Eliminating Objective Functions and Warm Starting Algorithms Using Projections in Multi-objective Optimization

by

Gökhan Kof

A Thesis Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Industrial Engineering

KOÇ ÜNİVERSİTESİ

July 29, 2024

Eliminating Objective Functions and Warm Starting Algorithms Using Projections in Multi-objective Optimization

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Gökhan Kof

and have found that it is complete and satisfactory in all respects,

and that any and all revisions required by the final

examining committee have been made.

Committee Members:

Prof. Fikri Karaesmen (Advisor)

Prof. Serpil Sayın (Co-Advisor)

Asst. Prof. Erhun Özkan

Assoc. Prof. Firdevs Ulus

Assoc. Prof. Barış Yıldız

Date:

To my family

ABSTRACT

Eliminating Objective Functions and Warm Starting Algorithms Using Projections in Multi-objective Optimization
Gökhan Kof
Master of Science in Industrial Engineering
July 29, 2024

Many real world optimization problems have more than a single objective. In order to solve these multi-objective problems exactly, usually one has to solve many single objective problems to obtain all the efficient solutions and nondominated points. As the size of the nondominated set increases, so does the computational effort required to solve multi-objective problems. One way to mitigate this and lessen the computational burden is to reduce the number of objective functions in a given problem. When there are redundant objective functions, this reduction is obvious and the only requirement then is to detect the redundant objectives. Otherwise, any reduction will result in information loss in the form of missing some nondominated points. In other words, the set of points that will be obtained from the reduced problem is going to be a representation, which is a subset of the nondominated set of the original problem. In this thesis, we develop a projection based metric in order to determine which objective to remove with the goal of keeping the information loss to a minimum. We base our approach on the characterization of a redundant objective for the linear case. We then assess the level of redundancy for an objective using its similarity to its projection. The performance of this method is evaluated on many sets of test problems from the literature as well as on generated correlated instances and the information loss is quantified using quality metrics. Our method demonstrates strong performance across the majority of tested problems. Moreover, we show that the representation we obtain after solving the reduced problem can be used to warm start the exact solution process of the original multi-objective problem. This way parts of the search region that do not contain any nondominated points can be eliminated and the number of objects that are needed to be maintained throughout the solution process can be reduced. Our preliminary experiments indicate that our initialization method is highly effective in achieving these reductions.

ÖZETÇE

Çok Amaçlı Optimizasyon Problemlerinde Yansımalar Kullanılarak Amaç Fonksiyonlarının Eksiltilmesi ve Çözüm Algoritmalarının Hızlı Başlatılması

Gökhan Kof
Endüstri Mühendisliği, Yüksek Lisans
29 Temmuz 2024

Gerçek hayatta karşılaşılan birçok eniyileme problemi birden fazla amaca sahiptir. Bu çok amaçlı problemleri tam çözmek ve tüm etkin çözümleri ve baskın noktaları elde etmek için genellikle birçok tek amaçlı problemin çözülmesi gerekir. Baskın kümenin boyutu arttıkça, çözüm için gereken hesaplama yükü de artacaktır. Bu hesaplama yükünü azaltmanın yollarından biri, amaç fonksiyonlarının sayısını eksiltmektir. Problemin gereksiz amaç fonksiyonları içerdiği durumda, bu eksiltmenin nasıl yapılacağı açıktır ve yapılması gereken gereksiz amaç fonsksiyonları tespit etmektir. Aksi takdirde, herhangi bir eksiltme, baskın nokta kümesinin tam olarak elde edilememesi şeklinde bir bilgi kaybına yol açar ve eksiltilmiş problemden elde edilecek baskın küme, asıl problemin baskın nokta kümesinin bir alt kümesi olan temsili çözümler olacaktır. Bu tezde, bilgi kaybını en aza indirebilme amacıyla hangi amaç fonksiyonunun eksiltileceğini belirlemek üzere yansıma tabanlı bir ölçüt geliştirilmiştir. Yaklaşım, doğrusal problemler için gereksiz amaç fonksiyonlarının özelliklerine dayanmaktadır. Bir amaç fonksiyonunun gereksizliği yansımasına benzerliği ölçülerek değerlendirilmiştir. Ölçütün performansı test problemleri ve çalışma kapsamında üretilen korelasyonlu problemler üzerinde sınanmış ve elde edilen temsili çözümler kalite ölçütleri kullanılarak değerlendirilmiştir. Yaklaşımımızın birçok problem türünde iyi performans gösterdiği görülmüştür. Ayrıca, eksiltilmiş problemi çözerek elde ettiğimiz temsili çözümlerin bazı özellikleri gösterilmiş ve bu çözümler asıl çok amaçlı problemin tam çözüm sürecini hızlı başlatmak için kullanılmıştır. Burada amaç baskın noktaların bulunmadığı bölgeleri çözüm sürecine dahil etmemek ve problemlerin tam çözüm süreçlerinde takip edilmesi gereken nesne sayısını azaltmaktır. Öncül deney sonuçları yaklaşımımızın gerekli nesne sayısını azaltmakta etkili olduğunu göstermiştir.

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisors Fikri Karaesmen and Serpil Sayın for their guidance and support during this study.

I am grateful to Erhun Özkan, Firdevs Ulus and Barış Yıldız for taking part in my thesis committee and for their suggestions and comments.

This thesis is supported by TÜBİTAK (The Scientific and Technological Research Council of Türkiye), project no. 122M422.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

MOO	Multi-objective Optimization
MOLP	Multi-objective Linear Program
MOILP	Multi-objective Integer Linear Program
$f = (f_1, \ldots, f_p)$	vector of objective functions
$\mathcal{X}$	feasible region of an optimization problem
$\mathcal{Y} = f(\mathcal{X})$	objective space
$y^1 < y^2$	$y_k^1 < y_k^2$ for $y_1, y_2 \in \mathbb{R}^p$ and $k \in \{1, \ldots, p\}$
$y^1 \leqq y^2$	$y_k^1 \leq y_k^2$ for $y_1, y_2 \in \mathbb{R}^p$ and $k \in \{1, \ldots, p\}$
$y^1 \leq y^2$	$y^1 \leqq y^2$ and $y^1 \neq y^2$
$\mathbb{R}^p_>$	$\{y \in \mathbb{R}^p : y > 0\}$
$\mathbb{R}^p_\geq$	$\{y \in \mathbb{R}^p : y \geq 0\}$
$\mathbb{R}^p_\geqq$	$\{y \in \mathbb{R}^p : y \geqq 0\}$
y^I	ideal point
y^N	nadir point
y^U	an upper bound on the nadir point ($y^U \geq y^N$)
y_{-j}	$(y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_p)$
$\mathcal{Y}_N$	Nondominated Set
$\bar{\mathcal{Y}}^q_N$	Nondominated Set with f_q removed
$\mathcal{X}_E$	Efficient Set
$\bar{\mathcal{X}}^q_E$	Efficient Set with f_q removed

Chapter 1

INTRODUCTION

Mathematical programming models have proven to be very useful for decision making in several different domains. Different entities often rely on expressing their problem mathematically and use optimization algorithms to decide on what the best course of action is for their organization. Building these models require a deep understanding of the real world situation and expressing the boundaries the decision makers have to stay within through mathematical constraints. However, another description is also required in terms of the criteria that need to be optimized within those boundaries. Thus, it is easy to see that most such entities do not have a clear cut single goal and they often need to consider trade-offs between different targets they want to achieve. Examples can be the service level vs. the amount of inventory for a manufacturing plant, capital requirements vs. investment duration for a bank, number of tellers vs waiting time for a postal office. In other words, many real world problems have more than one objective that needs to be considered by the decision makers and the stakeholders. Most of the time, these objectives are in conflict with each other, i.e. in order to get better in one objective, the decision makers need to sacrifice from the other.

A feasible solution to a multi-objective optimization (MOO) problem is said to dominate another solution if the value of the objective functions are better or equal for all objectives and strictly better for at least one objective. If, for a feasible solution, there is no other solution that dominates it, then, it becomes an efficient solution and the value of the objective functions yields a nondominated point. The set of all nondominated points makes up the nondominated set or the Pareto frontier.

There are mainly three classes of solution approaches for MOO problems. The first is the a priori methods. These require the decision makers to express their preferences before the actual solution process. This preference can be stated in different ways such as a hierarchy amongst objectives or a set of weights to express the importance of each objective in the eyes of the decision makers. The second one is interactive methods. These are designed to find a number of efficient solutions at first. These solutions are then shared with the decision maker and based on the feedback received, new solutions are obtained. This process proceeds until the decision makers are satisfied. The last group of methods is called a posteriori methods. These aim at finding efficient solutions. The decision makers then decide on which solution they prefer to implement. One variant of a posteriori methods is the representation approach, where a representative subset of the set of efficient solutions is obtained.

The exact solution for a multi-objective problem refers to finding all the nondominated points and a collection of efficient solutions that yield the nondominated points. There are two main types of exact solution approaches. These are algorithms that work in the decision space and the ones that work in the solution space. The decision space algorithms search the feasible region of the MOO problem while the solution space algorithms search the objective space which is the mappings obtained by evaluating each feasible point for all objectives. The objective space algorithms reformulate the MOO problem as a single objective problem using some parameters in order to find efficient solutions. This requires systematically searching the solution space and solving many single objective reformulations by changing the value of the parameters.

If the decision variables are only allowed to take integer values, then the problem becomes a multi-objective integer programming (MOILP) problem. Then, an exact solution will require solving many single objective integer programming (IP) problems. IPs are well studied in the literature and they are most of the time $\mathcal{NP}$-hard problems. As a result, solving a multi-objective variant of an IP requires solving many IP problems, making the multi-objective variant even more difficult.

As more objective functions are introduced, MOO problems become harder to solve. This is because another conflicting objective provides more ways for the feasible points to be nondominated. Therefore, an increase in the number of nondominated points of an MOILP problem requires more IPs to be solved. Literature shows that MOILP problems require a lot of resources especially when we have more than three objectives. For this reason, it is worth looking for ways to reduce the number of objectives for an MOILP problem. However, when we try to solve a MOILP problem using a fewer number of objective functions, we will not be able to get all of the nondominated points if there is no redundancy in the original set of objective functions.

Since the number of nondominated points can be quite large for large problems, finding a subset of these solutions can also be preferred. The subsets are called representations. Therefore, when we try to solve a multi-objective problem using a fewer number of objective functions, we get a representation of the original problem. Fewer efficient solutions will help with the computational burden of the solution process at the expense of some information loss. This information loss is the difference between the representation and the set of nondominated points. There are quality metrics developed in the literature in order to quantify and better understand the level of the information loss.

In this thesis, we provide a projection based method to reduce the number of objective functions for a MOILP problem and analyze its performance via extensive computational experiments on library instances and on newly generated test problems. By treating the Pareto front of the reduced problem as a representation of the original one, we compute a number of quality metrics as measures of the information loss. In fact, our approach is applicable to a more general class of MOO problems than the ones with integer variables. Focusing on MOILP problems makes the evaluation process more straight-forward. Our experiments show that the performance of our method is strong for the majority of the problems tested.

In Chapter 2, we provide a short summary of the literature in this area. In Chapter 3, we describe our method of objective reduction. In Chapter 4, we share some computational results where we test the effectiveness of our approach. In Chapter 5, we show some of the properties of the subset of nondominated points obtained by solving the reduced problem, how they can be useful in describing the search space for the original problem and how we can use them to initialize the search region for the original problem. Finally, we state our conclusions and possible future work that can be considered.

Chapter 2

LITERATURE REVIEW

In this chapter, we will mention basic concepts and the notations associated with multi-objective optimization problems. An overview of solution strategies used in different types of multi-objective problems will be given. Later, the definition of representations of Pareto sets and different metrics used to assess their quality will be introduced. Then, since we are interested in eliminating objective functions, redundant and essential objectives will be discussed and a survey of research, studies and results that can be found in the literature will be provided. Lastly, we will take a overview of the search space representations used in different algorithms.

2.1 Multi-objective Optimization

Multi-objective optimization (MOO) problems are optimization problems with more than one objective. While comparing two feasible solutions for an MOO problem, the decision maker (DM) needs to consider the objective values for each objective. If a solution is better in every objective compared to another solution, the choice is clear and the DM is better off picking the former. However, the interesting part of MOO problems comes from the tradeoff the DM needs to evaluate. One solution may be better in one objective but worse in another objective. Therefore, it is often not enough to try and find a single solution to MOO problems. Here, we can define Pareto optimality and dominance for MOO problems. Consider the following MOO problem:

$$
\begin{aligned}
(\textbf{MOO}) \quad & \text{"minimize"} && f(x) \\
& \text{subject to} && x \in \mathcal{X}
\end{aligned}
\tag{2.1}
$$

where $f(x) = (f_1(x), \ldots, f_p(x))$, $f_j : \mathbb{R}^n \mapsto \mathbb{R}, j \in \{1, \ldots, p\}$ and $\mathcal{X} \subseteq \mathbb{R}^n$.

The problem has p objectives and the quotes around the term minimize is to refer to the tradeoff we need to make in deciding on what the "best" solution is. If we were able to minimize every objective at the same time by a common feasible solution, that would mean the problem is in essence a single objective problem.

Usually, the set of feasible solutions for an MOO problem is called the **decision space** while the set of objective values, $\mathcal{Y} = f(\mathcal{X})$, is called the **objective space**.

For simplicity, we redefine the comparison operators for vectors as follows: Let y_j denote the jth component of vector y and $y^1, y^2 \in \mathbb{R}^p$.

- $y^1 \leqq y^2 \iff y^1_j \leq y^2_j, \forall j \in \{1, \ldots, p\}$,
- $y^1 \leq y^2 \iff y^1 \leqq y^2$ and $y^1 \neq y^2$,
- $y^1 < y^2 \iff y^1_j < y^2_j, \forall j \in \{1, \ldots, p\}$.

Along with the above, we define different subsets of $\mathbb{R}^p$ as follows:

- $\mathbb{R}^p_{\geqq} = \{y \in \mathbb{R}^p : y \geqq 0\}$, nonnegative orthant of $\mathbb{R}^p$,
- $\mathbb{R}^p_{\geq} = \{y \in \mathbb{R}^p : y \geq 0\} = \mathbb{R}^p_{\geqq} \backslash \{0\}$,
- $\mathbb{R}^p_{>} = \{y \in \mathbb{R}^p : y > 0\}$, positive orthant of $\mathbb{R}^p$.

Additionally, given a vector $y \in \mathbb{R}^p$, $y_{-k} \in \mathbb{R}^{p-1}$ denotes the vector y with its kth component omitted: $y_{-k} = (y_1, \ldots, y_{k-1}, y_{k+1}, \ldots, y_p)$.

Definition 2.1.1 *$x \in \mathcal{X}$ is an **efficient solution** if $\nexists x' \in \mathcal{X}$ such that $f(x') \leq f(x)$. In this case, $y = f(x) = (f_1(x), \ldots, f_p(x))$ is called a **nondominated point**. The set of all efficient solutions, $\mathcal{X}_E$, is called the **efficient set** or **Pareto Set** and the set of all nondominated points, $\mathcal{Y}_N$, is called the **nondominated set** or **Pareto Front**.*

Definition 2.1.2 *$x \in \mathcal{X}$ is a **weakly efficient solution** if $\nexists x' \in \mathcal{X}$ such that $f(x') < f(x)$. In this case, $y = f(x) = (f_1(x), \ldots, f_p(x))$ is called a **weakly nondominated point**. The set of weakly efficient solutions and weakly nondominated points are denoted as $\mathcal{X}_{wE}$ and $\mathcal{Y}_{wN}$, respectively.*

Clearly, $\mathcal{X}_E \subseteq \mathcal{X}_{wE}$ and $\mathcal{Y}_N \subseteq \mathcal{Y}_{wN}$.

Definition 2.1.3 *For the problem in* (2.1)*, the point* $y^I \in \mathbb{R}^p$ *where* $y_j^I = \min_{x\in\mathcal{X}} f_j(x)$, $\forall j \in \{1,\dots,p\}$*, is called the* ***ideal point*** *while* $y^N \in \mathbb{R}^p$ *where* $y_j^N = \max_{x\in\mathcal{X}_E} f_j(x)$, $\forall j \in \{1,\dots,p\}$*, is called the* ***nadir point****.*

It is important to know that, in a truly multi-objective setting, the ideal point cannot be part of the nondominated set. If it is, then the problem boils down to a single objective problem since the ideal solution gives the best results in terms of every objective. Another important point is that while computing the ideal point, having $\mathcal{X}$ or $\mathcal{X}_E$ as the feasible set is equivalent since $\min_{x\in\mathcal{X}} f_j(x) = \min_{x\in\mathcal{X}_E} f_j(x), \forall j \in \{1,\dots,p\}$, whereas this is not the case for the nadir point. If the objectives are maximized over the feasible set instead of the efficient set, an upper bound on the nadir point will be computed. For a problem with two objectives, we can look at Figure 2.1 to better understand these concepts.

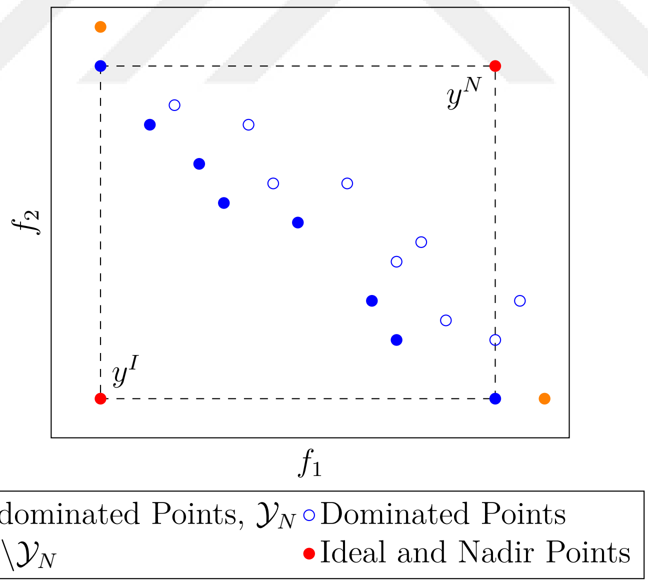


Figure 2.1: Nondominated, Ideal and Nadir Points for a 2-objective Problem

2.2 *Solutions of Multi-objective Optimization Problems*

The goal of multi-objective optimization is to assess the trade-offs between different solutions by computing the Pareto Front. This is usually done by solving a single objective problem with different parameters multiple times. These are called scalarizations. Here, we will briefly look at some scalarizations used in multi-objective optimization. To use these scalarizations to compute the entire Pareto Front, a systematic way of searching the parameter space is needed. This will be discussed later.

2.2.1 Weighted-sum Method

One way we can solve a multi-objective problem is to use a weight vector and combine all objectives into a single objective. Let $\lambda \in \mathbb{R}^p_{\geq}$ and consider the following formulation:

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{p} \lambda_j f_j(x) \\ \text{subject to} \quad & x \in \mathcal{X}. \end{aligned} \tag{2.2}$$

We can use the following theorems to generate nondominated solutions using different values of λ.

Theorem 2.2.1 ([Geoffrion, 1968]) *Let x^* be an optimal solution to problem* (2.2)*. Then, the following statements hold:*

- *if $\lambda \in \mathbb{R}^p_{\geq}$, $x^* \in \mathcal{X}_{wE}$,*
- *if $\lambda \in \mathbb{R}^p_{>}$, $x^* \in \mathcal{X}_{E}$.*

Theorem 2.2.2 ([Zadeh, 1963]) *If $\mathcal{X}$ is a convex set and $f_j : \mathbb{R}^n \mapsto \mathbb{R}, j \in \{1, \ldots, p\}$ are convex functions, then for an $x^* \in \mathcal{X}_E$, $\exists \lambda \in \mathbb{R}^p_{>}$ such that x^* is an optimal solution to Problem* (2.2)*.*

2.2.2 ε-Constraint Method

Another essential method used to find an efficient solution is the ε-constraint method. For this method, we replace the original multi-objective problem with the following problem:

$$\begin{aligned} \text{minimize} \quad & f_k(x) \\ \text{subject to} \quad & f_j(x) \leq \varepsilon_j \quad \forall j \in \{1, \ldots, p\} \quad j \neq k, \\ & x \in \mathcal{X}. \end{aligned} \tag{2.3}$$

Here, $k \in \{1, \ldots, p\}$ is any one the objective functions and $\varepsilon \in \mathbb{R}^{p-1}$.

Theorem 2.2.3 ([Haimes et al., 1971]) *If $x^* \in \mathcal{X}$ is the optimal solution of Problem* (2.3) *for some $\varepsilon \in \mathbb{R}^{p-1}$ and $k \in \{1, \ldots, p\}$, then x^* is a weakly efficient solution, $x^* \in \mathcal{X}_{wE}$. If it is the unique optimal solution, then x^* is an efficient solution, $x^* \in \mathcal{X}_E$.*

Theorem 2.2.4 ([Ehrgott, 2005]) *A feasible solution $\hat{x} \in \mathcal{X}$ is efficient if and only if there exists an $\varepsilon \in \mathbb{R}^{p-1}$ such that $\hat{x}$ is an optimal solution of Problem* (2.3), *$\forall k \in \{1, \ldots, p\}$.*

Since the optimal solution of Problem (2.3) may be weakly efficient, a modification is needed to obtain only efficient solutions. We can use an augmented formulation to avoid weakly efficient solutions by adding $\sum_{\substack{j=1 \\ j \neq k}}^{p} f_j(x)$ to the objective function of (2.3) after multiplying it with a small coefficient. Details of this approach can be found in [Mavrotas and Florios, 2013]. Alternatively, a two-stage formulation as in [Kirlik and Sayin, 2014] can be used:

$$\begin{aligned} P_k(\varepsilon) \quad z^* = \text{minimize} \quad & f_k(x) \\ \text{subject to} \quad & f_j(x) \leq \varepsilon_j, \quad \forall j \in \{1, \ldots, p\}, j \neq k \\ & x \in \mathcal{X}. \end{aligned} \tag{2.4}$$

$$\begin{aligned} Q_k(\varepsilon) \quad \text{minimize} \quad & \sum_{j=1}^{p} f_j(x) \\ \text{subject to} \quad & f_j(x) \leq \varepsilon_j, \quad \forall j \in \{1, \ldots, p\}, j \neq k \\ & f_k(x) = z^* \\ & x \in \mathcal{X}. \end{aligned} \tag{2.5}$$

Formulations given in (2.4) and (2.5) can be used to generate efficient solutions. By changing the value of ε systematically, we can obtain the entire set of nondominated points, $\mathcal{Y}_N$. Also, note that convexity is not required for the ε-constraint method to work.

2.2.3 Hybrid Method

It is also possible to combine both weighted sum and ε-constraint approaches in a single formulation. Consider the following mathematical model:

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{p} \lambda_j f_j(x) \\ \text{subject to} \quad & f_j(x) \leq \varepsilon_j \quad \forall j \in \{1,\dots,p\} \\ & x \in \mathcal{X}. \end{aligned} \tag{2.6}$$

Theorem 2.2.5 ([Guddat, 1985]) *A feasible solution $\hat{x} \in \mathcal{X}$ is an efficient solution if and only if it is the optimal solution to problem* (2.6) *for some $\lambda \in \mathbb{R}^p_>$ and $\varepsilon \in \mathbb{R}^{p-1}$.*

Similar to the ε-constraint method, the hybrid approach does not require any convexity assumptions.

2.2.4 Weighted Tchebycheff Method

The weighted Tchebycheff norm of a p-dimensional vector y is defined as $||y||^w_\infty = \max_{j \in 1,\dots,p} w_j |y_j|$. Now consider the following formulation:

$$\begin{aligned} \text{minimize} \quad & ||f(x) - y^I||^w_\infty \\ \text{subject to} \quad & x \in \mathcal{X}. \end{aligned} \tag{2.7}$$

The following results have been shown for Tchebycheff scalarization.

Theorem 2.2.6 ([Bowman, 1976]) *Let $\hat{x} \in \mathcal{X}$. $\hat{x} \in \mathcal{X}_E$ only if $\exists w \in \mathbb{R}^p_>$ such that $\hat{x}$ is an optimal solution of problem given in Problem* (2.7)*.*

Theorem 2.2.7 ([Bowman, 1976]) *Let $x^* \in \mathcal{X}$ be the unique optimal solution for Problem* (2.7) *for some $w \in \mathbb{R}^p_>$. Then, $x^* \in \mathcal{X}_E$.*

Using these theorems, we can change the value of w systematically to obtain all efficient solutions.

2.3 Multi-objective Discrete Optimization Problems

For the multi-objective discrete optimization (MODO) problems, we assume $\mathcal{X} \subseteq \mathbb{Z}^n$. Several algorithms are developed to solve these types of problems. Some studies focus on specific discrete problem types that have special structures such as shortest-path [Shi et al., 2017] and knapsack problems [Bazgan et al., 2009] with multiple objectives. There are also more generalized approaches to solving MODO problems. A group of these approaches works on the decision space using multi-objective branch-and-bound algorithms. A survey of these methods is provided in [Przybylski and Gandibleux, 2017]. Some of those algorithms can be found in [Klein and Hannan, 1982], [Kiziltan and Yucaoğlu, 1983], [Aguirre and Tanaka, 2009], [Belotti et al., 2013], [Forget et al., 2022]. There are also approaches that focus on the objective space. They start with an initial *search region* in the objective space and define smaller *search zones* to find new nondominated points. [Chalmet et al., 1986] provides an algorithm that works in the bi-objective case. Some of the more general algorithms can be found in [Boland et al., 2017a], [Ozlen and Azizoğlu, 2009], [Kirlik and Sayin, 2014], [Domínguez-Ríos et al., 2021], [Tamby and Vanderpooten, 2021]. In what follows, we will explain the two methods used in solving MODO problems, algorithmic components of which will be utilized later in the thesis.

2.3.1 *[Kirlik and Sayin, 2014]*

This is an algorithm that uses the ε-constraint method to generate solutions for multi-objective discrete optimization problems [Kirlik and Sayin, 2014]. We know from Section 2.2.2 that the solution to the two-stage model will be a nondominated point. In order to choose the ε values, the method considers the projections of nondominated points onto the $(p-1)$ dimensional space as depicted in Figure 2.2. [Kirlik and Sayin, 2014] uses regions they call rectangles in the projected space to manage the search region. The rectangles are defined as $\mathcal{B}^{p-1}(l,u) = \{\bar{y} \in \mathbb{R}^{p-1} | l \leqq$

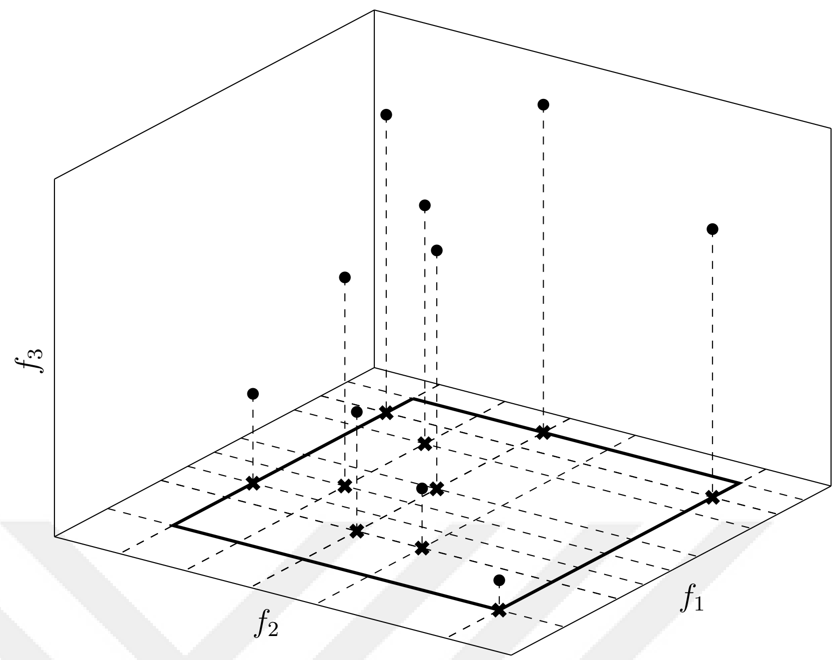


Figure 2.2: Nondominated points in $\mathbb{R}^3$ and their projections onto $f_1 - f_2$ plane

$\bar{y} \leqq u\}$ where $l, u \in \mathbb{R}^{p-1}$.

The algorithm maintains a list of such rectangles formed by the projections of nondominated points onto $(p-1)$-dimensional space. At any point in the algorithm, among the set of $(p-1)$-dimensional rectangles, the one that has the largest volume measure, defined below for $\mathcal{B}_j^{p-1}(l^j, u^j)$ as $V_j = \prod_{\substack{i=1 \\ i \neq k}}^{p} (u_i^j - \bar{y}_i^I)$, is chosen as the next rectangle. Then, a two-stage model described in (2.4) and (2.5) is solved to generate an efficient solution using $\mathcal{B}^{p-1}(l, u)$ by setting $\varepsilon_j = u_j, \forall j \in \{1, \ldots, p\}, j \neq k$ for some $k \in \{1, \ldots, p\}$.

Theorem 2.3.1 ([Kirlik and Sayin, 2014]) *For $\varepsilon \in \mathbb{R}^p$, an optimal solution to the two-stage formulations $P_k(\varepsilon)$ and $Q_k(\varepsilon)$ is efficient.*

Theorem 2.3.2 ([Kirlik and Sayin, 2014]) *For any efficient solution x^* of a multi-objective discrete optimization problem, there exists an $\varepsilon \in \mathbb{R}^{p-1}$ such that x^* is an optimal solution to two-stage formulations $P_k(\varepsilon)$ and $Q_k(\varepsilon)$ for some $k \in \{1, \ldots, p\}$.*

Theorem 2.3.1 and 2.3.2 guarantee the generation of efficient solutions.

