the fitness of individuals in NSGA-III. The reference points in NSGA-III are placed on the hyperplane restructured in each iteration based on the current population and expected to cover the feasible region in the objective space.

In the first stage of the sorting procedure of NSGA-III, nondominated fronts are identified as done in NSGA-II by calculating their usual domination ranks. All solutions are examined in regard to whether they are dominated by any other solution or not. After that, all nondominated solutions are ranked as one. The same procedure is repeated after these nondominated solutions are excluded. Nondominated solutions in the remaining set are ranked as two and the procedure is repeated until all individuals in the entire population are assigned a rank.

After the assignment of nondomination ranks, a front is formed for each rank. Fronts with lower nondomination ranks are added to the population $\mathcal{S}_t$ in order from level one to level l, where t is the iteration count during the evolution. This is done

until the number of solutions in $\mathcal{S}_t$ exceeds the predefined threshold level N which is equal to the half size of the combined population, consisting of the current population and an equally sized offspring population. If $|\mathcal{S}_t| = N$, no further operation is required and $\mathcal{S}_t$ is set as the population of the next generation. If $|\mathcal{S}_t| > N$, members from the fronts of level one to level $(l-1)$ are selected as the population of the next generation and the remaining members are selected out of those at level l. Reference points are used select members at this level in a well-spread way. Elitism is achieved by using this truncation methodology.

The reference set based algorithm uses a predefined set of reference points to maintain diversity among population members. In general, the algorithm not only emphasizes nondominated solutions, but also emphasizes population members associated with these reference points. If the reference point set is widely distributed over the objective space, the resulting population is also likely to be widely distributed on the Pareto optimal front. The reference point set can be predefined or provided preferentially by the decision maker. If no preference information is supplied by the decision maker, the structurally located reference points can be implemented, as in the systematic approach proposed by Das and Dennis (1998) where reference points are distributed uniformly on the hyperplane for diversity maintenance. In this approach, reference points are placed on a normalized hyperplane with intercept points equal to one on each objective axis. The total number of reference points H is determined using d number of divisions for each objective in the m-objective problem, as given in Equation 4.1. An example reference point set structure for four divisions in a three-objective problem is presented in Figure 4.2.

$$H = \binom{m+d-1}{d} \tag{4.1}$$

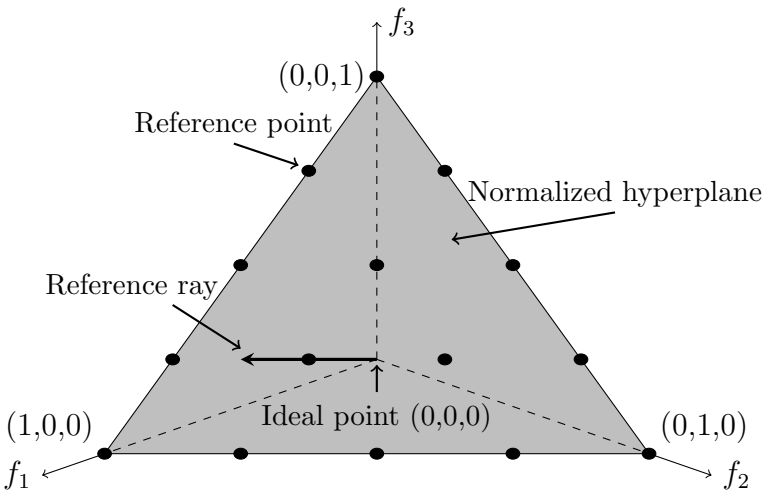

Figure 4.2. Reference points on the normalized hyperplane $d = 4, m = 3$.

The hyperplane creation procedure takes the population $\mathcal{S}_t$ as input and gives the reference point set as output. For a problem with all minimization objectives, the ideal point y^I is determined from $\mathcal{S}_t$ by finding the minimum values observed in $\mathcal{S}_t$ for each objective function and creating the vector composed of these values. Then, objective vectors of solutions in $\mathcal{S}_t$ are transformed by subtracting the ideal vector from them. In this way, the ideal vector becomes the origin point and the objective vectors of the solutions in $\mathcal{S}_t$ always have non-negative values. For each objective value, at least one solution in $\mathcal{S}_t$ is guaranteed to have a corresponding objective value that equals zero.

Second, extreme points of $\mathcal{S}_t$ for each objective axis are identified by finding a solution with a minimum achievement scalarizing function (ASF) value. It should be noted at this point that these extreme points have different meanings than the extreme points of a feasible region or the extreme supported points of a MOCO. ASF is taken from Miettinen and Mäkelä (2002) but not from Deb and Jain (2013) due to differences in the definitions given in these two articles.

ASF using a weight vector $\vec{w}$ is defined for a solution $\vec{x}$ as in Equation 4.2.

$$ASF(\vec{x})_{\vec{w}} = \max_{i=1,\ldots,m} w_i f_i(x_i) \tag{4.2}$$

By using unit vectors as the weight vectors, an extreme solution is obtained for the corresponding objective axis. In other words, extreme solutions for the objective axes are those that minimize the corresponding ASF. Eventually, m extreme solutions should be obtained.

These m extreme solutions are used to obtain the hyperplane equation using Gaussian elimination. The generated hyperplane contains all the extreme solutions. After that, each objective axis is normalized using the intercepts of the hyperplane. Objective space, therefore, solutions in $\mathcal{S}_t$ are transformed by dividing each objective axis by the corresponding intercept value. Thus, the intercept values and the total coordinates of all points on the hyperplane become equal to one. This procedure is illustrated in Figure 4.3.

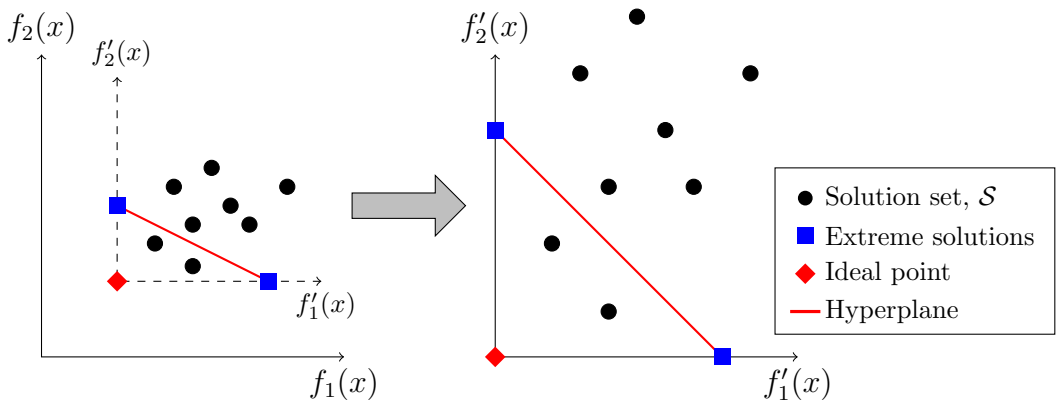

Figure 4.3. Construction of the hyperplane in normalized objective space.

An important note to take is the possibility that a single solution in $\mathcal{S}_t$ may become the extreme solution for multiple objective axes. In this case, an infinite number of hyperplane equations can be created due to this degeneracy, and therefore the reference point set becomes unstable. Another important remark is that the constructed hyperplane might have negative intercepts in some objective axes. After the hyperplane is constructed and the objective space is transformed, the reference points are identified and placed on the normalized hyperplane. As a result, the sum of the coordinates of all reference points is equal to one. This transformed objective space is used to determine which solution in $\mathcal{S}_t$ gets associated with which reference point.

In the association phase, the reference rays corresponding to each reference point are defined by starting from the origin point and passing through the corresponding reference points. Then, distance between the solutions in $\mathcal{S}_t$ and the reference rays is calculated. Each solution in $\mathcal{S}_t$ becomes associated with the reference point whose reference ray is the closest to this particular solution. It is important to emphasize that a reference point may have more than one solution associated with it, or similarly there may be no solution associated with it. The number of solutions associated with the reference point j is called the niche count ρ_j. At the end of the association phase, the reference point set $\mathcal{R}$ is said to consist of two mutually exclusive sets, associated reference set $\mathcal{R}_A$ and empty reference set $\mathcal{R}_E$. The association of the solutions set with the reference point set is illustrated in Figure 4.4 for an example with two-dimensional objective space. It should be noted that not all solutions need to be "behind" the fixed hyperplane.

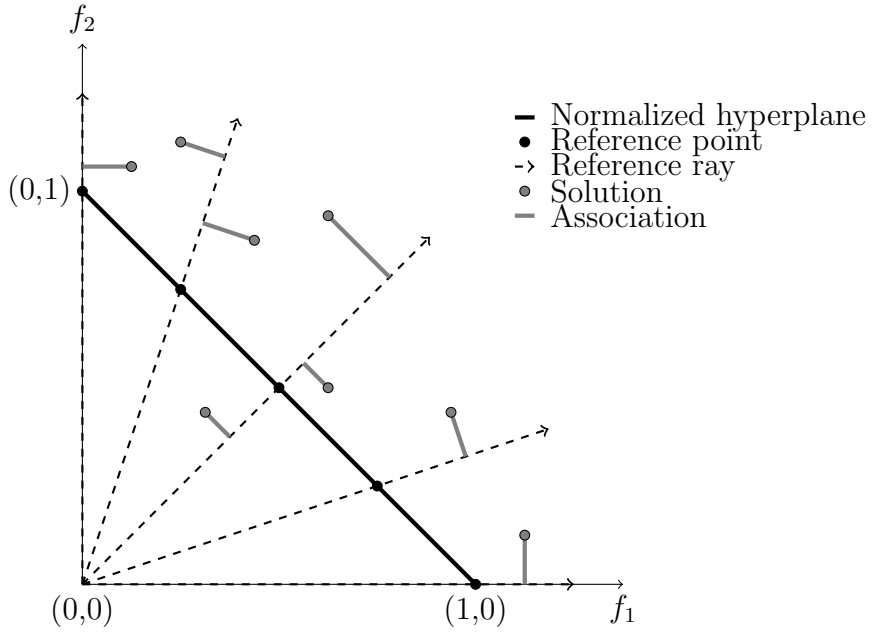

Figure 4.4. Association of the solution and reference point set.

A niche-preserving operation is followed to determine which solutions at the non-domination level l are added to the next generation population. Solution selection begins with reference points with the smallest niche counts. Each reference point adds its closest associated solution to $\mathcal{S}_t$, if it has not yet been added. If all reference points

are taken into account and more solutions need to be added, the closest second solutions are taken into account. This procedure is followed until all vacancies within $\mathcal{S}_t$ are filled and the next generation population of size equal to the predetermined population size parameter is formed.

NSGA-III has $\mathcal{O}(N^2m)$ complexity and does not require any additional parameters other than usual genetic parameters except for division number d and number of reference points H. The population size parameter is made dependent on the number of reference points. To avoid idle reference points, it is preferred to have more solutions than the number of reference points, and the algorithm is considered successful if it does not require a large population size. With these in mind, it is generally accepted to work with a population size equal to the number of reference points. In this way, the purpose of the algorithm is to achieve niche counts equal to one for all reference points.

Problems Encountered in NSGA-III and Proposal to Use a Fixed Hyperplane. In NSGA-III, as the existing population evolves with each iteration, the equation defining the hyperplane containing all reference points changes. Although the positions of the reference points remain unchanged relative to the hyperplane, it is claimed that altering the hyperplane equation makes the necessary adaption for the reference points to guide the search.

However a number of drawbacks of this implementation have been observed. For instance, since the members of the population in the early stages of the evolution might be all dominated, the hyperplane constructed from these solutions can be misleading. As a result, the reference points might be poorly structured and the algorithm may proceed in the wrong directions. In addition, when the hyperplane changes with the evolving population, the locations of the entire reference point set shift drastically and the sorting procedure used so far now becomes useless. In other words,it is highly probable that instabilities of the reference points and the association of population members disrupt the optimization efforts.

Furthermore, degeneracy is observed occasionally during hyperplane construction. This degeneracy may occur due to failing to obtain m distinct extreme solutions to construct the hyperplane or obtaining a hyperplane with negative intercept values for some objective axes. Finally, the construction of the hyperplane in each iteration imposes a heavy computational burden and makes the algorithm impractical for MaOPs.

A fundamental novelty in the proposed reference set based sorting algorithm compared to NSGA-III is the way to construct the hyperplane. To avoid the instabilities mentioned above, a fixed hyperplane is constructed at the very beginning of the algorithm and used throughout the entire evolutionary process contrary to the original approach. By producing a hyperplane wide enough to include all possible feasible solutions and using this hyperplane throughout evolution, the mentioned problems with the NSGA-III are expected to vanish. It is observed that the ultimate hyperplane obtained from the extreme points of all feasible solutions will never require an update when new solutions are introduced. When these extreme solutions are seeded in the initial population, the widest possible hyperplane can be obtained at the beginning of the algorithm.

An important remark is that the extreme solutions used to construct the hyperplane are not obtained by calculating some kind of ASF. These extreme solutions are actually the optimal solutions for separate objectives, thus individual minima. These solutions can also be found from the set of solutions called lexicographic optimal solutions. This means that m different objectives must be solved optimally in advance, each using a distinct lexicographic ordering. As a result, it can be said that obtaining the fixed hyperplane requires that the single-objective version of the problem be solved at reasonable times. It should be noted that the algorithm becomes hybridized in this way. When this is not possible, a heuristic approach can be implemented and near optimal solutions can be used. In short, the time consumption of the proposed algorithm should include the computation times required to obtain m single-objective optimal/near optimal solutions. In spite of this, using a fixed hyperplane shortens the computation time of the reference set based sorting algorithm. Instead of finding the

ideal solution, extreme solutions, equation and the intercepts of the hyperplane in each iteration; all these steps are taken only once in the beginning. Most importantly, the performance is improved since the algorithm does not have to search for the hyperplane over the course of generations, while a good one is readily available from the beginning.

4.2. Genetic Operators

The reference set based sorting, which is the main component of the algorithm, aims to provide sufficient representation for all reference points to achieve an all-embracing Pareto front approximation. The rest of the genetic operators and mechanisms within the proposed MaOEA are designed in a way to complement and sustain this endeavor to enhance its power and success.

This section describes the genetic operators and the key features of the proposed MaOEA. Parameters and operator schemes as well as the pseudo-codes for the key algorithm segments are going to be presented in the order in which they are executed in each iteration of evolution. It should be noted that prior to the beginning of the evolutionary process, single-objective problems must be solved and the initial population must be generated. Members of the initial population are randomly generated and repaired if they happen to be infeasible, except for the seed solutions found by solving the single-objective problems.

In this section, the parent pair selection mechanism is defined in Section 4.2.1. The path relinking recombination scheme is described in detail in Section 4.2.2. Mutation and immigration operators are explained in Section 4.2.3. Repair and local improvement procedures are provided in Section 4.2.4. Finally, stopping criteria and external population structure are given in Section 4.2.5.

4.2.1. Selection

Instead of an independent selection procedure that first creates a parent pool and then randomly selects parents from the pool, a selection mechanism has been

devised to facilitate the crossover scheme by choosing parents deliberately as couples that will enter recombination together. The target is to associate an equal number of solutions with each reference point, e.g., when the population size is specified equal to the number of reference points, all reference points need to have niche counts equal to one. Empty regions where reference points do not have any representative solutions must be populated. The target of the selection mechanism becomes to generate a parent couple whose offspring will be associated with a reference point whose niche count is equal to zero.

On that account, two solutions associated with reference points adjacent to a randomly selected empty reference point (target reference point) are selected as a parent couple. The neighborhood structure of the reference points is illustrated in a three-dimensional hyperplane in Figure 4.5. Six points within the 15 reference points are neighbors $\mathcal{R}_N$ of the reference point r. It is anticipated that a path built between these pairs during recombination is envisaged to generate an offspring suitable for the purpose.

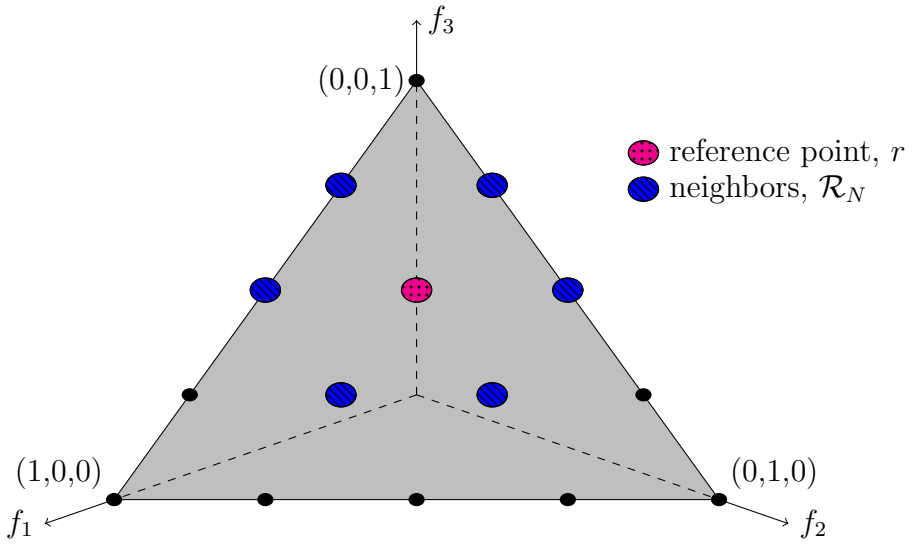

Figure 4.5. Reference point neighborhood structure, $d = 4, m = 3$.

On the other hand, there are some conditions required to successfully identify such parent pairs. For instance, to avoid short and inadequate paths, the parent pair cannot be formed by the same reference point or reference points that are themselves

adjacent to each other. When a suitable set of reference points is found, solutions that are associated with a minimum distance to the corresponding associated reference points are selected as the parent pair. In case the second parent cannot be found using this rule, it is selected by means of a binary tournament scheme. The decision in the binary tournament is based on the similarity/dissimilarity to the already selected parent, i.e. Hamming distance (Ribeiro and Resende, 2012).

In the crossover schemes, an offspring is produced by a parent pair, and the number of offspring generated in each iteration should be equal to the size of the current population in the reference set based sorting phase. As a result, not all the necessary parent pairs can be generated by the selection mechanism described above.

The rest of the parent pairs are generated by a variant of the binary tournament procedure that is modified to overcome the challenges posed by MaOPs. In this binary tournament procedure, the first parent is selected by randomly picking two solutions and comparing them according to the order they received in the reference set based sorting of the previous iteration. In this manner, solutions favored by the reference set based sorting algorithm have a higher chance of entering the recombination procedure. The second parent of the pair is selected by randomly picking two solutions and comparing them according to how different genotypes they have from the already selected first parent. At the end, the selection mechanism provides the recombination operator with the parent pairs and their target reference points. The pseudo-code of the parent pair selection, where the dissimilar candidate for second parent is preferred in the binary tournament scheme, is provided in Algorithm 2 in Figure 4.6.

In the development phase of the proposed algorithm, two alternative methods are considered to choose between the two candidates to form a parent pair with the already selected first parent. The first is to choose the similar candidate to recombine similar parents to obtain a “close” parent pair. On the other hand, by choosing the dissimilar candidate, a “far” parent pair can prevent possible premature convergence and create pressure for improvement by providing a diverse gene pool. As a result, both alternatives are tested in numerical experimentation in Section 5.2.

Algorithm 2: ***Select_Pairs***
Input: $population : \mathcal{P}$, $referencePoints : \mathcal{R} = \mathcal{R}_A \cup \mathcal{R}_E$
1. Set $parentPairs : \mathcal{P}_{pairs} = \emptyset$
2. Set $guidingReferencePoints : \mathcal{R}_G = \emptyset$
3. **For each** reference point r **in** $\mathcal{R}_E$:
4. Set $neighbors : \mathcal{R}_N$ from $\mathcal{R}_A$ in the neighborhood of r
5. **If** $\mathcal{R}_N.Count \geq 2$:
6. Choose two references points: r_1 and r_2 from $\mathcal{R}_N$
7. Set the closest associated solution of r_1 as p_1
8. Set the closest associated solution of r_2 as p_2
9. Add (p_1, p_2) to $\mathcal{P}_{pairs}$
10. Add r to $\mathcal{R}_G$
11. **Else if** $\mathcal{R}_N.Count = 1$:
12. Choose the reference point: r_1 from $\mathcal{R}_N$
13. Set the closest associated solution of r_1 as p_1
14. Choose two candidate solutions: s_1, s_2 randomly from $\mathcal{P}$
15. Set p_2 by $BinaryTournament(s_1, s_2)$ based on $Dissimilarity\_with\_p_1$
16. Add (p_1, p_2) to $\mathcal{P}_{pairs}$
17. Add r to $\mathcal{R}_G$
18. **Else if** $\mathcal{R}_N.Count = 0$:
19. **Continue**
20. **End if**
21. **End for each**
22. **While** $\mathcal{P}_{pairs}.Count < \mathcal{P}.Count$:
23. Choose two candidate solutions: s_1, s_2 randomly from $\mathcal{P}$
24. Set p_1 by $BinaryTournament(s_1, s_2)$ based on $Reference\_Set\_Based\_Sorting$
25. Choose two candidate solutions: s_1, s_2 randomly from $\mathcal{P}$
26. Set p_2 by $BinaryTournament(s_1, s_2)$ based on $Dissimilarity\_with\_p_1$
27. Add (p_1, p_2) to $\mathcal{P}_{pairs}$
28. **End while**
Output: $\mathcal{P}_{pairs}$, $\mathcal{R}_G$

Figure 4.6. Pseudo-code for parent pair selection algorithm.

In summary, the selection of the parent pairs consists of these two methods: reference point based selection and binary tournament. The binary tournament plays a complementary role and two alternative strategies are tested for it. The total number of parent pairs is equal to the population size and is constant throughout the evolutionary algorithm. The relative use of these two methods depends on the number of reference

points whose niche counts are equal to zero and therefore varies throughout the search. In addition to this variability, it is not guaranteed that for each reference point with zero niche count, there will be a suitable parent pair in each iteration. Based on the parameter setting, it can be argued that the majority of the parent pair generation might be handled by the binary tournament. As the evolutionary algorithm advances through iterations, the number of reference points with zero niche counts decreases because it is one of the main goals of the reference set based sorting algorithm.

4.2.2. Recombination

Recombination (crossover) operators, which play a key role in evolutionary approaches, become even more critical in MaOEAs. It is reported that conventional recombination schemes may be proven inadequate in MaOPs, since the offspring whose parents are close to the Pareto optimal front need not to be close to the front themselves. Therefore, devising successful recombination operators is regarded as another challenge of MaOPs (Jaimes and Coello, 2015).

The proposed algorithm employs a reference point guided path relinking as the recombination method. Path relinking suggested by Glover *et al.* (2000) is used to generate new high-quality solutions by exploring paths that connect previously found high quality elite solutions (Ribeiro and Resende, 2012). Starting from one of the solutions as the initiating solution, some move operations in the neighborhood space are performed towards the other guiding solution (Glover *et al.*, 2000). The neighborhood is restricted in such a way that each move introduces an attribute of the guiding solution that is not present in the initiating solution. For example, Beausoleil *et al.* (2008) use back-and-forward path relinking strategy applied to a multi-objective knapsack problem.

The implementation in this thesis uses a randomized mixed path relinking strategy (Glover *et al.*, 2004), in which two paths are initiated simultaneously from both parents. The moves selected at each step are guided by the reference points. The nodes of the path generated at each step correspond to the objective space positions of

candidate solutions for offspring. The path that connects parents is formed by creating new solutions from both parents through some move operations, which makes parents look more similar at every step.

In the path relinking scheme, starting from a random position of the chromosomes, the genes of the two parents at that position are examined. If the corresponding genes of the two parents are different, a flip operation is performed for both parents. Specifically, when using binary encoding as in the knapsack problem, the move corresponds to switching a binary gene in one parent from 0 to 1 and vice versa in the other. When using permutation encoding as in the TSP and QAP, the move for both parents corresponds to the copying of the other parent's gene of that specific point. In Figure 4.7, the move operations are illustrated for binary and permutation encoding.

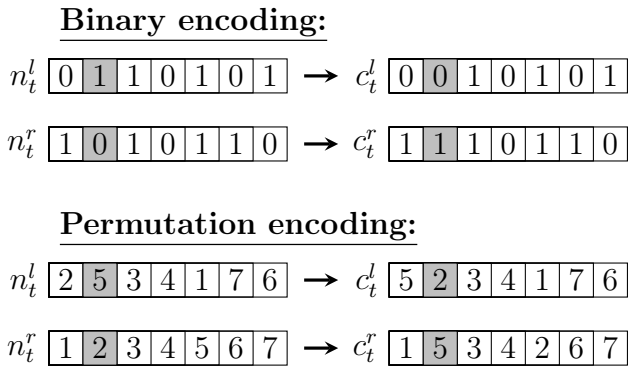

Figure 4.7. Move operations in binary and permutation encoding.

In each more iteration t, the end nodes of the two paths, namely n_t^l and n_t^r, create two candidate solutions, namely c_t^l and c_t^r, respectively. After each move operation, the resulting two candidate solutions are inspected and the move that yields the "better" candidate is performed by the corresponding parent, thereby forming a new node on the path. In the next move iteration, this new node replaces the parent from which it is obtained. Then, the move operation is applied for this new solution along with the solution at the end of the other path. Moves continue until all the genes of the two end points become the same meaning that the full path has been formed. At the end, the last node generated on the path is declared as the offspring solution. It should be noted that the procedure yields one offspring in this path relinking scheme.

In the comparison stage of the two candidate solutions, the feasibility is checked first. If only one candidate is feasible, this solution is chosen as the next node of the path. It must be noted that at most one of the candidates (the one obtained by switching the binary gene of 0 to 1) can be infeasible given the fact that the move is made from two feasible knapsack solutions. In TSP and QAP, both candidate solutions are always guaranteed to be feasible.

When both candidate solutions are feasible, the dominance is checked. This is very likely to be inconclusive when the conventional dominance relationship is applied. As a result, two alternative strategies, which differ in the way that they address this issue, have been developed and tested. Both strategies are named according to how the two candidate solutions in the nodes of a path are compared with each other: "*ϵ-path*" and "*reference point guided path*".

ϵ-path. The first strategy uses one of the dominance rules called ϵ-dominance suggested by Laumanns *et al.* (2002). From two objective vectors $\vec{f}, \vec{g} \in \mathbb{R}^m$, $\vec{f}$ is said to ϵ-dominate $\vec{g}$ for some $\epsilon > 0$, denoted as $\vec{f} >_\epsilon \vec{g}$, if and only if the condition in Equation 4.3 is satisfied for a minimization problem or the condition in Equation 4.4 for a maximization problem.

$$(1-\epsilon) f_i \leq g_i \quad \forall i \in \{1, ..., m\} \tag{4.3}$$

$$(1+\epsilon) f_i \geq g_i \quad \forall i \in \{1, ..., m\} \tag{4.4}$$

When using the ϵ-dominance principle, even if none of the solutions dominate the other, both have a certain ϵ-value, where they ϵ-dominate the other. Whichever solution has the smallest ϵ-value, this solution is better. The main advantage of using the ϵ-dominance principle is that ϵ is no longer a user-defined parameter, similar to the additive ϵ-indicator in Zitzler *et al.* (2003). A comparison using ϵ-dominance for three-objective problem with minimization objectives is demonstrated in Figure 4.8. This modification improves the ability to compare different individuals. It should be noted that the ϵ-domination principle is applied only in recombination, not in selection.

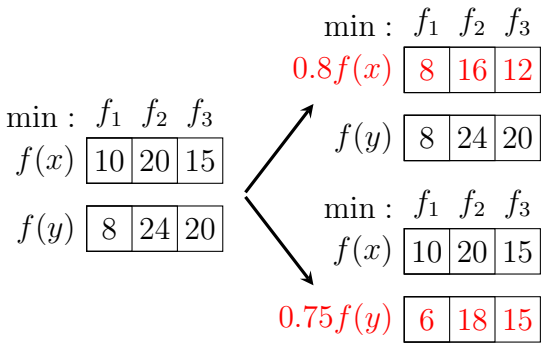

Figure 4.8. Demonstration of ϵ-dominance.

Reference point guided path. In the second path relinking strategy, candidate solutions are compared based on their move-values δ. Denoting the Euclidean distance between the parent node and the guiding reference ray (the ray starting from origin and passing through the target reference point) as d_t, and similarly, the distance between the candidate node and the guiding ray as d_{t+1}, the move-value δ is calculated as their difference.

The candidate solution with a minimum move-value, in other words whose move operation shortens the distance to the guiding reference ray more than the other, is chosen as the next node of the path. An illustrative example of how the move-values are measured is shown in Figure 4.9.

