Multicriteria Linear Optimisation with Applications in Sustainable Manufacturing

vorgelegt von
Dipl.-Math.

Sebastian Schenker

Von der Fakultät II - Mathematik und Naturwissenschaften
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
– Dr. rer. nat. –

genehmigte Dissertation

Promotionsausschuss:
Vorsitzender: Prof. Dr. Martin Henk
Gutachter: Prof. Dr. Martin Skutella
Gutachter: Prof. Dr. Ralf Borndörfer
Gutachter: Prof. Dr. Stefan Ruzika

Tag der wissenschaftlichen Aussprache: 23.10.2019

Berlin 2019

ABSTRACT

Multicriteria optimisation is concerned with optimising several objectives simultaneously. Assuming that all objectives are equally important but conflicting, we are interested in the set of nondominated points, i.e., the image set of solutions that cannot be improved in any objective without getting worse off in another one. This thesis comprises three parts. The first part considers the computation of nondominated vertices of multicriteria linear programs. We present a cutting plane algorithm for finding nondominated vertices based on enumerating the vertices of a weight space polyhedron and describe its implementation in PolySCIP, a solver for multicriteria optimisation problems which we developed as a part of this dissertation. The focus of the second part is on multicriteria integer programs. In this part we investigate a partitioning of the set of nondominated points given by the set of points whose projections remain nondominated if one of the objectives is discarded and the set of points whose projections become dominated if one of the objectives is discarded. We present characteristics of this partitioning and show that the first mentioned partition can be used as an approximation of the entire set of nondominated points. The third part of this thesis considers two real-world applications in the context of sustainable manufacturing. The first problem regards bicycle frame manufacturing via a bicriteria integer programming formulation for which the non-dominated points are computed via PolySCIP. The second problem considers a tricriteria assignment problem in the context of scenario analysis. We classify this problem, show its hardness, and compute the set of nondominated points as well a partitioning of the set of nondominated points for given real-world data via PolySCIP.

ZUSAMMENFASSUNG

Mehrkriterielle Optimierung befasst sich mit Optimierungsproblemen, bei denen mehrere in Konflikt stehende Zielfunktionen gleichzeitig optimiert werden. Unter der Annahme, dass alle Zielfunktionen gleichwertig sind, betrachten wir einen Bildpunkt als nicht-dominiert, sofern er in keiner Zielfunktion verbessert werden kann ohne sich in einer anderen Zielfunktion zu verschlechtern. In dieser Arbeit interessieren wir uns für die Menge der nicht-dominierten Punkte für ein gegebenes mehrkriterielles Optimierungsproblem. Diese Dissertation lässt sich in drei Teile gliedern. Der erste Teil beschäftigt sich mit dem Berechnen von nicht-dominierten Eckpunkten im Kontext mehrkriterieller linearer Programmierung. Nach einführenden Erläuterungen präsentieren wir einen Algorithmus, der auf der Enumeration von Eckpunkten eines Gewichtsraumpolyeders basiert. In diesem Zusammenhang

beschreiben wir ebenfalls die zugehörige Implementation in PolySCIP, einem von uns entwickelten Löser für mehrkriterielle Optimierungsprobleme. Der zweite Teil dieser Arbeit beschäftigt sich mit mehrkriterieller ganzzahliger Programmierung. In diesem Teil untersuchen wir eine Partitionierung der nicht-dominierten Punkte in zwei Teilmengen. Die erste Partition beinhaltet alle nicht-dominierten Punkte, deren Projektion nicht-dominiert bleibt, sofern man eine Zielfunktion außer Betracht lässt. Die zweite Partition beinhaltet alle nicht-dominierten Punkte, deren Projektion dominiert wird, sofern man eine Zielfunktion außer Betracht lässt. Wir charakterisieren diese Partitionierung und zeigen, dass die erstgenannte Partition die Menge der nicht-dominierten Punkte approximiert. Der dritte Teil dieser Arbeit betrachtet zwei Anwendungen aus dem Bereich der nachhaltigen Produktion. Die erste Anwendung betrifft ein auf Fahrradrahmen bezogenes Produktionsproblem, welches als ganzzahliges Programm mit zwei Zielfunktionen modelliert wird. Für die gegebenen Instanzen berechnen wir die Menge der nicht-dominierten Punkte mittels PolySCIP und interpretieren die zugehörigen Resultate. Die zweite Anwendung betrifft ein Optimierungsproblem mit drei Zielfunktionen aus dem Bereich der Szenarioanalyse. Wir zeigen, dass es zur Klasse der NP-schweren Probleme gehört und berechnen die Menge der nicht-dominierten Punkte sowie die zugehörigen Partitionen für die gegebenen Instanzen ebenfalls mittels PolySCIP.