Theorem 2.3.3 ([Kirlik and Sayin, 2014]) *Let x^* be an optimal solution of two-stage formulation associated with rectangle $\mathcal{B}_j^{p-1}(l^j, u^j)$, $P_k(u^j)$ and $Q_k(u^j)$. Then, $y^* = f(x^*)$ is the only nondominated point that is mapped into $\mathcal{B}_j^{p-1}(y^*_{-k}, u^j)$.*

After finding a nondominated point using a rectangle $\mathcal{B}_j^{p-1}(l^j, u^j)$, Theorem 2.3.3 allows the reduction of the search region by eliminating rectangles contained in $\mathcal{B}_j^{p-1}(l^j, u^j)$.

Theorem 2.3.4 ([Kirlik and Sayin, 2014]) *For rectangle $\mathcal{B}_j^{p-1}(l^j, u^j)$, if $P_k(u^j)$ is infeasible, then there is no nondominated point that is mapped into the rectangle $\mathcal{B}_j^{p-1}(y^I_{-k}, u^j)$.*

Theorem 2.3.4 enables the algorithm to eliminate empty rectangles that are contained in $\mathcal{B}_j^{p-1}(y^I_{-k}, u^j)$.

It is also shown that the algorithm is finite and is able to find all the nondominated points of an MODO problem.

2.3.2 [Domínguez-Ríos et al., 2021]

This algorithm uses an augmented Tchebycheff model to generate nondominated solutions. The augmented Tchebycheff model is given as follows:

$$\begin{aligned} &\text{minimize} \quad \left(\max_{j \in \{1,\dots,p\}} \quad (w_j |f_j(x) - y_j^I|) + \rho \sum_{j=1}^{p} w_j |f_j(x) - y_j^I| \right) \\ &\text{subject to} \quad x \in \mathcal{X}, \end{aligned} \tag{2.8}$$

where $\rho \in \mathbb{R}_>$ denotes the augmentation coefficient. The model described in (2.8) is used in [Holzmann and Smith, 2018] and can be used to generate nondominated points.

After finding a nondominated point, [Domínguez-Ríos et al., 2021] divides the solution space into smaller partitions. To achieve this, the algorithm makes use of a splitting procedure called p-partition. They define rectangles, calling them boxes, similar to the ones in [Kirlik and Sayin, 2014]. However, since [Kirlik and Sayin, 2014] uses projections of nondominated points, the rectangles are $(p-1)$ dimensional while in [Domínguez-Ríos et al., 2021], boxes keep all p dimensions. They are defined as $\mathcal{B}^p(l,u) = \{y \in \mathbb{R}^p | l \leqq y \leqq u\}$ where $l, u \in \mathbb{R}^p$.

The p-partition procedure takes a box, $\mathcal{B}^p(l,u)$, and a nondominated point, y, and creates p nonoverlapping new boxes. For $j \in \{1,\ldots,p\}$, they define the set $\mathcal{B}^p_j(l^j,u^j) = \{y' \in \mathcal{B}^p(l,u) | y'_j < y_j \wedge y'_i \geq y_i, \forall i > j\}$ and call it a box with direction j if it is nonempty. In the $\mathcal{B}^p_j(l^j,u^j)$ notation the points l^j and u^j can be computed as follows:

$$\begin{aligned} l^j &= (l_1,\ldots,l_j,\hat{y}_{j+1},\ldots,\hat{y}_p) \\ u^j &= (u_1,\ldots,u_{j-1},\hat{y}_j,u_{j+1},\ldots,u_p) \end{aligned} \tag{2.9}$$

where $\hat{y}_j = \max\{y_j,l_j\}, \forall j \in \{1,\ldots,p\}$ to account for the case where $y \notin \mathcal{B}^p(l,u)$. They also define $\mathcal{B}^p_0$ as $\{y' \in \mathbb{R}^p | y' \geqq \hat{y}\}$ and show that in this way a partition of the original rectangle is formed.

The idea of having partitions with directions functions as a balancing measure so that the generated nondominated points are picked from different directions in order to reach a better representation [Domínguez-Ríos et al., 2021].

The information about the current region to be explored using problem (2.8) is incorporated into the weights w. After the problem given in (2.8) is solved, the search space is divided using the p-partition method and a priority value is computed for each partition. This priority value is a scaled volume measure which is computed as follows:

$$scaled(\mathcal{B}^p(l,u)) = \prod_{j=1}^{p} \frac{u_j - l_j}{y^U_j - y^I_j} \tag{2.10}$$

where y^U is an over-estimator of the nadir point. Following the partitioning of the search space, a new box is selected and the procedure is repeated until all of the unexplored boxes are exhausted.

Since the search is distributed over each direction, this algorithm becomes an anytime algorithm. This means throughout the solution process, when the algorithm is stopped at any time, the solutions obtained so far will form a good representation. We explain what a representation means in the next section.

2.4 Representation of the Pareto Set

It is not uncommon for a multi-objective problem of large scale to have too many efficient solutions for a decision maker to individually consider. In those cases, the Pareto set can be summarized to simplify the effort spent to decide which course of action will be taken. Here, we define what is called a representation and how to assess its quality and performance when treated as a replacement for the whole Pareto set.

Although there are different interpretations of what can be considered as a representation for an MOO problem, we will refer to the definition given in [Sayın, 2000] throughout this thesis, which indicates that we understand a representation as a finite discrete set that is a subset of the nondominated set, $\mathcal{Y}_R$ is a **representation** if $\mathcal{Y}_R \subseteq \mathcal{Y}_N$.

In addition to those given in [Sayın, 2000], several metrics are developed in order to assess the quality of a solution set, mostly in the evolutionary multi-objective literature (see e.g. [Zitzler et al., 2003]). In that case, the solutions are not guaranteed to be nondominated points and some of them are not meaningful for the representations. A select group of representation quality metrics used in this thesis are given below. In our computational work, the exact nondominated set $\mathcal{Y}_N$ is available in all of the test instances. Therefore, the metrics utilize this information.

2.4.1 Overall Nondominated Vector Generation Ratio (ONVGR)

This metric is used to compare the size of the representation to the size of the Pareto set [Jiang et al., 2014]. These types of metrics are called capacity metrics because of what they are trying to achieve. ONVGR is computed as follows:

$$\text{ONVGR} = \frac{|\mathcal{Y}_R|}{|\mathcal{Y}_N|} \tag{2.11}$$

2.4.2 Coverage Error (α)

To assess the quality of a representation, convergence metrics are used. Here, we give the definition for the coverage error as introduced in [Sayın, 2000]. The coverage error is computed as $\max_{y\in\mathcal{Y}_N}\min_{\hat{y}\in\mathcal{Y}_R} d(y,\hat{y})$, where $d(y,\hat{y})$ is a metric that measures the distance between y and $\hat{y}$. The measure relies on the following intuition: each element y of $\mathcal{Y}_N$ is represented by a closest element $\hat{y}$ in $\mathcal{Y}_R$ and the distance between them is the error. The coverage error of a representative set $\mathcal{Y}_R$ is determined by the maximum of these errors. In this study, we have used the Tchebycheff distance divided by the range of each objective as $d(.,.)$, leading to:

$$\max_{y\in\mathcal{Y}_N}\min_{\hat{y}\in\mathcal{Y}_R}\max_{j\in\{1,\ldots,p\}}\frac{|\hat{y}_j - y_j|}{r_j} \tag{2.12}$$

where r_j is the range of objective j ($r_j = \max_{x\in\mathcal{X}_E} f_j(x) - \min_{x\in\mathcal{X}_E} f_j(x)$).

2.4.3 Additive Epsilon (ε_+)

Additive epsilon (ε_+) quantifies the amount of shift required for a set of points to weakly dominate another set of points [Zitzler et al., 2003]. In other words, additive epsilon indicator can be computed between any two sets. Therefore, the nondominated set is not required as an input to come up with an additive epsilon indicator value. The formula for computing the additive epsilon indicator of $\mathcal{Y}_R$ with respect to the reference set $\mathcal{Y}_N$, when $\mathcal{Y}_N$ is available, can be calculated as $\max_{y\in\mathcal{Y}_N}\min_{\hat{y}\in\mathcal{Y}_R}\max_{j\in\{1,\ldots,p\}} \hat{y}_j - y_j$. In our computational experiments, we use the scaled Tchebycheff distance as we did in the coverage error. Then, the additive epsilon indicator for a representation $\mathcal{Y}_R$ is computed as follows:

$$\varepsilon_+(\mathcal{Y}_R) = \max_{y\in\mathcal{Y}_N}\min_{\hat{y}\in\mathcal{Y}_R}\max_{j\in\{1,\ldots,p\}}\left(\frac{\hat{y}_j - y_j}{r_j}\right). \tag{2.13}$$

2.4.4 Range Ratio (RR)

A good representation should cover the spread of efficient solutions well. Therefore, comparing the range of objective values over the representation to that of the Pareto set should give useful insights about the quality of the representation. Here, we define range ratio as the average range coverage by the representation [Sayın, 2024]. It is computed as follows:

$$\mathrm{RR} = \frac{1}{p}\sum_{j\in\{1,\ldots,p\}}\frac{\max_{\hat{y}\in\mathcal{Y}_R}\hat{y}_j - \min_{\hat{y}\in\mathcal{Y}_R}\hat{y}_j}{\max_{y\in\mathcal{Y}_N} y_j - \min_{y\in\mathcal{Y}_N} y_j}. \tag{2.14}$$

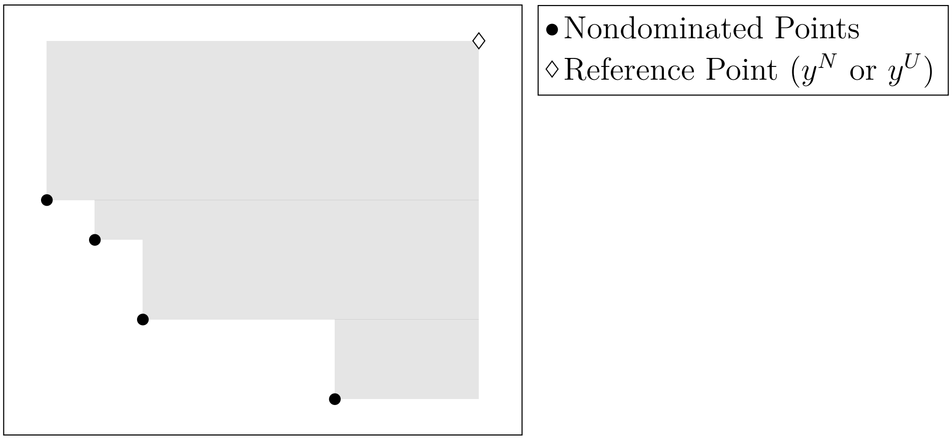


Figure 2.3: The Hypervolume Indicator for a Set of Nondominated Points

2.4.5 Hypervolume (HV) and Hypervolume Ratio

The Hypervolume indicator measures the volume of the union of the region that consists of hyperrectangles with nondominated points as lower vertices and a common upper vertex. This common vertex is the reference point for the computations, which can be set to the nadir point or its overestimation. Figure 2.3 illustrates the region that is used to compute the hypervolume. In the computation of the hypervolume measure for the representations, we use the sweeping algorithm provided by [Fonseca et al., 2006].

Note that hypervolume can be computed for a representation or the Pareto set itself. Hypervolume ratio is another representation quality metric which is computed as the ratio of the hypervolume of the representation to that of the Pareto Front ($\mathrm{HV}(\mathcal{Y}_R)/\mathrm{HV}(\mathcal{Y}_N)$).

Since these metrics are an indication of how well the representation can perform if reported to a decision maker instead of the Pareto Front, we will use them to evaluate the approaches developed in this thesis.

2.5 *Redundant Objectives, Essential Objectives and Dimension Reduction*

Regardless of the method chosen, computational studies and experiments show that as the number of objective functions increases, finding the Pareto set becomes exponentially more difficult. This is because adding objective functions introduces more ways for solutions to become efficient and thus the size of the nondominated set grows exponentially. For this reason, determining whether an objective function is required in a problem formulation can be worth the effort.

An objective is **redundant (or non-essential)** if excluding it does not change the set of efficient solutions. If an objective is not redundant, it is called an **essential objective**. Dimension reduction for MOO problems refers to removing objectives from the problem formulation. Depending on whether the removed objective is an essential or redundant objective, the set of efficient solutions obtained from the reduced problem may change.

[Gal and Leberling, 1977] provided the sufficient condition for an objective to be redundant within the context of multi-objective linear programming (MOLP) problems, for which $\mathcal{X} = \{x \in \mathbb{R}^n_{\geqq} | Ax \geq b\}$.

$$\begin{aligned} (\textbf{MOLP}) \quad \text{“minimize”} \quad & f(x) = ((c^1)^T x, \dots, (c^p)^T x) = Cx \\ \text{subject to} \quad & Ax \geq b, \\ & x \geq 0, \end{aligned} \tag{2.15}$$

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c^j \in \mathbb{R}^n, \forall j \in \{1, \dots, p\}$ and $C \in \mathbb{R}^{p \times n}$.

A theory is also given that describes what makes an objective redundant and a procedure is developed to show whether an objective is redundant.

Theorem 2.5.1 ([Gal and Leberling, 1977]) *An objective function for a multi-objective linear optimization problem given in* (2.15)*, with the objective coefficient vector* $c^q, q \in \{1, \dots, p\}$*, is redundant if and only if the following holds true:*

$$c^q = \sum_{\substack{j=1 \\ j \neq k}}^{p} \lambda_j c^j, \quad \lambda_j \geq 0, \quad \forall j \in \{1, \dots, p\}, \quad j \neq q. \tag{2.16}$$

There is also a characterization provided with the following theorem that provides a necessary and sufficient condition.

Theorem 2.5.2 *An objective function for a multi-objective linear optimization problem with the objective coefficient vector $c^q, q \in \{1, \ldots, p\}$, is redundant if and only if*

1. *$\forall \hat{x} \in \mathcal{X}_E$, $\exists \lambda \in \mathbb{R}^{p-1}_{>}$ such that $\hat{x}$ is the optimal solution of the following problem:*

$$\begin{aligned} \textit{minimize} \quad & \sum_{\substack{j=1 \\ j \neq q}}^{p} \lambda_j (c^j)^T x_j \\ \textit{subject to} \quad & x \in \mathcal{X}. \end{aligned} \tag{2.17}$$

2. *$\forall \hat{x} \in \bar{\mathcal{X}}^q_E$, $\exists \lambda \in \mathbb{R}^{p}_{>}$ such that $\hat{x}$ is the optimal solution of the following problem:*

$$\begin{aligned} \textit{minimize} \quad & \sum_{j=1}^{p} \lambda_j (c^j)^T x_j \\ \textit{subject to} \quad & x \in \mathcal{X}, \end{aligned} \tag{2.18}$$

where $\bar{\mathcal{X}}^q_E$ is the efficient set of the problem obtained when objective q is removed.

[Agrell, 1997] argues that the definition provided by [Gal and Leberling, 1977] is incomplete and is solely based on the geometric interpretation of objectives. Hence, a new definition of redundancy is provided based on the conflict between objectives. This leads to the definition of pairwise conflict between objectives. Two objectives are not in conflict if the difference between their solution images can be scaled by a positive real number. The conflict and redundancy is discussed in the context of MOLPs and it has been shown that the geometric interpretation discussed by [Gal and Leberling, 1977] does not imply the existence of conflict based redundancy. An illustrative example in two dimensional space with two independent objective coefficient vectors and a third clearly linearly dependent objective coefficient vector is provided. The example shows that although the third objective is redundant according to [Gal and Leberling, 1977], it should still be kept since it is in conflict with other objectives and its value at the ideal point is different. This approach focuses on decision analysis and argues that the objectives removed using the ideas presented in [Gal and Leberling, 1977] should not be ignored and still be presented to the DM although it is still possible to generate all the efficient solutions without those objectives.

[Thoai, 2012] explains a procedure to arrive at a set of extreme objectives. These are the objectives that are sufficient to obtain the efficient solutions of the original problem. The reduced problem can be solved and the resulting set of efficient solutions can be used to find the efficient solutions of the original problem by some affine transformation. Each objective is tested to be obtained from a linear combination of other objectives. This follows from [Gal and Leberling, 1977]. Then, the initially empty set of extreme objectives are updated until all objectives are processed.

[Malinowska, 2006] also provides necessary conditions for an objective to be redundant when the objective functions are continuous and convex and the feasible region is compact and convex. The study continues with guidelines on how to incorporate the necessary conditions into reducing objectives in multi-objective linear problems. The procedure presented can be used to remove non-essential objectives. [Malinowska and Torres, 2008] applies the same method and provides computational examples when applied to multi-objective linear optimization problems.

There are also a number of studies in evolutionary multi-objective optimization that address the objective reduction problem. [López Jaimes et al., 2008] proposes negative correlation to explain the conflict between different objectives. They provide an unsupervised feature selection method that works with a set of non-redundant objectives. This method is similar to k-neighbors in that the objectives are grouped into several neighborhoods. Amongst these neighborhoods, the most compact one that is measured through the correlation between objectives within a neighborhood is replaced by a central objective which is the average objective in that neighborhood. Using this unsupervised selection method, they first remove any redundant objective from the objective set and then try to find a subset of objectives of predetermined size that gives the minimum error measured using two quality indicators (δ-error and inverted generational distance) the distance between a representation and a placeholder for the nondominated set obtained using an evolutionary algorithm. The feature selection technique used in the study is formerly developed by [Mitra et al., 2002].

[López Jaimes et al., 2014] argues that the performance of multi-objective evolutionary algorithms scale poorly to problems with many objectives. In order to help with that, they group the objectives into several subsets. This corresponds to forming several subproblems from the original MOO problem. The algorithm follows from the original framework developed by [Aguirre and Tanaka, 2009]. In this evolutionary algorithm, the initial population is formed using disjoint subspaces of objectives and subsequent generations are formed using a fraction of the objectives from parent subspaces. Crossover and mutation operators are also applied during the algorithm. Performance is compared to Nondominated Sorting Genetic Algorithm II (NSGA-II).

[Brockhoff and Zitzler, 2009] looks at the change in dominance structure after an objective is removed. Conflict is defined for sets of objective functions. If an objective is not redundant, some nondominated points will become dominated when we remove that objective. Therefore, two sets of objectives are conflicting if the induced set of nondominated points are different. Otherwise, they are non-conflicting. In order to quantify the level of conflict, they again look at the change in dominance structure and see how much a solution value should be corrected in order to preserve the set of nondominated points. This error value, δ, is computed by looking at the point that became dominated when a set of objectives is removed. From that point forward, two methods of objective reduction are proposed. First is to remove a set of objectives so that the error does not exceed δ. This problem is called δ-Minimum Objective Subset (δ-MOSS). The other one is to remove a predetermined number of objectives with minimum error, which is the problem to find the Minimum Objective Set of Size k with Minimum Error (k-EMOSS). Both problems are shown to be $\mathcal{NP}$-hard and an exact method along with a greedy heuristic algorithm are provided to solve both problems. They test the performance of the methods described on test instances of 0-1 knapsack problems with different number of objectives and reductions. The quality of the set of generated points are compared using hypervolume indicator without the availability of the Pareto Front.

[Cheung and Gu, 2014] uses negative correlation values between objective vectors as the level of conflict between objectives. They formulate a problem to find weights that will minimize the correlation between objectives after the transformation. It has been shown that the formulated problem can preserve the dominance structure as much as possible in the case of multiple essential objective functions. A new metric is also developed to assess the performance of the objective reduction method by comparing the Pareto set of the reduced problem with that of the original. The method is later integrated into a heuristic genetic algorithm. [Cheung et al., 2016] uses the same objective reduction method in evolutionary multi-objective optimization where the weights are updated according to the solutions generated. [Vázquez et al., 2018] gives different mixed-integer linear programming formulations for δ-MOSS and k-EMOSS models. Change in dominance structure is shown using examples and computational time and other results are compared using different sets of instances.

[Deb et al., 2005] uses principal component analysis (PCA) to decide on which objectives to remove in an evolutionary setting. The method starts with an initial population of solutions. Then, the correlation among different objectives are computed using the initial population. From the correlation matrix, the eigenvalues are extracted. Along each principal component, behaviors of different objectives are observed and the ones having the most conflict that will enrich the diversity in the set of generated solutions are picked. A threshold is used to manage processing principal components up until the total contribution to the solution diversity is satisfactory. The conflict information computed from the PCA is incorporated into NSGA-II. The performance is tested on previously presented many objective optimization problems [Deb et al., 2001].

Much of the work on objective reduction in MOO is focused on MOLPs. In this thesis, we develop an objective reduction approach based on geometric arguments that takes into account the possibility of some information loss. Although our arguments would be valid for MOLPs as well, we focus on MOILP problems. There are two reasons for this. First, having MOILPs with known nondominated sets makes the assessment of the proposed approach straight-forward. Second, we will discuss ways to leverage the nondominated set of the reduced MOILP problem to describe the search region of the original problem with p objectives in the later parts of this thesis. For this reason, in the next section, we will give a brief overview of some of the work that has been done on describing the search region for MOO problems.

The general form of an MOILP problem is given below:

$$\begin{aligned} (\textbf{MOILP}) \quad \text{“minimize”} \quad & f(x) = ((c^1)^T x, \dots, (c^p)^T x) \\ \text{subject to} \quad & Ax \geq b, \\ & x \in \mathbb{Z}^n_{\geqq}. \end{aligned} \tag{2.19}$$

2.6 Search Region Representations

One of the crucial components of a multi-objective optimization algorithm is the representation and update of the search region. It can be said that the better performing algorithms are usually the ones that are also good at the computation and management of the search region. These approaches differentiate themselves with the way they split the search region and their scalarization methods. A similar classification is also provided in [Karsu and Ulus, 2022].

2.6.1 $(p-1)$-dimensional Split Algorithms

These algorithms work on the projected search space by choosing an objective and projecting the nondominated points onto the $(p-1)$-dimensional space of the other remaining objectives. At each iteration, they control the values of the remaining objectives to create bounds. Consequently, these approaches generally use the ε-constraint methods. [Mavrotas, 2009] creates $(p-1)$-dimensional grids to represent the search region. These grids are computed as equal sized search zones by splitting the initial search region defined by the ideal and nadir points. [Mavrotas and Florios, 2013] and [Zhang and Reimann, 2014] tried to improve this method by implementing detection of the empty grids. [Ozlen and Azizoğlu, 2009] develops bounds dynamically as they generate new nondominated points instead of enumerating search zones from the whole search region in the beginning of their algorithm.

The method is later improved in [Ozlen et al., 2011] using recursion. [Dhaenens et al., 2010] extends the Parallel Partitioning Method from bi-objective to multi-objective problems. They consider subproblems with a subset of objectives and partition the higher dimensional search space with the nondominated points obtained from the reduced subproblems. Further examples of these $(p-1)$-split algorithms can be found in [Lokman and Köksalan, 2013, Kirlik and Sayin, 2014, Boland et al., 2017b, Turgut et al., 2019, Laumanns et al., 2006].

2.6.2 p-dimensional Split Algorithms

These algorithms keep all p dimensions in the representation of the search region and, consequently, all the search zones are also in full p dimensions. Scalarization methods are chosen appropriately to be able to use all p dimensions. Alternatively, in considering each p-dimensional search zone, some dimensions can be ignored in order to enable the usage of other scalarization methods such as the ε-constraint method. [Bektaş, 2018] uses a disjunctive programming approach and weighted sum scalarization to develop an iterative p-split algorithm. [Holzmann and Smith, 2018] and [Domínguez-Ríos et al., 2021] use a modified weighted Tchebycheff scalarization to generate nondominated points. They use the search zones to adjust the search parameters in the scalarization method. [Klamroth et al., 2015] starts with a stable set that is defined as a set of feasible points that do not dominate each other. It is then shown that the whole search region can be described as a union of search zones associated with each point in the stable set. Every search zone contains the set of feasible points that dominate the associated point in the stable set. Then, an equivalent description is derived using a set of local upper bounds. Then, it has been shown that the search can be restricted by creating a stronger representation of the search region using maximal upper bounds and an algorithm is provided in order to update the search region and removing any redundancies when a new nondominated point is found. [Dächert et al., 2017] contribute to [Klamroth et al., 2015] by introducing defining points that can be used to update the search region and adding a neighborhood relation between upper bounds to make the computations more efficient since they prevent creating redundant search zones. Later, [Tamby and Vanderpooten, 2021] uses a similar search space characterization and ε-constraint method by ignoring components of upper bounds, leading to a computationally competitive exact algorithm.

Chapter 3

ELIMINATING OBJECTIVES

In this chapter, we will explain the concepts and ideas used in coming up with our dimension reduction method. First, we will explain subspace and cone projections, how they are computed and how they relate to the concept of redundant objectives. Then, we will mention cosine similarity and, subsequently, measure it for an objective coefficient vector and its projection onto the subspace created by the other objective coefficient vectors, and explain how it relates to the problem of dimension reduction. Next, we will present our method for eliminating objectives for an MOILP problem by putting together the concepts we previously laid out. We will also describe the intuition behind our approach in determining which objective to remove. Finally, we will set up a slightly modified two-stage ε-constraint model for generating nondominated points of the original problem after removing an objective function.

3.1 Redundant Objectives and Their Projections

Previously, we gave the characterization of a redundant objective by [Gal and Leberling, 1977] for an MOLP problem. Given a set of objective coefficient vectors $(c^1, c^2, \ldots, c^p)$, one of which is redundant for an MOLP, we are able to determine which objective to remove and still be able to determine the Pareto set for all p objectives. For an MOLP, a redundant objective is closely related to its projection onto a subspace generated by the other objectives. This is based on the redundancy characterization provided by [Gal and Leberling, 1977]. Moreover, we are able to draw a conclusion about an objective by assessing its distance to its projection. Consequently, this distance measure can be used as an indicator of objective redundancy in MOLP problems and as a pseudo indicator to objective redundancy in other types of problems. In the remaining part of this chapter, we will show how this distance can be computed and use it to build a method for objective reduction in MOILP problems. In the next chapter, we put our approach to test in extensive computational experiments.

3.2 Subspace and Cone Projection

A projection in linear algebra is a linear transformation of a vector using a projection matrix, P, such that when applied multiple times the result will be the same [Strang, 2016]. Consider a vector a and its projection onto an object, $\mathcal{O}$, $proj_{\mathcal{O}}a$. Then, we can say that $proj_{\mathcal{O}}a = Pa = Pproj_{\mathcal{O}}a$. In other words, $P^2 = P$, if the linear transformation is applied to itself, the result will not change.

The orthogonal projection of a vector onto an object $\mathcal{O}$ is computed by finding the closest point in the object $\mathcal{O}$ [Strang, 2016]. This is achieved by setting the error vector, $a - proj_{\mathcal{O}}a$, to be perpendicular to $\mathcal{O}$. From this point on, when we mention projections, we will refer specifically to orthogonal projections.

3.2.1 Projection Onto a Subspace

A subspace can be described by using a basis. Let $\{b^1, \ldots, b^m\} \in \mathbb{R}^n$ be linearly independent vectors. We know that a subspace created by $Span(\{b^1, \ldots, b^m\})$ is made up of linear combinations of these vectors. Then, $Span(\{b^1, \ldots, b^m\})$ is equal to the column space of matrix $B = [b^1, b^2, \ldots, b^m]$, where the $\{b^1, \ldots, b^m\}$ are columns of B.

Let $\beta \in \mathbb{R}^m$ be a vector of coefficients associated with columns of B. Let $S(B)$ denote the column space of B, [Koç, 1995]. Let the projection of $a \in \mathbb{R}^n$ onto $S(B)$ be given by $proj_{S(B)}a = B\beta$. Observe that $proj_{S(B)}a \in S(B)$.

Since the error vector, $(a - proj_{S(B)}a)$, is orthogonal to the subspace $S(B)$, it should also be orthogonal to each vector in $S(B)$. Then, we can write that $\left\langle a - proj_{S(B)}a, b^j \right\rangle = \langle a - B\beta, b^j \rangle = 0, \forall j \in \{1, \ldots, m\}$. In matrix form, we get the

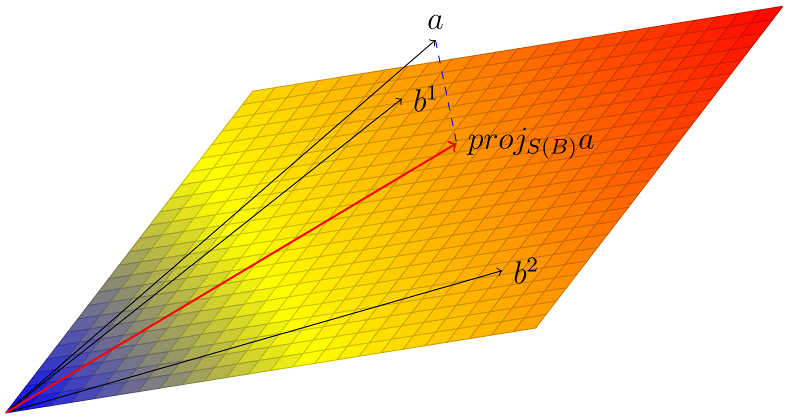


Figure 3.1: Projection onto a plane

following:

$$\begin{aligned} B^T(a - B\beta) &= 0 \\ B^T a - B^T B\beta &= 0 \\ B^T B\beta &= B^T a \\ \beta &= (B^T B)^{-1} B^T a. \end{aligned} \tag{3.1}$$

Now that we have the coefficients, β, we can compute the projection as $proj_{S(B)}a = B\beta = B(B^T B)^{-1}B^T a$. We can extract the projection matrix from this equation as $P = B(B^T B)^{-1}B^T$. P can be used to compute the projection of an n-dimensional vector x onto $S(B)$ as Px. An example is shown in Figure 3.1 where $B = [b^1, b^2]$ and the $Span(\{b^1, b^2\})$ is the plane shown in the figure and the red vector is the projection of vector a onto the plane, $proj_{S(B)}a$.