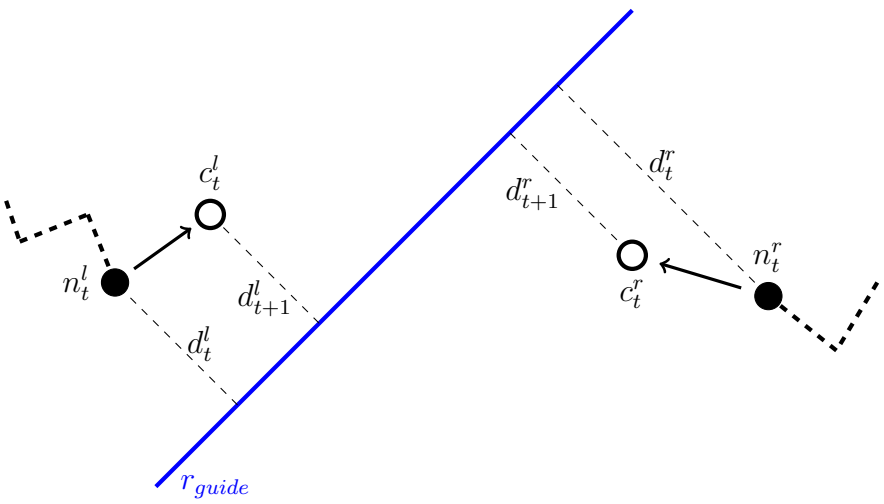

Figure 4.9. Reference point guided path relinking.

Algorithm 3: ***Reference_Point_Guided_Path_Relinking***
Input: $parentPairs : \mathcal{P}_{pairs}$, $referencePoints : \mathcal{R} = \mathcal{R}_A \cup \mathcal{R}_E$, $guidingReferencePoints : \mathcal{R}_G$
1. Set $offspring : \mathcal{P}_{off} = \emptyset$
2. **For each** $parentPair : (p_1, p_2)$ **in** $\mathcal{P}_{pairs}$:
3. Set $r_g = \mathcal{R}_G[0]$
4. **If** $r_g \neq \emptyset$
5. Set n_{total} as total difference in genotypes of p_1 and p_2
6. Set $n_1^l = p_1$ and $n_1^r = p_2$
7. Set $path = \emptyset$
8. **For** t **in** $(1, n_{total})$:
9. Apply $Move\_Operation(n_t^l, n_t^r)$ to obtain (c_t^l, c_t^r)
10. $\delta^l = Euclidean\_Distance(r_g, c_t^l) - Euclidean\_Distance(r_g, n_t^l)$
11. $\delta^r = Euclidean\_Distance(r_g, c_t^r) - Euclidean\_Distance(r_g, n_t^r)$
12. **If** $\delta^l < \delta^r$:
13. Add c_t^l to $path$
14. Set $n_{t+1}^l = c_t^l$ and keep n_t^r as n_{t+1}^r
15. **Else**:
16. Add c_t^r to $path$
17. Set $n_{t+1}^r = c_t^r$ and keep n_t^l as n_{t+1}^l
18. **End if**
19. Find $r_{associate}$ as association of the most recent node in $path$
20. **If** $r_{associate} \in \mathcal{R}_E$: **Break**
21. **End for**
22. Select s_{off} as the most recent node in $path$
23. $\mathcal{R}_G = \mathcal{R}_G \backslash r_g$
24. **Else**:
25. Select s_{off} from a generic crossover between (p_1, p_2)
26. **End if**
27. Add s_{off} to $\mathcal{P}_{off}$
28. **End for each**
Output: $\mathcal{P}_{off}$

Figure 4.10. Pseudo-code for reference point guided path relinking.

Another important aspect of the proposed path relinking crossover strategies is that it is not always required to form the full path. After each move, if the most recent node happens to be associated with a reference point (does not have to be the guiding reference point) whose niche count equals to zero, the corresponding solution

is stated as the offspring and crossover is terminated. In this way, reference points with zero niche count are removed and the time requirement of the crossover operation is decreased by avoiding to form the full path. The pseudo-code of the reference point guided path relinking recombination scheme is shown in Algorithm 3 in Figure 4.10.

As a final note, when the binary tournament is used to determine the parent pair in the absence of guiding reference points, a crossover scheme which is computationally less expensive is preferred to substitute the path relinking. (e.g. uniform crossover for the knapsack problem, two-order crossover for TSP and QAP). Selection mechanisms and recombination schemes are summarized in Table 4.1.

Table 4.1. Selection and recombination alternatives.

Selection	**Recombination**
Reference point based + ↪ Binary tournament 1: *close* ↪ Binary tournament 2: *far*	↪ Path relinking 1: ϵ-*path* ↪ Path relinking 2: *guided-path*
↪ Binary tournament 1: *close* ↪ Binary tournament 2: *far*	Uniform (binary encoding) Two-order (permutation encoding)

4.2.3. Mutation and Immigration

In evolutionary algorithms, the mutation operator is widely used to prevent premature convergence and provide a diverse population. In the implementation of the knapsack problem, the proposed MaOEA uses bit-flip mutation. Following recombination, each newly generated offspring undergoes a bit-flip mutation and randomly selected binary genes are reversed using a predetermined mutation rate. After the mutation, each offspring gets associated with a reference point, and if the offspring is infeasible, it enters a repair procedure. Among the feasible offspring solutions, those randomly selected ones enter the local improvement procedure based on the local improvement rate parameter.

For TSP and QAP, a variation of the mutation operator, the immigration operator aims to increase diversity in the population by bringing new solutions that are significantly different from existing solutions. In each iteration, a predetermined percentage of offspring is produced by the participation of immigrants. For the production phase of immigrants, Ahuja *et al.* (2000) propose a process using historical frequency information similar to the long term memory in the tabu search literature (Glover and Laguna, 1998). All past and present information regarding how many times each gene is assigned to each location of the chromosome is stored in the $n \times n$ matrix called the population history. Higher values in the population history matrix indicate that the corresponding assignment has been observed in many individuals whereas lower values indicate fewer individuals in the past and existing populations have the corresponding assignments in their permutations. The methodology targets to create new assignments with low values in the population history matrix by selecting genes in random order and distributing them to the lowest possible chromosome location assignment. This procedure allows searching unexplored regions of the objective space.

It is observed that immigrants obtained with this methodology are likely to be dominated and be expelled in the ranking phase. Therefore, all immigrant solutions are generated randomly and subjected to a local improvement operation immediately after they are created. In this way, unexplored regions of the objective space are extensively investigated and successfully incorporated into the population, which improves overall quality.

4.2.4. Repair and Local Improvement

Due to the capacity constraints in the knapsack, it is possible to produce infeasible offspring during the recombination process. To repair infeasible offspring two approaches are tested. The first approach is a "*greedy repair*" as in Zitzler and Thiele (1999). In this repair strategy, an infeasible solution is turned into a feasible solution by taking into account all the items included in the knapsack. The calculation is done to decide which items can be removed with a minimum loss in profit values in the objective functions. This means that for each item, the following value is calculated

in Equation 4.5 and all items are sorted in ascending order based on their q_j values. Items are removed until the capacity constraints are satisfied.

$$q_j = \max_{i,k} \left\{ \frac{p_{ij}}{w_{jk}} \right\} \quad \forall j \tag{4.5}$$

The repair procedure is followed by a similar greedy local improvement procedure. Unlike the repair procedure, items that are not included in the knapsack are taken to account in the local improvement procedure. These items are sorted in descending order of the calculated values as in Equation 4.5 and added accordingly, unless the capacity constraints are violated.

An alternative strategy is reference point guided weighted repair. For both repair and local improvement procedures, the items are sorted by taking the reference points into consideration. In this way, both of repair and local improvement procedures are guided by the reference point to which the corresponding offspring is associated. When a reference point $\vec{r} = [r_1, ..., r_m] : \sum_{i=1}^{m} r_i = 1, \forall r_i \geq 0$ is used to guide procedures, knapsack items are sorted by calculating the $q_j(\vec{r})$ values for each item as in Equation 4.6. In this way, the coordinate of the associated reference point is used to assess the importance of the knapsack items.

$$q_j(\vec{r}) = \frac{\sum_i r_i p_{ij}}{\max_k \{w_{jk}\}} \quad \forall j \tag{4.6}$$

High $q_j(\vec{r})$ value in the local improvement procedure indicates that the inclusion of item j into the knapsack provides great benefit in the direction of $\vec{r}$ whereas low $\vec{r}$ value in the repair procedure means that item j can be removed from the knapsack with no major loss that would take away from $\vec{r}$. The repair and local improvement procedures obtained in this manner are called "*weighted repair*".

For TSP and QAP, repair method is not required, and a greedy two-exchange neighborhood local search is used as the local improvement procedure. In this local search, neighboring solutions are generated by swapping two genes in the permutation

of an existing solution. If a candidate neighbor dominates the existing solution, the existing solution is converted to that candidate neighbor and the same local search procedure is restarted. This local improvement process continues until a nondominated individual in the neighborhood is reached.

Although repairs are made to every infeasible offspring immediately after the mutation step, it is not certain that every offspring enters a local improvement procedure. The proportion of offspring entering this procedure and the proportion of genes entering the mutation procedure will be investigated in the numerical analyses regarding the development stages of the proposed algorithm in Section 5.2.

4.2.5. Stopping Criteria and External Population

There are several alternatives to terminate the evolution in the proposed MaOEA. Limitations can be applied to the number of iterations, solution generation or repair calls, or there might be time limits. Based on the preliminary analysis, all algorithms are terminated when $500 \times N$ (population size) repair calls are made in the knapsack problem, to obtain a fair comparison (Zhang and Li, 2007). In the absence of repair calls, as in the cases of TSP and QAP, the algorithms are terminated when the number of iterations reaches 500.

Another important feature of the proposed MaOEA is keeping an external population which is basically an archive structure that stores all the nondominated points created throughout evolution. Archive mechanism is reported to be very useful for evolutionary algorithms whose sorting principles are not based on the dominance relationship, e.g., MOEA/D (Zhang and Li, 2007). Although nondominated sorting based algorithms do not have to use external populations in their original framework, it is presumed that archiving is necessary for MaOPs, since the huge number of nondominated points that get eliminated from the internal populations throughout iterations is a waste that may cause loss of important information. The computational effort required to update the external population can be reduced by limiting its size or applying some clustering techniques.

4.3. Constraint Handling Techniques

One of the main goals of this thesis is to develop a MaOEA that can cope with various different optimization problems. Different combinatorial problems have their own specific objective function and constraint structures that need be handled carefully. Differences in problem structure can affect both the design and operators of both MOEAs and MaOEAs. In some problem structures, repair operation may not always be possible or require great computational effort. Moreover, infeasible solutions, although ultimately undesirable, can provide useful information to the population during evolution. This section explains the necessary modifications and alternatives to constraint handling techniques, instead of repairing an infeasible solution as soon as it is created. The multi-constrained MaOKP is chosen for the analysis in this section since it is the benchmark problem which uses a repair procedure.

The comprehensive survey by Coello (2002) provides a discussion of well-known constraint handling techniques used with evolutionary algorithms. The review ranges from elementary changes in penalizing techniques to some sophisticated hybrid methods. Although there are numerous approaches to constraint handling techniques in evolutionary algorithms, most of these techniques can be classified into the following two categories: methods based on preserving the feasibility of solutions and methods based on penalty functions.

In the first category, it is ensured that only feasible offspring solution are created and that individuals always remain feasible. This is done by crossover operators designed to reproduce only feasible offspring or some repair mechanisms. The *FHFR* described above is in this category. Two versions of the repair method called "greedy repair" and "weighted repair" are provided for MaOKP, as described in Section 4.2.4. In both versions, items included in the knapsack are removed until all capacity constraints are met and an infeasible solution is converted into a feasible solution.

In the method of using penalty functions, the constrained problem is transformed into an unconstrained problem. For a particular solution, the amount of violation

is added (or subtracted) to the objective function value as a penalty. In this way, infeasible solutions are allowed to exist in the population, but with a penalty, calculated by adjusting the amount of violation by means of a weight. The penalty is so that the corresponding solution becomes undesirable compared to a feasible solution. In the implementation, the total violation amount is used to penalize all objective functions.

The penalty coefficient can be static or dynamic. In static penalties, the penalty coefficient does not depend on the stage of the algorithm and remains constant throughout the entire evolutionary process. The penalty coefficient should be low enough to allow some infeasible solutions to exist in the population but also high enough to prevent them from outweighing feasible solutions.

It is claimed that dynamic penalties work better than static penalties. By increasing the penalty coefficient over the course of the algorithm, genetic materials obtained from infeasible solutions are used at the beginning of the algorithm and these infeasible solutions are alienated from the population towards to the last generations of the algorithm. When using dynamic penalty approach, it is important to derive good dynamic penalty functions since the approach itself is very sensitive to changes in parameter values. As careful parameter search and tuning is necessary to obtain reliable results, a preliminary study for determining appropriate coefficient for both static and dynamic penalty approaches has been carried out. In the dynamic case, the penalty amount is increased linearly by the iteration count and the age of the corresponding solution.

Another approach for using dynamic penalties is adaptive penalty functions where the penalty amount is controlled by the information obtained from the evolutionary algorithm. In this thesis, the adaptive penalty function used originates from a method called self-adaptive penalty (Woldesenbet *et al.*, 2009). In the self-adaptive penalty approach, the amount of penalty is determined by the number of feasible solutions in the population.

For these three penalty approaches, the transformed objective function for a particular solution $\vec{x}$ is provided in Equation 4.7. Since the objectives are of maximization

type in the multi-constrained MaOKP, the penalty amount, $P(\vec{x})$, is subtracted from all objective functions.

$$\tilde{f}_i(\vec{x}) = \sum_{j=1}^{n} p_{ij}x_j - P(\vec{x}) \quad \forall i \tag{4.7}$$

where

$$P(\vec{x}) = \begin{cases} penaltyCoef \times V(\vec{x}) & \text{static penalty} \quad (4.8) \\ penaltyCoef \times t \times age(\vec{x}) \times V(\vec{x}) & \text{dynamic penalty} \quad (4.9) \\ penaltyCoef \times (1 - r_f) \times V(\vec{x}) & \text{adaptive penalty} \quad (4.10) \end{cases}$$

$V(\vec{x})$ is the total constraint violation amount of a particular solution $\vec{x}$ and it is calculated as in Equation 4.11 :

$$V(\vec{x}) = \sum_{k} \left(\sum_{j=1}^{n} w_{jk}x_j - C_k \right) \tag{4.11}$$

For the dynamic penalty approach in Equation 4.9, the current iteration number t and the age of the corresponding solution $age(\vec{x})$ are used as multipliers.

In adaptive penalty approach in Equation 4.10, the feasibility ratio r_f, is obtained by dividing the number of feasible solutions by the population size. If the number of feasible solutions in the population is low, the coefficient $(1 - r_f)$ approaches to one and infeasible solutions are penalized in proportion to the total amount of constraint violations. In this way, infeasible solutions become undesirable and the algorithm seeks to find feasible solutions. On the other hand, if the number of feasible solutions in the population is high, the coefficient $(1 - r_f)$ approaches to zero and the penalty amount for infeasible solutions decreases to zero. Infeasible solutions with good objective function values become desirable. The feasibility ratio value appears to fluctuate over the duration of the algorithm and converge after several iterations.

The last used technique is different from the others since no transformation has been made to the objective functions of infeasible solutions. In this method, feasible

solutions are always preferred to those that are infeasible. This technique is similar but not the same with superiority of feasible solution method in Qu and Suganthan (2011). In the implementation, feasible solutions always dominate infeasible solutions. If there is a feasible solution in the population, it means that infeasible solutions can never have rank one. Comparison of infeasible solutions among themselves is made by their violations. Infeasible solutions are mapped to a space called violation space where each dimension is represented by a constraint. In the violation space, objective functions become the minimization of violations and the nondomination principle is applied accordingly when two infeasible solutions are compared.

In numerical analysis on this subject in Section 5.3, the performance of all the constraint handling techniques mentioned in this section is evaluated. The names of these six alternatives are (i) *greedy repair*, (ii) *weighted repair*, (iii) *static penalty*, (iv) *dynamic penalty*, (v) *adaptive penalty* and (vi) *feasible preferred*.

Methods (i) and (ii) always ensure the feasibility of solutions using different repair procedures. For these methods, the proposed MaOEA is applied directly without any changes. For the rest of the methods which allow infeasible solutions, the path relinking recombination scheme is modified. When comparing two candidate solutions during the construction of the path, regardless of the infeasibility of a candidate, it is compared based on the relevant criteria (i.e. distance from the guiding reference point, or dominance). Additionally, when an infeasible solution is eliminated, greedy repair procedure is performed before the solution is checked for entering the external population. Similarly, all infeasible solutions in the final population enter greedy repair procedure in termination of the algorithm.

In methods (iii) to (v), the objective functions of the infeasible solutions are transformed using some penalty functions. As a result, their location on the hyperplane and their association with the reference points depend on how the penalty amount is defined during the algorithm. In dynamic and adaptive penalty approaches, the location and association of an infeasible solution change as the penalty amount changes. It is suspected that it may cause disorder for the association results and two alternatives

are experimented in numerical analysis. Based on some preliminary analysis, it is decided to map infeasible solutions according to their original objective function values so that their association remains the same during evolution.

4.4. Alternative Positioning for the Reference Set

This section includes the search for alternative positioning methods for the reference point set. Since the reference set based sorting constitutes the core of the proposed MaOEA, alternative ideas that can contribute to its quality should be carefully investigated. It should be noted that the solutions are first sorted according to the nondomination principle and the reference set based sorting is applied after the fronts are formed. In this way, reference set based sorting is used as a tie breaker for the solutions of the same rank. On the other hand, since it is one the main challenges observed in many-objective optimization, the number of nondominated solutions can be large and reference set based sorting becomes a decisive stage in the assessment of solutions. A successful Pareto approximation and the diversity of its members heavily rely on the reference set based sorting. As a result, its attributes and requirements are always kept mind when developing operators and mechanisms.

In the implementation, the population size and number of reference points are fixed as equal. In this way, the ultimate goal of the algorithm becomes associating one solution for each reference point. As a result, the distribution of the reference points governs the distribution of the population in the objective space.

No matter how well the reference points spread, they may still fail to represent the Pareto front effectively. In many real-life problems, Pareto optimal fronts have irregular shapes including degeneracy, discontinuity and nonlinearity (Ishibuchi *et al.*, 2019). Combinatorial optimization problems with discrete feasible solutions spaces are examples of problems in nature. Due to these irregularities in Pareto front shapes, some reference points will have no solutions that can be associated with them. Since it is often not possible to recognize this landscape in advance, it may be useful to update the positions of the reference points during the search. In other words, although the

systematic mesh approach proposed by Das and Dennis (1998) is used at the beginning, this can be changed during the course of the evolutionary algorithm. The update mechanism must be generic so that it can be applied to various different types of problems.

As a matter of fact, while using the proposed algorithm in its basic form (*FHFR*), it seems that a large number of reference points remain unassociated during the algorithm even though the complementary selection and recombination schemes attempt to fill these reference points. This is simply because there is no feasible nondominated solution that can be associated with one of these reference points. This means that the algorithm must be equipped to detect these reference points and take the necessary measures to avoid further effort. In the implementation of this thesis, when a reference point remains unassociated for a certain number of iterations, it is concluded that its surrounding region in objective space is void and its position needs to be updated.

Removing these reference points may be a straightforward approach. Two alternatives can be considered at this stage. Since the population size is determined equal to the number of reference points, the decrease in the number of reference points implies a decrease in the population size. In this way, the goal of holding one associated solution for each reference point is maintained and a significant improvement in time consumption is achieved. On the other hand, a decrease in population size can be expected to lead to poor performance. As a result, fixing the population size while reducing the number of reference points may be a second alternative. In this way, a smaller reference point set can reduce time consumption while preventing a possible deterioration in performance. Although these alternatives do not promise much improvement in performance, they are retained to check if they can lower the time requirement without a deterioration in quality. These methods are referred to as "*expiring references and population*" (ER1) and "*expiring references*" (ER2) in the numerical results.

More promising approaches are developed by updating the locations of reference points with void association rather that discarding them. While doing this, it is ensured that the reference points are always on the original fixed hyperplane.

Reference set update procedure works on the reference set as the second stage of the algorithm after the reference set based sorting stage works on the solution set to establish the population of the next iteration. This procedure, which relocates some reference points based on the information gathered during the first stage, is designed as an independent module so that it can be used in a plug-in manner. It must be noted that the reference set update procedure does not necessarily need to be applied for every iteration. When this stage is deactivated, the algorithm with a fixed reference set is obtained.

In fact, activating the reference set update procedure for the first time is delayed by some "learning period" (Asafuddoula *et al.*, 2017) thus allowing the population that is mostly composed of dominated solutions to mature in the initial iterations. The population is said to be sufficiently mature when the number of solutions in its first front has grown large enough such that nondominated points begin to be truncated. In other words, reference set update procedure is applied for the first time when the front count in the current population drops to one. In this manner, solution associations to the initial uniform map of reference points get settled to convey meaningful information. In many-objective problems, the front count rapidly decreases to one as the number of nondominated points is extremely high.

By the same line of reasoning, after each reference set update epoch, the reference set is allowed to settle for some time until the evolution process becomes more stable in terms of how the solutions are associated with the reference points. Preliminary experiments supported this concept showing that when the update procedure is applied at each iteration, a deterioration in performance occurs due to its overuse. As a result, the update procedure is applied only at the end of "cycles". Based on the preliminary numerical studies, when there is no change in how the solution set is associated with the reference point set for 10 iterations, the cycle is said to be complete and the reference point update procedure is activated.

In the implementation of this thesis, when it is detected that no solution is associated with a particular reference point throughout a cycle, this reference point is

relocated near another densely populated reference point. In this manner, it is hoped that the new reference point relieves the crowd and create balanced niche counts for the reference set based sorting algorithm. When a solution is chosen randomly and its associated reference point is selected, similar to a roulette wheel selection mechanism, reference points with higher association counts are given a higher chance. The new reference point is positioned between the selected reference point and one of its randomly chosen neighbors. The exact location is given by a randomly generated proportion that determines the convex combination of selected neighboring reference points. In this way, the new reference point is also guaranteed to be located on the fixed hyperplane. This approach is referred to as "*mobile references*" in numerical analysis and the resulting proposed MaOEA will yield the second version of the proposed algorithm with fixed hyperplane and mobile reference points: *FHMR*. The pseudo-code of the reference set update procedure in *FHMR* is shown in Algorithm 4 in Figure 4.11. An important remark on the reference set update procedure is that it is independent of the problem and the same in all benchmark problems.

Algorithm 4: ***Reference_Set_Update_Procedure*** **(for *FHMR*)**
Input: $population : \mathcal{P}$, $referencePoints : \mathcal{R} = \mathcal{R}_A \cup \mathcal{R}_E$, $Association(\mathcal{P}, \mathcal{R})$
1. Detect empty reference points throughout the cycle $\mathcal{R}_{DEL}$ from $\mathcal{R}_E$
2. **For each** reference points r **in** $\mathcal{R}_{DEL}$:
3. Select a solution s randomly from $\mathcal{P}$
4. Set $r_1 = Association(s, \mathcal{R})$
5. Select a reference point r_2 randomly from the neighborhood of r_1
6. Relocate $r = \alpha r_1 + (1 - \alpha) r_2$ using a random $\alpha : 0 \leq \alpha \leq 1$
7. Update the neighborhoods of r, r_1 and r_2
8. Update $Association(\mathcal{P}, \mathcal{R})$
9. **End for each**
Output: $\mathcal{R}$, $Association(\mathcal{P}, \mathcal{R})$

Figure 4.11. Pseudo-code of the reference set update procedure in *FHMR*.

4.5. Co-evolutionary Reference Set Algorithm

While the reference point update procedure in Section 4.4 occasionally relocates some of the reference points, the co-evolutionary approach defines a coherent and natural mechanism that allows for more flexible adaptation of the entire reference set. In the

co-evolutionary structure of the proposed MaOEA, the solution set and the reference point set are defined as two concurrently evolving populations. Solutions are evaluated based on how strongly they are associated with the reference point set, and reference points evolve based on how many solutions are associated with them. The evolution of a reference point corresponds to an update of its location on the fixed hyperplane. The co-evolutionary structure shows both cooperative and competitive aspects, as the balance between cooperation and competition is necessary to avoid stagnation in the algorithm and to ensure that the co-evolutionary framework is self-adaptive. It is also independent of the problem and the same in all benchmark problems.

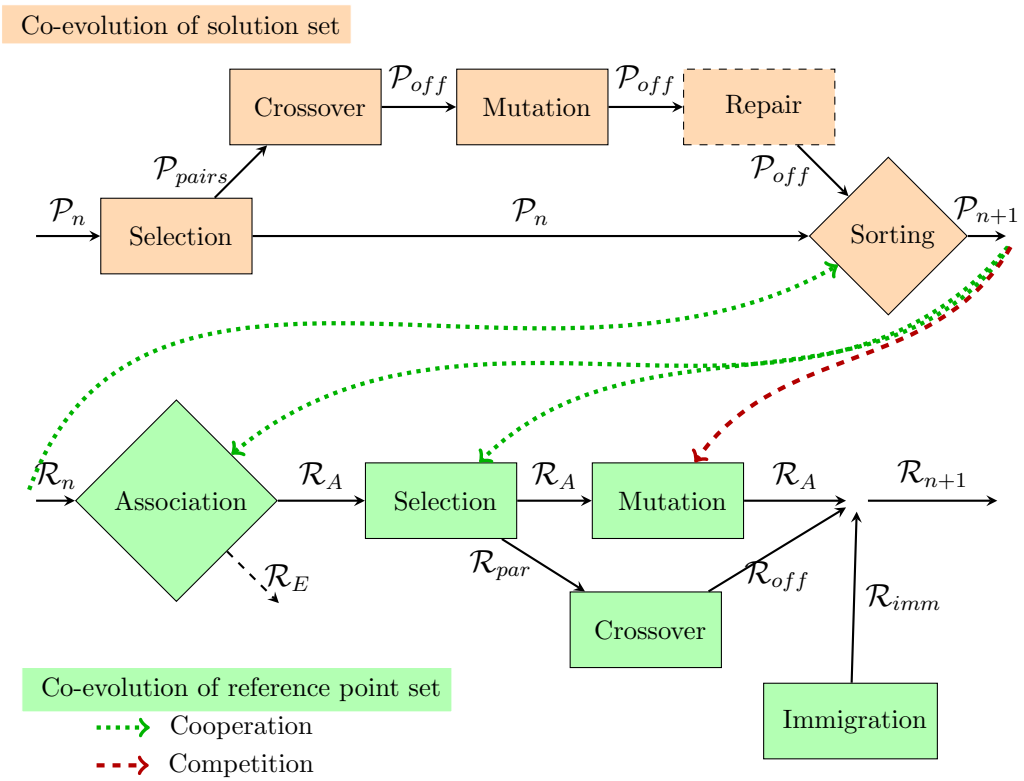

Figure 4.12. Flow chart of the co-evolution of solution and reference point sets.

In general, the two species evolve in a cooperative manner, and the members of both sets are rewarded if they manage to associate decisively with the counter set. On the other hand, they also involve in competition to diversify the search area. The diagram in Figure 4.12 displays the sequential progress of evolution processes and their co-evolutionary relationship. All interactions, except for the mutation of the reference set, include cooperative features. All these interactions are essential to improve the

performance of both species. It is worth mentioning that the evolution of the two sets have different paces, i.e. while the evolution of solution set occurs at each iteration, the evolution of reference set is activated only at the end of the cycles.

The reference set evolution resembles to a classic evolutionary approach with typical genetic operators such as selection, crossover, immigration and mutation. The pseudo-code for the main framework of the evaluation of the reference set is presented in Algorithm 5 in Figure 4.13.

Algorithm 5: Reference_Set_Co-evolution (for *FHCo*)
Input: $referencePoints : \mathcal{R} = \mathcal{R}_A \cup \mathcal{R}_E$, $Association(\mathcal{P}, \mathcal{R})$
1. $\mathcal{R}_{par} :=$ **Selection**$(\mathcal{R})$
2. $\mathcal{R}_{off} :=$ **Crossover**$(\mathcal{R}_{par})$
3. $\mathcal{R}_{img} :=$ **Immigration**
4. **Mutation**$(\mathcal{R}_A)$
5. $\mathcal{R} \leftarrow \mathcal{R}_A \cup \mathcal{R}_{off} \cup \mathcal{R}_{img}$
6. Update $Association(\mathcal{P}, \mathcal{R})$
Output: $\mathcal{R}$, $Association(\mathcal{P}, \mathcal{R})$

Figure 4.13. Pseudo-code of the reference set co-evolution in *FHCo*.

It is important to recall that in the parametric setting, the number of reference points is equal to the population size, so each reference point is intented to be exactly associated with one solution. The number of solutions associated with a reference point is denoted as its niche count. It is aimed to balance niche counts. The algorithm tries to fill empty reference points if possible. As a result, the reference point set is divided into two subsets: associated reference points $\mathcal{R}_A$ and empty reference points $\mathcal{R}_E$. A predefined proportion of associated reference points mutates and changes their locations on the hyperplane to force the evolution process to search diverse regions of the objective space. Empty reference points on the other hand, are replaced by newly generated reference points. This can happen either by introducing new offspring or with immigrated reference points. $\mathcal{R}_E$ will be completely replaced by these new reference points.

4.5.1. Selection in Co-evolution

Binary tournament is used for selection. Two candidates are randomly selected from the reference point set, then the one with higher niche count is preferred. In the case of a tie, the distance to the closest solution is measured and the one with greater distance is selected. In order to give each reference point a chance, the choice for the second parent is random.