ACKNOWLEDGEMENTS

I would like to thank Prof. Dr. Ralf Borndörfer and Prof. Dr. Martin Skutella for making this thesis possible by giving me a position as a research assistant at ZIB and at TU Berlin for several years and by giving me guidance in my intellectual endeavours throughout this time. I would like to acknowledge that the funding for my position came via the collaborative research centre 1026 and MATHEON research centre. I would like to thank all my previous colleagues at TU Berlin, at ZIB, and at the collaborative research centre for creating an enjoyable work and research environment. Special thanks go to Michael Bastubbe, Jan Heiland, Oliver Kley, Matthias Miltenberger, Robert Schwarz, Felipe Serrano, Thomas Schlechte, and Ingmar Vierhaus for proofreading different parts of this thesis.

I would like to give a very special *thank you* to my wife Federica for her continuous encouragement, support, and patience.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS

IP integer program 6

LP linear program 6

MNDP mono-nondominated point 36

MOP multicriteria optimisation problem

NDP nondominated point 6

PNDP poly-nondominated point 35

NOTATION

$[n]$	finite set of natural numbers $\{1, \ldots, n\}$
$\mathbb{N}_+$	set of positive natural numbers $\{1, 2, 3, \ldots\}$
$\mathbb{R}_{\geq 0}$	set of nonnegative real numbers
$\mathbb{R}_+$	set of positive real numbers
$\mathbb{1}$	all-ones vector (of appropriate dimension)
$\mathcal{X}$	feasible domain
$\mathcal{Y}$	image of the feasible domain in objective space
$\mathcal{N}$	entire set of nondominated points
$\mathcal{N}_s$	set of supported nondominated points
$\mathcal{N}_u$	set of unsupported nondominated points
$\mathcal{N}_p$	set of poly-nondominated points
$\mathcal{N}_m$	set of mono-nondominated points
$(c_1, \ldots, c_k, \mathcal{X})$	k-criteria optimisation problem $\min\limits_{x \in \mathcal{X}} (c_1^\top x, \ldots, c_k^\top x)$
$(\lambda^\top C, \mathcal{X})$	single objective optimisation problem $\min\limits_{x \in \mathcal{X}} (\sum_{i=1}^k \lambda_i c_i^\top x)$

1

INTRODUCTION

> "Begin at the beginning," the King said gravely, "and go on till you come to the end: then stop."
>
> *Lewis Carroll*, Alice in Wonderland

1.1 MULTICRITERIA OPTIMISATION

Multicriteria optimisation is concerned with optimising several objectives simultaneously. It can be considered as a generalisation of single objective optimisation with numerous applications that range from health care [52], economics and social sciences [81] to traffic and logistics [71]. Roughly speaking, any real-world application that can be modelled as an optimisation problem will likely be subject to several objectives provided that sufficient data is available. In this thesis, we assume that different objectives are equally important and that they generally conflict each other. The economist and sociologist Vilfredo Pareto (1848 – 1923) described a situation of a society in which an increased economic satisfaction of some members implies a decrease of economic satisfaction for others [62]. Although Pareto did not consider abstract objectives, the main principle now often named in honour of him stays the same: An outcome is Pareto optimal if it cannot be improved in at least one objective without deteriorating another. A more formal version will be given in Definition 2.1 in Section 2.2 in terms of *efficient solution* and *nondominated point*. Due to conflicting objectives, a given multicriteria optimisation problem will generally result in several nondominated points which brings us to another assumption used throughout this thesis. We will assume that a *decision maker* has no a priori preference distinguishing some nondominated points from others implying that the entire set of nondominated points is of interest to us. However, it will turn out that for larger instances the number of nondominated points might become enormous and presenting such a set of nondominated points to a decision maker would result rather in confusion than in a well-informed decision. In this regard, we adopt the attitude that computing the entire set of nondominated points is desirable as long as this set is not too large. Otherwise, an approximation, i.e. a subset of the entire set of nondominated points, might be a more appropriate result for a decision maker.