An alternative derivation of the projection of a vector a onto $S(B)$ is possible by considering the distance of the vector to its projection. This approach may provide a more intuitive explanation of our objective reduction approach.

Proceeding with the same logic of using orthogonality, we know that the error vector, $(a - proj_{S(B)}a)$, is perpendicular to the subspace and the projection will provide the closest point to a on the subspace. Therefore, we want to compute β by solving $\min_{\beta \in \mathbb{R}^m} h(\beta) = \|a - B\beta\|^2$. Let $h(\beta) = (a - B\beta)^T(a - B\beta)$. Then, $\nabla_\beta h(\beta) = 2B^T B\beta - 2B^T a$. By setting the gradient equal to zero and solving for β, we obtain:

$$\nabla_\beta h(\beta) = 2B^T B\beta - 2B^T a = 0$$
$$B^T B\beta = B^T a$$
$$\beta = (B^T B)^{-1} B^T a.$$

Thus, the projection is given by:

$$proj_{S(B)} a = B(B^T B)^{-1} B^T a. \tag{3.2}$$

Note that $(B^T B)^{-1}$ only exists if B has full column rank, meaning the columns of B are linearly independent.

Example: projection onto a line The subspace of a single vector is a line passing through the origin. Given a vector $a \in \mathbb{R}^n$, we want to find its projection onto a line created by vector $b \in \mathbb{R}^n$. Since the projection will belong to the subspace generated by b, it is going to be a scalar multiple of b. Let us call this scalar β. Then, using the definition given before, we can write the following:

$$\left\langle a - proj_{S(b)} a, b \right\rangle = 0$$
$$\langle a - \beta b, b \rangle = 0$$

where $S(b)$ is the subspace of vector b which is a line passing through the origin and b. As can be seen in Figure 3.2, the vector $(a - \beta b)$ is orthogonal to b, which means $\langle a - \beta b, b \rangle = 0$. Also, the projection will mark the closest point to a on the line. We can proceed to find:

$$(a - \beta b)^T b = 0$$
$$a^T b - \beta b^T b = 0$$
$$\beta b^T b = a^T b$$
$$\beta = \frac{a^T b}{b^T b}$$

Thus, $proj_{S(b)} a = \frac{a^T b}{b^T b} b$.

We can also see the projection matrix, $P = \frac{bb^T}{b^T b}$. Observe that P is $n \times n$ and the projection is obtained as Pa so the result will be another n-dimensional vector.

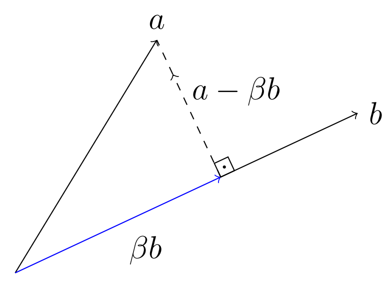


Figure 3.2: Projection onto a line

Now, consider an MOLP with p objectives and the projection of objective q onto the subspace generated by the span of the other objectives. The objective coefficient matrix, C, from Equation (2.15) is a $p \times n$ matrix. Let $\bar{C}^q$ be the $(p-1) \times n$ matrix that is the same as C except that row q is removed. The subspace created by the objective coefficient vectors other than c^q is equal to the column space of the $[\bar{C}^q]^T$. Plugging in $B = [\bar{C}^q]^T$ into Equation (3.2), we get the following expression for the projection of c^q onto $S(\bar{C}^q)$.

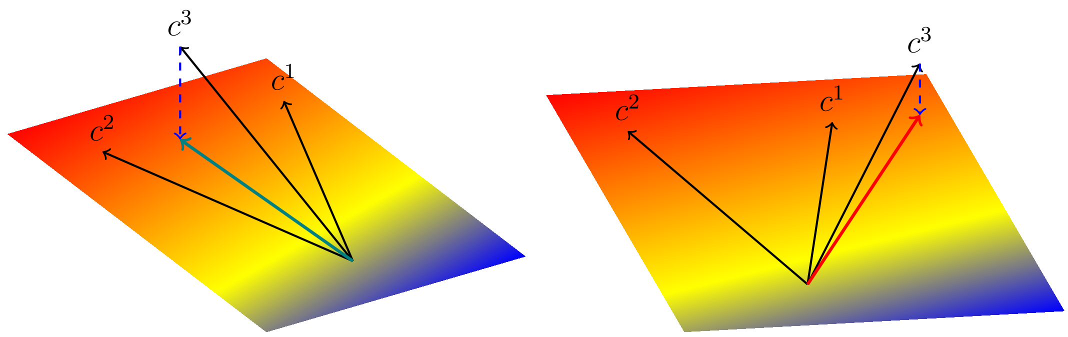
$$\beta^q = ([\bar{C}^q][\bar{C}^q]^T)^{-1}[\bar{C}^q]c^q \tag{3.3}$$

$$proj_{S([\bar{C}^q]^T)}c^q = [\bar{C}^q]^T\beta^q = [\bar{C}^q]^T([\bar{C}^q][\bar{C}^q]^T)^{-1}[\bar{C}^q]c^q \tag{3.4}$$

Proposition 3.2.1 *Given a MOLP problem as described in Equation (2.15), consider replacing the objective coefficient vector c^q with the expression given in Equation (3.4). Then, the new objective function added instead of $f_q(x)$ is redundant if $\beta^q \in \mathbb{R}^{p-1}_{\geq}$.*

Proof. To show that the new objective function, $(c^q)^T([\bar{C}^q]^T([\bar{C}^q][\bar{C}^q]^T)^{-1}[\bar{C}^q])x$, is redundant, we can set the $\lambda \in \mathbb{R}^{p-1}$ values from Theorem (2.5.1) equal to the β values used in computing the projection. Then, $proj_{S([\bar{C}^q]^T)}c^q = \sum_{\substack{j=1 \\ j \neq q}}^{p} \beta^q_j c^j$. Therefore, by Theorem (2.5.1), the new objective function is redundant as long as $\beta^q \in \mathbb{R}^{p-1}_{\geq}$. □

Note that, we require $\beta^q \in \mathbb{R}^{p-1}_{\geq}$. This is not guaranteed with subspace projection. An example of this can be seen in Figure 3.3. On the left, replacing c^3 with its subspace projection and solving the new multi-objective problem will result in a valid representation while, in the picture on the right, the subspace projection of c^3



(a) (b)

Figure 3.3: Subspace Projection: (a) positive β coefficients, (b) negative β coefficients.

is not a valid replacement in the multi-objective sense for it because β has negative coefficients and, in the newly formed problem, we will obtain points that are not nondominated for the original problem.

Next, we will explain how to find the projection of a vector onto a polyhedral cone and use it to find a replacement for c^q, which will make it a redundant objective.

3.2.2 Projection Onto a Polyhedral Cone

In this case, we want to find the projection onto a polyhedral cone which is formed by the conical combinations of the columns of B, where the cone generated by the columns of B is defined as:

$$cone(B) = \{x \in \mathbb{R}^n | x = B\lambda, \lambda \in \mathbb{R}^m_{\geqq}\}$$

We still want to find the point that is an element of $cone(B)$ which is a polyhedral cone and is closest to the vector to be projected. For that, we need to solve the following problem:

$$\min_{\beta \in \mathbb{R}^m_{\geqq}} \|a - B\beta\|^2$$

This is a quadratic optimization problem whose solution is no longer analytically available and can be obtained using a solver. The difference between a subspace projection with negative coefficients values in β and the cone projection is depicted in Figure 3.4.

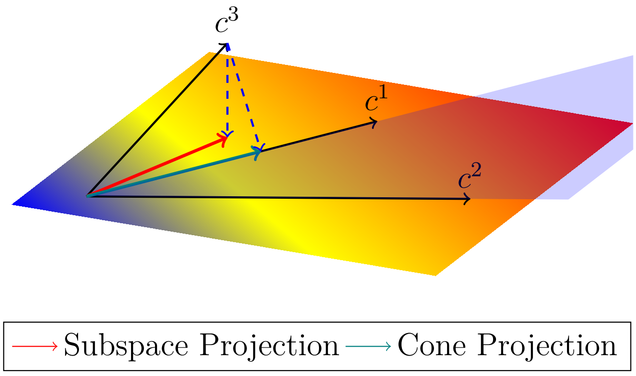


Figure 3.4: Cone projection vs. subsapce projection

Replacing an objective coefficient vector with a vector taken from the cone created by the remaining objective coefficient vectors will make that objective redundant. Among all the possible replacements, the cone projection is the one that is closest to the original objective coefficient vector. Following Proposition 3.2.1, it can be said that replacing an objective coefficient vector with its projection onto the polyhedral cone formed by the remaining coefficient vectors will make it a redundant objective.

The distance of an objective coefficient vector to its closest redundant counterpart will provide a way of assessing the level of redundancy for an objective function. Next, we will explain cosine distance and cosine similarity in order to measure the similarity of an objective coefficient vector to its redundant counterparts, which will help us define our method of objective reduction.

3.3 Cosine Similarity and Cosine Distance

Cosine similarity of two vectors is the dot product of the two vectors divided by their norms. Let $a, b \in \mathbb{R}^n$. Then, their cosine similarity is computed as:

$$sim(a, b) = \frac{a^T b}{||a|| \, ||b||}$$

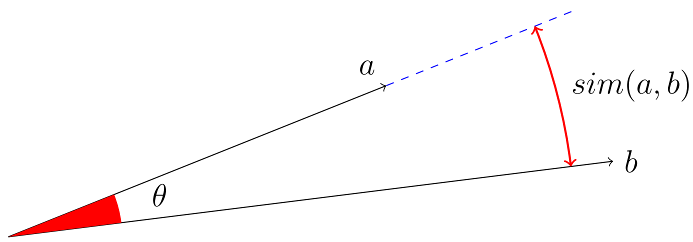


Figure 3.5: Cosine Similarity between two vectors, $sim(a,b) = \cos(\theta)$

It is easy to see that, if $a = sb$ for some scalar s, $sim(a,b) = 1$. Therefore, the cosine distance between two vectors a and b is defined as:

$$d_{\cos}(a,b) = 1 - sim(a,b).$$

Cosine similarity/distance is related to the angle between two vectors and is used quite often in machine learning.

We will use the distance between objective coefficient vectors and the cone projections to assess the level of redundancy for each objective function. The reason for this choice of cosine similarity is its effectiveness and frequent use in exploratory data analysis and the fact that it is agnostic about the scale/norm of each vector. If we rewrite the dot product of vectors a and b as $\|a\|\|b\|\cos(\theta)$, we will see that the cosine similarity is equal to the cosine of the angle between two vectors, which is independent of the norms of the vectors. Next, we explain the ideas behind our elimination method.

3.4 Intuition behind the Elimination Method

Consider the set of objective coefficient vectors shown in Figure 3.3a. Let θ_3 be the angle between the objective coefficient vector c^3 and its projection onto $cone([\bar{C}^3]^T)$.

Consider a modified version of our MOILP where we replace objective coefficient vector c^3 by its projection $proj_{cone([\bar{C}^3]^T)}c^3$, the green vector in Figure 3.3a. In that case, we can say that the new problem has a redundant objective and it is the one we just added instead of c^3 since $proj_{cone([\bar{C}^3]^T)}c^3$ is a conic combination of c^1 and c^2.

Let θ_3 be the angle between c^3 and $proj_{cone([\bar{C}^3]^T)}c^3$. We can compute the cosine similarity and cosine distance between c^3 and $proj_{cone([\bar{C}^3]^T)}$ as follows:

$$sim(c^3, proj_{cone([\bar{C}^3]^T)}c^3) = \cos(\theta_3) = \frac{\|proj_{cone([\bar{C}^3]^T)}c^3\|}{\|c^3\|}$$

$$d_{\cos}(c^3, proj_{cone([\bar{C}^3]^T)}c^3) = 1 - sim(c^3, proj_{cone([\bar{C}^3]^T)}c^3)$$

In our example, cone projection provides a way to transform c^3 into a redundant objective by taking its projection and we can be sure that it is in fact redundant because it is linearly dependent on c^1 and c^2 and $\beta^3 \in \mathbb{R}^2_{\geq}$. We can find other redundant objective coefficient vectors from the same cone but what is special about the cone projection is that it is the one that is closest to c^3.

$d_{\cos}(c^q, proj_{cone([\bar{C}^q]^T)}c^q)$ can be interpreted as the cosine distance to redundancy for c^q. Let θ_q be the angle between c^q and $proj_{cone([\bar{C}^q]^T)}c^q)$. Then,

$$sim(c^q, proj_{cone([\bar{C}^q]^T)}c^q) = \cos(\theta_q) = \frac{\|proj_{cone([\bar{C}^q]^T)}c^q\|}{\|c^q\|}$$

$$d_{\cos}(c^q, proj_{cone([\bar{C}^q]^T)}c^q) = 1 - sim(c^q, proj_{cone([\bar{C}^q]^T)}c^q).$$

Definition 3.4.1 (Cosine Distance to Redundancy) *For an MOILP with the objective coefficient matrix C and given an objective function q, its **cosine distance to redundancy** is defined as:*

$$d_{\cos}(c^q, proj_{cone([\bar{C}^q]^T)}c^q) = 1 - sim(c^q, proj_{cone([\bar{C}^q]^T)}c^q). \tag{3.5}$$

The intuition behind our approach is that the closer an objective coefficient vector is to redundancy, the effect of removing that objective is going to be less compared to other objectives, evaluated by comparing the nondominated sets of the reduced and original problems.

We can show that the cone projection ensures to generate a redundant objective for an MOILP. Let $y_{-q} = (y_1, \ldots, y_{q-1}, y_{q+1}, \ldots, y_p)$.

Theorem 3.4.1 (Redundant objectives in MOILP) *Let $c^q = \sum_{\substack{j=1 \\ j\neq q}}^{p} \lambda_j c^j$ for an MOILP where $\lambda \in \mathbb{R}^{p-1}_{\geq}$. Then, $f_q(x) = (c^q)^T x$ is redundant.*

Proof. Assume f_q is not redundant. Then, $\exists y' \in \mathcal{Y}_N \backslash \bar{\mathcal{Y}}^q_N$. Since $y' \notin \bar{\mathcal{Y}}^q_N$, $\exists y''$ such that $y''_{-q} \leq y'_{-q}$. Then, $\sum_{\substack{j=1 \\ j\neq q}}^{p} \lambda_j y''_j \leq \sum_{\substack{j=1 \\ j\neq q}}^{p} \lambda_j y'_j$, which means $y''_q \leq y'_q$. Then, $y'' < y'$, which contradicts $y' \in \mathcal{Y}_N$. □

Next, we will explain how to determine which objective to remove using the approach that we have just described.

3.5 Metric for determining which objective to remove

For our metric, we will compute the projections first. Let q be the index of the objective we consider to remove from a MOILP with the corresponding objective function f_q and the vector of objective coefficients, c^q. Thus, $f_q(x) = (c^q)^T x$. Recall that matrix C of objective function coefficients is given by:

$$C = \begin{bmatrix} — & (c^1)^T & — \\ & \vdots & \\ — & (c^p)^T & — \end{bmatrix}$$

We assume that all of the objective coefficient vectors are linearly independent. Otherwise, we can check C^T for redundancy using Theorem 2.5.1 and eliminate the ones that are redundant. Let $\bar{C}^q$ denote the matrix obtained from C by removing the q^{th} row. We compute the coefficients β^q by solving $\min_{\beta^q \in \mathbb{R}^{p-1}_{\geq}} \|c^q - [\bar{C}^q]^T \beta^q\|$. β^q can then be used to build the projection of c^q onto $cone([\bar{C}^q]^T)$.

Once we have the value of β^q, we can compute the cone projection as $[\bar{C}^q]^T \beta^q$ and the cosine distance between c^q and $proj_{cone([\bar{C}^q]^T)} c^q$ will be equal to:

$$d_{\cos}(c^q, proj_{cone([\bar{C}^q]^T)} c^q) = 1 - sim(c^q, [\bar{C}^q]^T \beta^q).$$

To pick which objective to remove, we determine q^* as follows:

$$q^* = \arg\max_{j \in \{1,\dots,p\}} \cos(\theta_j) = \arg\min_{j \in \{1,\dots,p\}} d_{\cos}(c^q, proj_{cone([\bar{C}^q]^T)} c^q), \qquad (3.6)$$

where θ_j is the angle between c^j and $proj_{cone([\bar{C}^j]^T)} c^j$. This will indicate the objective which has the smallest distance to redundancy. We propose solving a reduction of the original problem by removing this objective function.

3.6 Solving the Reduced Problem and Ensuring Nondominatedness in the Original

It is obvious that eliminating one of the objective functions of the given problem and forgetting about it completely will not only lead to not being able to compute some nondominated points, but the nondominated points of the reduced problem, when lifted back into the original space, do not have to be nondominated. There can be more than one feasible point that have the same values for the objective functions that are kept in the reduced problem and different values for the one that has been removed. That means only one of those feasible solutions is efficient and the rest are weakly dominated. In order to guarantee finding a nondominated point of the original problem using only $p-1$ objectives, we propose using the following two-stage formulation that excludes objective function q:

$$\begin{array}{lll} \bar{P}_k^q(\varepsilon) & \min & y_k^* = f_k(x) \\ & \text{subject to} & f_i(x) \leq \varepsilon_i, \quad i = 1, \dots, p \quad i \neq k, i \neq q \\ & & x \in \mathcal{X}, \end{array} \tag{3.7}$$

$$\begin{array}{lll} \bar{Q}_k^q(\varepsilon) & \min & \sum_{i=1}^{p} f_i(x) \\ & \text{subject to} & f_i(x) \leq \varepsilon_i, \quad i = 1, \dots, p \quad i \neq k, i \neq q \\ & & f_k(x) = y_k^*, \\ & & x \in \mathcal{X}. \end{array} \tag{3.8}$$

Note that while the ε-constraints are only written for every objective except objective q in both stages, it is included in the objective function of the second stage formulation, $\bar{Q}_k^q(\varepsilon)$.

Theorem 3.6.1 *The optimal solution, x^*, for the two-stage formulation, $(\bar{P}_k^q(\varepsilon), \bar{Q}_k^q(\varepsilon))$, for $\varepsilon \in \mathbb{R}^{p-2}$ is a nondominated point of the original MOILP.*

Proof. Suppose x^* is not efficient. Then, $\exists x' \in \mathcal{X}$ s.t. $f(x') \leq f(x^*)$. Then, x' is a feasible solution for $\bar{P}_k^q(\varepsilon)$ since $f_i(x') \leq f_i(x^*) \leq \varepsilon_i, i = 1, ..., p$ and $i \neq q$, $i \neq k$. Also, $f_k(x') \leq f_k(x^*)$. Since x^* is the optimal solution of $\bar{P}_k^q(\varepsilon)$, $f_k(x') = f_k(x^*) = y_k^*$. Otherwise, if $f_k(x') < f_k(x^*)$, x^* is not optimal. Then, x' is also a feasible solution of $\bar{Q}_k^q(\varepsilon)$. Summing over all i, we get $\sum_{i=1}^p f_i(x') < \sum_{i=1}^p f_i(x^*)$, which contradicts the optimality of x^* for $\bar{Q}_k^q(\varepsilon)$. Then, x^* must be efficient. □

Let $\bar{\mathcal{X}}_E^q$ and $\bar{\mathcal{Y}}_N^q$ be the set of efficient solutions and nondominated points, respectively, obtained using the above formulation with an MOILP when objective q

is removed. Theorem 3.6.1 establishes that $\bar{\mathcal{Y}}_N^q \subseteq \mathcal{Y}_N$.

With that, we now have a way of obtaining a representation after an objective reduction. In the next chapter, we study how well this approach performs on various problem classes.

Chapter 4

COMPUTATIONAL EXPERIMENTS

In this chapter, we will explain how we tested the performance of our elimination method. We used problem instances from [Kirlik, 2014], [Mesquita-Cunha et al., 2022] and [Tamby and Vanderpooten, 2021]. We have also generated some problem instances with correlated objective vectors in order to mimic scenarios where eliminating objectives is likely to be more beneficial. We applied our method and evaluated the outcomes as explained below.

4.1 Testing Methodology

Our experimental framework relies on the following expectation. If none of the objectives in an MOILP is redundant, when we remove an objective, solve the reduced problem as previously explained, and lift the solution back to the original space, we will get a representation of the nondominated set of the original problem.

For a given problem instance, we test our proposed method that identifies the objective function to be removed using cone projections. Although theoretically not justified, we also compute the objective function to be removed based on the subspace projections and report the performance of both approaches.

For benchmarking purposes, we find the entire nondominated set of each test instance. In addition, we find all the nondominated points for all possible reduced problems obtained by removing any one of the objective functions. This way we are able to identify the objective function removal that achieves maximum representational quality in a given metric. We can then compare the representation obtained after removing the objective indicated by our method and the best possible representation obtained after removing an objective.

In the plots that display the performance of our method, we included both the subspace and cone projection based metrics. For each problem set, we group similar instances together based on the number of variables or constraints and take the average metric value of best and worst case scenarios and results of our method. In addition, for each problem, we look at removal of which objective will give the best and worst representation based on each one of the quality metrics. Then, we look at the performance of our method in that same metric benchmarked with respect to the best and worst possible objective function removal. Below we provide a pseudocode that explains our benchmarking procedure for the ε_+ indicator. The procedure for other metrics is similar.

Algorithm 4.1.1 Benchmarking Procedure for Additive Epsilon Indicator (ε_+)

```
Y_N, X_E ← solve(MOILP)
for q = 1, ..., p do
    Ȳ^q_N, X̄^q_E ← solve(MOILP^{-q})
end for
best ← min({ε+(Ȳ^q_N, Y_N)}_{q=1}^p)
worst ← max({ε+(Ȳ^q_N, Y_N)}_{q=1}^p)
```

$\mathcal{Y}_N, \mathcal{X}_E \leftarrow$ **solve**$(MOILP)$
for $q = 1, \ldots, p$ **do**
$\quad \bar{\mathcal{Y}}_N^q, \bar{\mathcal{X}}_E^q \leftarrow$ **solve**$(MOILP^{-q})$
end for
best $\leftarrow \min(\{\varepsilon_+(\bar{\mathcal{Y}}_N^q, \mathcal{Y}_N)\}_{q=1}^p)$
worst $\leftarrow \max(\{\varepsilon_+(\bar{\mathcal{Y}}_N^q, \mathcal{Y}_N)\}_{q=1}^p)$

Amongst the metrics we use, for ONVGR, RR and hypervolume ratio metrics, a higher value is better, while for ε_+ and coverage error, a lower value indicates higher quality in the representation.

We have also looked at the correlation of cosine similarity between an objective function and its cone projection ($\cos(\theta)$) versus each individual quality metric directly. This will allow us to see if there is a relationship between $\cos(\theta)$ and the quality metrics we used. If the value of $\cos(\theta)$ is higher, we expect that objective function to be close to being redundant. Therefore, we anticipate to get higher quality representations by removing an objective function with a higher $\cos(\theta)$ value. Therefore, as $\cos(\theta)$ gets larger, the quality metrics should improve. Since we want to have a lower value for coverage error and additive epsilon indicator, we expect to see a negative correlation between $\cos(\theta)$ and these metrics. For the remaining metrics, we anticipate a positive correlation with $\cos(\theta)$. We can observe that this is the case for almost all of the test problems as indicated by the correlation coefficients in Table A.1 and the correlation plots in Figures A.1 to A.8 in Appendix A. While nearly all of the correlations supported our expectations, there were exceptions. However, it is worth noting that in cases where a weak correlation was observed, the values of the metrics themselves were often very close to each other, suggesting a limited range of variation and potentially explaining the lack of correlation.

We have used the Julia programming language (v1.10.0) [Bezanson et al., 2017] and JuMP.jl (v1.12) package [Dunning et al., 2017] for modeling, Ipopt (a interior point line search filter method [Wächter and Biegler, 2006]) to compute the cone projections and CPLEX (v22.1.1) to solve the original multi-objective problems and their reduced forms. Since JuMP.jl itself does not have the capability to solve multi-objective problems, we have implemented the exact MOILP solution algorithm from [Tamby and Vanderpooten, 2021]. Our implementation of [Tamby and Vanderpooten, 2021] as well as other MOILP algorithms were later used to add multi-objective solution capabilities to JuMP.jl in the form of another package, MultiObjectiveAlgorithms.jl, which is provided at `https://github.com/jump-dev/MultiObjectiveAlgorithms.jl`. We have used high performance computer clusters with Intel(R) Xeon(R) Gold 2640 @ 2.50GHz and 64GB RAM for running all the computations. All of the source codes developed for this thesis are provided at `https://github.com/kofgokhan/RAMODO`.

4.2 Problem Types Used in Testing

We have gathered MOILP instances from the literature to test our method. These problem sets contain different problem types such as the single constraint binary knapsack, binary knapsack with multiple constraints, assignment, traveling salesman and generalized form of integer linear programming problems.

4.2.1 Multi-objective Knapsack Problem (MOKP)

The formulation for the multi-objective knapsack problem is as follows:

$$\max \quad \sum_{i=1}^{n} v_i^k x_i, \qquad k \in \{1, \ldots, p\} \tag{4.1}$$

$$\text{subject to} \quad \sum_{i=1}^{n} w_i x_i \leq W, \tag{4.2}$$

$$x_i \in \{0, 1\}, \qquad \forall i \in \{1, \ldots, n\} \tag{4.3}$$

where v_i^k is the profit for item i in objective k, w_i is the weight of item i and W is the size of the knapsack.

4.2.2 Multi-objective Assignment Problem (MOAP)

The formulation for the multi-objective assignment problem is as follows:

$$\min \quad \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij}^k x_{ij}, \qquad k \in \{1, \ldots, p\} \tag{4.4}$$

$$\text{subject to} \quad \sum_{i=1}^{n} x_{ij} = 1, \qquad \forall j \in \{1, \ldots, n\} \tag{4.5}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad \forall i \in \{1, \ldots, n\} \tag{4.6}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i, j \in \{1, \ldots, n\} \tag{4.7}$$

where c_{ij} is the cost of assigning task i to server j in objective k.

4.2.3 Multi-objective Integer Linear Problem (MOILP)

The formulation for the multi-objective integer linear programming problem is as follows:

$$\min \quad \sum_{i=1}^{n} c_i^k x_i, \qquad k \in \{1, \ldots, p\} \tag{4.8}$$

$$\text{subject to} \quad \sum_{i=1}^{n} a_{ji} x_i \leq b_j, \qquad \forall j \in \{1, \ldots, m\} \tag{4.9}$$

$$x \in \mathbb{Z}^n_{\geqq}. \tag{4.10}$$

4.2.4 Multi-objective Traveling Salesman Problem (MOTSP)

TSP is the problem of finding the lowest cost tour that visits all nodes in a given network and visits every city only once. Miller-Tucker-Zemlin [Miller et al., 1960] formulation for the MOTSP is given below:

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}^{k} x_{ij}, && k \in \{1,\ldots,p\} && (4.11)\\
\text{subject to} \quad & \sum_{\substack{i=1\\ i\neq j}}^{n} x_{ij} = 1, && \forall j \in \{1,\ldots,p\} && (4.12)\\
& \sum_{\substack{j=1\\ j\neq i}}^{n} x_{ij} = 1, && \forall i \in \{1,\ldots,p\} && (4.13)\\
& u_i - u_j + n x_{ij} \leq n-1, && \forall i,j \in \{2,\ldots,p\}, i \neq j && (4.14)\\
& x_{ij} \in \{0,1\}, && \forall i,j \in \{1,\ldots,p\}, i \neq j && (4.15)\\
& u_i \in \mathbb{R}. && \forall i \in \{2,\ldots,p\} && (4.16)
\end{aligned}
$$

where c_{ij} is the cost of traveling from node i to j and x_{ij} indicates whether the salesman will travel from node i to j. The auxiliary variable u_i denotes the city that will be visited after city i.

4.3 Problem Sets

The computational experiments are conducted using problem sets from previous studies in the literature. Below is a table that summarizes the problem sets and the instances they contain. The numerical results of these experiments can be found in Tables B.1-B.11 in Appendix B. These results are summarized in Figures 4.1 to 4.11.

We have also generated our own correlated instances. These represents scenarios where objective reduction could be potentially more meaningful. The numerical results of these experiments can be found in Tables B.12 and B.13 in Appendix B. These results are summarized in Figures 4.12 and 4.13.