4.5.2. Crossover in Co-evolution

The crossover scheme involves a convex combination of the locations of the two parent reference points in the objective space to generate the location of an offspring reference point. By denoting the locations of the parent points as $L(\vec{r}_x)$ and $L(\vec{r}_y)$, the location of the offspring reference point $L(\vec{r_o})$ is calculated by Equation 4.12.

$$L(\vec{r}_o) = \alpha L(\vec{r}_x) + (1 - \alpha)L(\vec{r}_y) \tag{4.12}$$

where, $\alpha : 0 \leq \alpha \leq 1$ is a randomly generated non-negative coefficient. In other words, the location of the offspring reference point is in the line segment drawn between its parents. Since all existing reference points on the normalized hyperplane, the offspring reference points produced by this scheme are also located on the normalized hyperplane.

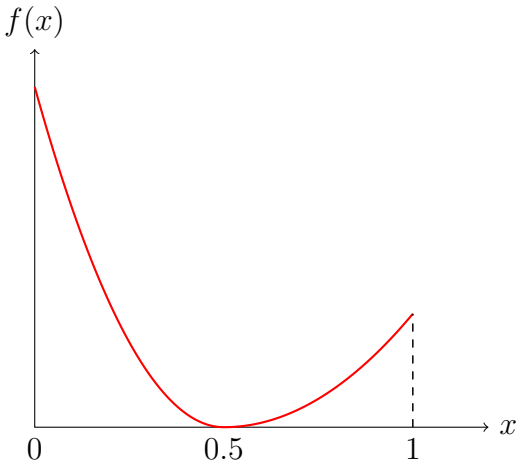

Figure 4.14. PDF of the convex combination ratio.

In the preliminary numerical analysis to determine the ratio α, it is observed that using a uniform random ratio produces an offspring which does not bear any common features with either of its parents. There must be bias towards either one of the parents. Additionally, it is observed that the bias towards the parent selected with binary tournament must be more. For this purpose, a quadratic piecewise probability density function (PDF) as shown in Figure 4.14 is used to generate convex combination ratio. The formulations of the probability density function $f(x)$, and the cumulative density function $F(x)$ are presented in Equation 4.13 and 4.14, respectively.

$$f(x) = \begin{cases} 18(x-0.5)^2, & 0 \leq x \leq 0.5. \\ 6(x-0.5)^2, & 0.5 < x \leq 1. \end{cases} \tag{4.13}$$

$$F(x) = \begin{cases} 6x^3 - 9x^2 + \frac{9}{2}x, & 0 \leq x \leq 0.5. \\ 2x^3 - 3x^2 + \frac{3}{2}x + \frac{3}{4}, & 0.5 < x \leq 1. \end{cases} \tag{4.14}$$

4.5.3. Immigration in Co-evolution

In order to explore different regions of the objective space, new reference points with random coordinates are also generated. These reference points are called immigrant reference points and they are used to complement the crossover scheme in replacing some portion of the empty reference points. The relative ratio at which these two methods are used to replace empty reference points depends on the current state of evolution. The ratio of immigrant count to crossover count is equal to the ratio of number of empty reference points to the number of associated reference points. In this way, the immigration process shows a self-adaptive behavior to reduce its use as the share of empty reference points decreases throughout the evolutionary process. In order to obtain better immigrants, twice as many are created and a binary tournament is applied between them. Immigrants with a prospective associated solution are selected.

4.5.4. Mutation in Co-evolution

As the number of nondominated points increases exponentially in many-objective problems, the task of a posteriori approaches becomes more difficult since avoiding premature convergence becomes even more challenging. In the application of a co-evolutionary structure, it is natural to prefer solutions/reference points that are closely associated with the counter set. However, there is a clear threat that the reference points and their best associated solutions will "stick" together so that they are never eliminated in the sorting procedure. This will result in the stagnation of the co-evolutionary framework and thus a premature convergence.

Mutation is used to avoid this kind of behavior and to constitute the competitive part of the co-evolutionary structure. In order to generate as distinct nondominated points as possible, the associated reference points also move on the hyperplane and search different regions of the objective space. The mutation is applied only to a percentage, not to all associated reference points. This percentage is equal to the associated reference point percentage. In this manner, as the percentage of associated reference points increases during the evolution, the use of the mutation also increases.

In order to direct the search to the unexplored regions of the objective space, mutated reference points move away from their closest associated solutions. The opposite direction on the line segment between the reference point and its closest associated solution is used. The length of this move is critical since large values cause the reference point to drift away and small values are ineffective. After investigating different alternatives in a preliminary study, it is decided to use a random coefficient between 0 and 1 to determine the move length. In this way, the move length can be at most equal to the distance between the corresponding reference point and its closest associated solution. The mutated reference points are also ensured to stay on the normalized hyperplane.

5. NUMERICAL EXPERIMENTATION

This chapter contains all the numerical experimentation conducted to verify, evaluate and compare the performance of the proposed algorithm, as well as its detailed investigation relative to the different features and properties of the many-objective combinatorial optimization problems. One or more of the benchmark problem sets are used for each of these analyses, as required.

First, the experimental setting is provided in Section 5.1. The development stages of the proposed fixed hyperplane fixed reference point set (*FHFR*) algorithm are illustrated in Section 5.2 using MaOKP and MaOQAP. Section 5.3 and 5.4 present the experiments on MaOKP with different constraint handling/reference point positioning strategies and reference point association behavior of three proposed MaOEA versions. In Section 5.5, performance evaluation of three proposed MaOEA versions is made using all three benchmark problems. Additional experimentation involving the construction of the fixed hyperplane with near optimal solutions is shown for MaOTSP and MaOQAP in Section 5.6. Experimentation including test problems with correlated objectives and test problems having different objective function characteristics are presented in Section 5.7 and 5.8, respectively. In Section 5.9, the numerical results for finding robust solutions are illustrated. Finally, numerical research on alternative limit specification for hypervolume calculation is explained in Section 5.10.

5.1. Experimental Setting

The most important issue in the experimental setting is the size of the reference point set and the population adjusted accordingly. In the numerical studies for the proposed MaOEA, the sizes of the solution set is equal to the size of the reference point set. In this way, the goal of the algorithm becomes associating exactly one solution with each reference point. It should be reminded that when using the systematic simplex lattice design by Das and Dennis (1998), the number of reference points depends on the objective and division counts as in Equation 4.1. The number of reference points

required for different division counts corresponding to each objective count is given in Table 5.1. Here, the options shown with asterisks are chosen as the reasonable reference point set and solution population sizes for each objective count.

Table 5.1. Setting of the reference point set.

Division count (d)	**Objective count (m)**							
	2	**4**	**6**	**8**	**10**	**15**	**20**	**30**
2	3	10	21	36	55	120*	210*	465*
3	4	20	56	120*	220*	680	1540	4960
4	5	35	126*	330	715	3060	8855	40920
7	8	120*	792	3432	11440	...	...	...
119	120*	...	...	...	...	...	...	...

All algorithms are coded in C# programming language in Microsoft Visual Studio 2017 and all experiments are carried out on a PC with 3.60 GHz Intel® Core™ i7-3820 CPU and 16 GB RAM running on a 64-bit Windows 10 operating system. In MaOKP, all algorithms are terminated when $500 \times N$ (population size) repair calls are made (Zhang and Li, 2007). Based on preliminary fine-tuning experiments, algorithms are terminated when the iteration count hits 500 in MaOTSP and MaOQAP.

5.2. Development Phases of the Fixed Reference Set Algorithm

This section includes the development stages of the proposed fixed hyperplane fixed reference point set (*FHFR*) algorithm. Experiments on alternative genetic operators and fine-tuning of algorithm parameters will be demonstrated. MaOKP is used for this experimentation phase. The performances of alternative genetic operators are evaluated using the Hypervolume (or Hyperarea) Ratio (HR). For a particular set of points in objective space, HR is defined as the ratio of the volume dominated by this set to the hypercube volume where the set points are located. HR for each algorithm is calculated using Monte Carlo simulation, generating 10^5 random coordinate points uniformly in the hypercube. High HR results indicate not only a close approximation of the true Pareto front, but also a diverse and well-distributed approximation.

5.2.1. Genetic Operators

There are three selection mechanisms and two crossover schemes to be tested. The selection mechanism are random selection (*random*), binary tournament selection (*bt*) and reference point based selection (*ref*). Random selection is used to observe whether searching for alternative selection mechanisms has the potential to improve performance, and the classic binary tournament forms a standard example which is easy to adopt. The crossover schemes are uniform (*uniform*) and ϵ-dominance path relinking (ϵ-*path*). The former is presented as a standard crossover method, frequently used in binary encoding, and is evaluated as a control group against path relinking method. In this phase of the analysis, only ϵ-*path* strategy is tested to observe the potential of path relinking.

Additionally at this stage of the analysis, the mutation and local improvement rates are fine-tuned. The fine-tuning step is implemented for a six-objective 500-item knapsack problem. Two levels are experimented for the mutation rate and three levels for the local improvement rate. The first level for the local improvement rate corresponds to bypassing the local improvement altogether whereas the third level corresponds to the implementation of local improvement to every recently generated offspring. At the second level, the probability of applying local improvement increases linearly at each iteration.

NSGA-III (Deb and Jain, 2013) is used as the base case for the experiments to observe the progress attained by introducing genetic operator alternatives. It should be noted that the algorithms that implement these genetic operators employ the fixed hyperplane concept, whereas NSGA-III is in its original form. HR results for the fine-tuning step are presented in Table 5.2. Since the objectives are maximization in MaOKP, the origin and the ideal point obtained from the single-objective optimal solutions are used in HR calculations. The best result is shown with an asterisk. Best results are obtained when the mutation rate is set at 0.004 and local improvement increases linearly at a rate equal to 0.01. In the rest of the numerical experimentation, this setting will always be used.

Table 5.2. HR results for fine-tuning, $m = 6$, $n = 500$.

Mutation rate	**Local imp. rate**	**Selection & crossover**		
		NSGA-III	**random & uniform**	**ref & ϵ-path**
0.004	0	0.383	0.568	0.648
	$0.01 \times t$	0.493	0.653	0.675*
	1	0.554	0.595	0.628
0.04	0	0.344	0.517	0.559
	$0.01 \times t$	0.444	0.621	0.643
	1	0.527	0.604	0.652

Next, the performances of selection and crossover alternatives are tested for larger number of objectives using 500-item knapsack problems from two to 15 objectives. In this way, their performance is examined with respect to the increase in the number of objectives. Table 5.3 displays the mean HR results in 10 replications. In Table 5.4, these results are normalized using the mean HR results of NSGA-III. In this way, the improvement achieved through selection and crossover alternatives can be observed in relative percentages allowing easier interpretation. In both tables, the best result for each objective count is displayed with an asterisk.

Table 5.3. HR results for selection & crossover, $n = 500$.

Selection & crossover	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.873	0.704	0.493	0.444	0.357	0.129
random & uniform	0.987*	0.855	0.653	0.589	0.499	0.207
bt & uniform	0.983	0.883	0.666	0.594	0.506	0.206
ref & uniform	0.984	0.851	0.657	0.566	0.452	0.182
random & ϵ-path	0.978	0.803	0.498	0.463	0.371	0.188
bt & ϵ-path	0.986	0.886*	0.673	0.635*	0.522	0.445*
ref & ϵ-path	0.978	0.878	0.675*	0.612	0.523*	0.294

Table 5.4. Relative HR results for selection & crossover, $n = 500$.

Selection & crossover	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
random & uniform	113.1%*	121.5%	132.5%	132.8%	139.7%	160.4%
bt & uniform	112.6%	125.4%	135.1%	133.7%	141.6%	160.0%
ref & uniform	112.7%	120.9%	133.1%	127.5%	126.6%	141.5%
random & ϵ-path	112.0%	114.0%	101.1%	104.3%	103.9%	146.2%
bt & ϵ-path	112.9%	125.8%*	136.4%	143.1%*	146.2%	345.2%*
ref & ϵ-path	112.0%	124.7%	137.0%*	137.8%	146.4%*	228.5%

From these results, it can be claimed that a significant improvement is achieved using a fixed hyperplane compared to its original implementation in NSGA-III. As the number of objectives increases, the relative HR results also increase, indicating that using the fixed hyperplane together with alternative selection and crossover methods responds better to the objective increase and is therefore more suitable for MaOKP. Although the results are close to each other except for the 15-objective case, it is observed that path relinking strategy stands out when combined with a selection method other than *random*.

Another analysis can be done this time by fixing the number of objectives ($m = 10$) and changing the number of knapsack items. In this way, the effect of problem size on performances can be observed. Relative HR results are shown in Table 5.5. The results support the argument that the main source of complexity in MaOPs is not due to an increase in the size of the problem, but an increase in the number of objectives.

The main analysis for the development of the proposed algorithm (*FHFR*) covers all genetic operator alternatives, including alternatives for repair and local improvement produces. All algorithm alternatives use fixed hyperplane obtained using single-objective optimal solutions. Since this modification alone significantly improves performance, a version called "*fixed NSGA-III*" is also provided in numerical results.

Table 5.5. Relative HR results for selection & crossover, $m = 10$.

Selection & crossover	**Item count (n)**			
	250	**500**	**750**	**1000**
NSGA-III	100.0%	100.0%	100.0%	100.0%
random & uniform	122.5%	139.7%	142.1%	158.5%
bt & uniform	121.1%	141.6%	140.9%	163.9%
ref & uniform	114.8%	126.6%	135.6%	154.5%
random & ϵ-path	68.8%	104.0%	136.0%	157.4%
bt & ϵ-path	135.3%*	146.2%	152.1%*	171.1%*
ref & ϵ-path	130.9%	146.4%*	148.3%	167.3%

Different from the original NSGA-III, in fixed NSGA-III, single-objective optimal solutions are provided to the initial population, and the algorithm uses the fixed hyperplane throughout evolution while all the remaining mechanisms are as in the original.

Based on previous experiments, the benefits of reference set based parent pair selection and path relinking recombination have been clearly observed. As a result, the binary tournament is used only when there are not enough guiding reference points. In this way, the binary tournament becomes a complementary and integral part of the reference set based parent pair selection.

Two versions are experimented for the binary tournament, where either similar/close parents (*close*) or dissimilar/far parents (*far*) are favored. Apart from these, random selection is still retained as a reference. Two versions of path relinking are experimented, namely path relinking using ϵ-dominance (*ϵ-path*) and reference point guided path relinking (*guided-path*). There are two repair and local improvement procedures called *greedy* and *weighted*.

Table 5.6 displays the mean HR results in 10 replications for MaOKP with 500 items. In Table 5.7, HR results are normalized again using the HR results of NSGA-III. The best result for each objective count is shown with an asterisk.

Table 5.6. HR results for genetic operator alternatives, $n = 500$.

Selection, crossover, repair/local Imp.	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.885	0.663	0.521	0.384	0.339	0.117
NSGA-III (fixed), greedy	0.965	0.769	0.605	0.463	0.412	0.148
NSGA-III (fixed), weighted	0.984	0.758	0.569	0.435	0.412	0.139
close, ϵ-path, greedy	0.986	0.783	0.604	0.486	0.427	0.163
close, ϵ-path, weighted	0.984	0.743	0.546	0.430	0.360	0.135
close, guided-path, greedy	0.979	0.778	0.599	0.441	0.379	0.117
close, guided-path, weighted	0.985	0.858*	0.699*	0.623*	0.532	0.241*
far, ϵ-path, greedy	0.987*	0.760	0.588	0.482	0.411	0.166
far, ϵ-path, weighted	0.986	0.739	0.539	0.434	0.362	0.135
far, guided-path, greedy	0.981	0.776	0.595	0.467	0.388	0.142
far, guided-path, weighted	0.986	0.828	0.691	0.617	0.541*	0.193

Table 5.7. Relative HR results for genetic operator alternatives, $n = 500$.

Selection, crossover, repair/local imp.	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	100%	100%	100%	100%	100%	100%
NSGA-III (fixed), greedy	109%	116%	116%	121%	122%	127%
NSGA-III (fixed), weighted	111%	114%	109%	113%	122%	119%
close, ϵ-path, greedy	111%	118%	116%	127%	126%	139%
close, ϵ-path, weighted	111%	112%	105%	112%	106%	116%
close, guided-path, greedy	111%	117%	115%	115%	112%	100%
close, guided-path, weighted	111%	130%*	134%*	162%*	157%	207%*
far, ϵ-path, greedy	112%*	115%	113%	126%	121%	142%
far, ϵ-path,weighted	111%	112%	103%	113%	107%	116%
far, guided-path, greedy	111%	117%	114%	122%	115%	121%
far, guided-path, weighted	111%	125%	133%	161%	160%*	165%

It should be restated that a significant improvement is achieved using the fixed hyperplane compared to the original implementation. As the number of objectives increases, the performance gap in relative HR results also increases, suggesting that the proposed genetic operator alternatives react better to the objective increase and

are therefore more suitable for MaOKP. It can be argued that it is necessary to have more than six objectives to clearly observe the difference in performance.

It is observed that when complemented with *weighted* repair/local improvement strategy, *guided-path* crossover becomes more powerful and does better than other combinations. While the performance of *close* binary tournament is slightly better than the *far* binary tournament, it is decided to keep both alternatives for further analysis.

5.2.2. External Population

External populations are mainly used to store nondominated solutions found during the search. Although the proposed MaOEA is based on nondominated sorting where nondominated solutions are given the highest priorities to be kept in the population, nondominated solutions might also be eliminated in some iterations. To protect valuable information obtained during the course of the evolution, an external population is stored and reported for performance assessment. All HR results given up to now have been calculated based on final populations. As external populations also include solutions in the final generation, an improvement in HR results is natural. Here, the contribution of an external population structure included in the algorithm will be tested. The comparison of HR results is presented in Table 5.8 and 5.9 for the original NSGA-III, for the algorithm obtained only by fixing the hyperplane and for two algorithm versions that have shown successful results in previous analyses.

5.2.3. Testing of the Proposed Configuration for FHFR on MaOKP

As seen in Table 5.8 and 5.9, the algorithm using *reference set based parent pair* selection complemented by *far* binary tournament, *reference point guided path relinking* recombination, and *weighted* repair/local improvement procedure surpasses others in every problem instance. As a consequence, this configuration becomes the first version of the *proposed algorithm* using fixed hyperplane with fixed reference point set (*FHFR*).

Table 5.8. HR results, internal vs. external.

Population	**Selection, crossover, repair/local imp.**	**Objective count (m)**					
		2	**4**	**6**	**8**	**10**	**15**
Internal	NSGA-III	0.885	0.663	0.521	0.384	0.339	0.117
	NSGA-III (fixed), greedy	0.965	0.769	0.605	0.463	0.412	0.148
	close, guided-path, weighted	0.985	0.858	0.699	0.623	0.532	0.241
	far, guided-path, weighted	0.986*	0.828	0.691	0.617	0.541	0.193
External	NSGA-III	0.901	0.756	0.591	0.454	0.395	0.173
	NSGA-III (fixed), greedy	0.979	0.863	0.722	0.575	0.451	0.207
	close, guided-path, weighted	0.985	0.890	0.772	0.645	0.544	0.288
	far, guided-path, weighted	0.986*	0.893*	0.783*	0.662*	0.560*	0.301*

Table 5.9. Relative HR results, internal vs. external.

Population	**Selection, crossover, repair/local imp.**	**Objective count (m)**					
		2	**4**	**6**	**8**	**10**	**15**
Internal	NSGA-III	100%	100%	100%	100%	100%	100%
	NSGA-III (fixed), greedy	109%	116%	116%	121%	122%	127%
	close, guided-path, weighted	111%	130%	134%	162%	157%	207%
	far, guided-path, weighted	111%*	125%	133%	161%	160%	165%
External	NSGA-III	102%	114%	113%	118%	117%	148%
	NSGA-III (fixed), greedy	111%	130%	139%	150%	133%	178%
	close, guided-path, weighted	111%	134%	148%	168%	161%	247%
	far, guided-path, weighted	111%*	135%*	150%*	173%*	165%*	258%*

The next stage of numerical analysis also includes MOEA/D (Zhang and Li, 2007). The performance of *FHFR* is compared to MOEA/D using both weighted sum (ws) and Tchebycheff (tc) approaches for decomposition. To see the progress made, the results of NSGA-III and fixed NSGA-III (with greedy repair and local improvement) are also presented. The mean HR and relative HR results are shown in Table 5.10 and 5.11, respectively. It can be seen that the proposed algorithm *FHFR* outperforms other algorithms including MOEA/D. It can also be noted that even the fixed NSGA-III performs better than MOEA/D, that is, using a fixed hyperplane significantly improves performance.

Table 5.10. HR results for *FHFR*, $n = 500$.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.901	0.756	0.591	0.454	0.395	0.173
NSGA-III (fixed)	0.979	0.863	0.722	0.575	0.451	0.207
FHFR	0.986*	0.893*	0.783*	0.662*	0.560*	0.301*
MOEA/D (ws)	0.985	0.818	0.617	0.455	0.389	0.202
MOEA/D (tc)	0.929	0.802	0.595	0.415	0.359	0.167

Table 5.11. Relative HR results for *FHFR*, $n = 500$.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	100%	100%	100%	100%	100%	100%
NSGA-III (fixed)	109%	114%	122%	127%	114%	120%
FHFR	109%*	118%*	133%*	146%*	142%*	174%*
MOEA/D (ws)	109%	108%	104%	100%	98%	117%
MOEA/D (tc)	103%	106%	101%	92%	91%	97%

The mean time requirement results for a single replication in Table 5.12 show that *FHFR* requires less time compared to MOEA/D. This is due to the fact that every generated solution is checked to see whether it is a nondominated solution to be added to the external population or not in MOEA/D. In nondominated sorting based algorithms, only solutions that are not dominated by the current population are checked for the external population. For all algorithms, it is observed that the formation of the external population takes most of the total time. The last row reports the computation times by the *CPLEX* solver to find single-objective optimal solutions. included . Computation times for all algorithms, except for NSGA-III, include these reported times.

Another analysis can be made by fixing the objective count ($m = 10$) but changing the knapsack item count. In this way, the effect of problem size on performances can be observed. It appears that HR results shown in Table 5.13 are not severely affected by the knapsack item count compared to the objective count.

Table 5.12. Computation time (sec) for *FHFR*, $n = 500$.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.5	0.6	0.7	0.9	2.1	1.2
NSGA-III (fixed)	0.6	0.8	1.0	1.2	2.6	2.0
FHFR	2.1	2.6	3.4	3.8	7.5	5.2
MOEA/D (ws)	0.9	2.7	4.6	5.2	12.5	10.0
MOEA/D (tc)	1.3	2.7	4.3	5.1	11.3	9.9
CPLEX	0.2	0.3	0.4	0.5	0.6	0.8

Table 5.13. HR results for *FHFR*, $m = 10$.

Algorithm	**Objective count (m)**			
	250	**500**	**750**	**1000**
NSGA-III	0.399	0.395	0.322	0.330
NSGA-III (fixed)	0.481	0.451	0.411	0.388
FHFR	0.593*	0.560*	0.519*	0.521*
MOEA/D (ws)	0.441	0.389	0.352	0.351
MOEA/D (tc)	0.416	0.359	0.322	0.315

As a further analysis, the Pareto front approximations are evaluated using a cardinality-based performance index, Coverage of Two Sets $C(\mathcal{S}_1, \mathcal{S}_2)$ proposed by Zitzler and Thiele (1999). This performance index can be used to compare two solution sets from different algorithms as in Equation 2.10. The results are provided for the six-objective 500-item MaOKP in Table 5.14. The ratios in the row of an algorithm show how many members of the external populations of other algorithms are dominated by members of the corresponding algorithm's external population. In short, the high values in the row of an algorithm indicate the success of the algorithm compared to other algorithms. The ratios in the column of an algorithm, on the contrary, show how much of the external population members of the corresponding algorithm are dominated by members of another algorithm's external population.

The results reveal that the members of the Pareto front approximation of NSGA-III are often dominated by others. While the two versions of MOEA/D are more

Table 5.14. Coverage of Two Sets $C(\mathcal{S}_1, \mathcal{S}_2)$ results, $m = 6$.

Algorithm	**NSGA-III**	**NSGA-III (fixed)**	**FHFR**	**MOEA/D (ws)**	**MOEA/D (tc)**
NSGA-III	-	0.008	0.001	0.001	0.002
NSGA-III (fixed)	0.822	-	0.001	0.017	0.002
FHFR	0.662	0.308	-	0.148	0.147
MOEA/D (ws)	0.944	0.668	0.002	-	0.022
MOEA/D (tc)	0.725	0.400	0.016	0.203	-

successful than *FHFR* to dominate the original and fixed NSGA-III members, comparisons between *FHFR* and two versions of MOEA/D show that the members of *FHFR* are rarely dominated by members of any other algorithm.

5.2.4. Testing of the Proposed Configuration for FHFR on MaOQAP

In the early stages of developing of the proposed MaOEA, extensive numerical tests are also carried out using QAP in order to verify its quality and to observe its response to different problem types. QAP instances with six objectives are used at this stage. Test problems are taken from QAPLIB (Burkard *et al.*, 1997). The selected instances cover a wide range of problem sizes and properties, and obey the triangular inequality in distance matrices. The original flow matrix given in QAPLIB for each of the instances is taken and used for the first objective function of the problem. The rest of the objective functions are generated by randomly distorting the original flow matrix.

Obviously, some changes need to be made in the algorithm when dealing with QAP. The main alteration is made by replacing binary encoding with permutation encoding. Consequently, the genetic operators such as crossover, mutation, and local improvement are affected, although the main framework of the algorithm remains the same. While the reference point guided path relinking remains in use, the two-order crossover replaces the uniform crossover. There is no need for a repair method, as there is no possibility to create infeasible offspring. The mutation procedure is replaced by

immigration and two-opt local improvement is used. After preliminary analysis, the immigration rate is set to 0.04 and local improvement rate is increased linearly with a rate of 0.01 per iteration. Algorithms are terminated after 500 iterations. It should be noted that when the immigration procedure is not adaptable for the use of other algorithms, e.g. MOEA/D, it is omitted. Necessary alterations are also made in the dominance relationship and performance assessment, since QAP is a minimization problem.

Relative HR results shown in Table 5.15 indicate that the proposed algorithm continues to perform better than other evolutionary algorithms. Time requirements are shown in Table 5.16. It should be noted that since it is not possible to solve single-objective QAP optimally, a genetic algorithm (GA) is used to obtain near optimal solutions, which are seeded into the initial population, to construct the fixed hyperplane. The last row reports the computation times of 10 replications of the GA to find near optimal solutions. Computation times for all algorithms, except the original NSGA-III, include these reported times.

Table 5.15. Relative HR results for *FHFR* in MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**
NSGA-III	100%	100%	100%	100%	100%
NSGA-III (fixed)	112%	127%	113%	119%	106%
FHFR	113%*	150%*	137%*	158%*	143%*
MOEA/D (ws)	109%	118%	59%	88%	128%
MOEA/D (tc)	108%	132%	90%	105%	116%

Similar inferences on the performance of the proposed algorithm can be made using the Coverage of Two Sets $C(\mathcal{S}_1, \mathcal{S}_2)$ metric in Table 5.17. The clear success of the proposed algorithm in both performance metrics verifies that *FHFR* is applicable to combinatorial MaOPs.

Table 5.16. Computation time (sec) for *FHFR* in MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**
NSGA-III	39	56	54	212	473
NSGA-III (fixed)	50	105	106	463	1268
FHFR	63	130	121	534	1385
MOEA/D (ws)	59	106	103	357	945
MOEA/D (tc)	57	108	104	359	951
10 replications of GA	12	52	55	265	817

Table 5.17. Coverage of Two Sets $C(\mathcal{S}_1, \mathcal{S}_2)$ results for *ste36a* , $m = 6$.

Algorithm	**NSGA-III**	**NSGA-III (fixed)**	**FHFR**	**MOEA/D (ws)**	**MOEA/D (tc)**
NSGA-III	-	0.123	0.078	0.165	0.412
NSGA-III (fixed)	0.324	-	0.078	0.164	0.374
FHFR	0.448	0.309	-	0.325	0.503
MOEA/D (ws)	0.285	0.155	0.057	-	0.373
MOEA/D (tc)	0.228	0.221	0.155	0.237	-

5.3. Experiments on Constraint Handling and Reference Point Positioning

In this section, the proposed algorithm (*FHFR*) is tested under different constraint handling techniques and reference point positioning strategies. The goal in expanding the problem structure is to claim that the proposed algorithm can be applied to a wider spectrum of optimization problems.

The numerical analysis for performance evaluation of different strategies related to constraint handling and reference point positioning are handled simultaneously because these two issues are regarded to be interdependent. Reference point sets are particularly important when examining irregular Pareto front structures that may result from highly constrained discrete solution spaces. As a result, numerical analysis on these two issues has been done together and multi-constrained MaOKP is used in these experiments.

The names of the six constraint handling techniques are *greedy repair*, *weighted repair*, *static penalty*, *dynamic penalty*, *adaptive penalty* and *feasible preferred*.