Problem Set	MOKP	MOAP	TSP	MOILP
[Kirlik, 2014]	$p = 3, 4, 5$	$p = 3$	-	$p = 3, 4, 5$
[Tamby and Vanderpooten, 2021]	$p = 3, 4, 5, 6$	$p = 3, 4$	-	-
[Mesquita-Cunha et al., 2022]	$p = 3, 4$	-	-	-
[Bektaş, 2018]	-	-	$p = 3, 4$	-

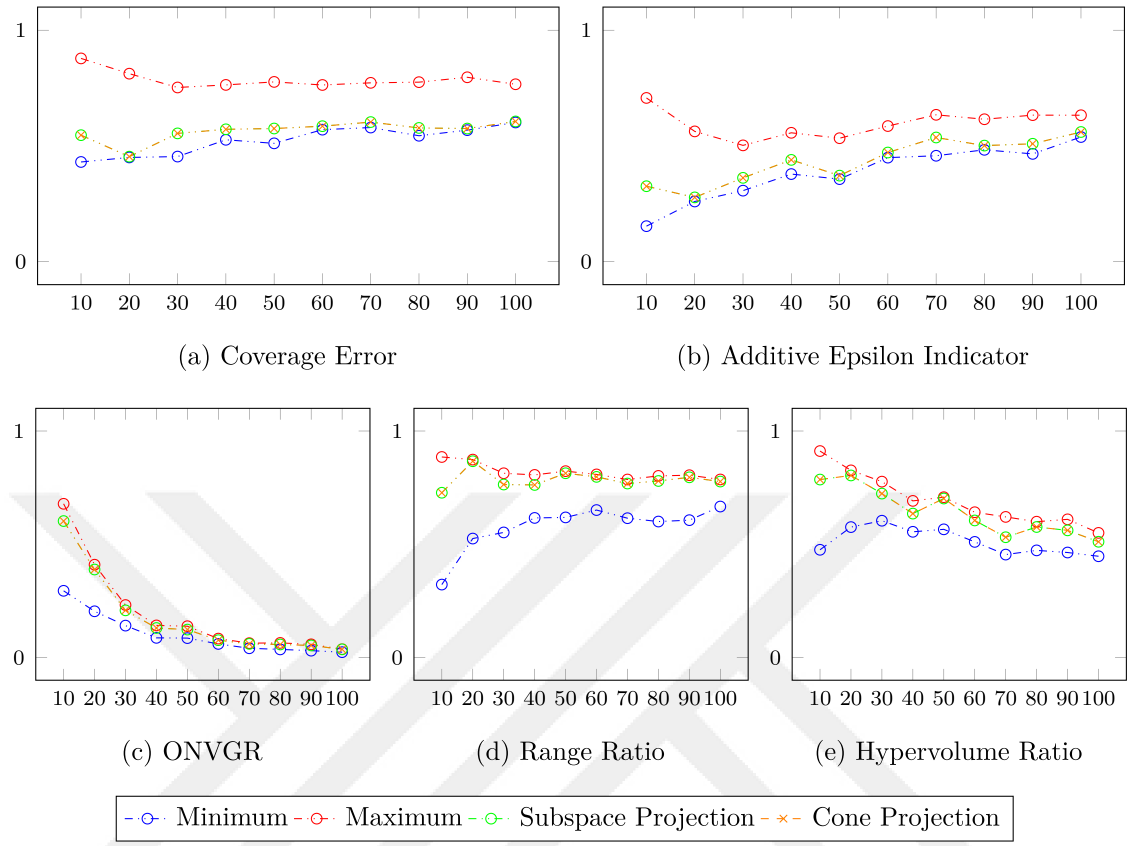
Table 4.1: Problem sets used in the computational experiments

4.3.1 [Kirlik, 2014]

In the problem instances found by [Kirlik, 2014], there are MOKP, MOAP and MOILP instances. These instances were used in [Kirlik and Sayin, 2014] and [Kirlik and Sayın, 2015].

The MOKP instances have 3, 4, 5 objectives. The number of items range from 10 to 100 for the tri-objective instances, 10 to 40 for 4 objective instances and from 10 to 20 for 5 objective instances with increments of 10. The parameters v_i^k and w_i are generated randomly from [1, 1000] interval and the weight capacity is set to nearest largest integer of the half of the total weights generated, $\lceil 0.5 \sum_{j=1}^{n} w_j \rceil$. There are 10 instances in each category. The results for the knapsack problems are given in Figure 4.1, 4.2, 4.3.

As can be seen, our method seems to perform well in this problem set. We are mostly very close to the best possible line. It is also worth mentioning that there is no difference between the subspace and cone projection for these instances. We observe that this is mostly because the ordering of the objective function indices using our cosine similarity based metric is not altered although the projections are not exactly always the same. For the instances with 3 and 4 objectives, we can observe that as the problem size increases, our method performs better. Especially, in metrics like coverage error and range ratio, the performance is very close to the best possible in large instances. For 5 objectives, the number of instances and the problem size is limited. Still, we can observe good results in ε_+ and hypervolume ratio for instances with 20 items.

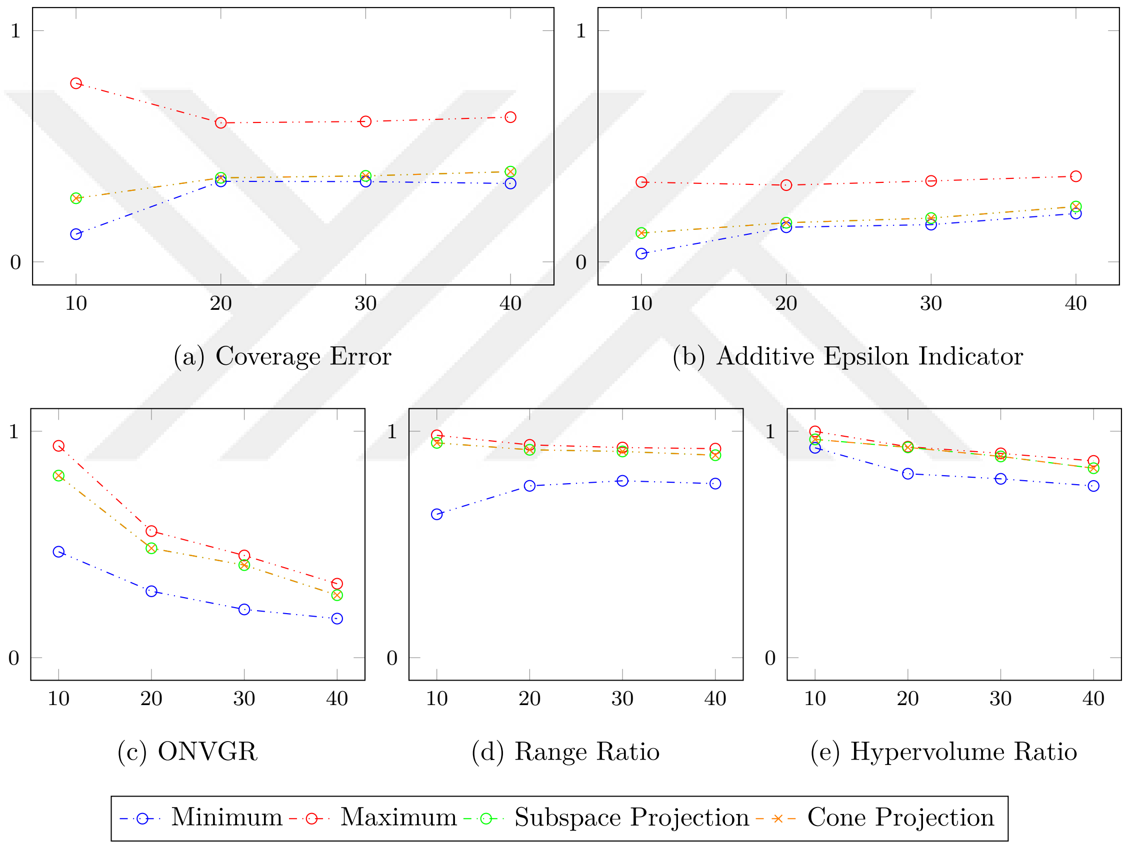


(a) Coverage Error (b) Additive Epsilon Indicator

(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

Figure 4.1: [Kirlik, 2014] MOKP instances with $p = 3$

In Figure A.1, when we aggregate all MOKP instances in [Kirlik, 2014], we can observe that the values for the quality metrics against $\cos(\theta)$ values are inline with our expectations. The corrrelation between quality metrics and the similarity measure is also high with absolute values around 0.50, as displayed in Table A.1.



(a) Coverage Error

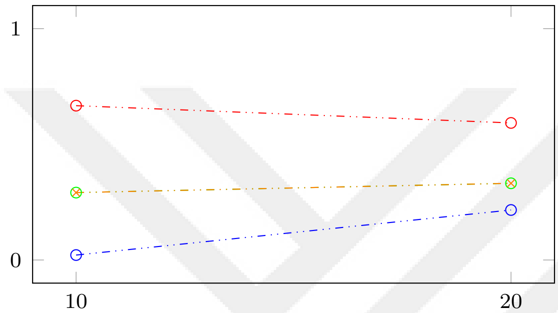
(b) Additive Epsilon Indicator

(c) ONVGR

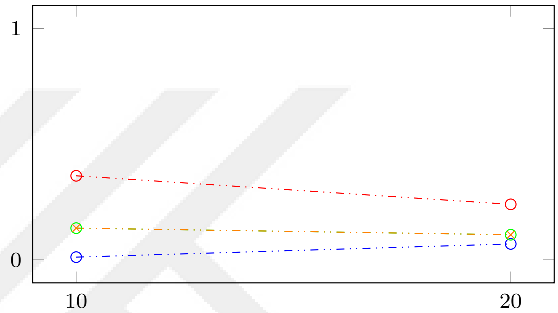
(d) Range Ratio

(e) Hypervolume Ratio

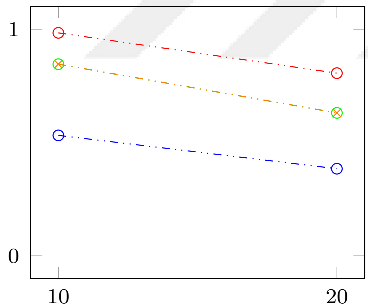
Figure 4.2: [Kirlik, 2014] MOKP instances with $p = 4$



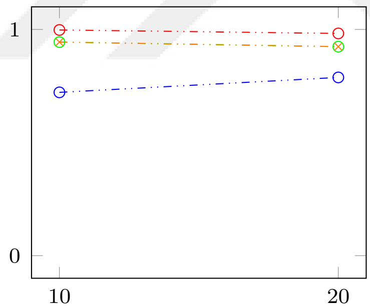
(a) Coverage Error



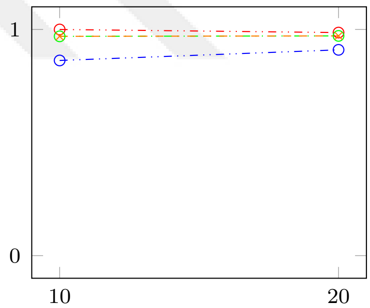
(b) Additive Epsilon Indicator



(c) ONVGR



(d) Range Ratio



(e) Hypervolume Ratio

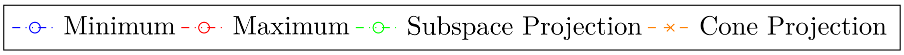


Figure 4.3: [Kirlik, 2014] MOKP instances with $p = 5$

MOAP instances have 3 objectives with the number of tasks/servers ranging from 5 to 50 with increments of 5. The objective coefficients are drawn from the interval [1, 20]. Again, each category has 10 instances. The results for the assignment problems are given in Figure 4.4.

In the MOAP instances, we cannot see the same level of performance as we saw in the MOKP instances. This can be due to the structure of the feasible region in the assignment problem or simply because there are too many nondominated points in these instances, particularly as n grows. We can also see that there is not a significant difference between the best and the worst possible performance levels. Perhaps, these instances are too symmetric due to the problem structure and data generation scheme.

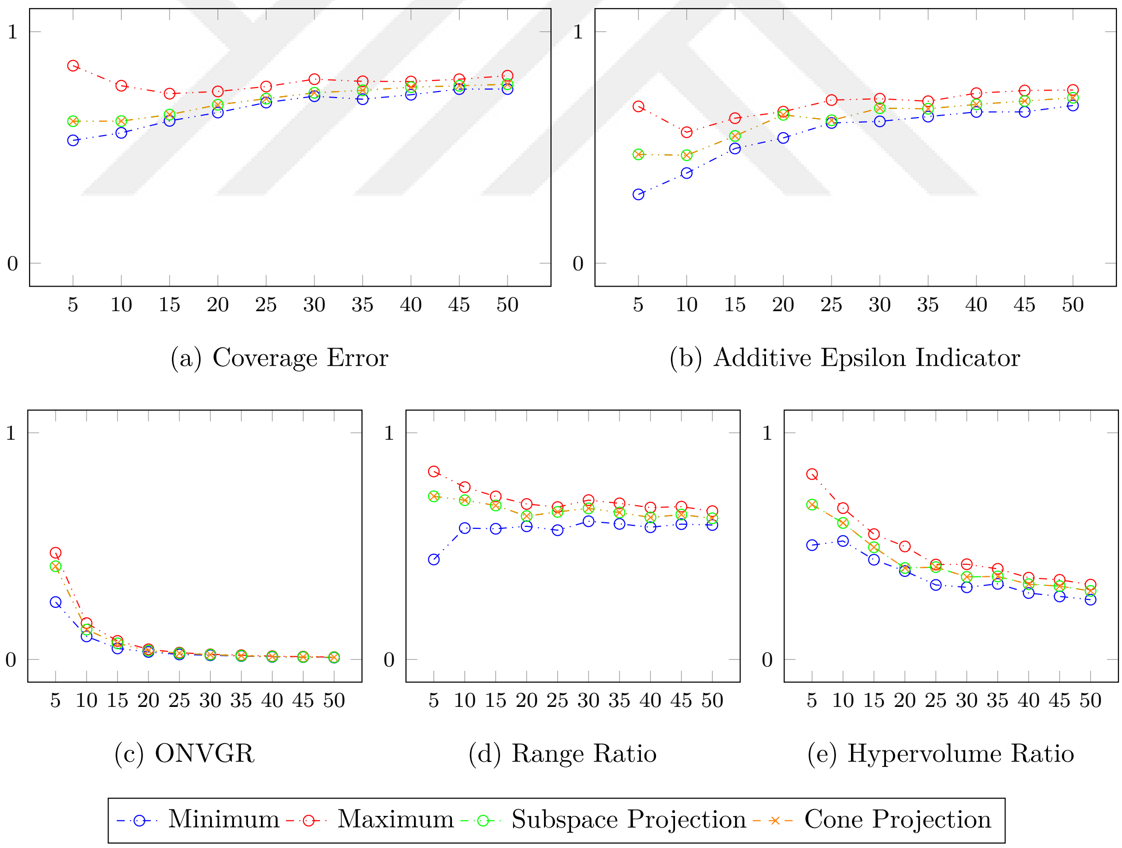


Figure 4.4: [Kirlik, 2014] MOAP instances

Figure A.2 shows no significant relationship between $\cos(\theta)$ and metric values.

The poorest correlation values are observed in this group across all problem sets, as displayed in Table A.1. This further reinforces that we require more testing to see the effect of our method in these type of instances as most points are close to each other.

In the MOILP instances with 3, 4 and 5 objectives, the number of constraints for each problem is equal to half of the number of variables for the same problem. The number of variables for the tri-objective problems range from 10 to 100 with increments of 10. Hence, the number of constraints are 5, 10, ..., 50, respectively. The number of variables for the problems with 4 and 5 objectives range from 10 to 80 and from 10 to 40, respectively. All the parameters are drawn from the interval [-100, 100]. There are 10 instances in every category. The results for the integer linear programming problems are given in Figure 4.5, 4.6, 4.7.

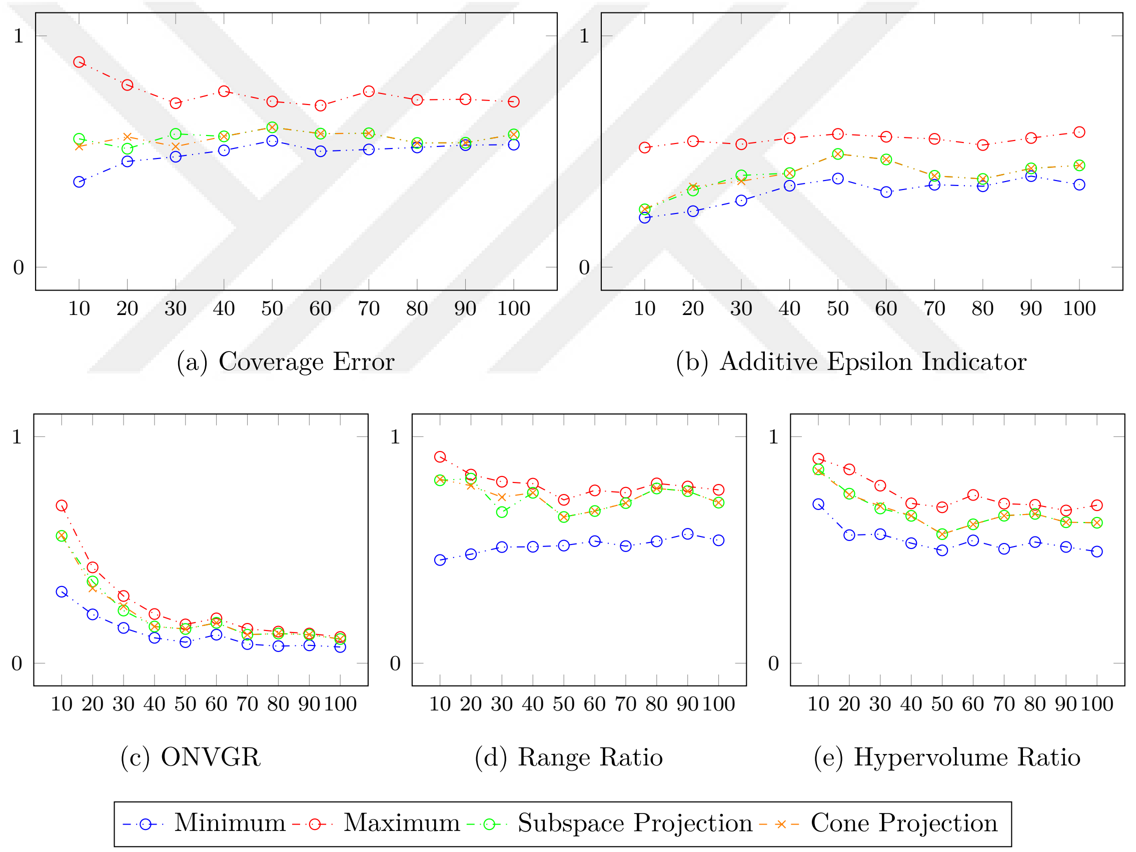


Figure 4.5: [Kirlik, 2014] MOILP instances with $p = 3$

For the first time so far, there is a difference between the performance of subspace and cone projection based methods in these instances. We see that both perform fairly well in this set. Especially in large problems, in metrics such as coverage error, approximation error and range ratio, our methods are close to the best possible lines.

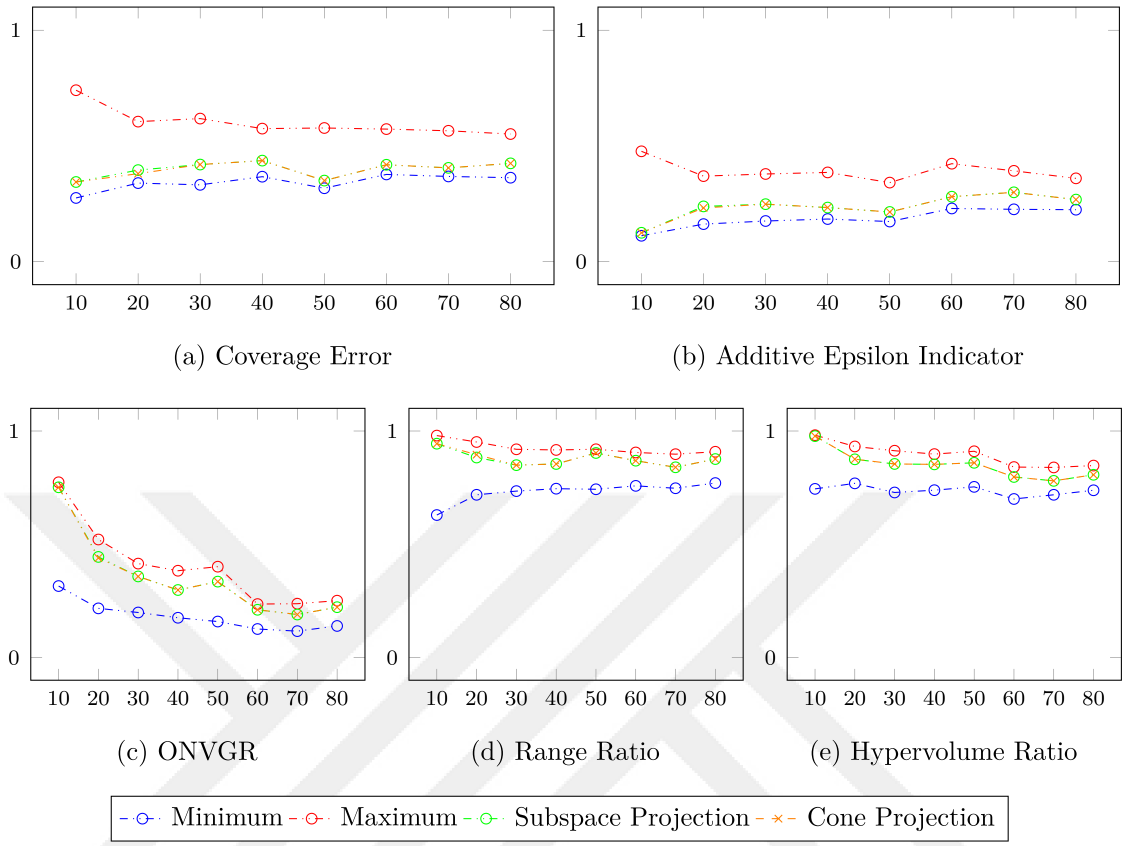
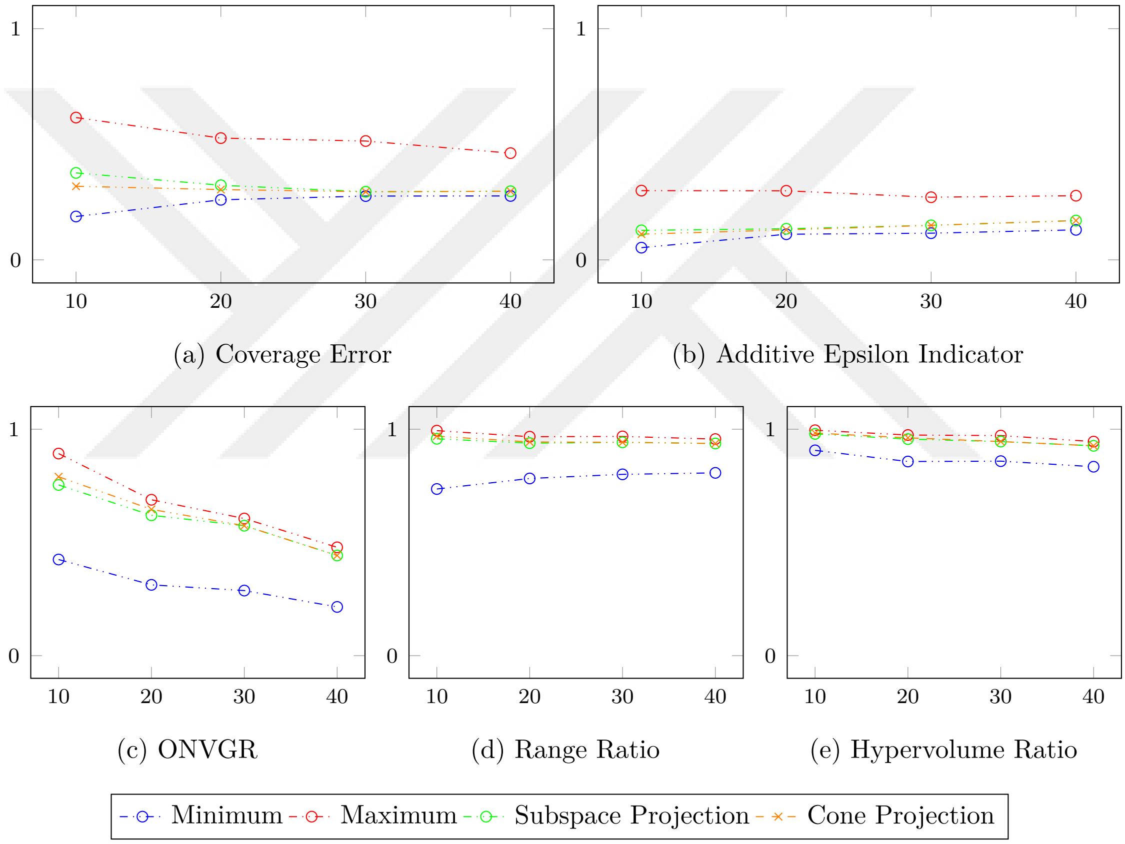


Figure 4.6: [Kirlik, 2014] MOILP instances with $p = 4$

Only for the set of instances with 5 objectives, our performance values seem to be worse compared to the previous results. This can be due to the smaller size of these instances as the number of variables are only 10 and 20 and the number of the nondomiated points are much less compared to the other instances. When we consider all the MOILP instances in [Kirlik, 2014] together, Figure A.3 indicates that the quality metrics with respect to $\cos(\theta)$ adhere to our expectations.



(a) Coverage Error

(b) Additive Epsilon Indicator

(c) ONVGR

(d) Range Ratio

(e) Hypervolume Ratio

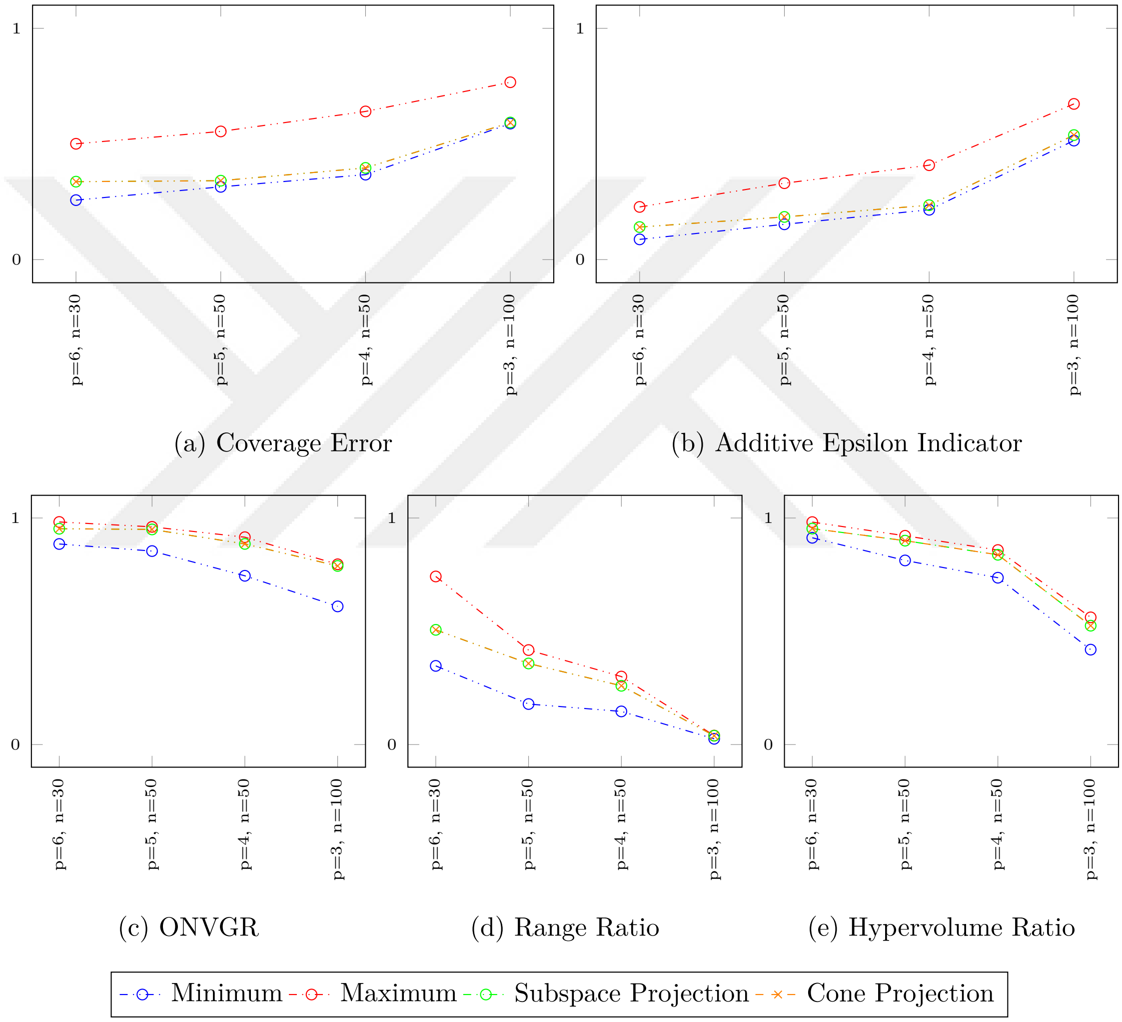
Figure 4.7: [Kirlik, 2014] MOILP instances with $p = 5$

4.3.2 [Tamby and Vanderpooten, 2021]

In the problem set provided by [Tamby and Vanderpooten, 2021], there are MOKPs and MOAPs. MOKP instances are with 3, 4, 5 and 6 objectives having 100, 50, 50, 30 items, respectively. The parameters v_i^k and w_i are generated randomly from [1, 100] interval and the weight capacity is set to nearest largest integer of the half of the total weights generated, $\lceil 0.5 \sum_{j=1}^{n} w_j \rceil$ as it was in [Kirlik and Sayin, 2014]. The results for the MOKPs are given in Figure 4.8. Similar to the problem set [Kirlik and Sayin, 2014], our performance in [Tamby and Vanderpooten, 2021] MOKP instances is strong and the result are very close to the best possible lines. Figure A.4 shows that even though the values for $\cos(\theta)$ is in a smaller range, the difference in quality of the representation can be significant. The correlation between the quality metrics and $\cos(\theta)$ is quite high in absolute values, as displayed in Table A.1.