Four options are used for reference point positioning. The first option for algorithm configuration is the basic form of the proposed algorithm, which uses a fixed positioned reference point set on the fixed hyperplane, *FHFR*. In the second and third options, when it is concluded that a region in the objective space is void during evolution , the reference points in these regions are removed, with or without a companionship of a removal of solutions in the population. In the case where it is decided to remove a solution, it is the last qualified solution in the most recent reference point based sorting. The second option is called "*expiring references and population*" (*ER1*) and the third option is called "*expiring references*" (*ER2*), respectively. Finally in the fourth option, the algorithm occasionally relocates some reference points on the fixed hyperplane, as described in detail in Section 4.4, called "*fixed hyperplane and mobile reference points*, *FHMR*.

The experimentation is done in two levels for the capacity C_k of the knapsack: 50% or 20%, and different levels for the number of capacity constraints. HR values are calculated using external populations. The mean HR results in 10 replications are displayed in Table 5.18. The columns represent different constraint handling techniques and rows represent different reference point positioning strategies. The differences between the alternatives can be clearly seen when the number of objectives is high, and therefore only the results of problems with 15 objectives are presented. The reference point positioning strategy that yields the best result for each constraint capacity-count setting is always *FHMR*, while the constraint handling technique that yields the best result in each setting is shown in italics. The overall best result with each setting is shown with a dagger. Note that the cells corresponding to *weighted repair* and *FHFR* are the original version of the proposed algorithm and marked with an asterisk.

In Table 5.19, HR values are normalized using HR result of the algorithm alternative using *weighted repair* and *FHFR*. In this way, the amount of improvement achieved with alternative algorithm settings can be observed in percentages and inter-

Table 5.18. HR results for constraint handling & reference positioning, $m = 15$.

C_k	Constraint count	Algorithm	Greedy repair	Weighted repair	Static penalty	Dynamic penalty	Adaptive penalty	Feasible preferred
50%	1	FHFR	0.299	*0.301**	0.289	0.283	0.289	0.273
		ER1	0.266	*0.272*	0.235	0.234	0.235	0.230
		ER2	0.300	*0.301*	0.286	0.284	0.286	0.276
		FHMR	0.317	*0.326*†	0.306	0.301	0.306	0.297
	15	FHFR	0.072	0.069*	0.074	*0.096*	0.086	0.069
		ER1	0.069	0.063	0.069	*0.087*	0.081	0.063
		ER2	0.072	0.069	0.074	0.092	*0.093*	0.069
		FHMR	0.092	0.091	0.098	*0.118*†	0.109	0.090
20%	1	FHFR	*0.273*	0.272*	0.254	0.250	0.255	0.218
		ER1	0.220	*0.236*	0.192	0.201	0.205	0.172
		ER2	*0.276*	0.271	0.248	0.252	0.258	0.223
		FHMR	*0.296*†	*0.296*†	0.279	0.271	0.283	0.252
	15	FHFR	0.026	0.024*	0.032	0.049	*0.057*	0.024
		ER1	0.024	0.022	0.025	0.047	*0.050*	0.022
		ER2	0.026	0.024	0.025	0.049	*0.051*	0.024
		FHMR	0.046	0.043	0.052	0.069	*0.075*†	0.043

preted more straightforwardly. As a further analysis, Pareto front approximations are evaluated using Coverage of Two Sets $C(\mathcal{S}_1, \mathcal{S}_2)$. Results are given for two algorithm settings in Table 5.20. Superior results are shown with asterisks.

When the alternatives for different reference point positioning methods are examined, it is observed that the results are independent of the number of constraints and how tight the constraints are. While ER1 has the lowest performance, ER2 is as successful as the original algorithm. Significant improvement in HR results is observed when mobile references method is used and this method clearly demonstrates superior performance. Additionally, it is observed that reference point positioning methods and constraint handling techniques have no influence on each other.

In the analysis of constraint handling techniques, when the number of objectives increases or constraints become tighter, dynamic and adaptive penalty methods

Table 5.19. Relative HR results for constraint handling & reference positioning, $m = 15$, 15 constraints, $C_k = 20\%$.

Algorithm	**Greedy repair**	**Weighted repair**	**Static penalty**	**Dynamic penalty**	**Adaptive penalty**	**Feasible preferred**
FHFR	109%	*100%**	134%	205%	237%	99%
ER1	101%	92%	103%	197%	210%	91%
ER2	109%	102%	104%	207%	212%	101%
FHMR	192%	182%	219%	289%	*315%*†	181%

Table 5.20. Coverage of Two Sets $C(\mathcal{S}_1, \mathcal{S}_2)$ results, $m = 15$.

Objective count (m)	**Constraint count**	**C_k%**	**FHFR**	**FHMR**
			Weighted repair	**Adaptive penalty**
15	1	50	0.135*	0.041
		20	0.067*	0.053
	3	50	0.020	0.052*
		20	0.018	0.059*
	5	50	0.016	0.092*
		20	0.012	0.203*
	15	50	0.007	0.112*
		20	0.005	0.233*

achieve higher performance. However, the effects of using different constraint handling techniques appear to be more limited than using different reference point positioning methods. As a result, it is decided to continue using *weighted repair* strategy in both the *FHFR* and *FHMR* versions of the proposed MaOEA.

5.4. Reference Point Association Behavior

In this section, an analysis is performed on MaOKP to see the effectiveness of the reference point update procedures. The three main versions of the proposed MaOEA, *FHFR*, *FHMR* and *FHCo*, are evaluated based on how they manage to associate the solutions set with the reference point set. The average associated reference point percentages in the termination stage of the algorithms are shown in Table 5.21. From

the results, the success of the reference point update procedures, especial the version with the co-evolutionary approach, can be observed. It should noted that *FHCo* is designed to be self-adaptive so it does not require parametric search and can be easily adapted to different problem types.

Table 5.21. Percentage of associated reference points in MaOKP.

Algorithm	**Objective count (m)**					
	6	**8**	**10**	**15**	**20**	**30**
FHFR	27.3%	38.2%	39.4%	45.7%	57.6%	42.1%
FHMR	54.5%	65.8%	54.2%	73.7%	78.4%	75.2%
FHCo	63.3%	79.8%	56.3%	79.3%	92.1%	90.3%

5.5. Comparisons Between Proposed and Existing MOEAs

This section describes the numerical experimentation conducted to evaluate the performance of the proposed MaOEA. The contribution of the different crucial aspects of the proposed MaOEA is assessed and discussed by comparing its performance with those of the following seven state-of-the-art MaOEAs whose key properties are outlined in Appendix B.

- NSGA-III (Deb and Jain, 2013),
- *A*-NSGA-III (Jain and Deb, 2014),
- A^2-NSGA-III (Jain and Deb, 2013),
- Two_Arch2 (Wang *et al.*, 2014),
- RPEA (Liu *et al.*, 2017),
- MOEA/D (Zhang and Li, 2007), and
- PICEA-w (Wang *et al.*, 2015).

Three versions of the proposed algorithm are included in the numerical analysis to explore the effects of the reference point set adaptation strategies. The first version of the proposed algorithm (*FHFR*) that uses a fixed reference point set is kept in the experimentation to demonstrate the benefits gained solely from using a fixed hyperplane.

In the second version, reference points can be repositioned during the evolutionary process. This dynamic reference set approach, detailed in Section 4.4, is abbreviated as *FHMR* where MR stands for mobile references. The third and final version of the proposed algorithm is the co-evolutionary algorithm described in detail in Section 4.5 and abbreviated as *FHCo*. Both *FHMR* and *FHCo* work on fixed hyperplane and are designed to bring further improvement on *FHFR*.

In order to make the algorithms comparable, necessary modifications and parameter settings are made. For instance, the population size (total of convergence archive and divergence archive sizes in the case of Two Arc_2) and the number of reference points (the number of decomposition vectors in the case of MOEA/D and PICEA-w) are kept the same for all algorithms. The exceptions are A-NSGA-III and A^2-NSGA-III, where the number of reference points is allowed to increase.

It should be noted that lexicographic optimal solutions are fed into the initial populations of all algorithms for a fair comparison. However, when single-objective optimal solutions are offered to an algorithm which works on hyperplanes, namely NSGA-III, A-NSGA-III or A^2-NSGA-III, its hyperplane becomes fixed throughout evolution, resulting into a new algorithm whose behavior changes drastically. Therefore, numerical experimentation includes both original and fixed hyperplane versions of these algorithms. For MOEA/D, two versions with different decomposition methods, weighted sum and Tchebycheff are included in the experiments. As a result, the numerical analysis covers a total of 14 algorithm versions.

The primary analysis carried out is to investigate the performance of these algorithms as the number of objectives increases. All algorithms are allowed to carry external populations, and evaluation is done on final external populations. The following subsections include experimentation on many-objective knapsack problem (MaOKP), many-objective traveling salesman problem (MaOTSP) and many-objective quadratic assignment problem (MaOQAP) in the respective order.

5.5.1. MaOKP Results

In the parameter setting of the knapsack problem, as in previous analyses, the values for profit p_{ij} and weight w_{jk} parameters are randomly specified using discrete uniform distribution in the interval $[10, 100]$. The capacity of the knapsack C_k is 50% of the sum of all the weights of the corresponding constraint. The analysis is performed using single constraint, 500-item knapsack problems with six to 30 objectives. The mean HR results in 10 replications are displayed in Table 5.22 and the standard deviations of HR results are provided in Table 5.23. In all tables, the last three rows correspond to three versions of the proposed MaOEAs.

In Table 5.24, the nonparametric Wilcoxon rank-sum test at 95% confidence level is carried out for each objective count instance in order to verify statistical differences between the HR results of algorithm alternatives. Similarly, Friedman's test is carried out to determine the overall ranks of the algorithms. Wilcoxon rank-sum test results are presented in three digits: *w-t-l* representing win, tie and lose, respectively. The average of Friedman's test results over all problem instances and the overall final results (given in parentheses) are provided.

Additionally, the behavior of algorithms as the number of objectives increases can be observed in terms of relative HR values in Table 5.25 and in Figure 5.1 where results are normalized according to NSGA-III results. In this way, the relative performances of the algorithms can be observed and another view is obtained to interpret the results.

These results reveal that all three algorithms, NSGA-III, *A*-NSGA-III and A^2-NSGA-III, benefit significantly when their hyperplanes are fixed. Among them, the fixed hyperplane version of A^2-NSGA-III performs the best. However, although this algorithm employs a reference set update mechanism, it is still surpassed by *FHFR*. In fact, *FHFR* is more successful than all its competitors and it is surpassed only by its adaptive versions. Its performance can be significantly enhanced by using mobile reference points (*FHMR*). In spite of this improvement, the reference point relocation strategy of *FHMR* remains rather restrictive compared with its co-evolutionary coun-

Table 5.22. Mean HR results for MaOKP.

Algorithm	**Objective count (m)**					
	6	**8**	**10**	**15**	**20**	**30**
NSGA-III	0.591	0.454	0.395	0.173	0.0320	0.0022
NSGA-III (fixed)	0.722	0.575	0.451	0.207	0.0429	0.0031
A-NSGA-III	0.532	0.418	0.348	0.167	0.0428	0.0034
A-NSGA-III (fixed)	0.762	0.590	0.524	0.229	0.0567	0.0044
A^2-NSGA-III	0.611	0.552	0.445	0.228	0.0626	0.0049
A^2-NSGA-III (fixed)	0.771	0.618	0.543	0.231	0.0655	0.0054
Two_Arch2	0.775	0.635	0.521	0.225	0.0664	0.0052
RPEA	0.735	0.495	0.412	0.214	0.0550	0.0043
MOEA/D (ws)	0.617	0.455	0.389	0.202	0.0408	0.0031
MOEA/D (tc)	0.595	0.415	0.359	0.167	0.0333	0.0022
PICEA-w	0.767	0.622	0.568	0.263	0.0722	0.0058
FHFR	0.783	0.662	0.560	0.301	0.0715	0.0055
FHMR	0.811	0.708	0.590	0.326	0.0844	0.0067
FHCo	0.835	0.735	0.631	0.352	0.0970	0.0078

Table 5.23. Standard deviation of HR results for MaOKP.

Algorithm	**Objective count (m)**					
	6	**8**	**10**	**15**	**20**	**30**
NSGA-III	0.0467	0.0586	0.0588	0.0050	0.0000	0.0007
NSGA-III (fixed)	0.0247	0.0360	0.0549	0.0148	0.0009	0.0005
A-NSGA-III	0.0367	0.0477	0.0519	0.0050	0.0005	0.0001
A-NSGA-III (fixed)	0.0233	0.0343	0.0493	0.0068	0.0006	0.0002
A^2-NSGA-III	0.0292	0.0375	0.0505	0.0052	0.0005	0.0010
A^2-NSGA-III (fixed)	0.0278	0.0221	0.0496	0.0073	0.0002	0.0000
Two_Arch2	0.0296	0.0299	0.0498	0.0138	0.0005	0.0007
RPEA	0.0250	0.0438	0.0455	0.0124	0.0004	0.0001
MOEA/D (ws)	0.0260	0.0504	0.0470	0.0124	0.0001	0.0002
MOEA/D (tc)	0.0428	0.0557	0.0523	0.0036	0.0001	0.0005
PICEA-w	0.0205	0.0220	0.0455	0.0064	0.0009	0.0005
FHFR	0.0258	0.0239	0.0487	0.0578	0.0001	0.0000
FHMR	0.0227	0.0289	0.0334	0.0468	0.0009	0.0007
FHCo	0.0252	0.0242	0.0281	0.0591	0.0003	0.0006

Table 5.24. Wilcoxon rank-sum test and Friedman test results on MaOKP.

Algorithm	**Wilcoxon rank-sum test, objective count (m)**						**Friedman test**
	6	8	10	15	20	30	
NSGA-III	0-4-9	0-4-9	0-6-7	2-0-11	0-0-13	0-1-12	12.5 (12)
NSGA-III (fixed)	5-1-7	5-3-5	3-4-6	3-3-7	3-1-9	2-2-9	9.333 (10)
A-NSGA-III	0-1-12	0-3-10	0-3-10	0-1-12	3-1-9	3-1-9	12.5 (12)
A-NSGA-III (fixed)	7-4-2	6-2-5	5-6-2	6-3-4	6-0-7	6-1-6	7 (7)
A^2-NSGA-III	1-3-9	5-1-7	2-5-6	6-3-4	7-0-6	5-3-5	8.333 (8)
A^2-NSGA-III (fixed)	7-4-2	6-4-3	7-5-1	6-3-4	8-0-5	8-1-4	5.333 (5)
Two_Arch2	7-4-2	8-3-2	7-2-4	4-5-4	9-0-4	7-4-2	5.667 (6)
RPEA	5-1-7	2-2-9	2-3-8	4-2-7	5-0-8	5-1-7	9.167 (9)
MOEA/D (ws)	1-3-9	0-4-9	0-6-7	3-1-9	2-0-11	2-1-10	11.333 (11)
MOEA/D (tc)	1-3-9	0-3-10	0-3-10	0-1-12	1-0-12	0-1-12	13.333 (14)
PICEA-w	7-4-2	8-2-3	8-4-1	10-1-2	11-0-2	9-2-2	4 (4)
FHFR	7-4-2	10-1-2	8-4-1	10-3-0	10-0-3	9-2-2	3.5 (3)
FHMR	12-1-0	12-0-1	9-4-0	11-2-0	12-0-1	12-0-1	2 (2)
FHCo	12-1-0	13-0-0	12-1-0	11-2-0	13-0-0	13-0-0	1(1)

Table 5.25. Relative HR results for MaOKP.

Algorithm	**Objective count (m)**					
	6	**8**	**10**	**15**	**20**	**30**
NSGA-III	100%	100%	100%	100%	100%	100%
NSGA-III (fixed)	122%	127%	114%	120%	134%	141%
A-NSGA-III	90%	92%	88%	97%	134%	153%
A-NSGA-III (fixed)	129%	130%	133%	132%	177%	198%
A^2-NSGA-III	103%	122%	113%	132%	196%	218%
A^2-NSGA-III (fixed)	131%	136%	137%	134%	205%	243%
Two_Arch2	131%	140%	132%	130%	207%	234%
RPEA	124%	109%	104%	124%	172%	192%
MOEA/D (ws)	104%	100%	98%	117%	128%	136%
MOEA/D (tc)	101%	92%	91%	97%	104%	97%
PICEA-w	130%	137%	144%	152%	226%	260%
FHFR	133%	146%	142%	174%	223%	246%
FHMR	137%	156%	149%	188%	264%	301%
FHCo	141%	162%	160%	203%	303%	349%

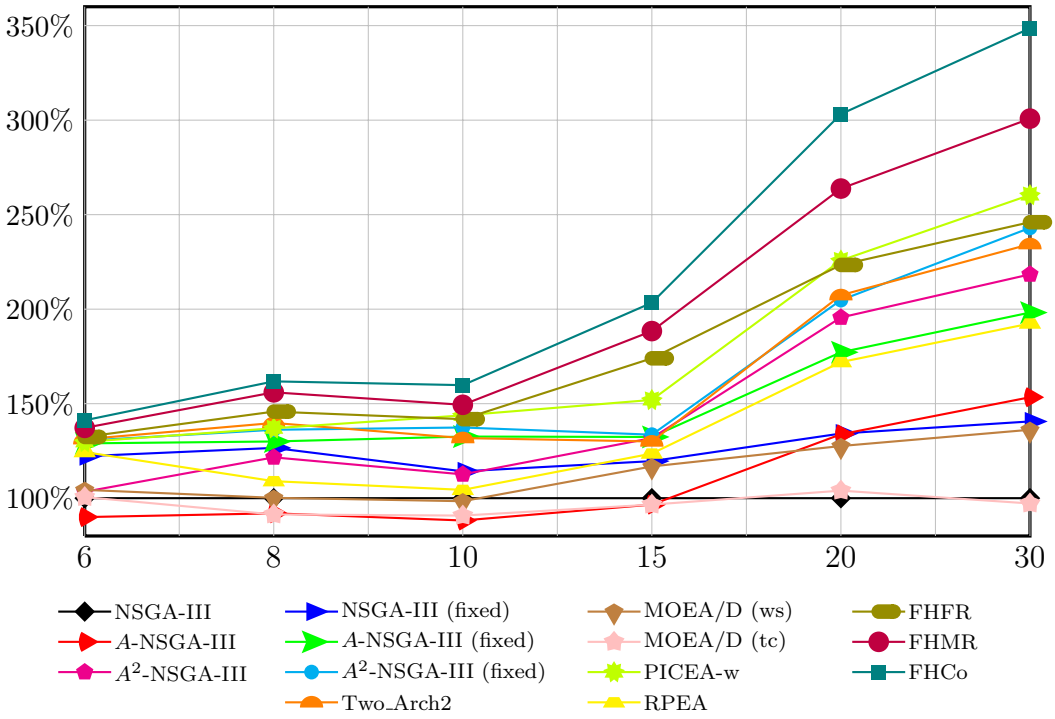

Figure 5.1. Relative HR values with regard to the objective count.

Table 5.26. Computation time (sec) for MaOKP.

Algorithm	**Objective count (m)**					
	6	**8**	**10**	**15**	**20**	**30**
NSGA-III	0.7	0.9	2.1	1.2	2.6	10.1
NSGA-III (fixed)	1.0	1.2	2.6	2.0	5.5	24.2
A-NSGA-III	1.4	1.9	4.1	3.1	7.0	31.4
A-NSGA-III (fixed)	1.6	2.1	4.4	3.9	12.4	65.3
A^2-NSGA-III	1.0	1.6	3.8	2.9	6.9	31.5
A^2-NSGA-III (fixed)	1.3	1.8	4.2	3.5	11.5	57.9
Two_Arch2	1.2	1.9	4.1	3.4	7.0	29.9
RPEA	0.8	1.0	2.3	1.6	3.5	16.4
MOEA/D (ws)	4.6	5.2	12.5	10.0	18.2	68.9
MOEA/D (tc)	4.3	5.1	11.3	9.9	17.8	67.1
PICEA-w	3.9	4.9	9.7	8.7	19.2	77.8
FHFR	3.4	3.8	7.5	5.2	12.7	54.5
FHMR	2.9	3.1	7.1	4.9	13.6	67.6
FHCo	4.1	4.6	9.5	8.4	18.1	83.4
CPLEX	*0.4*	*0.5*	*0.6*	*0.8*	*2.2*	*3.4*

terpart *FHCo*, which allows all reference points to evolve in a manner more responsive to the information gathered from the population about the shape of the Pareto front. As a result, the algorithm is capable of searching different and relevant regions in the objective space and producing a crowded external archive. For the proposed co-evolutionary algorithm, the relative HR results increase as the number of objectives increases, which indicates that it responds better to the objective count, and is therefore more suitable for many objectives. Finally, the Friedman's test results reveal a perfect score for the co-evolutionary algorithm *FHCo*. In summary, the proposed algorithm is the best approach to solve MaOKP among the tested algorithms.

Table 5.26 presents the time requirement for a single replication of each algorithm. The last row reports the computation times of the CPLEX solver to find single-objective optimal solutions. Computation times for all algorithms, except for the original versions of NSGA-III, A-NSGA-III and A^2-NSGA-III, include these reported times.

5.5.2. MaOTSP Results

This section includes the performance evaluation of the versions of the proposed algorithm, as well as the benchmark algorithms selected from the literature, on many-objective symmetric TSP (MaOTSP) and many-objective asymmetric TSP (MaOATSP). For MaOTSP, 100-city problem instances are produced with two to 15 objectives. The c_{ijk} values in the cost matrix are generated using the same methodology in "*kroa*" problem instances in TSPLIB (Reinelt, 1991) and they obey the triangular inequality. The locations of 100 cities are randomly generated in a rectangle with dimensions 4000 by 2000. Cost matrix values are calculated by rounding Euclidean distances to nearest integers. For MaOATSP, 50-city problem instances are produced with two to 15 objectives. Cost matrix values are randomly specified using discrete uniform distribution in the interval $[0, 1000]$ (Cirasella *et al.*, 2001) and do not necessarily meet the triangular inequality. In both problem types, cost matrices are reproduced for each objective using the same methodology so that the feasible region in objective space has the same scale and properties in each objective dimension.

Table 5.27. Mean HR results for MaOTSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.876	0.612	0.513	0.418	0.356	0.324
NSGA-III (fixed)	0.952	0.788	0.644	0.528	0.425	0.414
A-NSGA-III	0.873	0.617	0.517	0.429	0.396	0.389
A-NSGA-III (fixed)	0.934	0.741	0.589	0.489	0.417	0.421
A^2-NSGA-III	0.875	0.613	0.589	0.472	0.431	0.422
A^2-NSGA-III (fixed)	0.901	0.753	0.639	0.587	0.486	0.492
Two_Arch2	0.948	0.781	0.566	0.467	0.398	0.402
RPEA	0.881	0.612	0.468	0.367	0.327	0.298
MOEA/D (ws)	0.892	0.615	0.470	0.431	0.409	0.416
MOEA/D (tc)	0.916	0.699	0.527	0.464	0.382	0.411
PICEA-w	0.934	0.772	0.547	0.486	0.421	0.445
FHFR	0.964	0.804	0.657	0.522	0.448	0.452
FHMR	0.976	0.822	0.683	0.617	0.541	0.558
FHCo	0.985	0.841	0.703	0.664	0.596	0.601

Table 5.28. Standard deviation of HR results for MaOTSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.0140	0.0122	0.0159	0.0275	0.0428	0.0064
NSGA-III (fixed)	0.0120	0.0117	0.0134	0.0138	0.0265	0.0176
A-NSGA-III	0.0154	0.0150	0.0226	0.0160	0.0242	0.0308
A-NSGA-III (fixed)	0.0125	0.0134	0.0191	0.0322	0.0184	0.0200
A^2-NSGA-III	0.0139	0.0097	0.0309	0.0406	0.0555	0.0212
A^2-NSGA-III (fixed)	0.0167	0.0127	0.0052	0.0324	0.0194	0.0316
Two_Arch2	0.0089	0.0197	0.0223	0.0243	0.0466	0.0435
RPEA	0.0186	0.0211	0.0525	0.0415	0.0348	0.0032
MOEA/D (ws)	0.0090	0.0072	0.0323	0.0186	0.0271	0.0232
MOEA/D (tc)	0.0148	0.0131	0.0216	0.0283	0.0296	0.0154
PICEA-w	0.0129	0.0110	0.0140	0.0285	0.0226	0.0293
FHFR	0.0224	0.0118	0.0101	0.0128	0.0338	0.0410
FHMR	0.0116	0.0183	0.0169	0.0119	0.0167	0.0312
FHCo	0.0097	0.0147	0.0147	0.0158	0.0772	0.0119

Table 5.29. Relative HR results for MaOTSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	100%	100%	100%	100%	100%	100%
NSGA-III (fixed)	109%	129%	126%	126%	119%	128%
A-NSGA-III	100%	101%	101%	103%	111%	120%
A-NSGA-III (fixed)	107%	121%	115%	117%	117%	130%
A^2-NSGA-III	100%	100%	115%	113%	121%	130%
A^2-NSGA-III (fixed)	103%	123%	125%	140%	137%	152%
Two_Arch2	108%	128%	110%	112%	112%	124%
RPEA	101%	100%	91%	88%	92%	92%
MOEA/D (ws)	102%	100%	92%	103%	115%	128%
MOEA/D (tc)	105%	114%	103%	111%	107%	127%
PICEA-w	107%	126%	107%	116%	118%	137%
FHFR	110%	131%	128%	125%	126%	140%
FHMR	111%	134%	133%	148%	152%	172%
FHCo	112%	137%	137%	159%	167%	185%

Table 5.30. Mean HR results for MaOATSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.896	0.718	0.598	0.521	0.409	0.376
NSGA-III (fixed)	0.931	0.782	0.631	0.554	0.436	0.414
A-NSGA-III	0.936	0.743	0.609	0.549	0.437	0.432
A-NSGA-III (fixed)	0.955	0.801	0.669	0.580	0.506	0.447
A^2-NSGA-III	0.966	0.794	0.655	0.570	0.436	0.422
A^2-NSGA-III (fixed)	0.933	0.844	0.699	0.581	0.507	0.492
Two_Arch2	0.936	0.718	0.545	0.415	0.412	0.402
RPEA	0.902	0.627	0.525	0.414	0.331	0.298
MOEA/D (ws)	0.868	0.666	0.567	0.504	0.456	0.356
MOEA/D (tc)	0.873	0.659	0.526	0.483	0.433	0.378
PICEA-w	0.927	0.792	0.612	0.564	0.484	0.425
FHFR	0.956	0.788	0.601	0.581	0.486	0.452
FHMR	0.974	0.784	0.672	0.604	0.509	0.558
FHCo	0.973	0.871	0.688	0.603	0.583	0.601

Table 5.31. Standard deviation of HR results for MaOATSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.0139	0.0236	0.0172	0.0140	0.0279	0.0334
NSGA-III (fixed)	0.0115	0.0179	0.0147	0.0374	0.0371	0.0197
A-NSGA-III	0.0190	0.0186	0.0109	0.0156	0.0213	0.0267
A-NSGA-III (fixed)	0.0283	0.0115	0.0181	0.0175	0.0236	0.0402
A^2-NSGA-III	0.0239	0.0176	0.0088	0.0111	0.0488	0.0392
A^2-NSGA-III (fixed)	0.0102	0.0182	0.0149	0.0290	0.0148	0.0318
Two_Arch2	0.0166	0.0178	0.0376	0.0235	0.0562	0.0464
RPEA	0.0121	0.0137	0.0103	0.0245	0.0325	0.0061
MOEA/D (ws)	0.0105	0.0252	0.0228	0.0376	0.0463	0.0389
MOEA/D (tc)	0.0123	0.0169	0.0285	0.0220	0.0375	0.0251
PICEA-w	0.0128	0.0066	0.0129	0.0122	0.0254	0.0566
FHFR	0.0167	0.0205	0.0205	0.0197	0.0712	0.0197
FHMR	0.0131	0.0168	0.0158	0.0222	0.0073	0.0214
FHCo	0.0119	0.0284	0.0170	0.0193	0.0213	0.0132

Table 5.32. Relative HR results for MaOATSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	100%	100%	100%	100%	100%	100%
NSGA-III (fixed)	104%	109%	106%	106%	107%	110%
A-NSGA-III	104%	103%	102%	105%	107%	115%
A-NSGA-III (fixed)	107%	112%	112%	111%	124%	119%
A^2-NSGA-III	108%	111%	110%	109%	107%	112%
A^2-NSGA-III (fixed)	104%	118%	117%	112%	124%	131%
Two_Arch2	104%	100%	91%	80%	101%	107%
RPEA	101%	87%	88%	79%	81%	79%
MOEA/D (ws)	97%	93%	95%	97%	111%	95%
MOEA/D (tc)	97%	92%	88%	93%	106%	101%
PICEA-w	103%	110%	102%	108%	118%	113%
FHFR	107%	110%	101%	112%	119%	120%
FHMR	109%	109%	112%	116%	124%	148%
FHCo	109%	121%	115%	116%	143%	160%

Table 5.33. Wilcoxon rank-sum test and Friedman test results on MaOTSP.