MOAP instances have 3 and 4 objectives with 50 and 20 tasks/servers. Each parameter is drawn randomly from the interval [1, 15]. The results for the MOAPs are given in Figure 4.9.

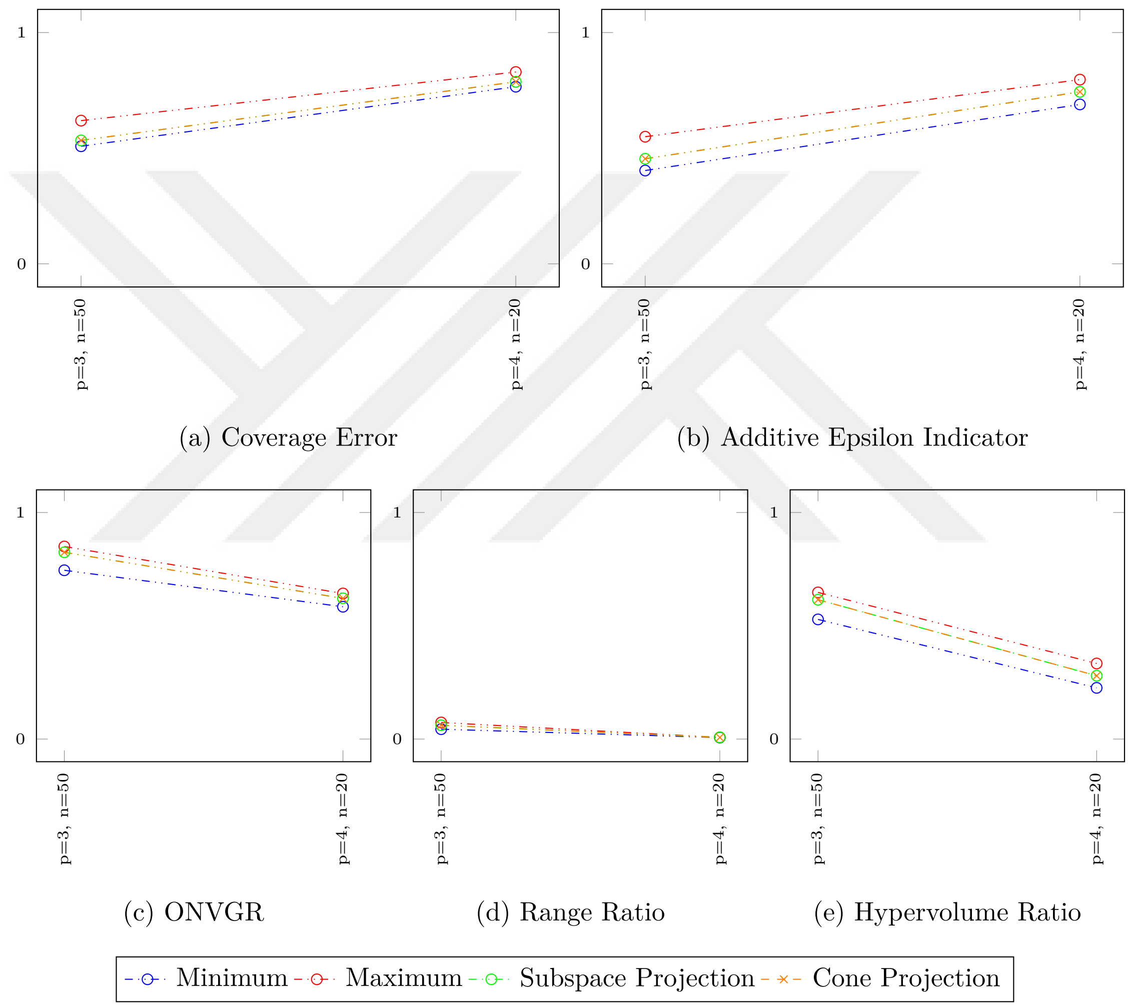
We see a similar situation in MOAPs that the best and worst case scenarios are close to each other, making it difficult to see the effectiveness of the reduction method. The performance of our approach in these instances seems to be slightly better than on [Kirlik, 2014] instances. Examining Figure A.5, we can see a strong correlation between $\cos(\theta)$ and quality metrics. This is different from the results we have seen in MOAP instances in [Kirlik, 2014]. Across all metrics, the absolute value of correlation is 0.74 and above in the desired direction, as displayed in Table A.1. This is the strongest correlation between the quality metrics and $\cos(\theta)$ across all problem sets.



(a) Coverage Error (b) Additive Epsilon Indicator

(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

Figure 4.8: [Tamby and Vanderpooten, 2021] MOKP instances



(a) Coverage Error

(b) Additive Epsilon Indicator

(c) ONVGR

(d) Range Ratio

(e) Hypervolume Ratio

Figure 4.9: [Tamby and Vanderpooten, 2021] MOAP instances

4.3.3 [Mesquita-Cunha et al., 2022]

This set of instances are referred to as MOKPs with multiple constraints. There are instances with 3 and 4 objective functions number of constraints range from 1 up to 15 with number of variables starting from only 10 and goes up to 100. There are a total of 12 different classes of problem sizes and a total of 360 problem instances. The coefficients for the objective function coefficients and the constraints are chosen randomly from [1, 100]. The results for these problems are given in Figure 4.10.

Across problems with different number of variables and different number of constraints, the performance of our approach is very consistent in these instances. We can see that the values for ONVGR is getting smaller meaning the reduced problems have much less nondominated points. Therefore, in all metrics, we are able to maintain a high quality with much smaller representations. In Figure A.6, we see that the correlation may not be as clear as the other problem sets for some quality metrics.

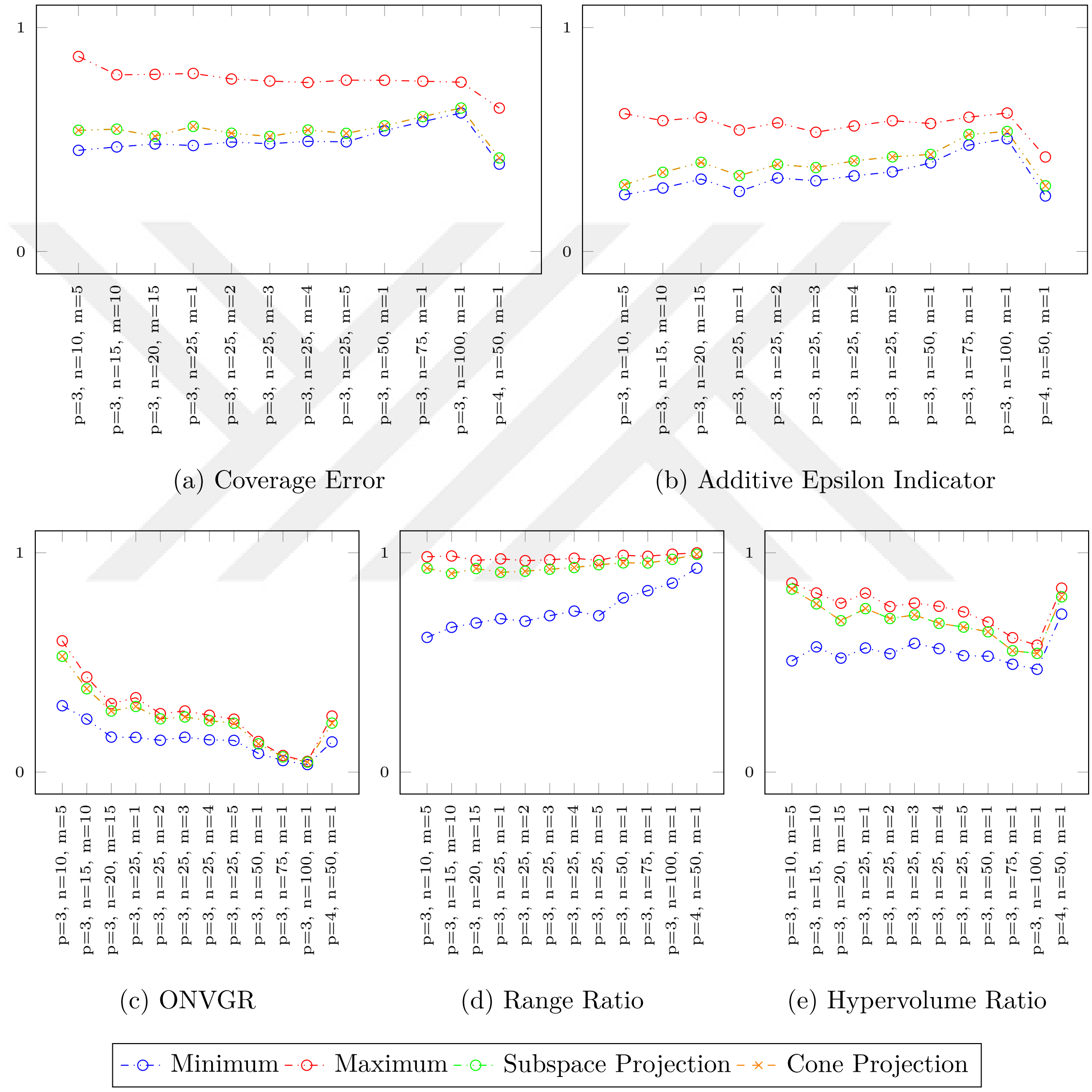
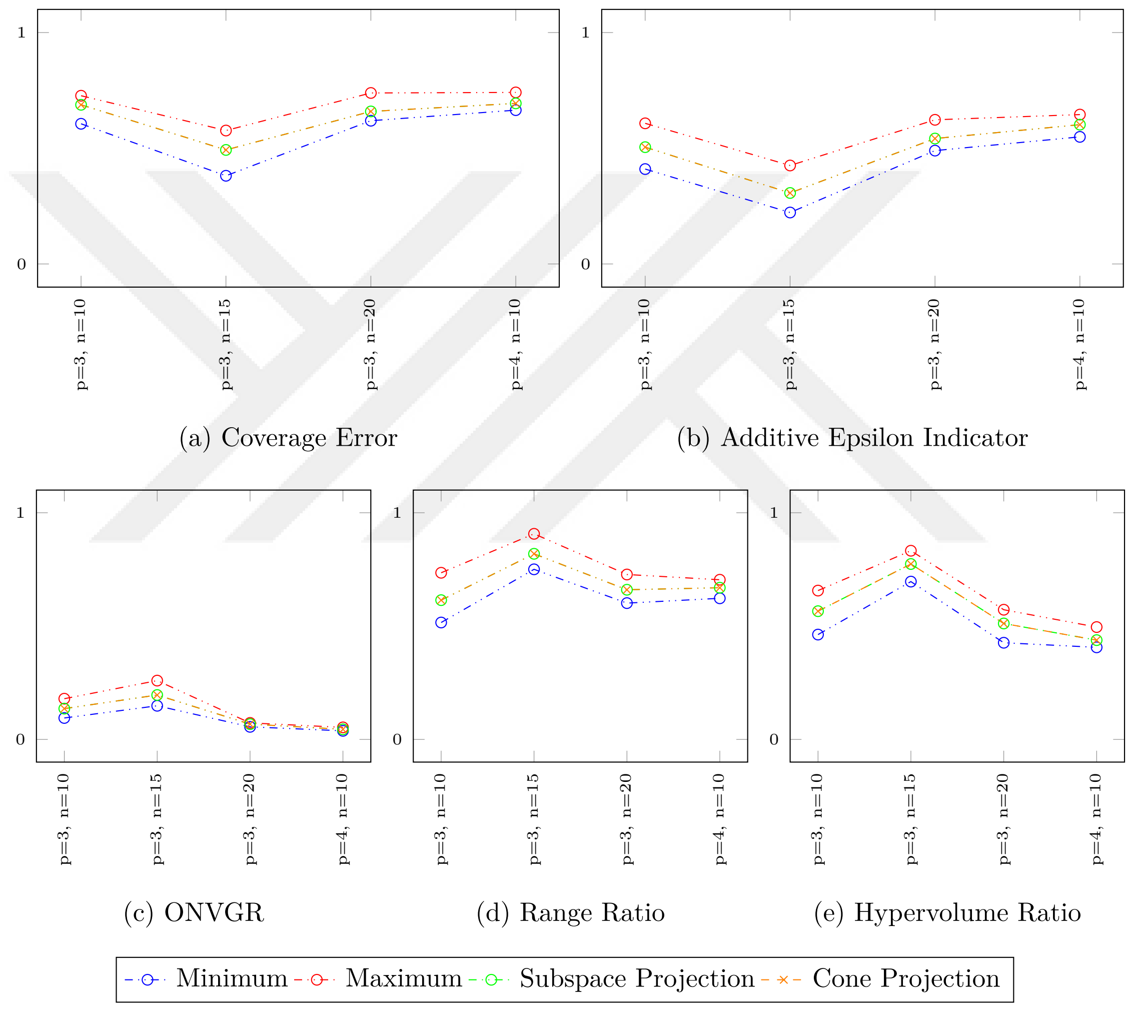


Figure 4.10: [Mesquita-Cunha et al., 2022] MOKP instances

4.3.4 [Bektaş, 2018]

There are 3 and 4 objective MOTSP instances in [Bektaş, 2018]. The tri-objective instances have 10, 15, and 20 customers and the 4 objective instances have 10 customers. Each category have 10 instances. All parameters are generated randomly from the interval [1, 20]. The results for the MOTSPs are given in Figure 4.11.

The results are similar to what we have seen with the assignment problems. The best and worst case scenarios are close to each other. Also, the size of the problems are small making it difficult to draw conclusion as opposed to previous problem types. Though, we can see that in larger problems ($p = 3, n = 20$ and $p = 4, n = 10$) coverage error is slightly closer to the best possible case. From Figure A.7, we can tell that the quality metrics follow the expected trend against $\cos(\theta)$. Nonetheless, there is need for further testing with larger sized problems.



(a) Coverage Error

(b) Additive Epsilon Indicator

(c) ONVGR

(d) Range Ratio

(e) Hypervolume Ratio

Figure 4.11: [Bektaş, 2018] MOTSP instances

4.4 Generating Correlated Problem Instances

In this section, we investigate the performance of our objective reduction method under the presence of correlated objective functions. This correlation often implies a degree of similarity between the objectives. We aim to simulate scenarios where objective reduction are potentially beneficial.

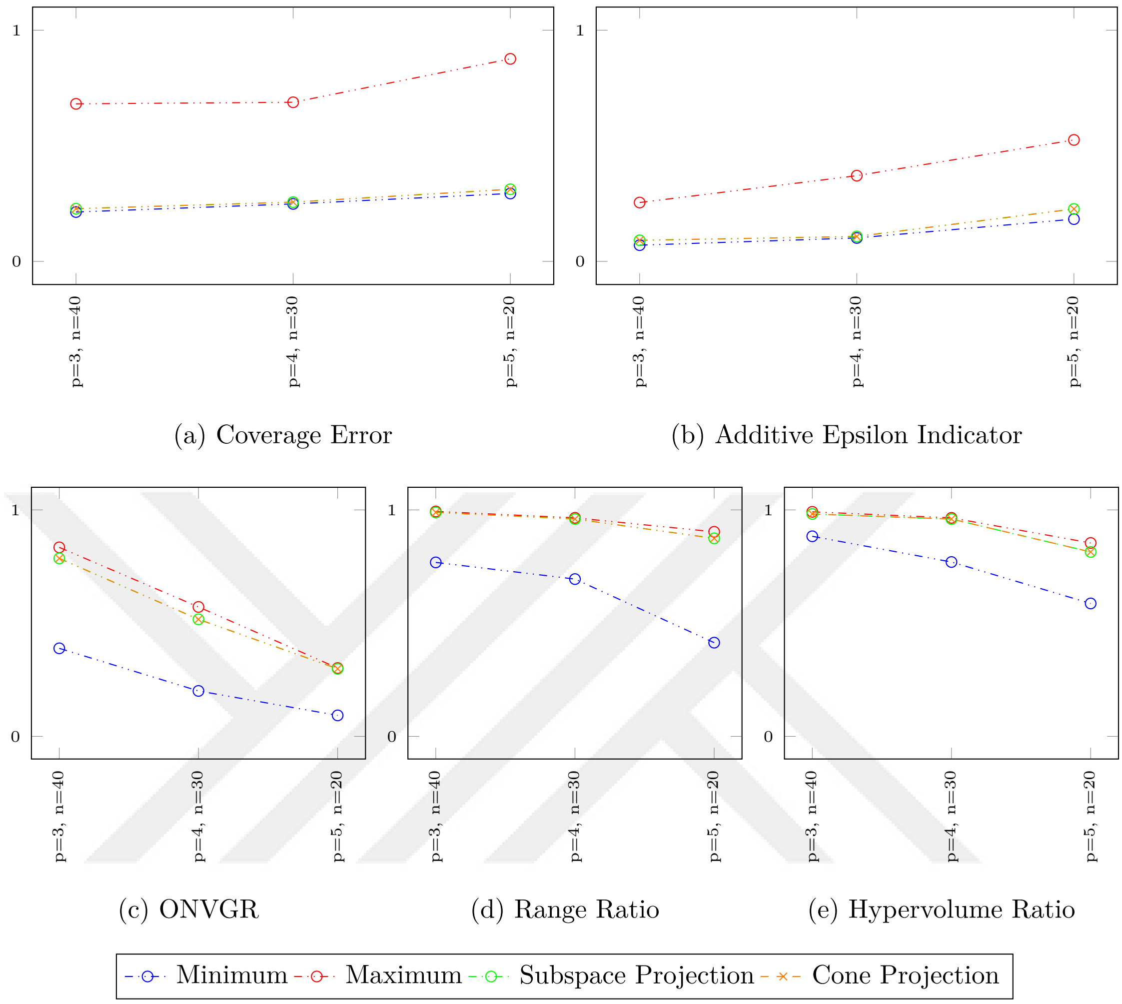
In order to generate instances with correlated objectives at a desired level, we have used an approach based on Cholesky decomposition. That way, we can remove any existing correlation in a randomly generated matrix and add a desired level to the matrix at hand.

Let A be a symmetric $n \times n$ matrix. A is said to have an LU decomposition if it can be written as $A = LU$. A is positive definite if and only if $x^T Ax > 0, \forall x \in \mathbb{R}^n, x \neq 0$. We can also say that A is symmetric positive definite if and only if all its eigenvalues are strictly positive. In the case where A is symmetric positive definite, the LU decomposition for A has a special structure: $A = LL^T$ is possible where L is a lower triangular matrix and all of the diagonal entries of L is strictly positive.

In order to generate an objective coefficient matrix with a desired level of correlation, we start by creating a randomly generated $(p \times n)$ matrix, C. We can first uncorrelate C by a reverse transformation. Let $R = corr(C)$ be the correlation between rows of C and R' be the matrix of desired levels of correlation. We can apply Cholesky decomposotion to $R = LL^T$ and uncorrelate C by multiplying L^{-1} with C. After that, we can introduce correlation by applying Cholesky decomposition to $R' = L'L'^T$. Then, the new objective coefficient matrix would be $C' = L'L^{-1}C$. Since we want the elements of C' to be integers, we round the values of the coefficients to the nearest integer and obtain $C' = \lfloor L'L^{-1}C \rceil$.

We have two classes of knapsack problems that we have generated using the method explained above. For both classes, there are three groups of instances with $(p = 3, n = 40), (p = 4, n = 30)$ and $(p = 5, n = 20)$. Each group has 10 instances in both classes. All the coefficients and w_i values are generated randomly from the interval [1, 1000] and the capacity of the knapsack is set to $\lfloor 0.5 \sum_{j=1}^{n} w_j \rfloor$

For scenarios where only one pair of objective functions are correlated, we have



(a) Coverage Error (b) Additive Epsilon Indicator

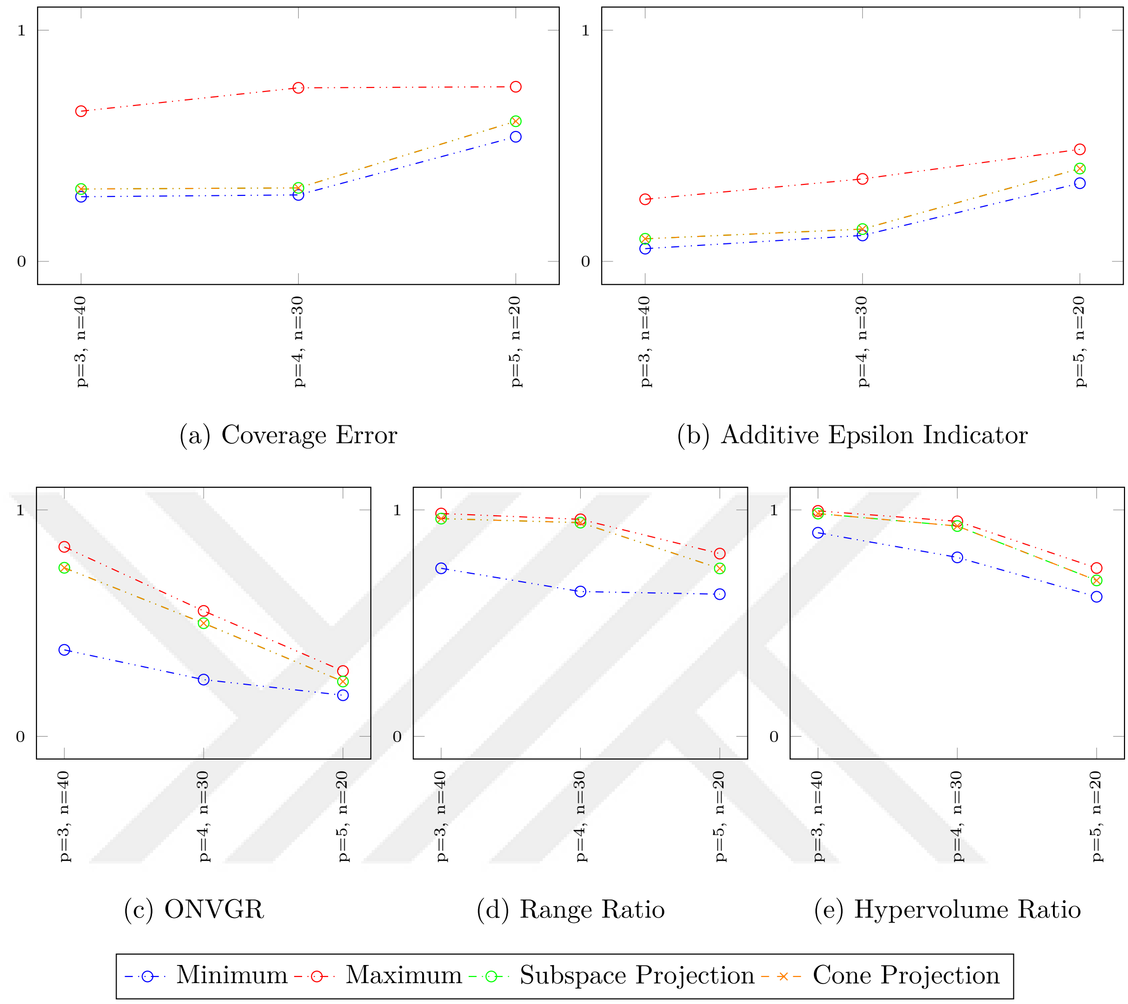
(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

Figure 4.12: Class-I correlated MOKP instances

generated a set of Class-I correlated instances. For these problems with 4 objectives, the targeted correlation matrix is given below:

$$R' = \begin{bmatrix} 1 & 0 & 0 & .8 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ .8 & 0 & 0 & 1 \end{bmatrix}$$

For scenarios where an objective function is correlated with two other objectives, we have generated a set of Class-II correlated instances. For these problems with 4 objectives, the targeted correlation matrix is given below:



(a) Coverage Error (b) Additive Epsilon Indicator

(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

Figure 4.13: Class-II correlated MOKP instances

$$R' = \begin{bmatrix} 1 & 0 & 0 & .5 \\ 0 & 1 & 0 & .5 \\ 0 & 0 & 1 & 0 \\ .5 & .5 & 0 & 1 \end{bmatrix}$$

The results for the correlated instances are given in Figures 4.12 and 4.13 and it seems that the outcome align with our expectation since the results are quite good and very close the the best case scenarios. We can also see that the results are similar to the knapsack instances from library instances. Therefore, our method is effective in both correlated and uncorrelated data generation schemes. Another point that should be mentioned is that the two plots Class-I and Class-II instances are similar to each other. We can see a slight divergence from the best case scenario in Class-II compared to Class-I. This should be expected since we have two objectives that are highly correlated in the Class-I instances which can be considered as alternatives to each other in an objective reduction scenario. In Class-II, there are two moderately correlated pairs of objective functions. Although the correlation is not as strong as Class-I, since we remove correlation from all the other pairs, some reductions are clearly better than others. When we look at the relationship between $\cos(\theta)$ and the quality metrics across both classes of instances in Figure A.8, we can conclude that they align with our expectations.

Overall, the performance of our objective reduction method is quite strong across a wide range of problems classes. We can now use this approach to reduce the number of objective functions and obtain a high quality representation. In the next chapter, we will use the nondominated points in the representation obtained from the reduced problem to warm start the solution procedure for the original problem.

Chapter 5

INITIALIZATION OF SEARCH REGION USING PROJECTION OF THE NONDOMINATED SET

In this chapter, we will describe a search space reduction method in which we use the nondominated points obtained by solving the reduced problem that we addressed previously. Each of these nondominated points can be used as a bound on a subset of the nondominated set of the original problem in the $(p-1)$-dimensional space. Bringing the regions induced by these bounds together, we can come up with a more specific description of the search region of the original problem. This way we can eliminate parts of the search region that does not contain any nondominated points and hence can allow for better initialization of the search procedure.

5.1 Properties of the Nondominated Points of the Reduced Multi-objective Problem

Recall that $\bar{\mathcal{Y}}_N^q$ to be the set of nondominated points obtained from solving the reduced multi-objective problem obtained by deleting the objective function f_q. If we use the ε-constraint scalarization described in Section 3.6, $\bar{\mathcal{Y}}_N^q \subseteq \mathcal{Y}_N$ will hold true. We will now show a special property of the points in $\bar{\mathcal{Y}}_N^q$.

Proposition 5.1.1 *Let* $q \in \{1, \ldots, p\}$. $\forall y \in \mathcal{Y}_N \backslash \bar{\mathcal{Y}}_N^q$, $\exists y' \in \bar{\mathcal{Y}}_N^q$ *such that* $y'_{-q} \leq y_{-q}$.

Proof. Let $f(\tilde{x}) = \tilde{y} \in \mathcal{Y}_N$ and $\tilde{y} \notin \bar{\mathcal{Y}}_N^q$. Also, let $\varepsilon_j = \tilde{y}_j, \forall j \in \{1, \ldots, p\}$ and $j \neq q$. Then, $\tilde{y}$ is an optimal solution to $(P_q(\varepsilon), Q_q(\varepsilon))$.

Let $k \in \{1, \ldots, p\}$ and $k \neq q$. Consider the two stage problem $(\bar{P}_k^q(\varepsilon_{-k}), \bar{Q}_k^q(\varepsilon_{-k}))$ with optimal solution $f(x^*) = y^*$. Then, $\tilde{x}$ is feasible for $(\bar{P}_k^q(\varepsilon_{-k}), \bar{Q}_k^q(\varepsilon_{-k}))$. Since x^* is optimal and $\tilde{x}$ is feasible for $(\bar{P}_k^q(\varepsilon_{-k}), \bar{Q}_k^q(\varepsilon_{-k}))$, $\sum_{\substack{j=1 \\ j\neq q}}^p y_j^* \leq \sum_{\substack{j=1 \\ j\neq q}}^p \tilde{y}_j$, which implies $y_{-q}^* \leqq \tilde{y}_{-q}$. Since $y^* \in \bar{\mathcal{Y}}_N^q$ and $\tilde{y} \notin \bar{\mathcal{Y}}_N^q$, $y^* \neq \tilde{y}$. Therefore, $y_{-q}^* \leq \tilde{y}_{-q}$. $\square$

Proposition 5.1.1 implies that, for every nondominated point in $\mathcal{Y}_N$ that is not an element in $\bar{\mathcal{Y}}_N^q$, we can find a nondominated point from $\bar{\mathcal{Y}}_N^q$ which dominates that point in the projected search space. We can use this information to cut parts of the search region enclosed by y^I and y^N (y^U if nadir point is not available).

We will use a similar notation from [Tamby and Vanderpooten, 2021, Klamroth et al., 2015, Dächert et al., 2017] to describe parts of the search region and let $D_{-q}(y) = \{\bar{y} \in \mathbb{R}^{p-1} | y_{-q} \leqq \bar{y}\}$. Using this notation, we can conclude that $\forall y \in \mathcal{Y}_N, \exists y' \in \bar{\mathcal{Y}}_N^q$ such that $y_{-q} \in D_{-q}(y')$. Hence, $\{y_{-q} | y \in \mathcal{Y}_N\} \subseteq \cup_{y' \in \bar{\mathcal{Y}}_N} D_{-q}(y')$.

Proposition 5.1.2 *The* $(p-1)$*-dimensional search space for an MOILP problem can be fully described by* $\cup_{y' \in \bar{\mathcal{Y}}_N} D_{-q}(y')$ *where* $q \in \{1, \ldots, p\}$.

Proof. Proof follows from 5.1.1. $\forall y \in \mathcal{Y}_N \backslash \bar{\mathcal{Y}}_N^q$, $\exists y' \in \bar{\mathcal{Y}}_N^q$ such that $y_{-q} \in D_{-q}(y')$. Therefore, $(\mathcal{Y}_N \backslash \bar{\mathcal{Y}}_N^q) \subseteq \cup_{y' \in \bar{\mathcal{Y}}_N^q} D_{-q}(y')$. By definition, $\forall y' \in \bar{\mathcal{Y}}_N^q, y'_{-q} \in D_{-q}(y')$. Hence, $\mathcal{Y}_N \subseteq \cup_{y \in \bar{\mathcal{Y}}_N^q} D_{-q}(y)$. $\square$

5.2 Description of the $(p-1)$*-Dimensional Search Region*

Using the previous results, we can now come up with a description for the search region for an algorithm that works in the $(p-1)$-dimensional search space.

Recall the definition of a $(p-1)$-dimensional box from Section 2.2, $\mathcal{B}^{p-1}(l, u) = \{y \in \mathbb{R}^{p-1} | l \leqq y \leqq u\}$. As explained in Section 2.5, one of the critical components of a multi-objective discrete optimization algorithm is the management and representation of the search space. As depicted in Figure 5.1, the subregions $D_{-q}(y)$ are overlapping introducing redundancies, which may lead to searching the same region more than once and finding repeated solutions. In order to eliminate this, we can convert the union of these regions to a union of disjoint boxes. For this, we first need to sort the nondominated points in $\bar{\mathcal{Y}}_N^q$ lexicographically.

Definition 5.2.1 (Lexicographic Order) *Consider two points* $y^1, y^2 \in \mathbb{R}^p$. $y^1 <_{lex} y^2$ *if* $y_i^1 < y_i^2$, *where* $i = \min\{t \in \{1, \ldots, p\} | y_t^1 \neq y_t^2\}$.

We will now define a lexicographic partitioning approach that is compatible with lexicographic order.

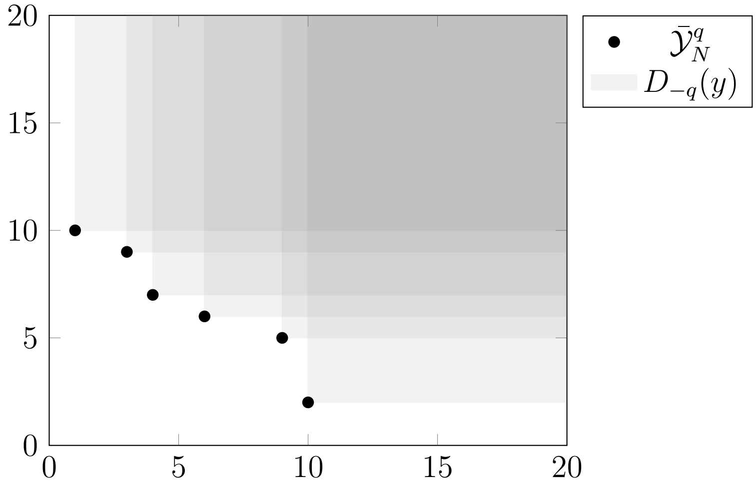


Figure 5.1: Search Region described by $\cup_{y \in \bar{\mathcal{Y}}_N^q} D_{-q}(y)$ in $\mathbb{R}^2$ for $p = 3$

Definition 5.2.2 (Lexicograhpic Partition) *For a box $\mathcal{B}^p(l, u)$ and some $y \in \mathbb{R}^p$, let $\bar{\mathcal{B}}_i^p(l^i, u^i) = \{y' \in \mathcal{B}^p(l, u) | y'_i < y_i \wedge y'_j \geq y_j, \forall j < i\}$, where $i \in \{1, \ldots, p\}$ and $\bar{\mathcal{B}}_0^p(y, u) = \{y' \in \mathcal{B}^p | y \leq y'\}$. Note that for $i > 1$, $l^i = (\hat{y}_1, \ldots, \hat{y}_{i-1}, l_i, l_{i+1}, \ldots, l_p)$ and $u^i = (u_1, \ldots, u_{i-1}, \hat{y}_i, u_{i+1}, \ldots, u_p)$ where $\hat{y}_i = \max(y_i, l_i), \forall i \in \{1, \ldots, p\}$.*

This partitioning is similar to the one introduced in [Domínguez-Ríos et al., 2021]. The differences can be seen in Figure 5.2. We can see the partitioning of the points that are not dominated by y as red, blue and green boxes with directions 1, 2, and 3, respectively. In lexicographic partition, we set all the points that are lower in their direction to a new partition. On the other hand, p-partition creates each partition with the points that have a better value in their direction and worse values in subsequent directions. For example, the red box in p-partition contains the points that are not dominated by y and that has a lower value in first objective and higher values for the 2^{nd} and 3^{rd} objectives. The red box in lexicographic partition contains the points that are not dominated by y and that has a lower value in the first objective.

Proposition 5.2.1 *$\bar{\mathcal{B}}_i^p(l^i, u^i) \cap \bar{\mathcal{B}}_j^p(l^j, u^j) = \emptyset, \forall i, j \in \{1, \ldots, p\}, i \neq j$.*

Proof. Let $y' \in (\bar{\mathcal{B}}_i^p(l^i, u^i) \cap \bar{\mathcal{B}}_j^p(l^j, u^j))$ where $i \neq j$ and $i < j$. Then, since $y' \in \bar{\mathcal{B}}_j^p(l^j, u^j)$, $y'_j < y$ and $y'_i \geq y_i$, since $i < j$. However, it is also true that $y'_i < y_i$, since

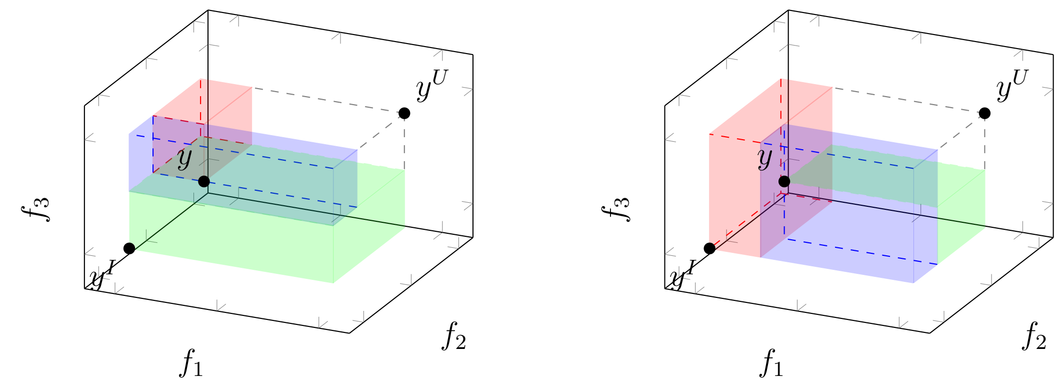


Figure 5.2: Left: p-partition, Right: lexicographic partition

$y' \in \bar{\mathcal{B}}^p_i(l^i, u^i)$. This is a contradiction and, therefore, $(\bar{\mathcal{B}}^p_i(l^i, u^i) \cap \bar{\mathcal{B}}^p_j(l^j, u^j)) = \emptyset$. □

Proposition 5.2.2 $\cup_{j=0}^{p} \bar{\mathcal{B}}^p_j(l^j, u^j) = \mathcal{B}^p(l, u)$

Proof. We first need to show that $\cup_{j=0}^{p} \bar{\mathcal{B}}^p_j(l^j, u^j) \subseteq \mathcal{B}^p$. Let $y' \in \cup_{j=0}^{p} \bar{\mathcal{B}}^p_j(l^j, u^j)$. Then, by definition $y' \in \mathcal{B}^p$. Next, we need to show that $\mathcal{B}^p \subseteq \cup_{j=0}^{p} \bar{\mathcal{B}}^p_j(l^j, u^j)$. Let $y' \in \mathcal{B}^p$. If $y' \geqq y$, then $y' \in \bar{\mathcal{B}}^p_0$. Otherwise, let i be the first index that satisfies $y'_i < y_i$. Then, $y' \in \bar{\mathcal{B}}^p_i(l^i, u^i)$. Thus, $\cup_{j=0}^{p} \bar{\mathcal{B}}^p_j(y) = \mathcal{B}^p$. □

We can show that the lexicographic partitioning creates a valid partition.

Proposition 5.2.3 *For a box $\mathcal{B}^p(l, u)$, the lexicographic partitions of $\mathcal{B}^p$ created using a point $y \in \mathcal{B}^p$, $\{\bar{\mathcal{B}}^p_j(l^j, u^j)\}_{j=0}^{p}\}$, forms a valid partition.*

Proof. Proof follows from Proposition 5.2.1 and Proposition 5.2.2. □

Note that, in $(p-1)$-dimensional space, the same partitioning can be performed for $\mathcal{B}^{p-1}(l, u)$ and $y_{-q} \in \mathcal{B}^{p-1}(l, u)$.

This partitioning method creates boxes with no overlap and that fully describes the $(p-1)$-dimensional search region. This is depicted in Figure 5.3, where we start with $\mathcal{B}^{p-1}(y^I_{-q}, y^N_{-q})$. As we iterate through the points in $\bar{\mathcal{Y}}^q_N$ in lexicographic order, we use lexicographic partitioning to construct our disjoint boxes. After we perform a lexicographic partitioning, we set $\bar{\mathcal{B}}^{p-1}_0(y_{-q}, u)$ aside, seen as the green regions in Figure 5.3, as these are the partition that contain the nondominated points that are dominated by the points in $\bar{\mathcal{Y}}^q_N$ in $(p-1)$-dimensional space. We keep the dotted regions for the remainder of the initialization procedure to create more green boxes for the remaining points in $\bar{\mathcal{Y}}_N^q$. Lexicographic sorting and lexicographic partitioning allows us to create the green regions in Figure 5.3 in a more structured manner since, otherwise, we might have initialized the search region disproportionately, which might not be as efficient.

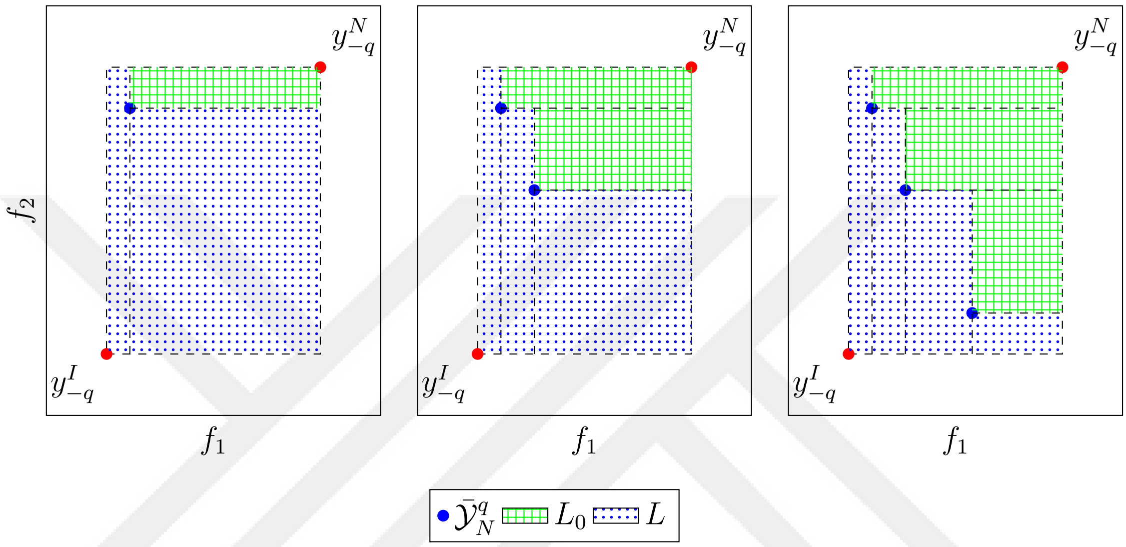


Figure 5.3: Initialization using $\bar{\mathcal{Y}}_N^q$ with lexicographic partitioning

Similar to [Domínguez-Ríos et al., 2021], after we perform a lexicographic partitioning operation, some boxes obtained maybe empty. Therefore, we can remove these empty boxes that have zero volume. We can compute the volume for $\bar{\mathcal{B}}_j^{p-1}(l^j, u^j)$ as $V_j = \prod_{\substack{i=1 \\ i \neq q}}^{p}(u_i^j - l_i^j), \forall j \in \{1, \ldots, p\}, j \neq q$.

5.3 Initialization of the search space

Using the ideas we have laid out in the previous section, we can now come up with an initialization procedure for the search region of an MOILP. This search region description can be used in an MOILP algorithm that operates on the $(p-1)$-dimensional search space.

We start with the region enclosed by y^I_{-q} and y^U_{-q}. Using the lexicographicly sorted nondominated points in $\bar{\mathcal{Y}}_N^q$, we split the search region into partitions and keep the ones that describe the region dominated by the points in $\bar{\mathcal{Y}}_N^q$ in $(p-1)$-dimensional space. The steps are provided in Algorithm 5.3.1.

Algorithm 5.3.1 Initialization of Search Region using $\bar{\mathcal{Y}}_N^q$

Require: y^I, y^U, $\bar{\mathcal{Y}}_N^q$

Ensure: $\{\bar{\mathcal{B}}_0^{p-1}(y_{-q})\}_{y\in\bar{\mathcal{Y}}_N^q}$

$\bar{\mathcal{Y}}_N^q \leftarrow$lexicographic_sort$(\bar{\mathcal{Y}}_N^q)$

$L \leftarrow \{\mathcal{B}^{p-1}(y^I_{-q}, y^U_{-q})\}$

$L_0 \leftarrow \emptyset$

for $y^j \in \bar{\mathcal{Y}}_N^q$ **do**

  for $\mathcal{B}^{p-1}(l,u) \in L$ **do**

    if $y_{-q} \leq u$ **then**

      $\hat{y}_i \leftarrow \max(y_i^j, l_i), \quad \forall i \in \{1,\dots,p\}, i \neq q$

      $\{\bar{\mathcal{B}}_i^{p-1}(l^i,u^i)\}_{\substack{i=0\\ i\neq q}}^{p} \leftarrow$lexicographic_partition$(\mathcal{B}^{p-1}(l,u), \hat{y}_{-q})$

      $L_0 \leftarrow L_0 \cup \{\bar{\mathcal{B}}_0^{p-1}(\hat{y}_{-q}, u)\}$

      $L \leftarrow L \cup \{\bar{\mathcal{B}}_i^{p-1}(l^i,u^i)\}_{\substack{i=1\\ i\neq q\\ V_i\neq 0}}^{p}$

      $L \leftarrow L \backslash \{\mathcal{B}^{p-1}(l,u)\}$

    end if

  end for

end for

return L_0

Example: [Kirlik and Sayin, 2014] In [Kirlik and Sayin, 2014], a $(p-1)$-dimensional rectangle, which is used to express part of a search space, is defined as $\mathcal{B}^{p-1}(l,u) = \{y \in \mathbb{R}^{p-1} | l \leqq y \leqq u\}$ where $l, u \in \mathbb{R}^{p-1}$. We can use the output of Algorithm 5.3.1 as the initial set of rectangles in [Kirlik and Sayin, 2014]. Let $k \in \{1,\dots,p\}$ be the index of the objective function chosen for the two stage model of the algorithm. With the way the rectangle list is maintained in the original algorithm, at each iteration, a search will occur using a rectangle $\mathcal{B}^{p-1}(l,u)$ and one of the following will happen, further reducing the search space:

1. A new nondominated solution y^* will be found and the rectangle $\mathcal{B}^{p-1}(l,u)$ along with every other rectangle that satisfy $l_i < y_i^* < u_i$ will be split across the dimension i and every rectangle $\mathcal{B}^{p-1}(l',u')$ satisfying $\mathcal{B}^{p-1}(l',u') \subseteq \mathcal{B}^{p-1}(y^*_{-k},u)$ will be discarded.

2. An existing nondominated solution y^* will be found and the rectangle $\mathcal{B}^{p-1}(l,u)$ along with every other rectangle that satisfy $l_i < y_i^* < u_i$ will be split across the dimension i and every rectangle $\mathcal{B}^{p-1}(l',u')$ satisfying $\mathcal{B}^{p-1}(l',u') \subseteq \mathcal{B}^{p-1}(y^*_{-k},u)$ will be discarded.

3. The first stage problem will be infeasible and every rectangle $\mathcal{B}^{p-1}(l',u')$ satisfying $\mathcal{B}^{p-1}(l',u') \subseteq \mathcal{B}^{p-1}(y^I_{-k},u)$ will be discarded.

Next, we can show that the initialization will eliminate parts of the search region that does not contain any nondominated points and will avoid solving infeasible models.

Proposition 5.3.1 *Using the initialization method given in Algorithm 5.3.1, [Kirlik and Sayin, 2014] algorithm will not solve any infeasible two-stage models.*

Proof. For any box $\bar{\mathcal{B}}_0^{p-1}(l,u)$ from the output L_0 of Algorithm 5.3.1, $\exists \hat{y} \in \bar{\mathcal{Y}}_N^q$ such that $\hat{y}_{-q} \leq u$. Then, we can conclude that, for any search zone, $\bar{\mathcal{B}}_0^{p-1}(l,u)$ obtained by splitting the initial partitions, $\exists \hat{y} \in \bar{\mathcal{Y}}_N^q$ such that $\hat{y}_{-q} \leq u'$. Then, $\hat{y}$ is a feasible point for $(P_k(u), Q_k(u))$. Since we know that $\hat{y}$ is nondominated, $(P_k(u), Q_k(u))$ cannot be infeasible. □

We have used the points in $\bar{\mathcal{Y}}_N^q$ to initialize the solution process for the original problem with p objectives. This requires solving the reduced problem obtained by removing objective q. We will now expand this idea to initialize the search region for not only the original problem with p objectives but also the reduced problem with $(p-1)$ objectives. By recursively applying the same idea, we can go down further and start the initialization using a bi-objective problem.

5.4 Recursive Initialization of the Search Region

For problems with many objectives, we can use the nondominated points of the reduced problem to initialize the search region. A follow up question might be to initialize the search region of the reduced problem by reducing it again. If possible, we can go all the way down to a bi-objective reduced problem and initialize the search region for an MOILP with $p \geq 3$. For $\Psi \subseteq \{1, \dots, p\}$, let $y_\Psi = (y_j)_{j \in \Psi}$.

Proposition 5.4.1 *Let $\Psi \subseteq \{1, \dots, p\} \wedge |\Psi| > 1$. Consider the following two-stage ε-constraint model where $k \in \Psi$.*

$$\begin{aligned} \bar{P}_k^\Psi(\varepsilon) \qquad & \min \quad y_k^* = f_k(x) \\ & \textit{subject to:} \quad f_j(x) \leq \varepsilon_j, \quad j \in \Psi, \quad j \neq k \\ & \qquad\qquad x \in \mathcal{X} \end{aligned} \tag{5.1}$$

$$\begin{aligned} \bar{Q}_k^\Psi(\varepsilon) \qquad & \min \quad \sum_{j=1}^{p} f_j(x) \\ & \textit{subject to:} \quad f_j(x) \leq \varepsilon_j, \quad j \in \Psi, \quad j \neq k \\ & \qquad\qquad f_k(x) = y_k^*, \\ & \qquad\qquad x \in \mathcal{X}. \end{aligned} \tag{5.2}$$

Let $y^* = f(x^*)$ be the optimal solution for the two-stage problem $(\bar{P}_k^\Psi(\varepsilon), \bar{Q}_k^\Psi(\varepsilon))$. Then, $y^* \in \mathcal{Y}_N$. Moreover, $y^* \in \bar{\mathcal{Y}}_N^\Psi$ where $\bar{\mathcal{Y}}_N^\Psi = \{y \in \mathcal{Y}_N | \nexists y' \in \mathcal{Y} : y'_\Psi \leq y_\Psi\}$.

Proof. Suppose $y^* \notin \mathcal{Y}_N$. Then, $\exists y' \in \mathcal{Y}$ such that $y' \leq y$. We know $y'_j \leq y^*_j \leq \varepsilon_j, \forall j \in \Psi \wedge j \neq k$. Then, y' is feasible for $\bar{P}_k^\Psi(\varepsilon)$ and $y'_k \leq y^*_k$. Since y^* is the optimal solution of $(\bar{P}_k^\Psi(\varepsilon), \bar{Q}_k^\Psi(\varepsilon))$, $y'_k = y^*_k$. Otherwise, y^* is not optimal. Therefore, y' is also feasible for $\bar{Q}_k^\Psi(\varepsilon)$. Summing over all $j \in \{1, \dots, p\}$, we get $\sum_{j=1}^p y'_j < \sum_{j=1}^p y^*_j$, which contradicts the optimality of y^*. Then, $y^* \in \mathcal{Y}_N$.

Suppose $y^* \notin \bar{\mathcal{Y}}_N^\Psi$. Then, $\exists \hat{y} \in \mathcal{Y}$ such that $\hat{y}_\Psi \leq y^*_\Psi$. Similarly, $\hat{y}_\Psi$ is feasible for $(\bar{P}_k^\Psi(\varepsilon), \bar{Q}_k^\Psi(\varepsilon))$. Summing over all $j \in \{1, \dots, p\}$, we get $\sum_{j=1}^p \hat{y}_j < \sum_{j=1}^p y^*_j$, which contradicts optimality of y^*. Hence, $y^* \in \bar{\mathcal{Y}}_N^\Psi$. □

Proposition 5.4.2 *Let $\Psi_1 \subset \Psi_2 \subseteq \{1, \dots, p\}$. Then, $\forall y^1 \in \bar{\mathcal{Y}}_N^{\Psi_1}, \nexists y^2 \in \bar{\mathcal{Y}}_N^{\Psi_2}$ such that $y^2_{\Psi_1} \leq y^1_{\Psi_1}$.*

Proof. Let $\varepsilon^1 \in \mathbb{R}^{|\Psi_1|}$ such that y^1 is the optimal solution to $(\bar{P}_k^{\Psi_1}(\varepsilon^1), \bar{Q}_k^{\Psi_1}(\varepsilon^1))$ where $k \in \Psi_1$. Suppose $y^2_{\Psi_1} \leq y^1_{\Psi_1}$. Then, y^2 is feasible for $(\bar{P}_k^{\Psi_1}(\varepsilon^1), \bar{Q}_k^{\Psi_1}(\varepsilon^1))$. Summing over all $j \in \Psi_1$, we get $\sum_{j \in \Psi_1} y^2_j < \sum_{j \in \Psi_1} y^1_j$. This contradicts the optimality of y^1 which completes the proof. □

An algorithm that leverages the initialization of the search space using the nondominated points of the reduced problem is provided in Algorithm 5.4.1. Here, $MOILP^{\Psi}$ represents the multi-objective problem that only uses the objectives $f_j(x)$ where $j \in \Psi$.

Algorithm 5.4.1 Recursive algorithm that uses search space initialization with $\bar{\mathcal{Y}}_N^{\Psi}$

Require: $\Psi \subset \{1, \ldots, p\} \wedge |\Psi| = 2$

Ensure: $\mathcal{Y}_N$

$\bar{\mathcal{Y}}_N^{\Psi} \leftarrow$ solve$(MOILP^{\Psi})$ ▷ Use a bi-objective algorithm

for $j \in \{1, \ldots, p\} \backslash \Psi$ **do**

  $\{\bar{\mathcal{B}}_0^{\Psi}(y_{\Psi})\}_{y \in \bar{\mathcal{Y}}_N^{\Psi}} \leftarrow$initialize_search_region$(\mathcal{B}^{\Psi}(y^I_{\Psi}, y^U_{\Psi}), \bar{\mathcal{Y}}_N^{\Psi})$

  $\bar{\mathcal{Y}}_N^{\Psi} \leftarrow$ solve$(MOILP^{\Psi}, \{\bar{\mathcal{B}}_0^{\Psi}(y_{\Psi})\}_{y \in \bar{\mathcal{Y}}_N^{\Psi}})$ ▷ Use an algorithm that operates in $(p-1)$-dimensional search space

  $\Psi \leftarrow \Psi \cup \{j\}$

end for

In the next section, we test the [Kirlik and Sayin, 2014] algorithm initialized by recursively solving reduced problems.

5.5 Preliminary Computational Experiments

For testing our initialization approach, we picked the knapsack problems from the problem set provided in [Kirlik, 2014]. These problems have 5 objectives and 20 items. We will compare the performance of [Tamby and Vanderpooten, 2021], [Kirlik and Sayin, 2014] and [Kirlik and Sayin, 2014] with the recursive initialization procedure previously provided. Due to the computational times and resources required to solve these problems, we set a time limit of 1 hour and compare the number of nondominated points found ($|\mathcal{Y}_R|$) by these algorithms. We also compare the maximum number objects required ($L_{\max}$) to keep track of the search region throughout the runs and the number of iterations (N) and the number of infeasible models solved (N_{inf}). Experiments are performed on a shared HPC cluster with Intel Xeon 2.1 GHz CPU and 64 GB of memory limit. All the algorithms are implemented in Julia v1.10.0 [Bezanson et al., 2017] using JuMP v1.31 [Dunning et al., 2017] and CPLEX v22.1.1.

The results of these experiments are summarized in Table 5.1.

Algorithm	$L_{\max}$	$\lvert\mathcal{Y}_R\rvert$	N	N_{inf}
[Kirlik and Sayin, 2014]	15,127,778.3	83.3	357.7	72.1
[Kirlik and Sayin, 2014] initialized using $\bar{\mathcal{Y}}^q_N$	7,357,217.5	112.5	269.1	0.0
[Tamby and Vanderpooten, 2021]	958.6	130.0	694.3	0.0

Table 5.1: [Kirlik, 2014] MOKP instance with $p = 5$ and $n = 20$

As can be seen in the results, the initialization reduces the number of objects significantly while still being much higher compared to [Tamby and Vanderpooten, 2021]. We believe one additional way to avoid these large number of objects can be to maintain separate lists of objects for each region created in the initialization. It is still likely that we will not be below the numbers seen in [Tamby and Vanderpooten, 2021] but it should make the list of objects more manageable.

Another observation is that we do not solve any infeasible problems. Since proving infeasiblity for integer programs are computationally expensive [Tamby and Vanderpooten, 2021], this can be considered another important benefit of the initialization procedure.

The reason for the lower number of iterations is that, with the initialization procedure, we also solve the reduced problem and the reported number of iterations are only for the problem with p objectives. This allows us to obtain many nondominated points by solving smaller problems and quickly obtain many nondominated points. It can also be seen that [Tamby and Vanderpooten, 2021] is able to fit much more iterations into the same timeframe. This is due to the fact that the algorithm is able to warmstart the mathematical models and more importantly there is not much time spent managing the search region due to the smaller number of objects maintained.

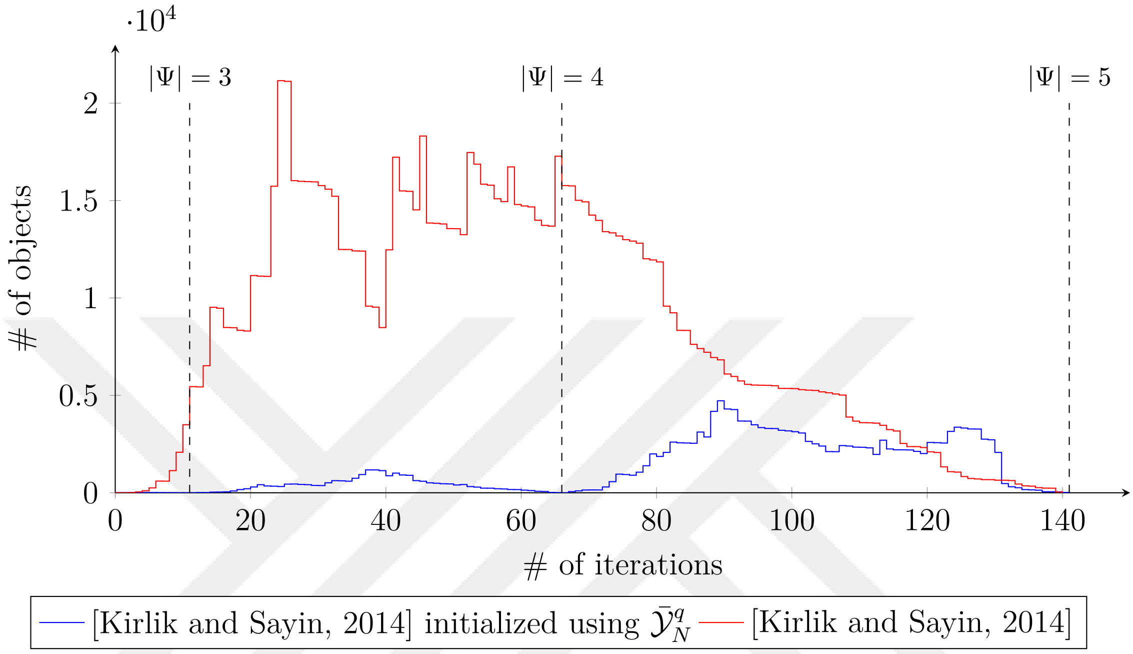


Figure 5.4: Comparison of # of objects with [Kirlik and Sayin, 2014]. Knapsack problem with $p = 5$

In order to compare the behavior of [Kirlik and Sayin, 2014] and its variant with the initialization, we can examine Figure 5.4 where we see the change in the number of objects for a smaller MOKP from [Kirlik and Sayin, 2014] with 5 objectives and 10 items. In this figure, the number of objects corresponds to the size of the list of boxes that needs to be kept during the execution of the algorithm. It can be seen that the maximum number of objects is far less than the original. Also, the boxes we maintain with $\Psi = 3$ and 4 are less complex with fewer dimensions.