Algorithm	**Wilcoxon rank-sum test, objective count (m)**						**Friedman test**
	2	4	6	8	10	15	
NSGA-III	0-4-9	0-4-9	2-2-9	1-2-10	1-1-11	1-0-12	12.833 (13)
NSGA-III (fixed)	8-3-2	8-3-2	9-1-3	9-1-3	3-7-3	2-6-5	5.167 (5)
A-NSGA-III	0-3-10	0-4-9	2-2-9	1-2-10	2-4-7	2-4-7	11.667 (12)
A-NSGA-III (fixed)	7-3-3	6-1-6	7-1-5	4-4-5	4-5-4	3-6-4	7 (7)
A^2-NSGA-III	0-3-10	0-4-9	7-1-5	4-4-5	4-6-3	3-7-3	8.5 (9)
A^2-NSGA-III (fixed)	4-1-8	6-2-5	9-1-3	11-0-2	11-0-2	11-0-2	5 (4)
Two_Arch2	8-3-2	8-2-3	6-0-7	4-4-5	2-7-4	2-6-5	8 (8)
RPEA	0-4-9	0-4-9	0-1-12	0-0-13	0-0-13	0-0-13	13.333 (14)
MOEA/D (ws)	2-3-8	0-4-9	0-1-12	1-3-9	2-7-4	2-7-4	10.333 (11)
MOEA/D (tc)	6-0-7	5-0-8	2-2-9	3-5-5	1-4-8	2-6-5	9.833 (10)
PICEA-w	7-1-5	7-3-3	5-0-8	4-4-5	4-5-4	6-4-3	6.667 (6)
FHFR	9-2-2	10-1-2	11-0-2	9-1-3	8-2-3	8-2-3	3.667 (3)
FHMR	12-1-0	12-0-1	12-0-1	12-0-1	12-0-1	12-0-1	2 (2)
FHCo	12-1-0	13-0-0	13-0-0	13-0-0	13-0-0	13-0-0	1 (1)

Table 5.34. Wilcoxon rank-sum test and Friedman test results on MaOATSP.

Algorithm	**Wilcoxon rank-sum test, objective count (m)**						**Friedman test**
	2	4	6	8	10	15	
NSGA-III	2-1-10	3-1-9	4-3-6	3-1-9	1-3-9	1-2-10	11.333 (11)
NSGA-III (fixed)	4-4-5	6-4-3	8-0-5	5-5-3	1-6-6	4-5-4	8.167 (9)
A-NSGA-III	4-5-4	5-0-8	4-3-6	5-1-7	2-5-6	4-6-3	7.833 (8)
A-NSGA-III (fixed)	7-4-2	10-1-2	10-1-2	7-4-2	9-3-1	5-5-3	4.333 (4)
A^2-NSGA-III	9-4-0	6-4-3	9-0-4	6-4-3	2-6-5	4-5-4	6 (6)
A^2-NSGA-III (fixed)	4-4-5	12-0-1	12-1-0	6-7-0	10-2-1	11-0-2	3.5 (3)
Two_Arch2	4-5-4	3-1-9	1-2-10	0-1-12	1-5-7	4-4-5	10.5 (10)
RPEA	2-1-10	0-0-13	0-1-12	0-1-12	0-0-13	0-0-13	13.5 (14)
MOEA/D (ws)	0-1-12	1-1-11	2-1-10	2-2-9	3-6-4	1-2-10	11.333 (11)
MOEA/D (tc)	0-1-12	1-1-11	0-2-11	2-1-10	1-6-6	1-2-10	12.167 (13)
PICEA-w	4-4-5	6-4-3	4-3-6	6-3-4	7-3-3	4-6-3	7 (7)
FHFR	9-2-2	6-5-2	4-3-6	9-2-2	6-3-4	7-3-3	5.167 (5)
FHMR	11-2-0	6-4-3	10-1-2	11-2-0	10-2-1	12-0-1	2.667 (2)
FHCo	11-2-0	13-0-0	12-1-0	11-2-0	13-0-0	13-0-0	1.5 (1)

Concorde TSP Solver (Applegate *et al.*, 2006) for MaOTSP and CPLEX Solver for MaOATSP are used to obtain single-objective optimal solutions. The results for mean HR, standard deviation of HR, and relative HR in 10 replications are displayed in Table 5.27 to 5.29 for MaOTSP, in Table 5.30 to 5.32 for MaOATSP. The results of Wilcoxon rank-sum test at 95% confidence level and Friedman's test are displayed in Table 5.33 for MaOTSP, and in Table 5.34 for MaOATSP.

The results for both problem types show that the proposed MaOEA has superior performance over the performance of other existing algorithms. From relative HR results, as the number of objectives increases, the gaps between all versions of the proposed MaOEA and the original NSGA-III also increase, indicating that all algorithm versions can better handle many objective problems more successfully. Thus, it has been shown that the proposed MaOEA works significantly better also for problems with permutation encoding.

Table 5.35. Computation time (sec) for MaOTSP and MaOATSP.

Algorithm	**MaOTSP, objective count (m)**				**MaOATSP, objective count (m)**			
	6	**8**	**10**	**15**	**6**	**8**	**10**	**15**
NSGA-III	48.0	55.0	80.6	76.2	54.4	63.9	94.0	92.1
NSGA-III (fixed)	49.5	56.7	80.8	81.0	60.4	70.5	98.0	98.3
A-NSGA-III	52.8	59.7	86.3	81.0	62.3	71.2	97.3	98.6
A-NSGA-III (fixed)	52.2	60.2	85.5	82.8	63.8	72.5	100.5	103.5
A^2-NSGA-III	54.9	62.8	90.3	83.3	63.0	71.1	103.0	100.6
A^2-NSGA-III (fixed)	53.3	59.8	86.7	86.0	65.2	70.9	101.1	105.2
Two_Arch2	64.8	74.5	102.8	98.9	73.3	85.7	119.2	116.9
RPEA	41.3	53.1	64.6	69.2	49.5	64.7	79.7	88.7
MOEA/D (ws)	51.0	58.2	74.8	75.9	62.2	72.0	89.6	96.5
MOEA/D (tc)	49.9	56.8	73.2	74.3	61.3	68.0	87.5	94.2
PICEA-w	68.4	79.2	127.8	109.1	78.9	90.8	142.5	129.9
FHFR	90.3	108.9	184.8	175.3	101.4	119.8	199.0	192.9
FHMR	95.3	112.6	193.3	180.9	103.6	124.8	210.5	197.8
FHCo	94.0	112.3	199.8	190.9	105.4	126.8	215.0	208.4
Concorde/CPLEX	3.2	3.9	4.4	10.5	5.1	6.7	8.2	12.4

Table 5.35 presents the mean time requirements for a single replication of each algorithm alternative. The last row reports the computation times of Concorde and CPLEX solvers to find single-objective optimal solutions for which all algorithms, except for *NSGA-III*, *A-NSGA-III* and A^2-*NSGA-III*, include.

5.5.3. MaOQAP Results

The performance of the proposed MaOEA is also evaluated for MaOQAP using problem instances with six and 10 objectives. Test problems are taken from QAPLIB (Burkard *et al.*, 1997). The selected instances cover a wide variety of problem sizes and properties, and obey the triangular inequality in distance matrices. It is chosen to solve the single-objective problems using a previously validated GA that provides the optimal results for a wide range of QAP instances in QAPLIB (Şahinkoç, 2014). Relative HR results are presented in Table 5.36 and Table 5.37, the results of Wilcoxon rank-sum test at 95% confidence level and Friedman's test are displayed in Table 5.38 and Table 5.39 for the problem instances with six and 10 objectives, respectively.

Table 5.36. Relative HR results for MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**
NSGA-III	100%	100%	100%	100%	100%
NSGA-III (fixed)	112%	127%	113%	119%	106%
A-NSGA-III	102%	118%	114%	117%	105%
A-NSGA-III (fixed)	114%	131%	124%	129%	111%
A^2-NSGA-III	101%	112%	134%	128%	121%
A^2-NSGA-III (fixed)	121%	161%	154%	131%	148%
Two_Arch2	85%	95%	72%	92%	98%
RPEA	73%	88%	61%	81%	87%
MOEA/D (ws)	109%	118%	59%	88%	128%
MOEA/D (tc)	108%	132%	90%	105%	116%
PICEA-w	107%	112%	94%	83%	104%
FHFR	113%	150%	137%	158%	143%
FHMR	111%	158%	159%	162%	173%
FHCo	135%	165%	181%	192%	204%

Table 5.37. Relative HR results for MaOQAP, $m = 10$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**
NSGA-III	100%	100%	100%	100%	100%
NSGA-III (fixed)	115%	129%	112%	120%	108%
A-NSGA-III	107%	126%	118%	123%	109%
A-NSGA-III (fixed)	126%	143%	138%	143%	122%
A^2-NSGA-III	97%	110%	132%	127%	120%
A^2-NSGA-III (fixed)	127%	169%	162%	137%	157%
Two_Arch2	94%	105%	80%	101%	109%
RPEA	67%	80%	56%	76%	81%
MOEA/D (ws)	130%	137%	71%	103%	150%
MOEA/D (tc)	128%	159%	110%	125%	138%
PICEA-w	124%	129%	109%	98%	123%
FHFR	131%	179%	163%	190%	171%
FHMR	121%	173%	177%	180%	188%
FHCo	164%	208%	221%	238%	257%

Table 5.38. Wilcoxon rank-sum test and Friedman test results on MaOQAP, $m = 6$.

Algorithm	**Wilcoxon rank-sum test, problem instance**					**Friedman test**
	had16	**scr20**	**nug20**	**kra30a**	**ste36a**	
NSGA-III	2-1-10	2-0-11	5-0-8	4-0-9	2-0-11	11.0 (12)
NSGA-III (fixed)	8-2-3	7-0-6	6-0-7	7-0-6	3-2-8	7.2 (6)
A-NSGA-III	3-1-9	5-1-7	7-0-6	6-0-7	3-2-8	8.6 (9)
A-NSGA-III (fixed)	11-0-2	8-1-4	8-0-5	8-1-4	6-0-7	5.6 (5)
A^2-NSGA-III	2-2-9	3-1-9	9-0-4	8-1-4	8-0-5	7.8 (7)
A^2-NSGA-III (fixed)	12-0-1	12-0-1	11-0-2	10-0-3	11-0-2	2.8 (2)
Two_Arch2	1-0-12	1-0-12	2-0-11	3-0-10	1-0-12	12.4 (13)
RPEA	0-0-13	0-0-13	0-1-12	0-0-13	0-0-13	13.8 (14)
MOEA/D (ws)	6-1-6	5-1-7	0-1-12	1-1-11	9-0-4	9.4 (10)
MOEA/D (tc)	6-1-6	8-1-4	3-0-10	5-0-8	7-0-6	8.0 (8)
PICEA-w	5-0-8	3-1-9	4-0-9	1-1-11	3-2-8	10.6 (11)
FHFR	9-1-3	10-0-3	10-0-3	11-0-2	10-0-3	3.8 (4)
FHMR	8-1-4	11-0-2	12-0-1	12-0-1	12-0-1	3.0 (3)
FHCo	13-0-0	13-0-0	13-0-0	13-0-0	13-0-0	1.0 (1)

Table 5.39. Wilcoxon rank-sum test and Friedman test results on MaOQAP, $m = 10$.

Algorithm	**Wilcoxon rank-sum test, problem instance**					**Friedman test**
	had16	**scr20**	**nug20**	**kra30a**	**ste36a**	
NSGA-III	3-0-10	1-0-12	3-0-10	2-1-10	1-0-12	12.0 (13)
NSGA-III (fixed)	5-0-8	5-1-7	6-0-7	5-0-8	2-2-9	9.4 (11)
A-NSGA-III	4-0-9	4-0-9	7-0-6	6-0-7	2-2-9	9.2 (10)
A-NSGA-III (fixed)	7-3-3	8-0-5	9-0-4	10-0-3	6-1-6	5.8 (5)
A^2-NSGA-III	1-1-11	3-0-10	8-0-5	8-0-5	5-0-8	8.8 (8)
A^2-NSGA-III (fixed)	8-2-3	10-0-3	10-1-2	9-0-4	10-0-3	4.4 (4)
Two_Arch2	1-1-11	2-0-11	2-0-11	2-1-10	2-2-9	11.6 (12)
RPEA	0-0-13	0-0-13	0-0-13	0-0-13	0-0-13	14.0 (14)
MOEA/D (ws)	11-1-1	7-0-6	1-0-12	4-0-9	9-0-4	7.6 (7)
MOEA/D (tc)	8-2-3	9-0-4	4-1-8	7-0-6	8-0-5	6.2 (6)
PICEA-w	7-1-5	5-1-7	4-1-8	1-0-12	6-1-6	9.0 (9)
FHFR	11-1-1	12-0-1	10-1-2	12-0-1	11-0-2	2.4 (2)
FHMR	6-0-7	11-0-2	12-0-1	11-0-2	12-0-1	3.6 (3)
FHCo	13-0-0	13-0-0	13-0-0	13-0-0	13-0-0	1.0 (1)

The results indicate the superior performance of the three versions of the proposed MaOEA, especially *FHCo*, compared with other existing algorithms. It has an increasing rate of HR results as the number of objectives or the problem size increases. Computation times are investigated in more detail, together with studies using near optimal solutions in Section 5.6.

5.6. Constructing the Fixed Hyperplane with Near Optimal Solutions

Finding single-objective optimal solutions (or, near optimal solutions as in this section) is crucial to the proposed MaOEA for two main reasons. First, the genetic materials of the single-objective optimal solutions are very useful for improving the quality of the initial and thus final populations. Secondly and more importantly, these single-objective optimal solutions are used to construct the fixed hyperplane used to place the reference point set. Properly positioning the hyperplane and the initial

reference point set is crucial for finding a successful and diverse Pareto approximation. Extreme solutions used at the construction phase of the hyperplane should result in such a hyperplane that all efficient solutions can be mapped onto it uniformly.

The proposed MaOEA uses lexicographic optimal solutions for each objective prior to the algorithm. However, for several combinatorial optimization problems, solving the single-objective problems may be very time consuming or even impossible. In such cases, it may be an option to skip finding single-objective optimal solutions and to initiate the evolutionary algorithm using near optimal solutions. This section investigates the corresponding level of deterioration in the quality of Pareto front approximation versus the savings in computational time.

In this context, numerical experimentation is carried out using MaOTSP and MaOATSP. Near optimal solutions are obtained by using different predefined gap levels from optimal solutions. Single-objective optimization is fathomed when the objective value of the best feasible solution has an objective value that is a predefined gap from the known optimal. Five gap levels are tested: 1%, 5%, 10%, 30% and 50%. These are gaps between the upper bound and the optimal, not between the lower and upper bounds. The experimentation is conducted with objective counts from two to 15 and the three versions of the proposed MaOEA.

It is observed that none of these near optimal solutions found prior to the evolutionary algorithm remain in the final population at the end of the evolutionary algorithm. During the course of evolution, they are eliminated by the reference set based sorting algorithm because they are either dominated solutions or have a weak association with the reference point set. In addition to their absence in the final population, these near optimal solutions are always absent in the external population. This implies that they are definitely dominated solutions. There are some solutions with better objective function values in all objectives, and the evolutionary algorithm is capable of detecting these solutions. However, this should not mean that they are completely useless since their most important mission is to construct the fixed hyperplane.

When the hyperplane is not constructed using the lexicographic optimal solutions, some nondominated solutions may be found outside the non-negative orthant of the m-dimensional normalized objective space. In other words, there are nondominated solutions with negative coordinate values in some dimensions of normalized objective space. As a matter of fact, solutions with negative coordinate values can be observed during evolution and in the external population. In this case, it can be argued that the fixed hyperplane does not span the Pareto front and the reference points cannot properly represent all nondominated solutions. For this reason, some modifications are made in the evolutionary process of the reference point set. In particular, the mutation operator is modified to allow mutated reference points to have negative values. In order to prevent large deviations from the non-negative orthant, there is a limit on the allowed negative values, and these limits are dynamically determined using solutions with negative coordinate values discovered so far. These limits are also used in the immigration operator. When all these modifications come into effect, an alternative co-evolutionary algorithm ($FHCo^-$) is developed to work with fixed hyperplanes constructed using near optimal solutions. It should also be noted that when $FHCo^-$ is used with a fixed hyperplane constructed using optimal solutions, it reduces to $FHCo$ since no negative coordinate values will be encountered during evolution.

Table 5.40 shows HR results for alternatives using different levels of gap percentages between the near optimal solutions and single-objective optimal solutions for symmetric TSP. In Table 5.41, relative HR results are obtained normalizing by the original versions of algorithms that use single-objective optimal solutions. In this way, one can observe the depreciation in performance caused by using near optimal solutions instead of real ones. HR and relative HR results are presented similarly for asymmetric TSP in Table 5.42 and Table 5.43, respectively.

Obviously, there is a decrease in HR results when near optimal solutions are used instead of optimal solutions and the amount of decrease increases when the gap between the objectives of near optimal and optimal solutions is increased. Better interpretation can be made by examining relative HR results. The depreciation in performance is similar in both MaOTSP and MaOATSP instances. Besides, it can be observed that

Table 5.40. HR results with near optimal solutions for MaOTSP.

Algorithm	**Gap %**	**Objective count (m)**					
		2	**4**	**6**	**8**	**10**	**15**
FHFR	optimal	0.964	0.804	0.657	0.522	0.448	0.452
	near 1%	0.956	0.792	0.640	0.501	0.432	0.419
	near 5%	0.949	0.783	0.630	0.490	0.424	0.407
	near 10%	0.914	0.735	0.594	0.450	0.411	0.376
	near 30%	0.893	0.718	0.574	0.433	0.382	0.354
	near 50%	0.841	0.684	0.526	0.387	0.349	0.277
FHMR	optimal	0.976	0.822	0.683	0.617	0.541	0.558
	near 1%	0.969	0.815	0.677	0.611	0.536	0.552
	near 5%	0.955	0.794	0.657	0.591	0.521	0.524
	near 10%	0.934	0.779	0.636	0.573	0.509	0.507
	near 30%	0.917	0.771	0.629	0.565	0.501	0.479
	near 50%	0.865	0.711	0.574	0.505	0.455	0.346
FHCo	optimal	0.985	0.841	0.703	0.664	0.596	0.601
	near 1%	0.980	0.836	0.698	0.659	0.592	0.594
	near 5%	0.970	0.819	0.680	0.638	0.579	0.566
	near 10%	0.951	0.800	0.674	0.637	0.574	0.554
	near 30%	0.931	0.792	0.660	0.613	0.556	0.536
	near 50%	0.901	0.759	0.607	0.563	0.522	0.419
FHCo$^-$	optimal	0.985	0.841	0.703	0.664	0.596	0.601
	near 1%	0.982	0.839	0.700	0.658	0.597	0.609
	near 5%	0.976	0.833	0.694	0.654	0.593	0.591
	near 10%	0.967	0.834	0.691	0.651	0.587	0.585
	near 30%	0.965	0.827	0.683	0.638	0.585	0.570
	near 50%	0.933	0.801	0.645	0.578	0.542	0.480

depreciation amounts are generally less than the given gap between the objectives of near optimal and optimal solutions. This means that the algorithms are capable of covering the deficiency in performance in some extend. The most striking observation is the fact that the co-evolutionary versions of the proposed MaOEA give impressive performance. When the gap is as small as one percent, their performances are as good as using optimal solutions. Their performances are always better than *FHFR* and *FHMR* and are more robust against increased gap percentage and objective count.

Table 5.41. Relative HR results with near optimal solutions for MaOTSP.

Algorithm	**Gap %**	**Objective count (m)**					
		2	**4**	**6**	**8**	**10**	**15**
FHFR	optimal	100%	100%	100%	100%	100%	100%
	near 1%	99%	98%	97%	96%	96%	93%
	near 5%	98%	97%	96%	94%	95%	90%
	near 10%	95%	91%	90%	86%	92%	83%
	near 30%	93%	89%	87%	83%	85%	78%
	near 50%	87%	85%	80%	74%	78%	61%
FHMR	optimal	100%	100%	100%	100%	100%	100%
	near 1%	99%	99%	99%	99%	99%	99%
	near 5%	98%	97%	96%	96%	96%	94%
	near 10%	96%	95%	93%	93%	94%	91%
	near 30%	94%	94%	92%	92%	93%	86%
	near 50%	89%	86%	84%	82%	84%	62%
FHCo	optimal	100%	100%	100%	100%	100%	100%
	near 1%	100%	99%	99%	99%	99%	99%
	near 5%	98%	97%	97%	96%	97%	94%
	near 10%	97%	95%	96%	96%	96%	92%
	near 30%	95%	94%	94%	92%	93%	89%
	near 50%	91%	90%	86%	85%	88%	70%
FHCo^{-}	optimal	100%	100%	100%	100%	100%	100%
	near 1%	100%	100%	100%	99%	100%	101%
	near 5%	99%	99%	99%	99%	99%	98%
	near 10%	98%	99%	98%	98%	99%	97%
	near 30%	98%	98%	97%	96%	98%	95%
	near 50%	95%	95%	92%	87%	91%	80%

Furthermore, the new co-evolutionary algorithm, *FHCo*$^{-}$ provides better performance than the original *FHCo*. This proves the success of operator modifications in the evolution of the reference point set. When single-objective optimal solutions are difficult to find, *FHCo*$^{-}$ can be used with a hyperplane constructed with near optimal solutions. Its performance even with 50% near optimal solutions is better than the performance of *FHFR* that uses optimal solutions.

Table 5.42. HR results with near optimal solutions for MaOATSP.

Algorithm	**Gap %**	**Objective count (m)**					
		2	**4**	**6**	**8**	**10**	**15**
FHFR	optimal	0.956	0.788	0.601	0.581	0.486	0.452
	near 1%	0.954	0.763	0.580	0.560	0.460	0.419
	near 5%	0.950	0.777	0.584	0.540	0.457	0.406
	near 10%	0.898	0.720	0.551	0.490	0.444	0.378
	near 30%	0.872	0.709	0.530	0.482	0.423	0.361
	near 50%	0.848	0.622	0.473	0.432	0.373	0.274
FHMR	optimal	0.974	0.784	0.672	0.604	0.509	0.558
	near 1%	0.966	0.782	0.656	0.602	0.506	0.541
	near 5%	0.964	0.768	0.645	0.579	0.482	0.531
	near 10%	0.939	0.755	0.616	0.571	0.484	0.500
	near 30%	0.920	0.749	0.631	0.544	0.477	0.488
	near 50%	0.861	0.643	0.514	0.483	0.426	0.344
FHCo	optimal	0.973	0.871	0.688	0.603	0.583	0.601
	near 1%	0.966	0.862	0.682	0.591	0.587	0.600
	near 5%	0.947	0.845	0.653	0.580	0.556	0.574
	near 10%	0.935	0.817	0.669	0.586	0.562	0.542
	near 30%	0.920	0.803	0.656	0.549	0.540	0.546
	near 50%	0.880	0.780	0.560	0.492	0.496	0.423
FHCo$^-$	optimal	0.973	0.871	0.688	0.603	0.583	0.601
	near 1%	0.960	0.876	0.678	0.606	0.586	0.607
	near 5%	0.954	0.855	0.666	0.580	0.575	0.591
	near 10%	0.932	0.832	0.642	0.566	0.565	0.574
	near 30%	0.907	0.814	0.645	0.565	0.546	0.571
	near 50%	0.880	0.784	0.568	0.494	0.496	0.465

Similar experimentation is also made using MaOQAP. Since QAP cannot be optimally solved even for the single-objective problem instances that are not too large. The assessment is made by putting a time limit on the GA that is used to obtain the near optimal solutions. In this way, it is aimed to see the effects of the time limits of the preprocess stage and ultimately to reduce the time requirement without decreasing the quality.

Table 5.43. Relative HR results with near optimal solutions for MaOATSP.

Algorithm	**Gap %**	**Objective count (m)**					
		2	**4**	**6**	**8**	**10**	**15**
FHFR	optimal	100%	100%	100%	100%	100%	100%
	near 1%	100%	97%	97%	96%	95%	93%
	near 5%	99%	99%	97%	93%	94%	90%
	near 10%	94%	91%	92%	84%	91%	84%
	near 30%	91%	90%	88%	83%	87%	80%
	near 50%	89%	79%	79%	74%	77%	61%
FHMR	optimal	100%	100%	100%	100%	100%	100%
	near 1%	99%	100%	98%	100%	99%	97%
	near 5%	99%	98%	96%	96%	95%	95%
	near 10%	96%	96%	92%	95%	95%	90%
	near 30%	94%	96%	94%	90%	94%	88%
	near 50%	88%	82%	76%	80%	84%	62%
FHCo	optimal	100%	100%	100%	100%	100%	100%
	near 1%	99%	99%	99%	98%	101%	100%
	near 5%	97%	97%	95%	96%	95%	96%
	near 10%	96%	94%	97%	97%	96%	90%
	near 30%	95%	92%	95%	91%	93%	91%
	near 50%	90%	90%	81%	82%	85%	70%
FHCo$^-$	optimal	100%	100%	100%	100%	100%	100%
	near 1%	99%	101%	99%	100%	101%	101%
	near 5%	98%	98%	97%	96%	99%	98%
	near 10%	96%	96%	93%	94%	97%	95%
	near 30%	93%	93%	94%	94%	94%	95%
	near 50%	90%	90%	83%	82%	85%	77%

As seen in the case of TSP, it is possible to achieve satisfactory results despite the absence of single-objective optimal solutions. Relative HR results are presented in Table 5.44 and computation times are shown in Table 5.45. The experimentation is conducted on the problem instances with six objectives. In these experiments, it is attempted to reduce the time requirement caused by the preprocess stage of single-objective GAs that are carried out separately for each objective. The number of replications of the single-objective GA is reduced from 10 to one, and it is observed that

Table 5.44. Relative HR results for time reduction in MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**
NSGA-III	100%	100%	100%	100%	100%
NSGA-III (fixed)	112%	127%	113%	119%	106%
A-NSGA-III	102%	118%	114%	117%	105%
A-NSGA-III (fixed)	114%	131%	124%	129%	111%
A^2-NSGA-III	101%	112%	134%	128%	121%
A^2-NSGA-III (fixed)	121%	161%	154%	131%	148%
Two_Arch2	85%	95%	72%	92%	98%
RPEA	73%	88%	61%	81%	87%
MOEA/D (ws)	109%	118%	59%	88%	128%
MOEA/D (tc)	108%	132%	90%	105%	116%
PICEA-w	107%	112%	94%	83%	104%
FHFR	113%	150%	137%	158%	143%
FHMR	111%	158%	159%	162%	173%
FHCo	135%	165%	181%	192%	204%
FHCo$^-$	135%	165%	181%	193%	203%
FHFR (single rep. of GA)	113%	146%	148%	155%	147%
FHMR (single rep. of GA)	110%	156%	153%	156%	165%
FHCo (single rep. of GA)	132%	162%	173%	187%	182%
FHCo$^-$ (single rep. of GA)	135%	165%	181%	191%	199%
FHFR (single rep. of GA, halved time)	113%	144%	147%	128%	136%
FHMR (single rep. of GA, halved time)	110%	155%	152%	150%	154%
FHCo (single rep. of GA, halved time)	131%	161%	172%	180%	171%
FHCo$^-$ (single rep. of GA, halved time)	135%	165%	180%	196%	194%

the single-objective GA is still capable of finding near optimal solutions sufficient for the purpose. The fixed hyperplane constructed in this way results in the same quality

Table 5.45. Computation time (sec) for time reduction in MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**
NSGA-III	39	56	54	212	473
NSGA-III (fixed)	50	105	106	463	1268
A-NSGA-III	47	62	59	208	361
A-NSGA-III (fixed)	57	129	121	391	1195
A^2-NSGA-III	41	50	44	182	455
A^2-NSGA-III (fixed)	54	101	88	365	1392
Two_Arch2	74	137	137	392	1113
RPEA	93	221	216	746	1508
MOEA/D (ws)	59	106	103	357	945
MOEA/D (tc)	57	108	104	359	951
PICEA-w	119	179	153	469	1223
FHFR	63	130	121	534	1385
FHMR	74	152	141	682	1687
FHCo	82	163	162	702	2081
FHCo$^-$	83	165	165	708	2083
FHFR (single rep. of GA)	53	85	75	301	662
FHMR (single rep. of GA)	62	104	93	437	921
FHCo (single rep. of GA)	69	112	108	443	1317
FHCo$^-$ (single rep. of GA)	70	114	111	451	1347
FHFR (single rep. of GA, halved time)	53	81	71	277	585
FHMR (single rep. of GA, halved time)	62	100	89	415	865
FHCo (single rep. of GA, halved time)	69	109	105	427	1286
FHCo$^-$ (single rep. of GA, halved time)	70	111	106	438	1314
single-objective problems (10 replications of GA)	12	52	55	265	817

approximation although there is a significant reduction in the time requirement. To test further time reduction, alternatives are obtained for each of the proposed MaOEA versions, where the time allowed for the preprocess GA is halved.

The experimentation has shown that in the event of difficulties in obtaining single-objective optimal solutions or when a great effort is required for this, it is possible to construct the hyperplane with near optimal solutions and still achieve satisfactory Pareto approximations. In the case of QAP, most of the computational effort are spent at this preprocessing stage. The results indicate that when the fixed hyperplane is constructed from the solutions obtained from shortened processes, there is no significant depreciation in the quality of the Pareto front approximation. Therefore, this is a valid compromising approach for problems like QAP, especially when *FHCo*$^{-}$ is used.

5.7. Test Problems with Correlated Objectives

In this section, the performances of the three versions of the proposed MaOEA and other existing state-of-the-art evolutionary algorithms are examined in the presence of a correlation between the objectives of a MaOP. For this purpose, MaOTSP and MaOATSP instances with 10 objectives are selected.

Many-objective TSP instances with correlated objectives are generated as follows: the first two objectives, $f_1(\vec{\mathbf{x}})$ and $f_2(\vec{\mathbf{x}})$, are generated as before. The rest of the objectives are generated using the methodology in Ishibuchi *et al.* (2011) to obtain correlated objectives as shown in Equation 5.1 and 5.2.

$$f_i(\vec{\mathbf{x}}) = \alpha f_1(\vec{\mathbf{x}}) + (1-\alpha) g_i(\vec{\mathbf{x}}), \qquad i = 3, 5, 7, 9 \tag{5.1}$$

$$f_i(\vec{\mathbf{x}}) = \alpha f_2(\vec{\mathbf{x}}) + (1-\alpha) g_i(\vec{\mathbf{x}}), \qquad i = 4, 6, 8, 10 \tag{5.2}$$

where correlation coefficient α is a positive real number in the $(0, 1)$ interval. In this formulation, $g_i(\vec{\mathbf{x}})$ represents random objectives and $f_i(\vec{\mathbf{x}})$ represents correlated objec-

tives. Two groups are obtained using this methodology: $\{f_i(\vec{\mathbf{x}}) : i = 1, 3, 5, 7, 9\}$ and $\{f_i(\vec{\mathbf{x}}) : i = 2, 4, 6, 8, 10\}$. When α is close to zero, $f_i(\vec{\mathbf{x}})$ values are almost the same as randomly generated objectives. When α is close to one, $f_i(\vec{\mathbf{x}})$ values become almost the same as other objectives in their groups. This methodology ensures to have a controlled correlation within the groups but not between them.

In computational experiments, four levels are examined for α, 0.2, 0.4, 0.6, and 0.8, where high values indicate strong correlation between the objectives. It should be noted that negative correlation values have not been tested. The reason is that the presence of negative correlations between objective functions makes the problem even more complicated as almost all feasible solutions become nondominated. As a result, it is not seen in the literature, and the question of how to generate test instances remains open. Performance comparison is done in the same way as before. HR and relative HR results are presented in Table 5.46 and Table 5.47 for symmetric TSP, and in Table 5.48 and Table 5.49 for asymmetric TSP.

Table 5.46. HR results with correlated objectives for MaOTSP.

Algorithm	**Correlation coefficient (α)**				
	0	**0.2**	**0.4**	**0.6**	**0.8**
NSGA-III	0.356	0.478	0.602	0.703	0.794
NSGA-III (fixed)	0.425	0.513	0.627	0.727	0.806
A-NSGA-III	0.396	0.496	0.620	0.723	0.812
A-NSGA-III (fixed)	0.417	0.514	0.637	0.742	0.844
A^2-NSGA-III	0.431	0.568	0.684	0.794	0.885
A^2-NSGA-III (fixed)	0.486	0.594	0.701	0.815	0.915
Two_Arch2	0.398	0.538	0.673	0.784	0.887
RPEA	0.327	0.420	0.521	0.604	0.680
MOEA/D (ws)	0.409	0.497	0.583	0.663	0.732
MOEA/D (tc)	0.382	0.484	0.572	0.623	0.673
PICEA-w	0.421	0.533	0.658	0.761	0.842
FHFR	0.448	0.593	0.723	0.823	0.905
FHMR	0.541	0.624	0.732	0.835	0.917
FHCo	0.596	0.660	0.743	0.854	0.943

Table 5.47. Relative HR results with correlated objectives for MaOTSP.

Algorithm	**Correlation coefficient (α)**				
	0	**0.2**	**0.4**	**0.6**	**0.8**
NSGA-III	100%	100%	100%	100%	100%
NSGA-III (fixed)	119%	107%	104%	103%	102%
A-NSGA-III	111%	104%	103%	103%	102%
A-NSGA-III (fixed)	117%	107%	106%	106%	106%
A^2-NSGA-III	121%	119%	114%	113%	112%
A^2-NSGA-III (fixed)	137%	124%	116%	116%	115%
Two_Arch2	112%	113%	112%	111%	112%
RPEA	92%	88%	87%	86%	86%
MOEA/D (ws)	115%	104%	97%	94%	92%
MOEA/D (tc)	107%	101%	95%	89%	85%
PICEA-w	118%	111%	109%	108%	106%
FHFR	126%	124%	120%	117%	114%
FHMR	152%	130%	121%	119%	116%
FHCo	167%	138%	123%	121%	119%

Table 5.48. HR results with correlated objectives for MaOATSP.

Algorithm	**Correlation coefficient (α)**				
	0	**0.2**	**0.4**	**0.6**	**0.8**
NSGA-III	0.409	0.513	0.638	0.724	0.796
NSGA-III (fixed)	0.436	0.542	0.656	0.737	0.798
A-NSGA-III	0.437	0.534	0.660	0.749	0.819
A-NSGA-III (fixed)	0.506	0.610	0.714	0.779	0.844
A^2-NSGA-III	0.436	0.537	0.640	0.719	0.779
A^2-NSGA-III (fixed)	0.507	0.578	0.670	0.758	0.827
Two_Arch2	0.412	0.519	0.637	0.724	0.801
RPEA	0.331	0.417	0.515	0.578	0.627
MOEA/D (ws)	0.456	0.518	0.587	0.637	0.665
MOEA/D (tc)	0.433	0.512	0.572	0.618	0.652
PICEA-w	0.484	0.569	0.663	0.750	0.803
FHFR	0.486	0.576	0.696	0.771	0.825
FHMR	0.509	0.610	0.706	0.786	0.843
FHCo	0.583	0.653	0.733	0.818	0.882

Table 5.49. Relative HR results with correlated objectives for MaOATSP.

Algorithm	**Correlation coefficient (α)**				
	0	**0.2**	**0.4**	**0.6**	**0.8**
NSGA-III	100%	100%	100%	100%	100%
NSGA-III (fixed)	107%	106%	103%	102%	100%
A-NSGA-III	107%	104%	104%	104%	103%
A-NSGA-III (fixed)	124%	119%	112%	108%	106%
A^2-NSGA-III	107%	105%	100%	99%	98%
A^2-NSGA-III (fixed)	124%	113%	105%	105%	104%
Two_Arch2	101%	101%	100%	100%	101%
RPEA	81%	81%	81%	80%	79%
MOEA/D (ws)	111%	101%	92%	88%	84%
MOEA/D (tc)	106%	100%	90%	85%	82%
PICEA-w	118%	111%	104%	104%	101%
FHFR	119%	112%	109%	107%	104%
FHMR	124%	119%	111%	109%	106%
FHCo	143%	127%	115%	113%	111%

As the correlation strength increases, the performance of nondominated sorting based algorithms (NSGA-III, *A*-NSGA-III, and A^2-NSGA-III) significantly improves. Likewise, the performance of the proposed algorithm also improves since it possesses Pareto dominance-based characteristic. When the objectives are highly correlated, the number of nondominated solutions decreases and the Pareto dominance comparison becomes more conclusive. On the contrary, the performance of MOEA/D relatively deteriorates, since it is a decomposition-based algorithm and uses static weight vector set. When there is a correlation between the objectives, most of the static weight vectors become useless. In other words, Pareto dominance reinstates its importance in the presence of correlation between objectives. This is an important fact since some level of correlation is always expected when dealing with real-world problems.

Obviously, if there is a significant correlation between the objectives of a MaOP, it is possible to reduce the number of objectives and transform the problem into a less dimensional problem where the difficulties encountered in many-objective optimization

vanish. As can be seen from the results, higher correlation leads to higher HR values for all algorithms and the difference between the algorithm alternatives become less pronounced.

5.8. Test Problems with Different Objective Scales

In this section, analyses on the performances of the proposed MaOEA and other examined algorithms are presented with problem instances with objective functions of different scales. All experimentation made so far include problem instances using the same distribution and scale in objective function parameters. For this purpose, many-objective TSP is chosen. TSP can have objective functions such as cost, distance, time, energy consumption, risk, and attractiveness, each defined at different scales.

In Lust and Teghem (2010a), four types of problem instances are produced for TSP. In the first type of problem instance, called Euclidean instances, c_{jk} cost values correspond to the Euclidean distance between two points randomly located in a two-dimensional plane. In this way, the cost parameter values obey the triangular inequality, $c_{xz} \leq c_{xy} + c_{yz}\ \forall x, y, z$, and they become symmetrical, $c_{xy} = c_{yx}\ \forall x, y$. Symmetric TSP instances used in this thesis fit such problem types. Second, in random instances, c_{jk} cost values are randomly generated using a uniform distribution. Asymmetric TSP instances used in this thesis fit the second type of problems. In the third type, in mixed instances, some objectives follow Euclidean instances, while others follow random instances. The last problem type is clustered instances where cities are randomly partitioned into clusters, and the objective is to visit the cities of each cluster consecutively. It should be noted that the last problem type is not used in experimentation as they correspond to a fundamentally different problem structure.

Many-objective TSP instances with two to 15 objectives are selected to observe the impact of the increase in the number of objectives on performances. In symmetric TSP, the locations of 100 cities are randomly generated in a rectangle with dimensions $(2000 \times i)$ by $(1000 \times i)$, for every objective $i = 1, 2, ..., 15$. In asymmetric TSP, 50×50 cost matrix values are randomly specified using discrete uniform distribution in the

interval $[1000 \times i - 1000, 2000 \times i - 1000]$, for every objective $i = 1, 2, ..., 15$. In mixed TSP, half of the objective functions are specified as in symmetric TSP, while the other half comes from asymmetric TSP.

Performance comparison is done in the same way as before. HR and relative HR results are presented in Table 5.50 and Table 5.51 for symmetric TSP, in Table 5.52 and Table 5.53 for asymmetric TSP, and in Table 5.54 and Table 5.55 for mixed TSP.

Table 5.50. HR results with different objective scales, MaOTSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.881	0.608	0.503	0.415	0.351	0.323
NSGA-III (fixed)	0.958	0.786	0.643	0.527	0.424	0.413
A-NSGA-III	0.878	0.611	0.508	0.426	0.387	0.388
A-NSGA-III (fixed)	0.939	0.737	0.586	0.481	0.416	0.420
A^2-NSGA-III	0.881	0.607	0.586	0.465	0.426	0.418
A^2-NSGA-III (fixed)	0.912	0.752	0.638	0.582	0.481	0.486
Two_Arch2	0.953	0.776	0.565	0.463	0.392	0.395
RPEA	0.893	0.606	0.459	0.361	0.321	0.291
MOEA/D (ws)	0.897	0.605	0.462	0.427	0.402	0.415
MOEA/D (tc)	0.919	0.682	0.518	0.463	0.374	0.408
PICEA-w	0.938	0.768	0.542	0.486	0.420	0.440
FHFR	0.969	0.800	0.651	0.515	0.446	0.444
FHMR	0.981	0.817	0.676	0.609	0.537	0.550
FHCo	0.986	0.836	0.702	0.661	0.591	0.595

The results display the ongoing superiority of the proposed MaOEA versions, especially *FHCo*. Nevertheless, the relative HR results of the proposed MaOEA are not as high in the mixed instances as in symmetric instances. As a matter of fact, a decrease in relative HR values is observed in almost all algorithms indicating that the performance of NSGA-III is relatively improved.

Additionally, the relative HR values in almost every row proceed similarly, i.e. algorithms respond similarly to the increase in the number of objectives. The differ-

Table 5.51. Relative HR results with different objective scales, MaOTSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	100%	100%	100%	100%	100%	100%
NSGA-III (fixed)	109%	129%	128%	127%	121%	128%
A-NSGA-III	100%	100%	101%	103%	110%	120%
A-NSGA-III (fixed)	107%	121%	116%	116%	118%	130%
A^2-NSGA-III	100%	100%	116%	112%	121%	129%
A^2-NSGA-III (fixed)	104%	124%	127%	140%	137%	150%
Two_Arch2	108%	128%	112%	112%	112%	122%
RPEA	101%	100%	91%	87%	92%	90%
MOEA/D (ws)	102%	100%	92%	103%	115%	129%
MOEA/D (tc)	104%	112%	103%	112%	107%	126%
PICEA-w	106%	126%	108%	117%	120%	136%
FHFR	110%	132%	129%	124%	127%	137%
FHMR	111%	134%	134%	147%	153%	170%
FHCo	112%	138%	140%	159%	169%	184%

Table 5.52. HR results with different objective scales, MaOATSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.894	0.716	0.592	0.519	0.401	0.371
NSGA-III (fixed)	0.931	0.781	0.625	0.550	0.433	0.409
A-NSGA-III	0.936	0.742	0.605	0.545	0.435	0.418
A-NSGA-III (fixed)	0.955	0.800	0.668	0.575	0.503	0.435
A^2-NSGA-III	0.965	0.793	0.649	0.567	0.431	0.413
A^2-NSGA-III (fixed)	0.931	0.841	0.693	0.581	0.497	0.480
Two_Arch2	0.935	0.715	0.542	0.413	0.407	0.389
RPEA	0.900	0.625	0.524	0.410	0.322	0.293
MOEA/D (ws)	0.866	0.665	0.564	0.496	0.452	0.355
MOEA/D (tc)	0.873	0.659	0.523	0.478	0.433	0.368
PICEA-w	0.926	0.790	0.606	0.559	0.483	0.417
FHFR	0.954	0.784	0.599	0.574	0.477	0.441
FHMR	0.972	0.781	0.669	0.601	0.500	0.551
FHCo	0.973	0.871	0.686	0.603	0.579	0.590

Table 5.53. Relative HR results with different objective scales, MaOATSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	100%	100%	100%	100%	100%	100%
NSGA-III (fixed)	104%	109%	106%	106%	108%	110%
A-NSGA-III	105%	104%	102%	105%	109%	113%
A-NSGA-III (fixed)	107%	112%	113%	111%	126%	117%
A^2-NSGA-III	108%	111%	110%	109%	108%	111%
A^2-NSGA-III (fixed)	104%	117%	117%	112%	124%	129%
Two_Arch2	105%	100%	92%	80%	102%	105%
RPEA	101%	87%	88%	79%	80%	79%
MOEA/D (ws)	97%	93%	95%	96%	113%	96%
MOEA/D (tc)	98%	92%	88%	92%	108%	99%
PICEA-w	104%	110%	102%	108%	120%	113%
FHFR	107%	110%	101%	111%	119%	119%
FHMR	109%	109%	113%	116%	125%	149%
FHCo	109%	122%	116%	116%	144%	159%

Table 5.54. HR results with different objective scales, mixed MaOTSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	0.873	0.648	0.547	0.463	0.396	0.348
NSGA-III (fixed)	0.959	0.788	0.610	0.531	0.421	0.395
A-NSGA-III	0.904	0.654	0.556	0.494	0.424	0.423
A-NSGA-III (fixed)	0.937	0.784	0.608	0.510	0.442	0.422
A^2-NSGA-III	0.933	0.714	0.606	0.527	0.402	0.390
A^2-NSGA-III (fixed)	0.931	0.790	0.682	0.566	0.501	0.486
Two_Arch2	0.949	0.727	0.557	0.418	0.411	0.403
RPEA	0.916	0.626	0.509	0.393	0.298	0.273
MOEA/D (ws)	0.880	0.618	0.509	0.449	0.446	0.384
MOEA/D (tc)	0.905	0.678	0.532	0.467	0.408	0.395
PICEA-w	0.909	0.759	0.568	0.499	0.428	0.439
FHFR	0.940	0.785	0.600	0.521	0.443	0.416
FHMR	0.956	0.792	0.657	0.619	0.498	0.548
FHCo	0.980	0.843	0.696	0.645	0.571	0.576

Table 5.55. Relative HR results with different objective scales, mixed MaOTSP.

Algorithm	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
NSGA-III	100%	100%	100%	100%	100%	100%
NSGA-III (fixed)	110%	122%	112%	115%	106%	113%
A-NSGA-III	103%	101%	102%	107%	107%	122%
A-NSGA-III (fixed)	107%	121%	111%	110%	112%	121%
A^2-NSGA-III	107%	110%	111%	114%	101%	112%
A^2-NSGA-III (fixed)	107%	122%	125%	122%	126%	140%
Two_Arch2	109%	112%	102%	90%	104%	116%
RPEA	105%	97%	93%	85%	75%	79%
MOEA/D (ws)	101%	95%	93%	97%	112%	110%
MOEA/D (tc)	104%	105%	97%	101%	103%	114%
PICEA-w	104%	117%	104%	108%	108%	126%
FHFR	108%	121%	110%	113%	112%	120%
FHMR	110%	122%	120%	134%	126%	157%
FHCo	112%	130%	127%	139%	144%	165%

ence between the algorithms becomes less significant as NSGA-III is not affected by objectives with different scales due to its normalization procedure.

5.9. Search for Robust Solutions

This section includes numerical testing of the capability of the proposed MaOEA versions, as well as other algorithm alternatives to achieve a Pareto approximation that contains successful solutions regarding different robust metrics. MaOKP and MaOQAP are used as benchmark problems for this purpose.

First, validation of the proposed MaOEA is done by comparing with the optimal solutions for stochastic and robust performance metrics, namely *expected cost*, *minimax cost*, *minimax absolute regret*, and *minimax relative regret* found by *CPLEX*. The ratio test in Table 5.56 presents the percentage values calculated by dividing the optimal values by the best objective values found in the Pareto approximation for the

corresponding performance metric. The results indicate that the Pareto approximation found by the proposed MaOEA (*FHFR* version) contains near optimal solutions for different performance metrics.

Table 5.56. Percentage results for the ratio test in MaOKP, $n = 500$.

Criterion	**Objective count (m)**					
	2	**4**	**6**	**8**	**10**	**15**
Expected Cost	100%	100%	100%	100%	99.6%	100%
Minimax Cost	100%	99.8%	99.5%	99.1%	97.9%	98.4%
Minimax Abs. Regret	100%	100%	99.2%	98.7%	98.3%	98.6%
Minimax Rel. Regret	100%	100%	99.4%	99.2%	98.0%	99.1%

Performance evaluation of algorithm alternatives is made with six-objective MaO-QAP instances since it is difficult to obtain any robust optimal solution with exact methods and evolutionary approaches are required. *FHCo*$^{-}$ is used as the co-evolutionary version of the proposed MaOEA.

Best robust objective values found in 10 replications by each algorithm alternative are presented in Table 5.57 for minimax cost, in Table 5.58 for minimax absolute regret, and in Table 5.59 for minimax relative regret. These values are normalized in Table 5.60 to 5.62 in the respective order. The ranking of the algorithms is provided next to normalized results. It should be noted that smaller percentage values are desired. The best result is shown with an asterisk for each robust metric.

Different versions of the proposed MaOEA have the best results in terms of different robust measures. Nevertheless, *FHCo*$^{-}$ is ranked as the best alternative based on Friedman's test, and the results indicate the superiority of the proposed MaOEA versions over other algorithms.

Table 5.57. Minimax cost results for MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**
NSGA-III	7722	226850	4692	170946	17936
NSGA-III (fixed)	6954	218920	4550	179192	17190
A-NSGA-III	7288	238780	4696	173226	17452
A-NSGA-III (fixed)	7116	215950	4842	172650	17552
A^2-NSGA-III	7646	227874	4836	176262	17946
A^2-NSGA-III (fixed)	7146	222436	4556	175628	17824
Two_Arch2	7156	219792	4730	169608	17320
RPEA	7582	243008	4872	170826	16732*
MOEA/D (ws)	7428	232924	4706	168840	17170
MOEA/D (tc)	7832	225352	4826	182152	18060
PICEA-w	6658	217560	4550	174648	16840
FHFR	6218	224986	4422*	172542	17736
FHMR	6122*	221778	4460	164992*	16834
FHCo$^-$	6176	206520*	4422*	169762	16872

Table 5.58. Minimax absolute regret results for MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**
NSGA-III	226	23168	272	14130	1544
NSGA-III (fixed)	206	22776	272	14892	1534
A-NSGA-III	208	23468	284	15000	1498
A-NSGA-III (fixed)	202	22088	268	14796	1452
A^2-NSGA-III	226	22794	292	14024	1560
A^2-NSGA-III (fixed)	216	21916*	262	13992	1498
Two_Arch2	216	22938	288	14236	1556
RPEA	218	25558	292	14020	1494
MOEA/D (ws)	208	22874	262	14356	1474
MOEA/D (tc)	224	22920	270	14604	1524
PICEA-w	206	21950	252	15034	1440
FHFR	196	23400	262	13990	1590
FHMR	190	22474	252	13566*	1410*
FHCo$^-$	184*	22442	246*	13628	1426

Table 5.59. Minimax relative regret results for MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**
NSGA-III	0.171	0.303	0.206	0.226	0.286
NSGA-III (fixed)	0.159	0.300	0.199	0.224	0.280
A-NSGA-III	0.148	0.294	0.208	0.238	0.282
A-NSGA-III (fixed)	0.145	0.284	0.199	0.231	0.275
A^2-NSGA-III	0.176	0.310	0.211	0.227	0.300
A^2-NSGA-III (fixed)	0.155	0.276	0.200	0.225	0.275
Two_Arch2	0.165	0.294	0.216	0.234	0.291
RPEA	0.172	0.320	0.214	0.229	0.275
MOEA/D (ws)	0.156	0.294	0.207	0.222	0.267
MOEA/D (tc)	0.175	0.300	0.212	0.224	0.276
PICEA-w	0.155	0.297	0.185	0.230	0.255
FHFR	0.147	0.291	0.207	0.223	0.281
FHMR	0.138	0.273	0.191	0.221*	0.251*
FHCo$^-$	0.135*	0.266*	0.184*	0.222	0.258

Table 5.60. Normalized minimax cost results for MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**	**Friedman test**
NSGA-III	100% (13)	100% (10)	100% (7)	100% (6)	100% (12)	9.6 (12)
NSGA-III (fixed)	90% (5)	97% (4)	97% (5)	105% (13)	96% (6)	6.6 (5)
A-NSGA-III	94% (9)	105% (13)	100% (8)	101% (9)	97% (8)	9.4 (11)
A-NSGA-III (fixed)	92% (6)	95% (2)	103% (13)	101% (8)	98% (9)	7.6 (7)
A^2-NSGA-III	99% (12)	100% (11)	103% (12)	103% (12)	100% (13)	12.0 (13)
A^2-NSGA-III (fixed)	93% (7)	98% (7)	97% (6)	103% (11)	99% (11)	8.4 (9)
Two_Arch2	93% (8)	97% (5)	101% (10)	99% (3)	97% (7)	6.6 (5)
RPEA	98% (11)	107% (14)	104% (14)	100% (5)	93% (1)	9.0 (10)
MOEA/D (ws)	96% (10)	103% (12)	100% (9)	99% (2)	96% (5)	7.6 (7)
MOEA/D (tc)	101% (14)	99% (9)	103% (11)	107% (14)	101% (14)	12.4 (14)
PICEA-w	86% (4)	96% (3)	97% (4)	102% (10)	94% (3)	4.8 (3)
FHFR	81% (3)	99% (8)	94% (1)	101% (7)	99% (10)	5.8 (4)
FHMR	79% (1)	98% (6)	95% (3)	97% (1)	94% (2)	2.6 (2)
FHCo$^-$	80% (2)	91% (1)	94% (1)	99% (4)	94% (4)	2.4 (1)

Table 5.61. Normalized minimax absolute regret results for MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**	**Friedman test**
NSGA-III	100% (13)	100% (11)	100% (10)	100% (7)	100% (11)	10.4 (11)
NSGA-III (fixed)	91% (5)	98% (6)	100% (9)	105% (12)	99% (10)	8.4 (8)
A-NSGA-III	92% (7)	101% (13)	104% (11)	106% (13)	97% (8)	10.4 (11)
A-NSGA-III (fixed)	89% (4)	95% (3)	99% (7)	105% (11)	94% (4)	5.8 (5)
A^2-NSGA-III	100% (14)	98% (7)	107% (13)	99% (6)	101% (13)	10.6 (14)
A^2-NSGA-III (fixed)	96% (9)	95% (1)	96% (4)	99% (4)	97% (7)	5.0 (3)
Two_Arch2	96% (10)	99% (10)	106% (12)	101% (8)	101% (12)	10.4 (11)
RPEA	96% (11)	110% (14)	107% (14)	99% (5)	97% (6)	10.0 (10)
MOEA/D (ws)	92% (8)	99% (8)	96% (5)	102% (9)	95% (5)	7.0 (6)
MOEA/D (tc)	99% (12)	99% (9)	99% (8)	103% (10)	99% (9)	9.6 (9)
PICEA-w	91% (6)	95% (2)	93% (2)	106% (14)	93% (3)	5.4 (4)
FHFR	87% (3)	101% (12)	96% (6)	99% (3)	103% (14)	7.6 (7)
FHMR	84% (2)	97% (5)	93% (3)	96% (1)	91% (1)	2.4 (2)
FHCo$^-$	81% (1)	97% (4)	90% (1)	96% (2)	92% (2)	2.0 (1)

Table 5.62. Normalized minimax relative regret results for MaOQAP, $m = 6$.

Algorithm	**had16**	**scr20**	**nug20**	**kra30a**	**ste36a**	**Friedman test**
NSGA-III	100% (11)	100% (12)	100% (7)	100% (8)	100% (12)	10.0 (11)
NSGA-III (fixed)	93% (9)	99% (11)	97% (5)	99% (5)	98% (9)	7.8 (8)
A-NSGA-III	87% (5)	97% (7)	101% (10)	105% (14)	99% (11)	9.4 (9)
A-NSGA-III (fixed)	85% (3)	94% (4)	97% (4)	102% (12)	96% (5)	5.6 (3)
A^2-NSGA-III	103% (14)	102% (13)	103% (11)	100% (9)	105% (14)	12.2 (14)
A^2-NSGA-III (fixed)	91%(7)	91% (3)	97% (6)	100% (7)	96% (6)	5.8 (4)
Two_Arch2	97% (10)	97% (8)	105% (14)	103% (13)	102% (13)	11.6 (13)
RPEA	101% (12)	106% (14)	104% (13)	101% (10)	96% (7)	11.2 (12)
MOEA/D (ws)	91% (8)	97% (6)	101% (9)	98% (3)	94% (4)	6.0 (5)
MOEA/D (tc)	102% (13)	99% (10)	103% (12)	99% (6)	96% (8)	9.8 (10)
PICEA-w	91% (6)	98% (9)	90% (2)	102% (11)	89% (2)	6.0 (5)
FHFR	86% (4)	96% (5)	101% (8)	99% (4)	98% (10)	6.2 (7)
FHMR	81% (2)	90% (2)	93% (3)	98% (1)	88% (1)	1.8 (2)
FHCo$^-$	79% (1)	88% (1)	90% (1)	98% (2)	90% (3)	1.6 (1)

5.10. Alternative Limit Specification for Hypervolume Calculation

Researchers have suggested that the performance comparisons depend on the location and size of the hypercube (Ishibuchi *et al.*, 2017). In this section, additional experimentation is carried out to analyze the effects of the choice for the boundaries of the hypercube.

In a MaOKP, where the objectives of the problem are maximization, it is convenient to use the ideal point as the upper limit and the origin as the lower limit for the hypervolume calculation. For minimization on the other hand, although it is appropriate to use the ideal point for the lower limit, the choice for the upper limit is ambiguous. An estimate for the nadir point can be used, but in that case the hypercube becomes too narrow. When the hypercube is too tight, the inconclusive area occupies most i.e., the region in which the Pareto approximation under examination is neither dominant nor dominated. This results in very small HR values and performance comparison of different Pareto approximations becomes difficult. On the other hand, very large hypercubes may weaken the accuracy of the Monte Carlo simulation by increasing its standard error. It is also suggested in the literature that different specifications for the hypervolume box may lead to different comparison results. It is claimed that Pareto approximations with a diverse solution set perform better in large hypercubes.

To analyze this issue, the performances of different algorithms on MaOTSP are evaluated using different upper limit specifications for the hypercube. The upper limits, which are 1%, 17%, 50%, and 100% worse than the nadir point estimate, are used to calculate HR results in 10 replications in Table 5.63. The ranking of the algorithms is provided next to HR results. Finding the nadir point is a difficult task in itself and numerous algorithms (Kirlik and Sayın, 2015) and evolutionary approaches (Deb and Miettinen, 2010) have been developed for this purpose. Here, the nadir point estimate is obtained by combining external populations of all algorithms.

From Table 5.63 it can be seen that different algorithms lead to significantly different HR values as the specified upper limit moves away from the estimated nadir

Table 5.63. HR results for hypervolume limit specification, MaOTSP, $m = 10$.