Chapter 6

CONCLUSION

In this thesis, we studied the ways of reducing the number of objective functions in MOILP problems. The motivation for this is the increase in computational effort required for MOILP problems as the number of objective functions increases. Therefore, with fewer objective functions, the problem becomes more manageable.

To start off, we studied the fundamentals of MOO problems and explained the key concepts. Previous work on redundant objectives lays out the ground work for objective reduction in MOO problems. Nevertheless, in most real world situations, there is rarely a redundant objective. Characteristics of the solution sets were used by the evolutionary MOO community to address the same problem of objective reduction. We introduce, on the other hand, an objective elimination method that is based on problem parameters. We study cases where there is no redundancy and the elimination will result in some information loss. To quantify this information loss, we treated the nondominated set of the reduced problem as a representation of the nondomianted set of the original problem and assessed its quality using some representation metrics from the literature.

Based on the key concepts required to understand redundancy of an objective function, a way to reduce objective functions was proposed to a well-known scalarization method to ensure that even with the reduced form, we are still able to find nondominated points from the original multi-objective problem.

We evaluated our reduction method on an extensive set of test problems from the literature. We used a range of problem types from different problem sets and showed that, for the majority of problems, our approach is an effective way to reduce the number of objective functions and obtained high quality representations for the nondominated set of the original problem. We also generated our instances with correlated objective function coefficients in order to mimic scenarios where objective reduction should be considered. We noted the cases where the choice of reduction makes a meaningful difference, our method works quite well.

Lastly, we discussed the properties of the solutions obtained from solving the reduced problem and used those to provide a description of the search region. These nondominated points are used to initialize the search region for an exact solution algorithm that works in the projected $(p-1)$-dimensional space. Using our initialization method leads to easier maintenance of the search region with a smaller list of objects. This is shown in our preliminary experiments where the number of objects required to be maintained reduced significantly.

In terms of potential future research direction, we can consider several alternatives. One of those alternatives is to extend our projection based reduction method to removing more than one objective function. This can be done in a recursive manner where the reduced problem obtained after removing one of the objective functions can be reduced further using the same projection based approach. An alternative to this is to remove multiple objectives at once. For this approach, angular distance to redundancy must be redefined not for a single but for a group of objective functions. It is also important to note that our objective reduction approach only considers the objective coefficient vectors and the feasible region of the problem is not accounted for. A new reduction method that also studies the properties of the feasible region is another potential research direction.

Expanding our initialization approach can be considered as another potential research direction. Using a recursive approach results in finding the same solutions multiple times during the course of the algorithm. These repeating solutions can be reduced by slightly modifying the MOILP problem after every recursive step by adding extra constraints, similar to [Ozlen et al., 2011]. However, this will lead to solving some infeasible models. We have opted to avoid any infeasible solutions but this is something that can be investigated. Most advanced solvers maintain the model state and things like branching history and node logs that can prove infeasibility quickly. That can be leveraged to avoid some repeated solutions.

Additionally, our initialization approach can be seen as a way of decomposing the search region with local ideal points and nadir points. There are methods previously mentioned in Section 2.6.2 that try to follow a similar approach to distribute the workload of the search of the objective space [Dhaenens et al., 2010]. This can be tested and supported by new ways of maintaining the search region, and communicating across processes that work on different parts of the objective space.

Another way to expand our initialization approach could be to convert the initialization into a complete exact solution algorithm. This can be done by considering the solutions of the reduced problem as lower bounds on the search region. These bounds can be updated whenever some of the nondominated points in $\bar{\mathcal{Y}}_N^q$ is dominated in $(p-1)$-dimensional space by a newly found nondominated point. This should trigger an update on the list of lower bounds and the algorithm can proceed until the list of lower bounds are exhausted. In order to use the lower bounds on $(p-1)$-dimensional space, we think scalarazation methods such as the hybrid method or Pascoletti-Serafini [Pascoletti and Serafini, 1984] can be adopted for $(p-1)$-dimensional space.

Overall, the projection based objective reduction approach introduced in this thesis provides an effective way of reducing the number of objective functions for an MOILP problem. This has been shown on an extensive set of problem instances from the literature as well as generated correlated instances. Additionally, the nondominated points of the reduced problem have been used to warm start the solution procedure for the original problem. Two different uses of the nondominated sets of the reduced problem have been shown to be beneficial to lessen the computational burden of solving MOILP problems and aid researchers and practitioners in their solution efforts.

BIBLIOGRAPHY

[Agrell, 1997] Agrell, P. J. (1997). On redundancy in multi criteria decision making. *European Journal of Operational Research*, 98:571–586.

[Aguirre and Tanaka, 2009] Aguirre, H. and Tanaka, K. (2009). Many-objective optimization by space partitioning and adaptive ϵ-ranking on mnk-landscapes. In Ehrgott, M., Fonseca, C. M., Gandibleux, X., Hao, J.-K., and Sevaux, M., editors, *Evolutionary Multi-Criterion Optimization*, pages 407–422, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Bazgan et al., 2009] Bazgan, C., Hugot, H., and Vanderpooten, D. (2009). Solving efficiently the 0–1 multi-objective knapsack problem. *Computers & Operations Research*, 36:260–279.

[Bektaş, 2018] Bektaş, T. (2018). Disjunctive programming for multiobjective discrete optimisation. *INFORMS Journal on Computing*, 30:625–633.

[Belotti et al., 2013] Belotti, P., Soylu, B., and Wiecek, M. M. (2013). A branch-and-bound algorithm for biobjective mixed-integer programs. *Optimization Online*, pages 1–29.

[Bezanson et al., 2017] Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98.

[Boland et al., 2017a] Boland, N., Charkhgard, H., and Savelsbergh, M. (2017a). A new method for optimizing a linear function over the efficient set of a multiobjective integer program. *European Journal of Operational Research*, 260(3):904–919.

[Boland et al., 2017b] Boland, N., Charkhgard, H., and Savelsbergh, M. (2017b). The quadrant shrinking method: A simple and efficient algorithm for solving tri-objective integer programs. *European Journal of Operational Research*, 260:873–885.

[Bowman, 1976] Bowman, V. J. (1976). On the relationship of the tchebycheff norm and the efficient frontier of multiple-criteria objectives. In Thiriez, H. and Zionts, S., editors, *Multiple Criteria Decision Making*, pages 76–86, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Brockhoff and Zitzler, 2009] Brockhoff, D. and Zitzler, E. (2009). Objective reduction in evolutionary multiobjective optimization: theory and applications. *Evolutionary computation*, 17:135–166.

[Chalmet et al., 1986] Chalmet, L., Lemonidis, L., and Elzinga, D. (1986). An algorithm for the bi-criterion integer programming problem. *European Journal of Operational Research*, 25(2):292–300.

[Cheung and Gu, 2014] Cheung, Y.-m. and Gu, F. (2014). Online objective reduction for many-objective optimization problems. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 1165–1171.

[Cheung et al., 2016] Cheung, Y.-M., Gu, F., and Liu, H.-L. (2016). Objective extraction for many-objective optimization problems: Algorithm and test problems; objective extraction for many-objective optimization problems: Algorithm and test problems. *IEEE Transactions on Evolutionary Computation*, 20:755.

[Deb et al., 2005] Deb, K., Saxena, D. K., et al. (2005). On finding pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems. *Kangal report*, 2005011:1–19.

[Deb et al., 2001] Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2001). Scalable test problems for evolutionary multi-objective optimization. Report, Zurich.

[Dhaenens et al., 2010] Dhaenens, C., Lemesre, J., and Talbi, E. (2010). K-ppm: A new exact method to solve multi-objective combinatorial optimization problems. *European Journal of Operational Research*, 200(1):45–53.

[Domínguez-Ríos et al., 2021] Domínguez-Ríos, M. A., Chicano, F., and Alba, E. (2021). Effective anytime algorithm for multiobjective combinatorial optimization problems. *Information Sciences*, 565:210–228.

[Dunning et al., 2017] Dunning, I., Huchette, J., and Lubin, M. (2017). Jump: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320.

[Dächert et al., 2017] Dächert, K., Klamroth, K., Lacour, R., and Vanderpooten, D. (2017). Efficient computation of the search region in multi-objective optimization. *European Journal of Operational Research*, 260:841–855.

[Ehrgott, 2005] Ehrgott, M. (2005). *Multicriteria Optimization*. Springer-Verlag.

[Fonseca et al., 2006] Fonseca, C., Paquete, L., and Lopez-Ibanez, M. (2006). An improved dimension-sweep algorithm for the hypervolume indicator. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1157–1163.

[Forget et al., 2022] Forget, N., Gadegaard, S. L., Klamroth, K., Nielsen, L. R., and Przybylski, A. (2022). Branch-and-bound and objective branching with three or more objectives. *Computers & Operations Research*, 148:106012.

[Gal and Leberling, 1977] Gal, T. and Leberling, H. (1977). Redundant objective functions in linear vector maximum problems and their determination. *European Journal of Operational Research*, 1:176–184.

[Geoffrion, 1968] Geoffrion, A. M. (1968). Proper efficiency and the theory of vector maximization. *Journal of Mathematical Analysis and Applications*, pages 618–630.

[Guddat, 1985] Guddat, J. (1985). *Multiobjective and Stochastic Optimization Based on Parametric Optimization*. (MATHEMATISCHE FORSCHUNG). Akademie-Verlag.

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

[Holzmann and Smith, 2018] Holzmann, T. and Smith, J. C. (2018). Solving discrete multi-objective optimization problems using modified augmented weighted tchebychev scalarizations. *European Journal of Operational Research*, 271:436–449.

[Jiang et al., 2014] Jiang, S., Ong, Y.-S., Zhang, J., and Feng, L. (2014). Consistencies and contradictions of performance metrics in multiobjective optimization. *IEEE Transactions on Cybernetics*, 44(12):2391–2404.

[Karsu and Ulus, 2022] Karsu, O. and Ulus, F. (2022). Split algorithms for multiobjective integer programming problems. *Computers & Operations Research*, 140:105673.

[Kirlik, 2014] Kirlik, G. (2014). Test instances for multiobjective discrete optimization problems. *URL: http://home. ku. edu. tr/~ moolibrary.*

[Kirlik and Sayin, 2014] Kirlik, G. and Sayin, S. (2014). A new algorithm for generating all nondominated solutions of multiobjective discrete optimization problems. *European Journal of Operational Research*, 232:479–488.

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

[Kiziltan and Yucaoğlu, 1983] Kiziltan, G. and Yucaoğlu, E. (1983). An algorithm for multiobjective zero-one linear programming. *Management Science*, 29:1444–1453.

[Klamroth et al., 2015] Klamroth, K., Lacour, R., and Vanderpooten, D. (2015). On the representation of the search region in multi-objective optimization.

[Klein and Hannan, 1982] Klein, D. and Hannan, E. (1982). An algorithm for the multiple objective integer linear programming problem. *European Journal of Operational Research*, 9:378–385.

[Koç, 1995] Koç, C. (1995). *Basic Linear Algebra*. Odtü Matematik Vakfı.

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

[Lokman and Köksalan, 2013] Lokman, B. and Köksalan, M. (2013). Finding all nondominated points of multi-objective integer programs. *Journal of Global Optimization*, 57:347–365.

[López Jaimes et al., 2008] López Jaimes, A., Coello Coello, C. A., and Chakraborty, D. (2008). Objective reduction using a feature selection technique. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, GECCO '08, page 673–680, New York, NY, USA. Association for Computing Machinery.

[López Jaimes et al., 2014] López Jaimes, A., Coello Coello, C. A., Aguirre, H., and Tanaka, K. (2014). Objective space partitioning using conflict information for solving many-objective problems. *Information Sciences*, 268:305–327. New Sensing and Processing Technologies for Hand-based Biometrics Authentication.

[Malinowska, 2006] Malinowska, A. (2006). Nonessential objective functions in linear multiobjective optimization problems. *Control and Cybernetics*, 35.

[Malinowska and Torres, 2008] Malinowska, A. B. and Torres, D. F. M. (2008). Computational approach to essential and nonessential objective functions in linear multicriteria optimization. *Journal of Optimization Theory and Applications*, 139:577–590.

[Mavrotas, 2009] Mavrotas, G. (2009). Effective implementation of the ε-constraint method in multi-objective mathematical programming problems. *Applied Mathematics and Computation*, 213:455–465.

[Mavrotas and Florios, 2013] Mavrotas, G. and Florios, K. (2013). An improved version of the augmented ε-constraint method (augmecon2) for finding the exact pareto set in multi-objective integer programming problems. *Applied Mathematics and Computation*, 219(18):9652–9669.

[Mesquita-Cunha et al., 2022] Mesquita-Cunha, M., Figueira, J. R., and Barbosa-Póvoa, A. P. (2022). New ϵ-constraint methods for multi-objective integer linear programming: A pareto front representation approach. *European Journal of Operational Research*.

[Miller et al., 1960] Miller, C. E., Tucker, A. W., and Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7:326–329.

[Mitra et al., 2002] Mitra, P., Murthy, C., and Pal, S. (2002). Unsupervised feature selection using feature similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):301–312.

[Ozlen and Azizoğlu, 2009] Ozlen, M. and Azizoğlu, M. (2009). Multi-objective integer programming: A general approach for generating all non-dominated solutions. *European Journal of Operational Research*, 199(1):25–35.

[Ozlen et al., 2011] Ozlen, M., Burton, B., and MacRae, C. (2011). Multi-objective integer programming: An improved recursive algorithm. *Journal of Optimization Theory and Applications*, 160.

[Pascoletti and Serafini, 1984] Pascoletti, A. and Serafini, P. (1984). Scalarizing vector optimization problems. *Journal of Optimization Theory and Applications*, 42:499–524.

[Przybylski and Gandibleux, 2017] Przybylski, A. and Gandibleux, X. (2017). Multi-objective branch and bound. *European Journal of Operational Research*, 260(3):856–872.

[Sayın, 2000] Sayın, S. (2000). Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming. *Mathematical Programming*, 87:543–560.

[Sayın, 2024] Sayın, S. (2024). Supported nondominated points as a representation of the nondominated set: An empirical analysis. *Journal of Multi-Criteria Decision Analysis*, 31.

[Shi et al., 2017] Shi, N., Zhou, S., Wang, F., Tao, Y., and Liu, L. (2017). The multi-criteria constrained shortest path problem. *Transportation Research Part E: Logistics and Transportation Review*, 101:13–29.

[Strang, 2016] Strang, G. (2016). *Introduction to Linear Algebra*. Wellesley - Cambridge Press, 5 edition.

[Tamby and Vanderpooten, 2021] Tamby, S. and Vanderpooten, D. (2021). Enumeration of the nondominated set of multiobjective discrete optimization problems. *INFORMS Journal on Computing*, 33:72–85.

[Thoai, 2012] Thoai, N. V. (2012). Criteria and dimension reduction of linear multiple criteria optimization problems. *Journal of Global Optimization*, 52:499–508.

[Turgut et al., 2019] Turgut, O., Dalkiran, E., and Murat, A. E. (2019). An exact parallel objective space decomposition algorithm for solving multi-objective integer programming problems. *Journal of Global Optimization*, 75:35–62.

[Vázquez et al., 2018] Vázquez, D., Fernández-Torres, M. J., Ruiz-Femenia, R., Jiménez, L., and Caballero, J. A. (2018). Milp method for objective reduction in multi-objective optimization. *Computers & Chemical Engineering*, 108:382–394.

[Wächter and Biegler, 2006] Wächter, A. and Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106:25–57.

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

[Zhang and Reimann, 2014] Zhang, W. and Reimann, M. (2014). A simple augmented ε-constraint method for multi-objective mathematical integer programming problems. *European Journal of Operational Research*, 234:15–24.

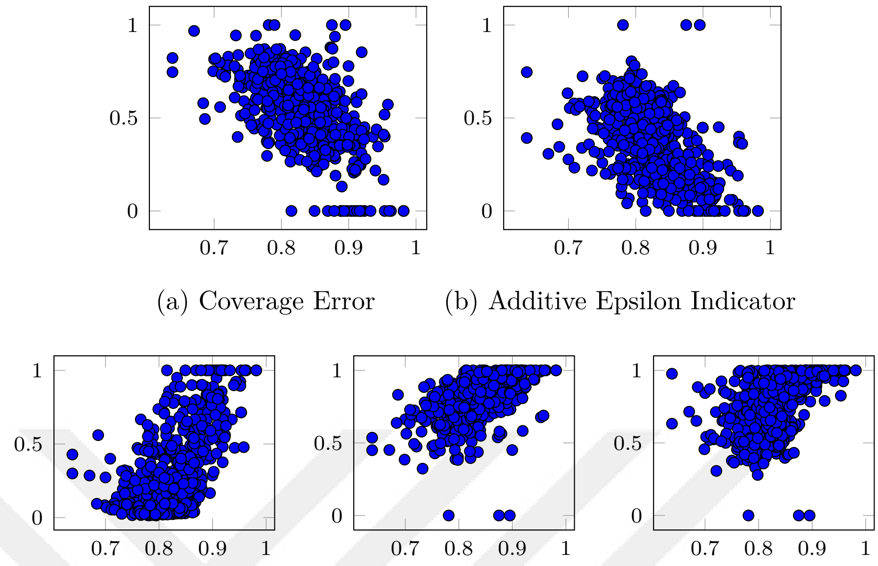
[Zitzler et al., 2003] Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., and da Fonseca, V. (2003). Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132.

Appendix A

CORRELATION ANALYSIS

Problem Set	Coverage Error	Additive Epsilon Indicator	ONVGR	Range Ratio	Hypervolume Ratio
[Kirlik, 2014], KP	-0.5571	-0.5300	+0.5734	+0.5131	+0.4835
[Kirlik, 2014], AP	-0.1592	+0.0425	-0.0556	+0.2831	-0.0188
[Kirlik, 2014], ILP	-0.4355	-0.4747	+0.5980	+0.4789	+0.4598
[Tamby and Vanderpooten, 2021], KP	-0.6597	-0.6239	+0.6179	+0.6192	+0.5809
[Tamby and Vanderpooten, 2021], AP	-0.8112	-0.7562	+0.7445	+0.7770	+0.7774
[Bektaş, 2018], TSP	-0.4256	-0.4623	+0.4203	+0.4015	+0.4337
[Mesquita-Cunha et al., 2022], KP	-0.4536	-0.3620	+0.1450	+0.3420	+0.2803
Correlated Instances, KP	-0.6330	-0.4723	+0.4008	+0.5641	+0.3537

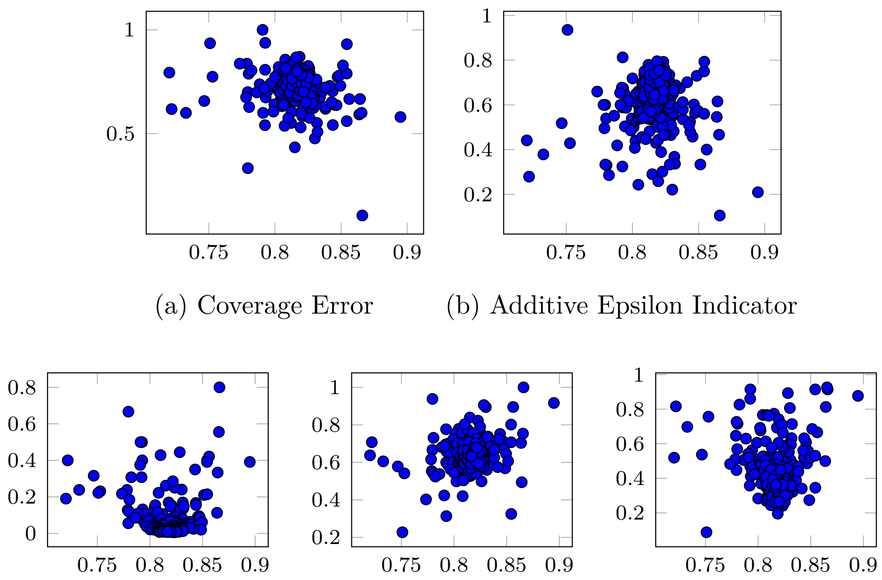
Table A.1: Correlation between quality metrics and $\cos(\theta)$



(a) Coverage Error (b) Additive Epsilon Indicator

(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

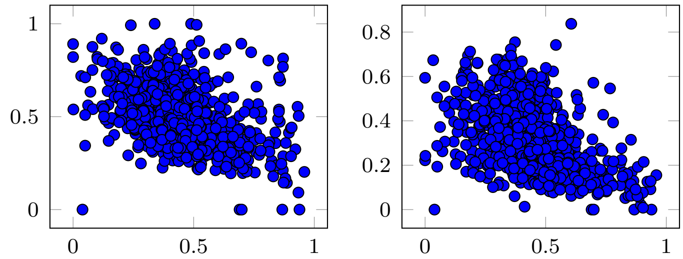
Figure A.1: [Kirlik, 2014] MOKP instances, $\cos(\theta)$ vs. quality metrics



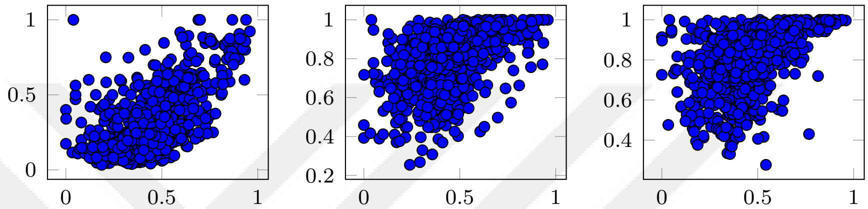
(a) Coverage Error (b) Additive Epsilon Indicator

(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

Figure A.2: [Kirlik, 2014] MOAP instances, $\cos(\theta)$ vs. quality metrics

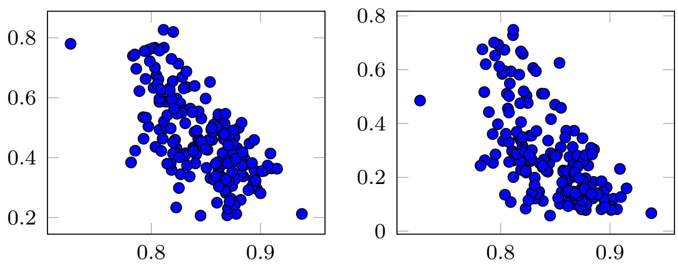


(a) Coverage Error (b) Additive Epsilon Indicator

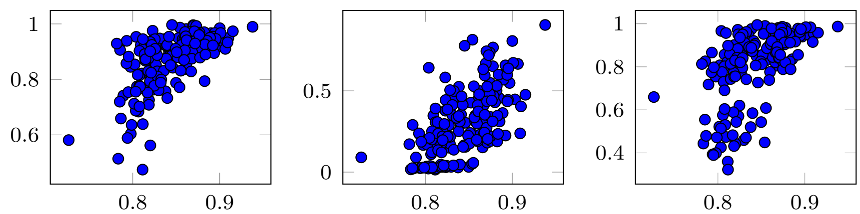


(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

Figure A.3: [Kirlik, 2014] MOILP instances, $\cos(\theta)$ vs. quality metrics

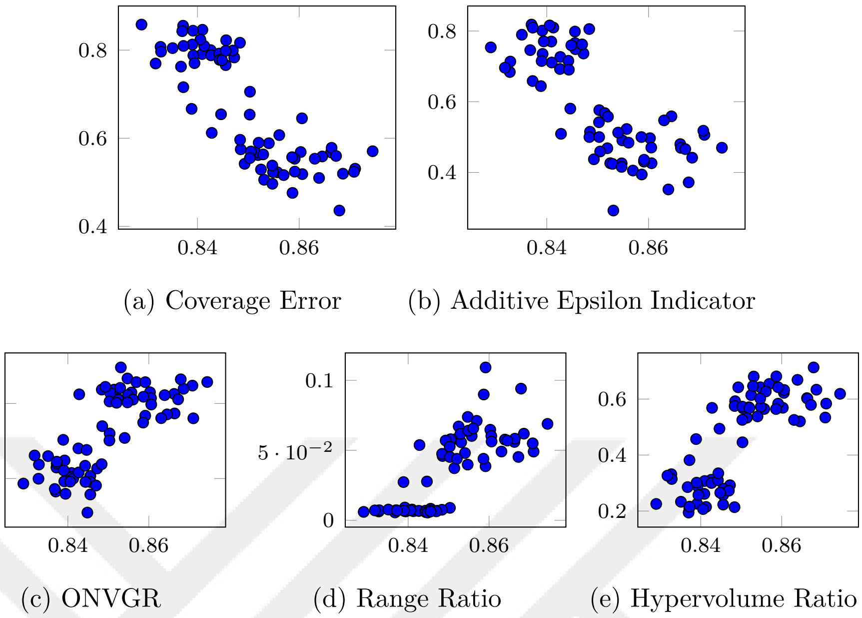


(a) Coverage Error (b) Additive Epsilon Indicator



(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

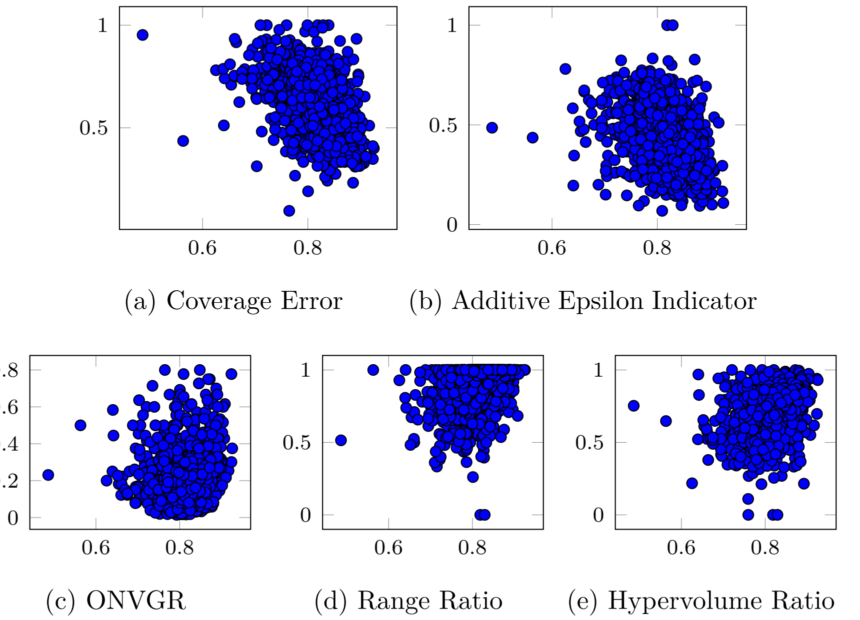
Figure A.4: [Tamby and Vanderpooten, 2021] MOKP instances, $\cos(\theta)$ vs. quality metrics



(a) Coverage Error (b) Additive Epsilon Indicator

(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

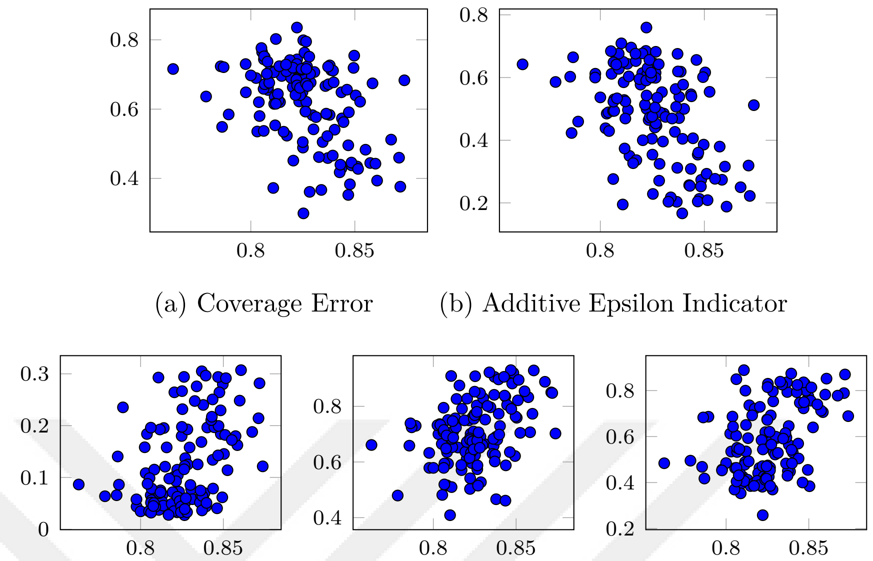
Figure A.5: [Tamby and Vanderpooten, 2021] MOAP instances, $\cos(\theta)$ vs. quality metrics