Algorithm	**Upper limit ratio to nadir point**			
	1.01	**1.17**	**1.5**	**2**
NSGA-III	0.009 (12)	0.028 (13)	0.234 (13)	0.356 (13)
NSGA-III (fixed)	0.011 (9)	0.033 (9)	0.278 (9)	0.425 (6)
A-NSGA-III	0.008 (13)	0.029 (12)	0.267 (12)	0.396 (11)
A-NSGA-III (fixed)	0.009 (11)	0.031 (11)	0.272 (11)	0.417 (8)
A^2-NSGA-III	0.007 (14)	0.033 (10)	0.284 (7)	0.431 (5)
A^2-NSGA-III (fixed)	0.012 (8)	0.049 (3)	0.354 (3)	0.486 (3)
Two_Arch2	0.014 (6)	0.034 (7)	0.277 (10)	0.398 (10)
RPEA	0.014 (7)	0.026 (14)	0.212 (14)	0.327 (14)
MOEA/D (ws)	0.017 (3)	0.035 (6)	0.286 (6)	0.409 (9)
MOEA/D (tc)	0.011 (10)	0.034 (8)	0.279 (8)	0.382 (12)
PICEA-w	0.015 (5)	0.041 (5)	0.303 (5)	0.421 (7)
FHFR	0.016 (4)	0.045 (4)	0.312 (4)	0.448 (4)
FHMR	0.017 (2)	0.052 (2)	0.356 (2)	0.541 (2)
FHCo	0.019 (1)	0.056 (1)	0.393 (1)	0.596 (1)

point and the hypercube gets larger. It is important to use the appropriate specification limits to differentiate the effects of different parameters and operators during the algorithm development phase. Nevertheless, it should be noted that all versions of the proposed MaOEA perform better than other algorithm alternatives in all upper limit specifications.

Table 5.64. HR results for hypervolume limit specification, *FHCo*, MaOTSP.

Objective count (m)	**Upper limit ratio to nadir point**			
	1.01	**1.17**	**1.5**	**2**
2	0.618	0.756	0.886	0.985
4	0.437	0.564	0.714	0.841
6	0.213	0.356	0.639	0.703
8	0.086	0.121	0.453	0.664
10	0.019	0.056	0.393	0.596
15	0.005	0.012	0.311	0.601

The choice of a suitable hypercube size largely depends on the number of objectives. From the HR results of *FHCo* presented in Table 5.64, it can be said that as the number of objectives increases, the hypercube needs to be larger. Based on the results, the most appropriate level for the upper limit ratio to nadir point appears as two, which has been used throughout the numerical studies presented in this chapter.

6. CONCLUSION

In this thesis, the challenges imposed by many-objectives in discrete search spaces are targeted in order to develop a successful MaOEA for combinatorial optimization problems.

The proposed MaOEA, which uses a reference point set based approach that employs reference based sorting along with elitist nondominated sorting, achieves to increase the efficiency of this kind of approaches by introducing several innovative ideas and strategies, especially related to the creation and management of the reference set. While reference set based approaches are considered as powerful methodologies when it comes to MaOEAs, this study demonstrates that their strength can be significantly enhanced by fixing the hyperplane that encompasses the reference points at the beginning. In the proposed algorithm, reference points are mapped on a hyperplane which is constructed at the beginning of the algorithm using lexicographic minima for the individual objectives, and its position in the objective space remains fixed throughout the algorithm. It has also been shown that the efficiency can be improved further by using evolutionary strategies and genetic operators specifically designed to complement the reference set concept. Reference point guided path relinking recombination scheme, complementing parent selection mechanisms, mutation and local improvement operators are the main features contributing to this success.

Nonetheless, the main strength of the proposed MaOEA comes from allowing the reference set to adapt to the underlying Pareto front by moving on a fixed hyperplane that has a good grip of the front. While the concept of co-evolution has been used in a few MaOEAs in the literature before, it has not been used for a reference set based approach in the sense it is proposed in this thesis. The proposed algorithm *FHCo* provides a practical and robust co-evolutionary structure that involves both cooperative and competitive aspects to achieve adaptation of the reference set to the topology of the Pareto front. The self-adaptive parametric nature not only precludes parameter fine-tuning efforts and the risk of overfitting to test problem instances, but also creates

a flexible and convenient way to balance diversity and convergence. Since the reference points are defined as coordinate points in the normalized objective space, the methods used in their evolution need not to be problem specific. For instance, a problem such as creating an infeasible reference point never arises. Thus, the development and use of the evolution of the reference set can be generic.

The main contributions of this research consist not only of proposing a successful reference set based algorithm, and introducing the concept of co-evolution of the reference points, but also verifying that it can be applied to a variety of different combinatorial problems in a sustainable manner. Hence, the research builds upon three well-known combinatorial optimization problems used as three main pillars.

The work done using MaOKP establishes the first pillar and leads to the methodological contribution to the field of evolutionary many-objective optimization to deal with combinatorial problems. MaOKP serves as a perfect benchmark problem in the development and performance evaluation of the proposed MaOEA, since the effects of genetic operators and parameter selections can easily be observed on MaOKP as run times are relatively low. Moreover, it is commonly used by various other studies experimenting on MOEAs and MaOEAs, hence it is possible to compare the performance of the proposed MaOEA with existing studies in the literature. The proposed MaOEA solves MaOKP very effectively. As such, to the best of our knowledge, the presented results are the best so far in the literature for MaOKP.

The second pillar, namely the research using MaOTSP as the benchmark problem, leads to the verification of the success of the proposed MaOEA under different problem-specific features of combinatorial problems. First of all, the proposed MaOEA can be easily extended to MaOTSP, once the genetic operators are adapted to handle permutation encoding instead of binary. Next, the experimentation on symmetric, asymmetric, and mixed TSP instances, correlated objectives and objectives with different scales reveal that the proposed MaOEA always takes the lead among other tested algorithms. MaOTSP also provides a controlled environment for testing the case where near optimal solutions for the individual objectives are used to construct the hyper-

plane. Such cases may occur when exact solutions for single-objective problems are not available or computationally too expensive. Numerical experimentation reveals that the amount of deterioration in the quality of the Pareto approximation remains restricted, and that the use of a fixed hyperplane persists as the best methodology even when near optimal solutions are used to construct it. Moreover, recognizing the fact that negative coordinate values might be required to define some of the reference vectors in such cases, a version $FHCo^{-}$ is developed where genetic operators are modified to allow reference points with negative coordinates. Thus, $FHCo^{-}$ becomes the proposed MaOEA for cases when one chooses to use near optimal solutions for hyperplane construction.

Finally, the third pillar involves the research regarding the modeling of optimization problems under scenario-based uncertainty as MaOPs. MaOQAP is used as a benchmark, not only for corroborating the findings concerning the use of near optimal solutions when constructing the fixed hyperplane, but also for demonstrating optimization under scenario-based uncertainty as a new and natural problem domain for MaOEAs. The proposed MaOEA can provide good solutions for several robust and stochastic performance metrics simultaneously.

Future work can be directed to several fruitful paths. One of these research directions is incorporating decision maker's preferences into the algorithm for an interactive approach. Since the great potential of co-evolving the reference set together with the solution set has been demonstrated, it is worthwhile to pursue this direction for possible new approaches. Optimization under scenario–based uncertainty is opened as new application area for MaOEA, which can lead to several interesting research questions including but not limited to scenario reduction, robustness measure based guidance strategies and similar. Finally, other combinatorial optimization problems with different properties and characteristics may be tested using the proposed method.

REFERENCES

Abdelaziz, F. B., S. Krichen, and J. Chaouachi, 1999, "A hybrid heuristic for multiobjective knapsack problems", in *Meta-heuristics*, pp. 205–212, Springer, New York.

Aguirre, H., and K. Tanaka, 2009, "Many-objective optimization by space partitioning and adaptive ε-ranking on mnk-landscapes", in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 407–422.

Ahuja, R. K., J. B. Orlin, and A. Tiwari, 2000, "A greedy genetic algorithm for the quadratic assignment problem", *Computers & Operations Research*, Vol. 27(10), pp. 917–934.

Aissi, H., C. Bazgan, and D. Vanderpooten, 2009, "Min–max and min–max regret versions of combinatorial optimization problems: A survey", *European Journal of Operational Research*, Vol. 197(2), pp. 427–438.

Alsheddy, A., and E. E. K. Tsang, 2010, "Guided pareto local search based frameworks for biobjective optimization", in *IEEE Congress on Evolutionary Computation*, pp. 1–8.

Alves, M. J., and M. Almeida, 2007, "Motga: A multiobjective tchebycheff based genetic algorithm for the multidimensional knapsack problem", *Computers & Operations Research*, Vol. 34(11), pp. 3458–3470.

Antonio, L. M., and C. A. C. Coello, 2016, "Indicator-based cooperative coevolution for multi-objective optimization", in *IEEE Congress on Evolutionary Computation*, pp. 991–998.

Antonio, L. M., and C. A. C. Coello, 2017, "Coevolutionary multiobjective evolutionary algorithms: Survey of the state-of-the-art", *IEEE Transactions on Evolutionary Computation*, Vol. 22(6), pp. 851–865.

Applegate, D., R. Bixby, V. Chvatal, and W. Cook, 2006, *Concorde tsp solver*, http://www.tsp.gatech.edu/concorde, accessed in August 2020.

Asafuddoula, M., T. Ray, and R. Sarker, 2014, "A decomposition based evolutionary algorithm for many objective optimization", *IEEE Transactions on Evolutionary Computation*, Vol. 19(3), pp. 445–460.

Asafuddoula, M., H. K. Singh, and T. Ray, 2017, "An enhanced decomposition-based evolutionary algorithm with adaptive reference vectors", *IEEE Transactions on Cybernetics*, Vol. 48(8), pp. 2321–2334.

Azad, M. A. K., A. M. A. Rocha, and E. M. Fernandes, 2014, "Improved binary artificial fish swarm algorithm for the 0–1 multidimensional knapsack problems", *Swarm and Evolutionary Computation*, Vol. 14, pp. 66–75.

Bader, J., and E. Zitzler, 2011, "Hype: An algorithm for fast hypervolume-based many-objective optimization", *Evolutionary Computation*, Vol. 19(1), pp. 45–76.

Bandyopadhyay, S., and A. Mukherjee, 2014, "An algorithm for many-objective optimization with reduced objective computations: A study in differential evolution", *IEEE Transactions on Evolutionary Computation*, Vol. 19(3), pp. 400-413.

Bazgan, C., H. Hugot, and D. Vanderpooten, 2009a, "Implementing an efficient fptas for the 0–1 multi-objective knapsack problem", *European Journal of Operational Research*, Vol. 198(1), pp. 47–56.

Bazgan, C., H. Hugot, and D. Vanderpooten, 2009b, "Solving efficiently the 0–1 multi-objective knapsack problem", *Computers & Operations Research*, Vol. 36(1), pp. 260–279.

Beausoleil, R. P., G. Baldoquin, and R. A. Montejo, 2008, "Multi-start and path relinking methods to deal with multiobjective knapsack problems", *Annals of Operations Research*, Vol. 157(1), pp. 105–133.

Branke, J., J. Branke, K. Deb, K. Miettinen, and R. Slowiński, 2008, *Multiobjective Optimization: Interactive and Evolutionary Approaches*, Vol. 5252, Springer Science & Business Media, Berlin.

Brockhoff, D., and E. Zitzler, 2006, "Are all objectives necessary? on dimensionality reduction in evolutionary multiobjective optimization", in *Parallel Problem Solving from Nature-PPSN IX*, pp. 533–542, Springer, New York.

Brockhoff, D., and E. Zitzler, 2007, "Improving hypervolume-based multiobjective evolutionary algorithms by using objective reduction methods", in *IEEE Congress on Evolutionary Computation*, pp. 2086–2093.

Burkard, R. E., S. E. Karisch, and F. Rendl, 1997, "Qaplib–a quadratic assignment problem library", *Journal of Global Optimization*, Vol. 10(4), pp. 391–403.

Captivo, M. E., J. Clımaco, J. Figueira, E. Martins, and J. L. Santos, 2003, "Solving bicriteria 0–1 knapsack problems using a labeling algorithm", *Computers & Operations Research*, Vol. 30(12), pp. 1865–1886.

Chand, S., and M. Wagner, 2015, "Evolutionary many-objective optimization: A quickstart guide", *Surveys in Operations Research and Management Science*, Vol. 20(2), pp. 35–42.

Charnes, A., and W. W. Cooper, 1957, "Management models and industrial applications of linear programming", *Management Science*, Vol. 4(1), pp. 38–91.

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

Cheng, R., Y. Jin, M. Olhofer, and B. Sendhoff, 2016, "A reference vector guided evolutionary algorithm for many-objective optimization.", *IEEE Transactions on Evolutionary Computation*, Vol. 20(5), pp. 773–791.

Cirasella, J., D. S. Johnson, L. A. McGeoch, and W. Zhang, 2001, "The asymmetric traveling salesman problem: Algorithms, instance generators, and tests", in *Workshop on Algorithm Engineering and Experimentation*, pp. 32–59.

Coello, C. A., 2000, "An updated survey of ga-based multiobjective optimization techniques", *ACM Computing Surveys (CSUR)*, Vol. 32(2), pp. 109–143.

Coello, C. A. C., 2002, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art", *Computer Methods in Applied Mechanics and Engineering*, Vol. 191(11), pp. 1245–1287.

Coello, C. A. C., G. B. Lamont, D. A. Van Veldhuizen, *et al.*, 2007, *Evolutionary Algorithms for Solving Multi-objective Problems*, Vol. 5, Springer, New York.

Coello, C. C., and M. R. Sierra, 2003, "A coevolutionary multi-objective evolutionary algorithm", in *IEEE Congress on Evolutionary Computation*, Vol. 1, pp. 482–489.

Corne, D. W., and J. D. Knowles, 2007, "Techniques for highly multiobjective optimisation: some nondominated points are better than others", in *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, pp. 773–780.

Cvetković, D., and I. C. Parmee, 2002, "Preferences and their application in evolutionary multiobjective optimization", *IEEE Transactions on Evolutionary Computation*, Vol. 6(1), pp. 42–57.

Czyżżak, P., and A. Jaszkiewicz, 1998, "Pareto simulated annealing–a metaheuristic technique for multiple-objective combinatorial optimization", *Journal of Multi-Criteria Decision Analysis*, Vol. 7(1), pp. 34–47.

da Silva, C. G., J. Clímaco, and J. Figueira, 2006, "A scatter search method for bi-criteria {0, 1}-knapsack problems", *European Journal of Operational Research*, Vol. 169(2), pp. 373–391.

Das, I., and J. E. Dennis, 1998, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems", *SIAM Journal on Optimization*, Vol. 8(3), pp. 631–657.

Day, R. O., and G. B. Lamont, 2005, "Multiobjective quadratic assignment problem solved by an explicit building block search algorithm–momga-iia", in *European Conference on Evolutionary Computation in Combinatorial Optimization*, pp. 91–100.

Deb, K., 2001, *Multi-objective Optimization Using Evolutionary Algorithms*, Vol. 16, John Wiley & Sons, New Jersey.

Deb, K., and D. E. Goldberg, 1989, "An investigation of niche and species formation in genetic function optimization", in *Proceedings of the Third International Conference on Genetic algorithms*, pp. 42–50.

Deb, K., and H. Jain, 2013, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints", *IEEE Transactions on Evolutionary Computation*, Vol. 18(4), pp. 577–601.

Deb, K., and K. Miettinen, 2010, "Nadir point estimation using evolutionary approaches: better accuracy and computational speed through focused search", in *Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems*, pp. 339–354, Springer, New York.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan, 2002, "A fast and elitist multiobjective genetic algorithm: Nsga-ii", *IEEE Transactions on Evolutionary Computation*, Vol. 6(2), pp. 182–197.

Deb, K., and D. Saxena, 2006, "Searching for pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems", in *Proceedings of the World Congress on Computational Intelligence*, pp. 3352–3360.

Deb, K., and D. K. Saxena, 2005, "On finding pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems", *Kangal report*, Vol. 2005011.

Deb, K., and J. Sundar, 2006, "Reference point based multi-objective optimization using evolutionary algorithms", in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pp. 635–642.

Deb, K., L. Thiele, M. Laumanns, and E. Zitzler, 2005, "Scalable test problems for evolutionary multiobjective optimization", in *Evolutionary Multiobjective Optimization*, pp. 105–145, Springer, New York.

Delort, C., and O. Spanjaard, 2010, "Using bound sets in multiobjective optimization: Application to the biobjective binary knapsack problem", in "International Symposium on Experimental Algorithms", Springer, pp. 253–265.

Ehrgott, M., and X. Gandibleux, 2000, "A survey and annotated bibliography of multiobjective combinatorial optimization", *OR-Spektrum*, Vol. 22(4), pp. 425–460.

Ehrgott, M., and X. Gandibleux, 2003, "Multiobjective combinatorial optimization—theory, methodology, and applications", in *Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys*, pp. 369–444, Springer, New York.

Ehrgott, M., and X. Gandibleux, 2004, "Approximative solution methods for multiobjective combinatorial optimization", *Top*, Vol. 12(1), pp. 1–63.

Ehrgott, M., and X. Gandibleux, 2007, "Bound sets for biobjective combinatorial optimization problems", *Computers & Operations Research*, Vol. 34(9), pp. 2674–2694.

Erlebach, T., H. Kellerer, and U. Pferschy, 2002, "Approximating multiobjective knapsack problems", *Management Science*, Vol. 48(12), pp. 1603–1612.

Figueira, J. R., A. Liefooghe, E.-G. Talbi, and A. P. Wierzbicki, 2010, "A parallel multiple reference point approach for multi-objective optimization", *European Journal of Operational Research*, Vol. 205(2), pp. 390–400.

Fishburn, P. C., 1974, "Exceptional paper—lexicographic orders, utilities and decision rules: A survey", *Management Science*, Vol. 20(11), pp. 1442–1471.

Fleming, P. J., R. C. Purshouse, and R. J. Lygoe, 2005, "Many-objective optimization: An engineering design perspective", in *International Conference on Evolutionary Multi-criterion Optimization*, pp. 14–32.

Florios, K., G. Mavrotas, and D. Diakoulaki, 2010, "Solving multiobjective, multiconstraint knapsack problems using mathematical programming and evolutionary algorithms", *European Journal of Operational Research*, Vol. 203(1), pp. 14–21.

Fonseca, C. M., and P. J. Fleming, 1998, "Multiobjective optimization and multiple constraint handling with evolutionary algorithms. i. a unified formulation", *IEEE*

Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, Vol. 28(1), pp. 26–37.

Gandibleux, X., and A. Freville, 2000, "Tabu search based procedure for solving the 0-1 multiobjective knapsack problem: the two objectives case", *Journal of Heuristics*, Vol. 6(3), pp. 361–383.

Gass, S., and T. Saaty, 1955, "The computational algorithm for the parametric objective function", *Naval Research Logistics Quarterly*, Vol. 2(1-2), pp. 39–45.

Geoffrion, A. M., J. S. Dyer, and A. Feinberg, 1972, "An interactive approach for multi-criterion optimization, with an application to the operation of an academic department", *Management Science*, Vol. 19(4), pp. 357–368.

Giagkiozis, I., R. C. Purshouse, and P. J. Fleming, 2013, "Towards understanding the cost of adaptation in decomposition-based optimization algorithms", in *IEEE International Conference on Systems, Man, and Cybernetics*, pp. 615–620.

Glover, F., and M. Laguna, 1998, "Tabu search", in *Handbook of Combinatorial Optimization*, pp. 2093–2229, Springer, New York.

Glover, F., M. Laguna, and R. Martí, 2000, "Fundamentals of scatter search and path relinking", *Control and Cybernetics*, Vol. 29(3), pp. 653–684.

Glover, F., M. Laguna, and R. Martí, 2004, "Scatter search and path relinking: Foundations and advanced designs", in *New Optimization Techniques in Engineering*, pp. 87–99, Springer, New York.

Goulart, F., and F. Campelo, 2016, "Preference-guided evolutionary algorithms for many-objective optimization", *Information Sciences*, Vol. 329, pp. 236–255.

Haimes, Y., L. Lasdon, and D. da Wismer, 1971, "On a bicriterion formation of the problems of integrated system identification and system optimization", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 1(3), pp. 296–297.

Hansen, M. P., and A. Jaszkiewicz, 1998, *Evaluating the quality of approximations to the non-dominated set*, IMM, Department of Mathematical Modelling, Technical University of Denmark.

Hua, Y., Y. Jin, and K. Hao, 2018, "A clustering-based adaptive evolutionary algorithm for multiobjective optimization with irregular pareto fronts", *IEEE Transactions on Cybernetics*, Vol. 49(7), pp. 2758–2770.

Huband, S., P. Hingston, L. Barone, and L. While, 2006, "A review of multiobjective test problems and a scalable test problem toolkit", *IEEE Transactions on Evolutionary Computation*, Vol. 10(5), pp. 477–506.

Iancu, D. A., and N. Trichakis, 2013, "Pareto efficiency in robust optimization", *Management Science*, Vol. 60(1), pp. 130–147.

Iorio, A. W., and X. Li, 2004, "A cooperative coevolutionary multiobjective algorithm using non-dominated sorting", in *Genetic and Evolutionary Computation Conference*, pp. 537–548.

Ishibuchi, H., N. Akedo, and Y. Nojima, 2013, "A study on the specification of a scalarizing function in moea/d for many-objective knapsack problems", in *International Conference on Learning and Intelligent Optimization*, pp. 231–246.

Ishibuchi, H., N. Akedo, and Y. Nojima, 2014a, "Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems", *IEEE Transactions on Evolutionary Computation*, Vol. 19(2), pp. 264–283.

Ishibuchi, H., N. Akedo, H. Ohyanagi, and Y. Nojima, 2011, "Behavior of emo algorithms on many-objective optimization problems with correlated objectives", in *IEEE Congress of Evolutionary Computation*, pp. 1465–1472.

Ishibuchi, H., L. He, and K. Shang, 2019, "Regular pareto front shape is not realistic", in *IEEE Congress on Evolutionary Computation*, pp. 2034–2041.

Ishibuchi, H., R. Imada, Y. Setoguchi, and Y. Nojima, 2017, "Reference point specification in hypervolume calculation for fair comparison and efficient search", in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 585–592.

Ishibuchi, H., Y. Tanigaki, H. Masuda, and Y. Nojima, 2014b, "Distance-based analysis of crossover operators for many-objective knapsack problems", in *International Conference on Parallel Problem Solving from Nature*, pp. 600–610.

Ishibuchi, H., N. Tsukamoto, and Y. Nojima, 2008, "Evolutionary many-objective optimization: A short review.", in *IEEE Congress on Evolutionary Computation*, pp. 2419–2426.

Ishibuchi, H., T. Yoshida, and T. Murata, 2003, "Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling", *IEEE Transactions on Evolutionary Computation*, Vol. 7(2), pp. 204–223.

Jaimes, A. L., and C. A. C. Coello, 2015, "Many-objective problems: challenges and methods", in *Springer Handbook of Computational Intelligence*, pp. 1033–1046, Springer, New York.

Jain, H., and K. Deb, 2013, "An improved adaptive approach for elitist nondominated sorting genetic algorithm for many-objective optimization", in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 307–321.

Jain, H., and K. Deb, 2014, "An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach.", *IEEE Transactions on Evolutionary Computation*, Vol. 18(4), pp. 602–622.

Jaszkiewicz, A., 2002a, "Genetic local search for multi-objective combinatorial optimization", *European Journal of Operational Research*, Vol. 137(1), pp. 50–71.

Jaszkiewicz, A., 2002b, "On the performance of multiple-objective genetic local search on the 0/1 knapsack problem-a comparative experiment", *IEEE Transactions on Evolutionary Computation*, Vol. 6(4), pp. 402–412.

Jaszkiewicz, A., 2018, "Many-objective pareto local search", *European Journal of Operational Research*, Vol. 271(3), pp. 1001–1013.

Jiang, X., M. Gong, H. Li, M. Zhang, and J. Li, 2017, "A two-phase multiobjective sparse unmixing approach for hyperspectral data", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 56(1), pp. 508–523.

Jones, D. F., S. K. Mirrazavi, and M. Tamiz, 2002, "Multi-objective meta-heuristics: An overview of the current state-of-the-art", *European Journal of Operational Research*, Vol. 137(1), pp. 1–9.

Jozefowiez, N., F. Semet, and E.-G. Talbi, 2008, "Multi-objective vehicle routing problems", *European Journal of Operational Research*, Vol. 189(2), pp. 293–309.

Justesen, P. D., and R. K. Ursem, 2010, "Many-objective distinct candidates optimization using differential evolution on centrifugal pump design problems", in *IEEE Congress on Evolutionary Computation*, pp. 1–8.

Keeney, R. L., and H. Raiffa, 1976, *Decision Analysis with Multiple Conflicting Objectives*, John Wiley & Sons, New Jersey.

Kirlik, G., and S. Sayın, 2014, "A new algorithm for generating all nondominated solutions of multiobjective discrete optimization problems", *European Journal of Operational Research*, Vol. 232(3), pp. 479–488.

Kirlik, G., and S. Sayın, 2015, "Computing the nadir point for multiobjective discrete optimization problems", *Journal of Global Optimization*, Vol. 62(1), pp. 79–99.

Klamroth, K., and M. M. Wiecek, 2000, "Dynamic programming approaches to the multiple criteria knapsack problem", *Naval Research Logistics*, Vol. 47(1), pp. 57–76.

Kleeman, M. P., R. O. Day, and G. B. Lamont, 2004, "Analysis of a parallel moea solving the multi-objective quadratic assignment problem", in *Genetic and Evolutionary Computation Conference*, pp. 402–403.

Knowles, J., and D. Corne, 2002a, "On metrics for comparing nondominated sets", in *IEEE Proceedings of the Congress on Evolutionary Computation*, Vol. 1, pp. 711–716.

Knowles, J., and D. Corne, 2003, "Instance generators and test suites for the multiobjective quadratic assignment problem", in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 295–310.

Knowles, J. D., and D. Corne, 2002b, "Towards landscape analyses to inform the design of hybrid local search for the multiobjective quadratic assignment problem.", *HIS*, Vol. 87, pp. 271–279.

Knowles, J. D., and D. W. Corne, 2000, "M-paes: A memetic algorithm for multiobjective optimization", in *IEEE Proceedings of the Congress on Evolutionary Computation*, Vol. 1, pp. 325–332.

Konak, A., D. W. Coit, and A. E. Smith, 2006, "Multi-objective optimization using genetic algorithms: A tutorial", *Reliability Engineering & System Safety*, Vol. 91(9), pp. 992–1007.

Korte, B., J. Vygen, B. Korte, and J. Vygen, 2012, *Combinatorial Optimization*, Vol. 2, Springer, New York.

Kouvelis, P., and G. Yu, 2013, *Robust Discrete Optimization and Its Applications*, Vol. 14, Springer Science & Business Media, Berlin.

Larichev, O., 1992, "Cognitive validity in design of decision-aiding techniques", *Journal of Multi-Criteria Decision Analysis*, Vol. 1(3), pp. 127–138.

Laumanns, M., L. Thiele, K. Deb, and E. Zitzler, 2002, "Combining convergence and diversity in evolutionary multiobjective optimization", *Evolutionary Computation*, Vol. 10(3), pp. 263–282.

Laumanns, M., L. Thiele, and E. Zitzler, 2006, "An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method", *European Journal of Operational Research*, Vol. 169(3), pp. 932–942.

Li, B., J. Li, K. Tang, and X. Yao, 2015, "Many-objective evolutionary algorithms: A survey", *ACM Computing Surveys (CSUR)*, Vol. 48(1), p. 13.

Liu, Q., Y. Jin, M. Heiderich, and T. Rodemann, 2019, "Adaptation of reference vectors for evolutionary many-objective optimization of problems with irregular pareto fronts", in *IEEE Congress on Evolutionary Computation*, pp. 1726–1733.

Liu, X.-F., Z.-H. Zhan, Y. Gao, J. Zhang, S. Kwong, and J. Zhang, 2018, "Coevolutionary particle swarm optimization with bottleneck objective learning strategy for many-objective optimization", *IEEE Transactions on Evolutionary Computation*, Vol. 23(4), pp. 587–602.

Liu, Y., D. Gong, X. Sun, and Y. Zhang, 2017, "Many-objective evolutionary optimization based on reference points", *Applied Soft Computing*, Vol. 50, pp. 344–355.

Liu, Z.-Z., Y. Wang, and P.-Q. Huang, 2020, "And: A many-objective evolutionary algorithm with angle-based selection and shift-based density estimation", *Information Sciences*, Vol. 509, pp. 400–419.

Loiola, E. M., N. M. M. de Abreu, P. O. Boaventura-Netto, P. Hahn, and T. Querido, 2007, "A survey for the quadratic assignment problem", *European Journal of Operational Research*, Vol. 176(2), pp. 657–690.

López-Ibánez, M., L. Paquete, and T. Stützle, 2004, "On the design of aco for the biobjective quadratic assignment problem", in *International Workshop on Ant Colony Optimization and Swarm Intelligence*, pp. 214–225.

López-Jaimes, A., and C. A. C. Coello, 2014, "Including preferences into a multiobjective evolutionary algorithm to deal with many-objective engineering optimization problems", *Information Sciences*, Vol. 277, pp. 1–20.

López-Jaimes, A., A. Oyama, and K. Fujii, 2013, "Space trajectory design: Analysis of a real-world many-objective optimization problem", in *IEEE Congress on Evolutionary Computation*, pp. 2809–2816.

Lu, H., and G. G. Yen, 2002, "Rank-density based multiobjective genetic algorithm", in *IEEE Proceedings of the Congress on Evolutionary Computation*, Vol. 1, pp. 944–949.

Lust, T., and J. Teghem, 2008, "Memots: a memetic algorithm integrating tabu search for combinatorial multiobjective optimization", *RAIRO-Operations Research*, Vol. 42(1), pp. 3–33.

Lust, T., and J. Teghem, 2010a, "The multiobjective traveling salesman problem: a survey and a new approach", in *Advances in Multi-Objective Nature Inspired Computing*, pp. 119–141, Springer, New York.

Lust, T., and J. Teghem, 2010b, "Two-phase pareto local search for the biobjective traveling salesman problem", *Journal of Heuristics*, Vol. 16(3), pp. 475–510.

Ma, X., X. Li, Q. Zhang, K. Tang, Z. Liang, W. Xie, and Z. Zhu, 2018, "A survey on cooperative co-evolutionary algorithms", *IEEE Transactions on Evolutionary Computation*, Vol. 23(3), pp. 421–441.

Marler, R. T., and J. S. Arora, 2004, "Survey of multi-objective optimization methods for engineering", *Structural and Multidisciplinary Optimization*, Vol. 26(6), pp. 369–395.

Miettinen, K., and M. M. Mäkelä, 2002, "On scalarizing functions in multiobjective optimization", *OR Spectrum*, Vol. 24(2), pp. 193–213.

Mkaouer, M. W., M. Kessentini, S. Bechikh, K. Deb, and M. Ó Cinnéide, 2014, "High dimensional search-based software engineering: Finding tradeoffs among 15 objectives for automating software refactoring using nsga-iii", in *Proceedings of the Conference on Genetic and Evolutionary Computation*, pp. 1263–1270.

Mohammadi, A., M. N. Omidvar, and X. Li, 2012, "Reference point based multi-objective optimization through decomposition", in *IEEE Congress on Evolutionary Computation*, pp. 1–8.

Mohammadi, A., M. N. Omidvar, X. Li, and K. Deb, 2014, "Integrating user preferences and decomposition methods for many-objective optimization", in *IEEE Congress on Evolutionary Computation*, pp. 421–428.

Mulvey, J. M., R. J. Vanderbei, and S. A. Zenios, 1995, "Robust optimization of large-scale systems", *Operations Research*, Vol. 43(2), pp. 264–281.

Murata, T., and A. Taki, 2009, "Many-objective optimization for knapsack problems using correlation-based weighted sum approach", in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 468–480.

Narukawa, K., and T. Rodemann, 2012, "Examining the performance of evolutionary many-objective optimization algorithms on a real-world application", in *IEEE Sixth International Conference on Genetic and Evolutionary Computing*, pp. 316–319.

Nguyen, S., M. Zhang, M. Johnston, and K. C. Tan, 2013, "Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative

coevolution genetic programming", *IEEE Transactions on Evolutionary Computation*, Vol. 18(2), pp. 193–208.

Okabe, T., Y. Jin, and B. Sendhoff, 2003, "A critical survey of performance indices for multi-objective optimisation", in *IEEE Congress on Evolutionary Computation*, Vol. 2, pp. 878–885.

Payne, G., and R. R. Carlson, 1975, "Gaussian expansions in distorted-wave—born-approximation calculations", *Physical Review C*, Vol. 12(4), p. 1260.

Peng, W., Q. Zhang, and H. Li, 2009, "Comparison between moea/d and nsga-ii on the multi-objective travelling salesman problem", in *Multi-objective Memetic Algorithms*, pp. 309–324, Springer, New York.

Polak, E., 1976, "On the approximation of solutions to multiple criteria decision making problems", in *Multiple Criteria Decision Making*, pp. 271–282, Springer, New York.

Praditwong, K., and X. Yao, 2006, "A new multi-objective evolutionary optimisation algorithm: The two-archive algorithm", in *IEEE International Conference on Computational Intelligence and Security*, Vol. 1, pp. 286–291.

Purshouse, R. C., and P. J. Fleming, 2003, "Evolutionary many-objective optimisation: An exploratory analysis", in *IEEE Congress on Evolutionary Computation*, Vol. 3, pp. 2066–2073.

Qu, B. Y., and P. N. Suganthan, 2011, "Constrained multi-objective optimization algorithm with an ensemble of constraint handling methods", *Engineering Optimization*, Vol. 43(4), pp. 403–416.

Reinelt, G., 1991, "Tsplib-a traveling salesman problem library", *ORSA Journal on Computing*, Vol. 3(4), pp. 376–384.

Ribeiro, C. C., and M. G. Resende, 2012, "Path-relinking intensification methods for stochastic local search algorithms", *Journal of Heuristics*, Vol. 18(2), pp. 193–214.

Rommelfanger, H., and R. Słowiński, 1998, "Fuzzy linear programming with single or multiple objective functions", in *Fuzzy Sets in Decision Analysis, Operations Research and Statistics*, pp. 179–213, Springer, New York.

Ruzika, S., and M. M. Wiecek, 2005 "Approximation methods in multiobjective programming", *Journal of Optimization Theory and Applications*, Vol. 126(3), pp. 473–501.

Şahinkoç, H. M., 2014, *Facility layout problem under uncertainty*, Master Dissertation, Boğaziçi University.

Sato, H., 2015, "Analysis of inverted pbi and comparison with other scalarizing functions in decomposition based moeas", *Journal of Heuristics*, Vol. 21(6), pp. 819–849.

Sato, H., H. Aguirre, and K. Tanaka, 2013, "Variable space diversity, crossover and mutation in moea solving many-objective knapsack problems", *Annals of Mathematics and Artificial Intelligence*, Vol. 68(4), pp. 197–224.

Sato, H., H. E. Aguirre, and K. Tanaka, 2004, "Local dominance using polar coordinates to enhance multiobjective evolutionary algorithms", in *IEEE Congress on Evolutionary Computation*, Vol. 1, pp. 188–195.

Sato, H., H. E. Aguirre, and K. Tanaka, 2007, "Controlling dominance area of solutions and its impact on the performance of moeas", in *International Conference on Evolutionary Multi-criterion Optimization*, pp. 5–20.

Sato, H., H. E. Aguirre, and K. Tanaka, 2010, "Pareto partial dominance moea and hybrid archiving strategy included cdas in many-objective optimization", in *IEEE Congress on Evolutionary Computation*, pp. 1–8.

Sayın, S., 2000, "Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming", *Mathematical Programming*, Vol. 87(3), pp. 543–560.

Sindhya, K., A. B. Ruiz, and K. Miettinen, 2011, "A preference based interactive evolutionary algorithm for multi-objective optimization: Pie", in *Evolutionary Multi-Criterion Optimization*, pp. 212–225.

Sinha, A., D. K. Saxena, K. Deb, and A. Tiwari, 2013, "Using objective reduction and interactive procedure to handle many-objective optimization problems", *Applied Soft Computing*, Vol. 13(1), pp. 415–427.

Snyder, L. V., 2006, "Facility location under uncertainty: a review", *IIE transactions*, Vol. 38(7), pp. 547–564.

Snyder, L. V., and M. S. Daskin, 2006, "Stochastic p-robust location problems", *IIE Transactions*, Vol. 38(11), pp. 971–985.

Sülflow, A., N. Drechsler, and R. Drechsler, 2007, "Robust multi-objective optimization in high dimensional spaces", in *International Conference on Evolutionary Multi-criterion Optimization*, pp. 715–726.

Tan, K. C., T. H. Lee, and E. F. Khor, 2002, "Evolutionary algorithms for multi-objective optimization: Performance assessments and comparisons", *Artificial Intelligence Review*, Vol. 17(4), pp. 251–290.

Tan, K. C., Y. Yang, and C. K. Goh, 2006, "A distributed cooperative coevolutionary algorithm for multiobjective optimization", *IEEE Transactions on Evolutionary Computation*, Vol. 10(5), pp. 527–549.

Tanigaki, Y., K. Narukawa, Y. Nojima, and H. Ishibuch, 2014, "Preference-based nsga-ii for many-objective knapsack problems", in *Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and 15th International Symposium on Advanced Intelligent Systems (ISIS)*, pp. 637–642.

Thiele, L., K. Miettinen, P. J. Korhonen, and J. Molina, 2009, "A preference-based evolutionary algorithm for multi-objective optimization", *Evolutionary Computation*, Vol. 17(3), pp. 411–436.

Tian, Y., R. Cheng, X. Zhang, F. Cheng, and Y. Jin, 2017, "An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility", *IEEE Transactions on Evolutionary Computation*, Vol. 22(4), pp. 609–622.

Ulungu, E., and J. Teghem, 1994, "Multi-objective combinatorial optimization problems: A survey", *Journal of Multi-Criteria Decision Analysis*, Vol. 3(2), pp. 83–104.

Ulungu, E. L., and J. Teghem, 1995, "The two phases method: An efficient procedure to solve bi-objective combinatorial optimization problems", *Foundations of Computing and Decision Sciences*, Vol. 20(2), pp. 149–165.

Van Veldhuizen, D. A., 1999, *Multiobjective evolutionary algorithms: classifications, analyses, and new innovations*, Tech. rep., DTIC Document.

Vianna, D. S., and J. E. C. Arroyo, 2004, "A grasp algorithm for the multi-objective knapsack problem", in *24th International Conference of the Chilean Computer Science Society*, pp. 69–75.

Visée, M., J. Teghem, M. Pirlot, and E. L. Ulungu, 1998, "Two-phases method and branch and bound procedures to solve the bi–objective knapsack problem", *Journal of Global Optimization*, Vol. 12(2), pp. 139–155.

Von Lücken, C., B. Barán, and C. Brizuela, 2014, "A survey on multi-objective evolutionary algorithms for many-objective problems", *Computational Optimization and Applications*, Vol. 58(3), pp. 707–756.

Wagner, T., N. Beume, and B. Naujoks, 2007, "Pareto-, aggregation-, and indicator-based methods in many-objective optimization", in *International Conference on Evolutionary Multi-criterion Optimization*, pp. 742–756, Springer, New York.

Wang, H., L. Jiao, and X. Yao, 2014, "Two_arch2: An improved two-archive algorithm for many-objective optimization", *IEEE Transactions on Evolutionary Computation*, Vol. 19(4), pp. 524–541.

Wang, R., R. C. Purshouse, and P. J. Fleming, 2013, "Preference-inspired coevolutionary algorithms for many-objective optimization", *IEEE Transactions on Evolutionary Computation*, Vol. 17(4), pp. 474–494.

Wang, R., R. C. Purshouse, and P. J. Fleming, 2015, "Preference-inspired coevolutionary algorithms using weight vectors", *European Journal of Operational Research*, Vol. 243(2), pp. 423–441.

Wickramasinghe, U. K., R. Carrese, and X. Li, 2010, "Designing airfoils using a reference point based evolutionary many-objective particle swarm optimization algorithm", in "IEEE Congress on Evolutionary Computation (CEC)", pp. 1–8.

Wierzbicki, A. P., 1980, "The use of reference objectives in multiobjective optimization", in *Multiple Criteria Decision Making Theory and Application*, pp. 468–486, Springer, New York.

Woldesenbet, Y. G., G. G. Yen, and B. G. Tessema, 2009, "Constraint handling in multiobjective evolutionary optimization", *IEEE Transactions on Evolutionary Computation*, Vol. 13(3), pp. 514–525.

Wolsey, L. A., and G. L. Nemhauser, 1999, *Integer and Combinatorial Optimization*, Vol. 55, John Wiley & Sons, New Jersey.

Wu, L., M. Shahidehpour, and Z. Li, 2011, "Comparison of scenario-based and interval optimization approaches to stochastic scuc", *IEEE Transactions on Power Systems*, Vol. 27(2), pp. 913–921.

Zadeh, L., 1963, "Optimality and non-scalar-valued performance criteria", *IEEE Transactions on Automatic Control*, Vol. 8(1), pp. 59–60.

Zeleny, M., 1973, "Compromise programming, multiple criteria decision-making", in *Multiple Criteria Decision Making*, pp. 263–301, University of South Carolina Press, Columbia.

Zhan, Z.-H., J. Li, J. Cao, J. Zhang, H. S.-H. Chung, and Y.-H. Shi, 2013, "Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems", *IEEE Transactions on Cybernetics*, Vol. 43(2), pp. 445–463.

Zhang, Q., and H. Li, 2007, "Moea/d: A multiobjective evolutionary algorithm based on decomposition", *IEEE Transactions on Evolutionary Computation*, Vol. 11(6), pp. 712–731.

Zhang, Y.-H., Y.-J. Gong, T.-L. Gu, H.-Q. Yuan, W. Zhang, S. Kwong, and J. Zhang, 2017, "Decal: Decomposition-based coevolutionary algorithm for many-objective optimization", *IEEE Transactions on Cybernetics*, Vol. 49(1), pp. 27–41.

Zhao, W., S. Alam, and H. A. Abbass, 2014, "Mocca-ii: A multi-objective co-operative co-evolutionary algorithm", *Applied Soft Computing*, Vol. 23, pp. 407–416.

Zhou, C., G. Dai, C. Zhang, X. Li, and K. Ma, 2018, "Entropy based evolutionary algorithm with adaptive reference points for many-objective optimization problems", *Information Sciences*, Vol. 465, pp. 232–247.

Zionts, S., and J. Wallenius, 1976, "An interactive programming method for solving the multiple criteria problem", *Management Science*, Vol. 22(6), pp. 652–663.

Zitzler, E., K. Deb, and L. Thiele, 2000, "Comparison of multiobjective evolutionary algorithms: Empirical results", *Evolutionary Computation*, Vol. 8(2), pp. 173–195.

Zitzler, E., M. Laumanns, and S. Bleuler, 2004, "A tutorial on evolutionary multiobjective optimization", in *Metaheuristics for Multiobjective Optimisation*, pp. 3–37, Springer, New York.

Zitzler, E., M. Laumanns, and L. Thiele, 2001, "Spea2: Improving the strength pareto evolutionary algorithm", *TIK-report*, Vol. 103.

Zitzler, E., M. Laumanns, L. Thiele, C. M. Fonseca, and V. G. da Fonseca, 2002, "Why quality assessment of multiobjective optimizers is difficult", in *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation*, pp. 666–674.

Zitzler, E., and L. Thiele, 1999, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach", *IEEE Transactions on Evolutionary Computation*, Vol. 3(4), pp. 257–271.

Zitzler, E., L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, 2003, "Performance assessment of multiobjective optimizers: An analysis and review", *IEEE Transactions on Evolutionary Computation*, Vol. 7(2), pp. 117–132.

APPENDIX A: Preliminary Analysis on Performance Assessment Methods

Table A.1. Comparison of Pareto front approximation pairs.

QAP	$C(\mathcal{S}_1, \mathcal{S}_2)$		R_3&R_4		**UL**		M_3^*	
instance	**MOEA**	**MaOEA**	**MOEA**	**MaOEA**	**MOEA**	**MaOEA**	**MOEA**	**MaOEA**
scr15	6.6%	10.1%	1.5%	4.9%	1226	1012	187769	114688
had16	2.1%	21.4%	4.4%	4.5%	12	12	2056	1487
els19	0.1%	18.6%	6.1%	6.6%	0	7217	137826680	95992927
had20	1.4%	34.0%	2.1%	4.6%	11	14	3421	2702
nug20	1.8%	29.9%	1.6%	4.6%	16	22	2498	1770
scr20	0.5%	35.8%	4.0%	7.8%	1210	1487	458547	254744
nug25	1.2%	28.8%	2.0%	4.9%	18	0	3428	2468
kra30a	3.9%	14.4%	1.4%	1.6%	469	910	116861	113357
kra30b	0.2%	14.7%	1.5%	1.7%	346	774	120575	106835
nug30	4.9%	16.2%	1.1%	2.3%	25	44	5669	5298
tho30	2.3%	20.0%	1.0%	2.7%	738	1281	194773	181310
ste36a	2.5%	22.6%	1.9%	4.7%	44	180	31619	27066

Table A.2. Comparison with the true Pareto front.

QAP	$C1_R$		**IGD**		**CE**		**HR**		**UL**		M_3^*	
instance	**MOEA**	**MaOEA**	**MOEA**	**MaOEA**	**MOEA**	**MaOEA**	**MOEA**	**MaOEA**	**MOEA**	**MaOEA**	**MOEA**	**MaOEA**
had8	0.371	0.994	28.22	6.25	98.04	53.58	0.828	0.923	0	0	313	243
nug8	0.425	0.947	46.37	4.81	141.69	56.64	0.754	0.836	18	13	554	447
scr8	0.487	0.920	9466.74	420	30713	13766	0.756	0.884	1253	935	136616	111591
had10	0.368	0.876	41.71	2.94	130.65	71.33	0.723	0.814	14	12	465	357
nug10	0.466	0.913	62.72	8.06	193.43	99.91	0.744	0.793	0	0	789	583
scr10	0.481	0.866	9152.78	1054.63	29533.15	17671.24	0.698	0.816	1476	1203	143464	111601
had12	0.301	0.843	70.31	3.91	275.12	125.39	0.649	0.794	16	14	816	592
nug12	0.245	0.851	95.83	7.73	360.42	150.74	0.606	0.758	22	20	1171	739
scr12	0.358	0.837	11644.98	1250.66	47118.57	22783.39	0.689	0.780	1614	1254	155947	97945

Table A.3. Comparison using decomposition vectors and robust measures.

QAP	**Decomposition vectors**			**Minimax cost**		**Minimax abs. regret**		**Minimax rel. regret**	
instance	**MOEA**	**MaOEA**	**Improvement**	**MOEA**	**MaOEA**	**MOEA**	**MaOEA**	**MOEA**	**MaOEA**
scr15	22.2%	70.6%	1.47%	107888	106594	28384	27222	0.412	0.383
had16	36.1%	63.5%	0.36%	6096	5936	300	250	0.101	0.086
els19	31.7%	68.3%	1.78%	36095512	33567766	18286324	5175728	1.174	0.910
had20	48.8%	50.8%	0.08%	11126	11126	554	444	0.093	0.076
nug20	21.8%	71.8%	0.11%	4394	4388	532	430	0.177	0.155
scr20	35.3%	64.7%	0.18%	241932	240752	79076	64730	0.637	0.460
nug25	31.7%	57.9%	0.19%	6416	6354	720	688	0.170	0.165
kra30a	17.9%	69.8%	0.36%	177730	173140	29830	26370	0.224	0.213
kra30b	18.3%	63.9%	0.21%	176570	173120	30460	28380	0.231	0.206
nug30	32.9%	57.1%	0.01%	10746	10688	1252	1244	0.162	0.161
tho30	27.0%	57.5%	0.04%	246586	245284	49346	45328	0.291	0.275
ste36a	32.9%	63.5%	0.08%	21272	20346	6476	5732	0.559	0.493

APPENDIX B: Implementation Details for Algorithms Used in Comparison

This section outlines a brief discussion about the properties and important aspects of some existing state-of-the-art evolutionary algorithms used in numerical experimentation throughout this chapter. Each of these algorithms possesses different characteristics and has been shown to have a satisfactory performance.

Adaptive Approach for Elitist Nondominated Sorting Genetic Algorithm, A-NSGA-III (Jain and Deb, 2014). This algorithm is based on the original NSGA-III and everything but the reference point structure has been kept the same. The reference point structure of NSGA-III is revised and adapted so that the algorithm becomes capable of finding better appropriate distribution of the reference points. Reference points are adjusted depending on the distribution of current solutions in each generation.

The adaption method has two stages: addition of new reference points near crowded reference points with high niche counts and occasional deletion of reference points with zero niche counts. Since all reference points are defined on a hyperplane, these two operations essentially relocate the reference points based on the niche counts.

In the addition stage, a simplex is built around a crowded reference point and a maximum of m new reference points are added, which are believed to share the niche count with the crowded reference point. The addition of new reference points continues until no reference point with niche count greater than one is left. An example of a reference point addition structure is presented in Figure B.1.

After the addition stage is over, new reference points with zero niche counts are deleted, but the original reference points are not. It means that the number of reference points might increase during the course of evolution.

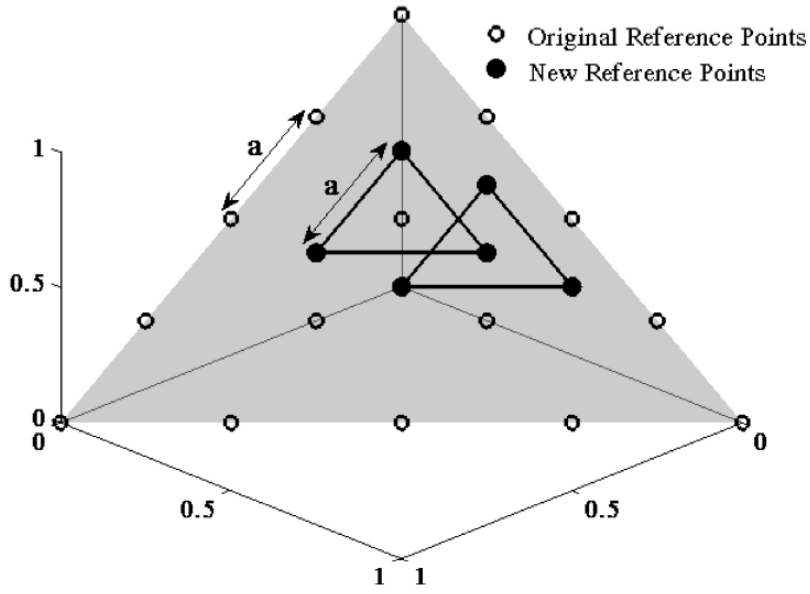

Figure B.1. Reference point structure in *A*-NSGA-III (Jain and Deb, 2014).

Improved Adaptive Approach for Elitist Nondominated Sorting Genetic Algorithm, A^2-NSGA-III (Jain and Deb, 2013). It is claimed that a more efficient approach is followed than *A*-NSGA-III. In this approach, the simplex structure results in addition of fewer new reference points, and the creation of new reference points is allowed only when the niche counts of crowded reference points remain constant for a predefined number (τ) of generations. An example of reference point addition structure is presented in Figure B.2.

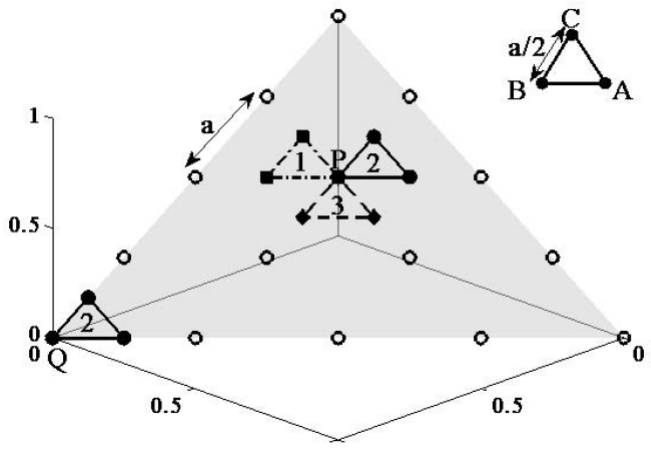

Figure B.2. Reference point structure in A^2-NSGA-III (Jain and Deb, 2013).

In this algorithm, there is a cap over the maximum number of reference points, and as a result, deletion process is triggered more often. In the implementation, the cap over the maximum number of reference point is 10 times the number of originally supplied reference points and the τ parameter is set to 10.

Improved Two Archive Algorithm, Two_Arch2 (Wang et al.*, 2014).* This algorithm, which is an example of reference set based approaches that use real reference points, separates the two goals of multi-objective optimization, namely convergence and divergence, explicitly. The solution set of each generation is divided into two groups: convergence archive (CA) and divergence archive (DA). The CA contains only nondominated solutions that once dominated existing archive members, and DA includes solutions with the greatest distances to CA members. Thus, solutions of the CA are adaptive real reference points, updated online.

An external population is maintained and reported for performance evaluation. The sizes of the CA and DA are kept equal to half the population sizes of other algorithms and equal. The binary additive epsilon-indicator ($I_{\epsilon+}$) (Zitzler *et al.*, 2003) is used as the selection principle for the CA, and the Pareto dominance-based selection principle is used for the DA. Moreover, lexicographic optimal solutions are provided at the beginning of the algorithm.

Reference Points Based Evolutionary Algorithm, RPEA (Liu et al.*, 2017).* Reference points are created as superior hypothetical solutions (i.e. virtual reference points) based on nondominated solutions with the largest crowding distances. For each objective, $\alpha \times N$ (population size) number of solutions with the largest crowding distances are found and reference points are generated based on these selected solutions. The distance between the generated reference point and the corresponding solution is calculated for each objective i as in Equation B.1:

$$\epsilon_i = \delta(f_i^{max} - f_i^{min}) \tag{B.1}$$

where f_i^{max} and f_i^{min} are the best and worst values observed for objective i in the current iteration, respectively. In the implementation, δ parameter is set to 0.05 and *alpha* parameter is set to $1/m$. Reference points are generated separately for each objective, so it is possible for a solution to produce multiple reference points.

Reference points are updated only in some iterations and the reference set regeneration frequency is determined by a predefined t_{grp} parameter which is set to one in the implementation, that is, the update procedure is triggered at each iteration. It is claimed that the reference points provide up-to-date information about the Pareto approximation at a moderate computation cost. At the end of each iteration, the combined population of the current and offspring population is truncated by calculating distance values with reference points. An example of reference point generation is presented in Figure B.3.

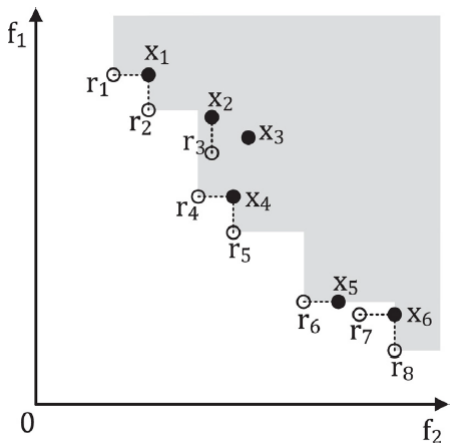

Figure B.3. Reference point generation in RPEA (Liu *et al.*, 2017).

Multi-Objective Evolutionary Algorithm Based on Decomposition, MOEA/D (Zhang and Li, 2007). MOEA/D is a decomposition-based evolutionary multi-objective algorithm. In MOEA/D, a multi-objective problem is decomposed into a number of single-objective problems identified using the same scalarizing function with different weight vectors. The number of weight vectors determines the population size because a single solution is kept for each single-objective problem defined by the corresponding weight vector.

The set of closest weight vectors of a given weight vector is defined as its neighborhood. The solution corresponding to each weight vector is generated by recombining a pair of parent solutions randomly selected from the neighborhood. The newly generated solution is compared with the existing solution of the corresponding weight vector and the existing solutions of its neighbors. It replaces those with a worse scalarizing function value in terms of the corresponding weight vector.

Different crossover schemes, repair, mutation and local improvement procedures can be adapted to MOEA/D, if no dominant solution is found in search, each solution is added to an external population. The decomposition approach used in MOEA/D is very important for the performance of the algorithm and different approaches may behave in differently depending on the characteristics of the problem. There are several decomposition approaches such as weighted sum, Tchebycheff and boundary intersection. The size of the neighborhoods of weight vectors is a user-defined parameter and must be carefully adjusted to avoid mating restriction.

MOEA/D has been shown to be computationally efficient and successful with a number of different problems (Ishibuchi *et al.*, 2014a). It is claimed that its decomposition-based structure handles many-objective problems better than approaches based on Pareto dominance. While this algorithm is categorized as a decomposition-based algorithm, there is a strong analogy between its weight vectors and the reference points.

In the implementation, the neighborhood structure of the decomposition weight vectors is determined to be the same with the reference point neighborhood structure of the proposed MaOEA. The weighted sum and Tchebycheff scalarizing functions are used, and the ideal point is used for the so-called reference point z^* mentioned on their paper.

Preference-Inspired Co-evolutionary Algorithm Using Weight Vectors, PICEA-w (Wang et al.*, 2015).* Existing population and offspring solutions are combined and truncation is applied to select the solutions of the next generation. The innovative part of this algorithm is that the application of the same procedure to decomposition weight

vectors. In each iteration, new weight vectors are generated and combined with the existing weight vector set. The combined set is then truncated based on information gathered from the current population.

It is a decomposition-based algorithm (i.e. the multi-objective problem is decomposed into a set of single-objective problems defined by different weighted scalarizing functions, Tchebycheff function in this case). In each iteration, weight vectors are generated randomly (Jaszkiewicz, 2002a).

A solution vector is said to be a neighbor of a weight vector if the angle between two vectors is less than a predefined θ parameter which is adjusted by current iteration number t and the maximum iteration number $maxGen$ in Equation B.2 to implement a local selection in the early stages of the algorithm, and a global selection in the late stages of the algorithm. In the implementation, $maxGen$ is set to 500.

$$\theta = \frac{\pi}{2} \times \frac{t}{maxGen} \tag{B.2}$$

The ranking matrix is produced by each candidate weighted scalarizing function to store the ranks of neighboring candidate solutions based on their performance. Candidate solutions are ordered lexicographically according to the ranking matrix. $(\mu + \lambda)$ elitist framework is used to select the solutions of the next generation. The selection of the weight vector is made using survived candidate solutions. In order for a weight to be selected, it must be one of the weights that a survived solution is best ranked. In case of multiple weights, it must be the furthest from the corresponding solution. Finally, an offline archive is used. A clustering technique called pruning archive (Zitzler *et al.*, 2001) is used to represent the Pareto approximation.