(a) Coverage Error (b) Additive Epsilon Indicator

(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

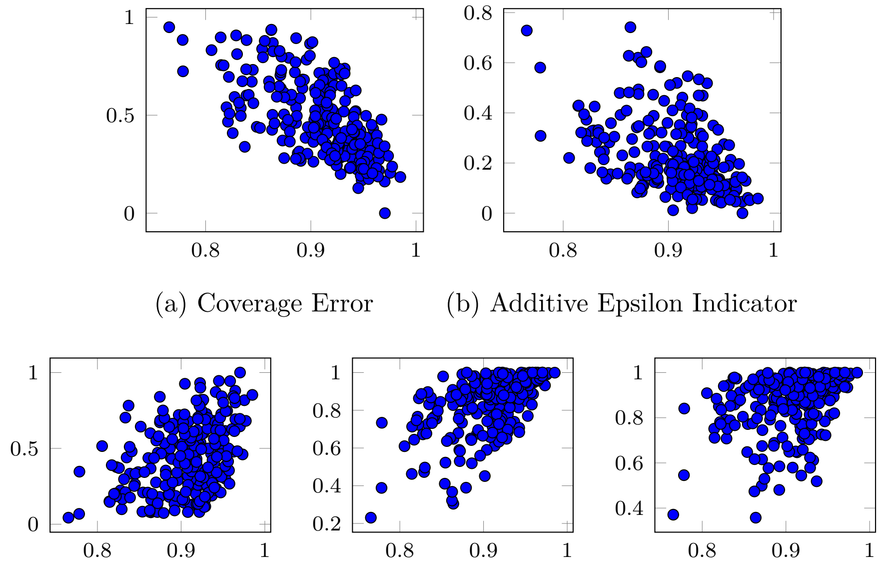
Figure A.6: [Mesquita-Cunha et al., 2022] MOKP instances, $\cos(\theta)$ vs. quality metrics



(a) Coverage Error (b) Additive Epsilon Indicator

(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

Figure A.7: [Bektaş, 2018] MOTSP instances, $\cos(\theta)$ vs. quality metrics



(a) Coverage Error (b) Additive Epsilon Indicator

(c) ONVGR (d) Range Ratio (e) Hypervolume Ratio

Figure A.8: Correlated MOKP instances, $\cos(\theta)$ vs. quality metrics

Appendix B

NUMERICAL RESULTS OF COMPUTATIONAL EXPERIMENTS

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	10	0.4304	0.8781	0.5462	0.5462
	20	0.4499	0.8121	0.4534	0.4534
	30	0.4537	0.7525	0.5539	0.5539
	40	0.5262	0.7635	0.5716	0.5716
	50	0.5109	0.7765	0.5750	0.5750
	60	0.5701	0.7632	0.5851	0.5851
	70	0.5792	0.7725	0.6028	0.6028
	80	0.5437	0.7754	0.5775	0.5775
	90	0.5680	0.7965	0.5749	0.5749
	100	0.6011	0.7664	0.6056	0.6056
Additive Epsilon Indicator	10	0.1528	0.7072	0.3255	0.3255
	20	0.2595	0.5626	0.2772	0.2772
	30	0.3064	0.5021	0.3617	0.3617
	40	0.3781	0.5565	0.4394	0.4394
	50	0.3562	0.5331	0.3713	0.3713
	60	0.4491	0.5856	0.4712	0.4712
	70	0.4572	0.6343	0.5362	0.5362
	80	0.4829	0.6151	0.5012	0.5012
	90	0.4653	0.6329	0.5090	0.5090
	100	0.5383	0.6324	0.5599	0.5599
ONVGR	10	0.2947	0.6787	0.6025	0.6025
	20	0.2043	0.4099	0.3887	0.3887
	30	0.1410	0.2318	0.2085	0.2085
	40	0.0870	0.1419	0.1305	0.1305
	50	0.0859	0.1384	0.1235	0.1235
	60	0.0602	0.0838	0.0767	0.0767
	70	0.0410	0.0637	0.0604	0.0604
	80	0.0360	0.0647	0.0578	0.0578
	90	0.0305	0.0585	0.0535	0.0535
	100	0.0238	0.0372	0.0364	0.0364
Range Ratio	10	0.3222	0.8853	0.7275	0.7275
	20	0.5253	0.8738	0.8666	0.8666
	30	0.5523	0.8134	0.7638	0.7638
	40	0.6164	0.8062	0.7620	0.7620
	50	0.6186	0.8228	0.8136	0.8136
	60	0.6511	0.8084	0.7976	0.7976
	70	0.6152	0.7860	0.7682	0.7682
	80	0.6004	0.8019	0.7794	0.7794
	90	0.6065	0.8049	0.7956	0.7956
	100	0.6666	0.7861	0.7769	0.7769
Hypervolume Ratio	10	0.4757	0.9115	0.7857	0.7857
	20	0.5759	0.8263	0.8038	0.8038
	30	0.6040	0.7766	0.7238	0.7238
	40	0.5551	0.6916	0.6355	0.6355
	50	0.5660	0.7079	0.7020	0.7020
	60	0.5106	0.6415	0.6052	0.6052
	70	0.4546	0.6209	0.5312	0.5312
	80	0.4731	0.5997	0.5763	0.5763
	90	0.4637	0.6101	0.5615	0.5615
	100	0.4472	0.5503	0.5111	0.5111

Table B.1: [Kirlik, 2014] MOKP instances with $p = 3$

Quality Measure	**Problem Size**	**Minimum**	**Maximum**	**Cone Projection**	**Subspace Projection**
Coverage Error	10	0.1197	0.7724	0.2751	0.2751
	20	0.3482	0.6014	0.3634	0.3634
	30	0.3471	0.6073	0.3718	0.3718
	40	0.3390	0.6259	0.3901	0.3901
Additive Epsilon Indicator	10	0.0356	0.3449	0.1248	0.1248
	20	0.1497	0.3319	0.1691	0.1691
	30	0.1610	0.3499	0.1893	0.1893
	40	0.2091	0.3700	0.2392	0.2392
ONVGR	10	0.4678	0.9355	0.8039	0.8039
	20	0.2932	0.5588	0.4831	0.4831
	30	0.2130	0.4513	0.4088	0.4088
	40	0.1725	0.3265	0.2759	0.2759
Range Ratio	10	0.6332	0.9819	0.9486	0.9486
	20	0.7588	0.9397	0.9180	0.9180
	30	0.7810	0.9279	0.9107	0.9107
	40	0.7686	0.9231	0.8940	0.8940
Hypervolume Ratio	10	0.9268	0.9992	0.9639	0.9639
	20	0.8123	0.9316	0.9288	0.9288
	30	0.7897	0.9022	0.8887	0.8887
	40	0.7582	0.8689	0.8366	0.8366

Table B.2: [Kirlik, 2014] MOKP instances with $p = 4$

Quality Measure	**Problem Size**	**Minimum**	**Maximum**	**Cone Projection**	**Subspace Projection**
Coverage Error	10	0.0214	0.6672	0.2914	0.2914
	20	0.2162	0.5926	0.3316	0.3316
Additive Epsilon Indicator	10	0.0113	0.3629	0.1367	0.1367
	20	0.0684	0.2396	0.1081	0.1081
ONVGR	10	0.5315	0.9840	0.8458	0.8458
	20	0.3844	0.8062	0.6308	0.6308
Range Ratio	10	0.7218	0.9977	0.9441	0.9441
	20	0.7879	0.9824	0.9239	0.9239
Hypervolume Ratio	10	0.8628	0.9998	0.9695	0.9695
	20	0.9098	0.9861	0.9714	0.9714

Table B.3: [Kirlik, 2014] MOKP instances with $p = 5$

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	5	0.5306	0.8534	0.6134	0.6134
	10	0.5632	0.7668	0.6140	0.6140
	15	0.6150	0.7325	0.6420	0.6420
	20	0.6509	0.7421	0.6845	0.6845
	25	0.6941	0.7632	0.7108	0.7108
	30	0.7215	0.7951	0.7362	0.7362
	35	0.7085	0.7858	0.7469	0.7469
	40	0.7279	0.7848	0.7605	0.7605
	45	0.7523	0.7949	0.7665	0.7665
	50	0.7524	0.8100	0.7733	0.7733
Additive Epsilon Indicator	5	0.2971	0.6774	0.4701	0.4701
	10	0.3893	0.5658	0.4664	0.4664
	15	0.4956	0.6265	0.5498	0.5498
	20	0.5412	0.6546	0.6409	0.6409
	25	0.6059	0.7048	0.6174	0.6174
	30	0.6127	0.7107	0.6697	0.6697
	35	0.6334	0.6998	0.6673	0.6673
	40	0.6535	0.7345	0.6862	0.6862
	45	0.6538	0.7461	0.7006	0.7006
	50	0.6814	0.7484	0.7146	0.7146
ONVGR	5	0.2533	0.4709	0.4111	0.4111
	10	0.1018	0.1603	0.1318	0.1318
	15	0.0487	0.0819	0.0712	0.0712
	20	0.0331	0.0449	0.0390	0.0390
	25	0.0222	0.0300	0.0273	0.0273
	30	0.0186	0.0227	0.0212	0.0212
	35	0.0152	0.0189	0.0166	0.0166
	40	0.0113	0.0144	0.0127	0.0127
	45	0.0104	0.0125	0.0111	0.0111
	50	0.0083	0.0103	0.0095	0.0095
Range Ratio	5	0.4411	0.8293	0.7197	0.7197
	10	0.5796	0.7601	0.7025	0.7025
	15	0.5765	0.7193	0.6793	0.6793
	20	0.5877	0.6858	0.6324	0.6324
	25	0.5702	0.6721	0.6509	0.6509
	30	0.6097	0.7030	0.6665	0.6665
	35	0.5980	0.6887	0.6482	0.6482
	40	0.5833	0.6702	0.6265	0.6265
	45	0.5970	0.6741	0.6392	0.6392
	50	0.5931	0.6542	0.6227	0.6227
Hypervolume Ratio	5	0.5044	0.8178	0.6831	0.6831
	10	0.5228	0.6677	0.6022	0.6022
	15	0.4400	0.5529	0.4951	0.4951
	20	0.3900	0.4985	0.4034	0.4034
	25	0.3287	0.4187	0.4074	0.4074
	30	0.3183	0.4201	0.3643	0.3643
	35	0.3335	0.3998	0.3663	0.3663
	40	0.2931	0.3607	0.3320	0.3320
	45	0.2781	0.3513	0.3235	0.3235
	50	0.2638	0.3302	0.3024	0.3024

Table B.4: [Kirlik, 2014] MOAP instances

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	10	0.3688	0.8870	0.5219	0.5549
	20	0.4572	0.7876	0.5634	0.5120
	30	0.4773	0.7087	0.5217	0.5762
	40	0.5048	0.7607	0.5653	0.5653
	50	0.5466	0.7163	0.6045	0.6045
	60	0.5009	0.6985	0.5772	0.5772
	70	0.5087	0.7604	0.5788	0.5788
	80	0.5176	0.7232	0.5368	0.5368
	90	0.5276	0.7258	0.5378	0.5378
	100	0.5296	0.7153	0.5738	0.5738
Additive Epsilon Indicator	10	0.2138	0.5170	0.2510	0.2499
	20	0.2420	0.5446	0.3481	0.3315
	30	0.2881	0.5313	0.3719	0.3967
	40	0.3520	0.5577	0.4066	0.4066
	50	0.3830	0.5760	0.4894	0.4894
	60	0.3247	0.5637	0.4655	0.4655
	70	0.3563	0.5546	0.3948	0.3948
	80	0.3497	0.5279	0.3813	0.3813
	90	0.3935	0.5583	0.4270	0.4270
	100	0.3566	0.5840	0.4400	0.4400
ONVGR	10	0.3155	0.6961	0.5653	0.5620
	20	0.2154	0.4229	0.3307	0.3607
	30	0.1557	0.2967	0.2515	0.2324
	40	0.1121	0.2170	0.1620	0.1620
	50	0.0929	0.1712	0.1510	0.1510
	60	0.1260	0.1978	0.1782	0.1782
	70	0.0845	0.1518	0.1258	0.1258
	80	0.0754	0.1395	0.1309	0.1309
	90	0.0789	0.1309	0.1260	0.1260
	100	0.0715	0.1157	0.1067	0.1067
Range Ratio	10	0.4554	0.9109	0.8119	0.8073
	20	0.4810	0.8323	0.7835	0.8144
	30	0.5126	0.8009	0.7328	0.6669
	40	0.5139	0.7925	0.7522	0.7522
	50	0.5191	0.7210	0.6453	0.6453
	60	0.5387	0.7626	0.6717	0.6717
	70	0.5168	0.7526	0.7068	0.7068
	80	0.5378	0.7936	0.7712	0.7712
	90	0.5709	0.7799	0.7595	0.7595
	100	0.5424	0.7652	0.7087	0.7087
Hypervolume Ratio	10	0.7021	0.9025	0.8482	0.8556
	20	0.5652	0.8554	0.7441	0.7481
	30	0.5695	0.7836	0.6937	0.6830
	40	0.5297	0.7055	0.6505	0.6505
	50	0.4981	0.6882	0.5696	0.5696
	60	0.5419	0.7425	0.6130	0.6130
	70	0.5049	0.7042	0.6510	0.6510
	80	0.5348	0.6989	0.6590	0.6590
	90	0.5132	0.6739	0.6222	0.6222
	100	0.4927	0.6971	0.6199	0.6199

Table B.5: [Kirlik, 2014] MOILP instances with $p = 3$

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	10	0.2748	0.7406	0.3438	0.3438
	20	0.3390	0.6048	0.3798	0.3952
	30	0.3315	0.6180	0.4192	0.4192
	40	0.3667	0.5746	0.4362	0.4362
	50	0.3168	0.5772	0.3499	0.3499
	60	0.3765	0.5725	0.4182	0.4182
	70	0.3678	0.5652	0.4048	0.4048
	80	0.3624	0.5511	0.4244	0.4244
Additive Epsilon Indicator	10	0.1113	0.4765	0.1243	0.1243
	20	0.1617	0.3692	0.2317	0.2383
	30	0.1751	0.3784	0.2479	0.2479
	40	0.1837	0.3852	0.2332	0.2332
	50	0.1728	0.3412	0.2142	0.2142
	60	0.2293	0.4229	0.2802	0.2802
	70	0.2257	0.3920	0.2992	0.2992
	80	0.2240	0.3596	0.2677	0.2677
ONVGR	10	0.3157	0.7738	0.7516	0.7516
	20	0.2172	0.5211	0.4396	0.4443
	30	0.1990	0.4148	0.3581	0.3581
	40	0.1757	0.3833	0.2982	0.2982
	50	0.1590	0.4012	0.3352	0.3352
	60	0.1261	0.2362	0.2109	0.2109
	70	0.1167	0.2378	0.1900	0.1900
	80	0.1396	0.2511	0.2224	0.2224
Range Ratio	10	0.6291	0.9798	0.9440	0.9440
	20	0.7185	0.9515	0.8955	0.8830
	30	0.7347	0.9193	0.8493	0.8493
	40	0.7454	0.9164	0.8545	0.8545
	50	0.7428	0.9195	0.9031	0.9031
	60	0.7579	0.9053	0.8690	0.8690
	70	0.7474	0.8977	0.8398	0.8398
	80	0.7702	0.9084	0.8762	0.8762
Hypervolume Ratio	10	0.7446	0.9816	0.9774	0.9774
	20	0.7689	0.9318	0.8757	0.8750
	30	0.7287	0.9130	0.8546	0.8546
	40	0.7388	0.8981	0.8528	0.8528
	50	0.7535	0.9105	0.8596	0.8596
	60	0.7000	0.8409	0.7972	0.7972
	70	0.7185	0.8392	0.7799	0.7799
	80	0.7384	0.8474	0.8069	0.8069

Table B.6: [Kirlik, 2014] MOILP instances with $p = 4$

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	10	0.1875	0.6155	0.3182	0.3761
	20	0.2596	0.5266	0.3041	0.3226
	30	0.2759	0.5140	0.2945	0.2945
	40	0.2764	0.4616	0.2967	0.2967
Additive Epsilon Indicator	10	0.0527	0.2994	0.1111	0.1276
	20	0.1107	0.2987	0.1298	0.1344
	30	0.1153	0.2709	0.1493	0.1493
	40	0.1297	0.2775	0.1700	0.1700
ONVGR	10	0.4246	0.8925	0.7901	0.7540
	20	0.3127	0.6885	0.6461	0.6200
	30	0.2875	0.6057	0.5740	0.5740
	40	0.2152	0.4785	0.4425	0.4425
Range Ratio	10	0.7360	0.9937	0.9694	0.9579
	20	0.7825	0.9670	0.9438	0.9386
	30	0.8007	0.9682	0.9423	0.9423
	40	0.8073	0.9565	0.9370	0.9370
Hypervolume Ratio	10	0.9068	0.9956	0.9831	0.9795
	20	0.8574	0.9745	0.9620	0.9565
	30	0.8590	0.9720	0.9456	0.9456
	40	0.8344	0.9454	0.9268	0.9268

Table B.7: [Kirlik, 2014] MOILP instances with $p = 5$

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	$p = 6, n = 30$	0.2571	0.5007	0.3370	0.3370
	$p = 5, n = 50$	0.3146	0.5541	0.3411	0.3411
	$p = 4, n = 50$	0.3673	0.6406	0.3964	0.3964
	$p = 3, n = 100$	0.5880	0.7668	0.5926	0.5926
Additive Epsilon Indicator	$p = 6, n = 30$	0.0874	0.2277	0.1407	0.1407
	$p = 5, n = 50$	0.1526	0.3303	0.1845	0.1845
	$p = 4, n = 50$	0.2154	0.4080	0.2352	0.2352
	$p = 3, n = 100$	0.5146	0.6731	0.5383	0.5383
ONVGR	$p = 6, n = 30$	0.8852	0.9825	0.9526	0.9526
	$p = 5, n = 50$	0.8540	0.9609	0.9495	0.9495
	$p = 4, n = 50$	0.7449	0.9152	0.8856	0.8856
	$p = 3, n = 100$	0.6098	0.7957	0.7883	0.7883
Range Ratio	$p = 6, n = 30$	0.3472	0.7419	0.5063	0.5063
	$p = 5, n = 50$	0.1788	0.4171	0.3578	0.3578
	$p = 4, n = 50$	0.1463	0.3001	0.2594	0.2594
	$p = 3, n = 100$	0.0254	0.0396	0.0390	0.0390
Hypervolume Ratio	$p = 6, n = 30$	0.9127	0.9819	0.9533	0.9533
	$p = 5, n = 50$	0.8126	0.9215	0.8999	0.8999
	$p = 4, n = 50$	0.7364	0.8585	0.8377	0.8377
	$p = 3, n = 100$	0.4189	0.5614	0.5247	0.5247

Table B.8: [Tamby and Vanderpooten, 2021] MOKP instances

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	$p = 4, n = 20$	0.5082	0.6192	0.5336	0.5336
	$p = 3, n = 50$	0.7658	0.8293	0.7871	0.7871
Additive Epsilon Indicator	$p = 4, n = 20$	0.4034	0.5496	0.4544	0.4544
	$p = 3, n = 50$	0.6891	0.7965	0.7436	0.7436
ONVGR	$p = 4, n = 20$	0.7449	0.8499	0.8244	0.8244
	$p = 3, n = 50$	0.5840	0.6425	0.6210	0.6210
Range Ratio	$p = 4, n = 20$	0.0434	0.0731	0.0609	0.0609
	$p = 3, n = 50$	0.0065	0.0073	0.0072	0.0072
Hypervolume Ratio	$p = 4, n = 20$	0.5280	0.6472	0.6150	0.6150
	$p = 3, n = 50$	0.2259	0.3335	0.2794	0.2794

Table B.9: [Tamby and Vanderpooten, 2021] MOAP instances

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	$p = 3, n = 10, m = 5$	0.4515	0.8707	0.5411	0.5411
	$p = 3, n = 15, m = 10$	0.4668	0.7888	0.5462	0.5462
	$p = 3, n = 20, m = 15$	0.4801	0.7908	0.5145	0.5145
	$p = 3, n = 25, m = 1$	0.4731	0.7950	0.5585	0.5585
	$p = 3, n = 25, m = 2$	0.4887	0.7702	0.5280	0.5280
	$p = 3, n = 25, m = 3$	0.4810	0.7611	0.5137	0.5137
	$p = 3, n = 25, m = 4$	0.4917	0.7545	0.5424	0.5424
	$p = 3, n = 25, m = 5$	0.4896	0.7650	0.5275	0.5275
	$p = 3, n = 50, m = 1$	0.5392	0.7645	0.5611	0.5611
	$p = 3, n = 75, m = 1$	0.5796	0.7602	0.6022	0.6022
	$p = 3, n = 100, m = 1$	0.6185	0.7559	0.6406	0.6406
	$p = 4, n = 50, m = 1$	0.3900	0.6404	0.4174	0.4174
Additive Epsilon Indicator	$p = 3, n = 10, m = 5$	0.2536	0.6152	0.2973	0.2973
	$p = 3, n = 15, m = 10$	0.2833	0.5841	0.3532	0.3532
	$p = 3, n = 20, m = 15$	0.3230	0.5990	0.3980	0.3980
	$p = 3, n = 25, m = 1$	0.2681	0.5422	0.3386	0.3386
	$p = 3, n = 25, m = 2$	0.3281	0.5746	0.3888	0.3888
	$p = 3, n = 25, m = 3$	0.3154	0.5323	0.3745	0.3745
	$p = 3, n = 25, m = 4$	0.3376	0.5606	0.4045	0.4045
	$p = 3, n = 25, m = 5$	0.3555	0.5839	0.4226	0.4226
	$p = 3, n = 50, m = 1$	0.3955	0.5712	0.4334	0.4334
	$p = 3, n = 75, m = 1$	0.4749	0.5997	0.5223	0.5223
	$p = 3, n = 100, m = 1$	0.5031	0.6182	0.5359	0.5359
	$p = 4, n = 50, m = 1$	0.2478	0.4219	0.2931	0.2931
ONVGR	$p = 3, n = 10, m = 5$	0.3027	0.5987	0.5282	0.5282
	$p = 3, n = 15, m = 10$	0.2417	0.4334	0.3799	0.3799
	$p = 3, n = 20, m = 15$	0.1595	0.3120	0.2785	0.2785
	$p = 3, n = 25, m = 1$	0.1585	0.3392	0.2993	0.2993
	$p = 3, n = 25, m = 2$	0.1451	0.2665	0.2428	0.2428
	$p = 3, n = 25, m = 3$	0.1590	0.2787	0.2512	0.2512
	$p = 3, n = 25, m = 4$	0.1471	0.2592	0.2344	0.2344
	$p = 3, n = 25, m = 5$	0.1445	0.2412	0.2231	0.2231
	$p = 3, n = 50, m = 1$	0.0852	0.1394	0.1290	0.1290
	$p = 3, n = 75, m = 1$	0.0524	0.0755	0.0694	0.0694
	$p = 3, n = 100, m = 1$	0.0344	0.0487	0.0455	0.0455
	$p = 4, n = 50, m = 1$	0.1377	0.2555	0.2233	0.2233
Range Ratio	$p = 3, n = 10, m = 5$	0.6141	0.9817	0.9299	0.9299
	$p = 3, n = 15, m = 10$	0.6601	0.9854	0.9056	0.9056
	$p = 3, n = 20, m = 15$	0.6799	0.9653	0.9279	0.9279
	$p = 3, n = 25, m = 1$	0.6995	0.9723	0.9107	0.9107
	$p = 3, n = 25, m = 2$	0.6879	0.9641	0.9157	0.9157
	$p = 3, n = 25, m = 3$	0.7131	0.9679	0.9250	0.9250
	$p = 3, n = 25, m = 4$	0.7341	0.9752	0.9326	0.9326
	$p = 3, n = 25, m = 5$	0.7127	0.9650	0.9455	0.9455
	$p = 3, n = 50, m = 1$	0.7943	0.9882	0.9539	0.9539
	$p = 3, n = 75, m = 1$	0.8272	0.9841	0.9533	0.9533
	$p = 3, n = 100, m = 1$	0.8616	0.9935	0.9698	0.9698
	$p = 4, n = 50, m = 1$	0.9296	0.9996	0.9929	0.9929
Hypervolume Ratio	$p = 3, n = 10, m = 5$	0.5071	0.8622	0.8342	0.8342
	$p = 3, n = 15, m = 10$	0.5710	0.8166	0.7671	0.7671
	$p = 3, n = 20, m = 15$	0.5197	0.7699	0.6902	0.6902
	$p = 3, n = 25, m = 1$	0.5659	0.8161	0.7454	0.7454
	$p = 3, n = 25, m = 2$	0.5393	0.7545	0.7001	0.7001
	$p = 3, n = 25, m = 3$	0.5870	0.7705	0.7165	0.7165
	$p = 3, n = 25, m = 4$	0.5627	0.7561	0.6782	0.6782
	$p = 3, n = 25, m = 5$	0.5306	0.7307	0.6611	0.6611
	$p = 3, n = 50, m = 1$	0.5283	0.6838	0.6397	0.6397
	$p = 3, n = 75, m = 1$	0.4916	0.6128	0.5532	0.5532
	$p = 3, n = 100, m = 1$	0.4685	0.5786	0.5412	0.5412
	$p = 4, n = 50, m = 1$	0.7203	0.8388	0.7992	0.7992

Table B.10: [Mesquita-Cunha et al., 2022] MOKP instances

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	$p = 3, n = 10$	0.6058	0.7270	0.6879	0.6879
	$p = 4, n = 10$	0.3811	0.5766	0.4928	0.4928
	$p = 3, n = 15$	0.6191	0.7390	0.6591	0.6591
	$p = 3, n = 20$	0.6647	0.7413	0.6939	0.6939
Additive Epsilon Indicator	$p = 3, n = 10$	0.4101	0.6077	0.5048	0.5048
	$p = 4, n = 10$	0.2223	0.4256	0.3068	0.3068
	$p = 3, n = 15$	0.4898	0.6230	0.5420	0.5420
	$p = 3, n = 20$	0.5492	0.6455	0.6020	0.6020
ONVGR	$p = 3, n = 10$	0.0946	0.1796	0.1358	0.1358
	$p = 4, n = 10$	0.1484	0.2595	0.1958	0.1958
	$p = 3, n = 15$	0.0555	0.0728	0.0675	0.0675
	$p = 3, n = 20$	0.0379	0.0531	0.0445	0.0445
Range Ratio	$p = 3, n = 10$	0.5157	0.7352	0.6144	0.6144
	$p = 4, n = 10$	0.7504	0.9070	0.8189	0.8189
	$p = 3, n = 15$	0.6013	0.7275	0.6603	0.6603
	$p = 3, n = 20$	0.6225	0.7044	0.6692	0.6692
Hypervolume Ratio	$p = 3, n = 10$	0.4625	0.6566	0.5657	0.5657
	$p = 4, n = 10$	0.6962	0.8324	0.7738	0.7738
	$p = 3, n = 15$	0.4265	0.5721	0.5115	0.5115
	$p = 3, n = 20$	0.4062	0.4956	0.4380	0.4380

Table B.11: [Bektaş, 2018] MOTSP instances

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	$p = 5, n = 20$	0.2135	0.6815	0.2279	0.2279
	$p = 4, n = 30$	0.2486	0.6882	0.2564	0.2564
	$p = 3, n = 40$	0.2937	0.8760	0.3120	0.3120
Additive Epsilon Indicator	$p = 5, n = 20$	0.0702	0.2546	0.0912	0.0912
	$p = 4, n = 30$	0.1013	0.3708	0.1077	0.1077
	$p = 3, n = 40$	0.1829	0.5254	0.2268	0.2268
ONVGR	$p = 5, n = 20$	0.3887	0.8346	0.7860	0.7860
	$p = 4, n = 30$	0.2006	0.5716	0.5169	0.5169
	$p = 3, n = 40$	0.0926	0.3017	0.2984	0.2984
Range Ratio	$p = 5, n = 20$	0.7682	0.9932	0.9889	0.9889
	$p = 4, n = 30$	0.6945	0.9650	0.9600	0.9600
	$p = 3, n = 40$	0.4138	0.9033	0.8744	0.8744
Hypervolume Ratio	$p = 5, n = 20$	0.8835	0.9915	0.9820	0.9820
	$p = 4, n = 30$	0.7699	0.9649	0.9597	0.9597
	$p = 3, n = 40$	0.5869	0.8540	0.8144	0.8144

Table B.12: Class-I correlated MOKP instances

Quality Measure	Problem Size	Minimum	Maximum	Cone Projection	Subspace Projection
Coverage Error	$p = 5, n = 20$	0.2797	0.6497	0.3130	0.3130
	$p = 4, n = 30$	0.2875	0.7507	0.3179	0.3179
	$p = 3, n = 40$	0.5391	0.7554	0.6062	0.6062
Additive Epsilon Indicator	p=5, n=20	0.0550	0.2686	0.0977	0.0977
	$p = 4, n = 30$	0.1123	0.3567	0.1397	0.1397
	$p = 3, n = 40$	0.3382	0.4846	0.4015	0.4015
ONVGR	p=5, n=20	0.3820	0.8369	0.7445	0.7445
	$p = 4, n = 30$	0.2506	0.5540	0.4997	0.4997
	$p = 3, n = 40$	0.1816	0.2885	0.2422	0.2422
Range Ratio	p=5, n=20	0.7421	0.9840	0.9615	0.9615
	$p = 4, n = 30$	0.6395	0.9585	0.9438	0.9438
	$p = 3, n = 40$	0.6278	0.8070	0.7416	0.7416
Hypervolume Ratio	p=5, n=20	0.8991	0.9956	0.9837	0.9837
	$p = 4, n = 30$	0.7904	0.9497	0.9287	0.9287
	$p = 3, n = 40$	0.6165	0.7433	0.6887	0.6887

Table B.13: Class-II correlated MOKP instances