1.2 SUSTAINABLE MANUFACTURING

This thesis is to some extent a product from working in the collaborative research centre 1026 *Sustainable Manufacturing*. The term *sustainability* for humankind is interpreted as the ability to meet its needs without a disruption to nature and society. Regarding manufacturing, the production path from individual components to a final end product consists of a large number of processes where each process consumes natural resources and energy. Sustainable manufacturing aims at incorporating the ecological and social footprint into the production chain, e.g. by incorporating the value of (renewable) resources, by considering the well-being of the labour force, et cetera. The idea is that stakeholders can limit their resource consumption and stay within responsible limits regarding economic, environmental, and social constraints by adopting sustainable manufacturing methods. Otherwise, these limits are exceeded implying that the ability of future generations to meet their own needs will be compromised irrevocably. From an optimisation perspective, we interpret economic, environmental, and social goals as conflicting objectives which are equally important leading us to multicriteria optimisation problems. Similarly, a nondominated point of such a multicriteria optimisation problem is interpreted as a potentially sustainable outcome. A decision maker which aims at finding a sustainable decision is consequently interested in the set of nondominated points.

1.3 THESIS OUTLINE AND CONTRIBUTION

Throughout this thesis, the term *linear optimisation* will subsume *linear programming* and *integer programming* and by multicriteria linear optimisation we refer to either a linear programming problem with several linear objectives or an integer programming problem with several linear objectives or both. In a nutshell, given a multicriteria linear optimisation problem we are interested in the set of nondominated points by either computing it entirely or by computing a subset of nondominated points which *represent* the entire set. We will examine this problem from a theoretical as well as practical perspective.

Chapter 2 introduces basic definitions and fundamental theorems and presents some well-known algorithmic approaches for computing nondominated points. This chapter establishes a basis for what is presented in the subsequent chapters.

Chapter 3 deals with computing nondominated vertices for multicriteria linear programs. The results in this chapter build upon basic polyhedral results. We show that any nondominated point can be represented by a combination of nondominated vertices (and nondominated rays) based on the resolution theorem for polyhedra. Subsequently, we present how to find any nondominated vertex algorithmically and, based on a weight space decomposition method, how to compute the entire set of nondominated vertices. Our main contribution in this chapter is to apply polyhedral theory which allows

us to adopt the resolution theorem for nondominated vertices and to present the weight space polyhedron method in the light of a *cutting plane* algorithm.

Chapter 4 deals with multicriteria integer programs. In this chapter we investigate a partitioning of the entire set of nondominated points into *poly-nondominated points* and *mono-nondominated points*. Firstly, we examine the relationship of these two point sets and their relationship to *supported* nondominated points. Subsequently, we show that for the general k-objective case the values of any mono-nondominated point is bounded from below and from above by poly-nondominated points. Furthermore, we show theoretically and empirically that the number of mono-nondominated points can largely exceed the number of poly-nondominated points. Moreover, we show in a computational study that the number of *irredundant boxes* that are generated by the set of poly-nondominated points is much lower than previously assumed. The partitioning approach in this chapter builds upon work by Tenfelde-Podehl [76]. Our main contribution is to generalise this partitioning approach to the k-objective case and to correct some previously made inaccurate observations regarding the number of poly-nondominated points as well as the number of irredundant boxes which are generated by the set of poly-nondominated points.

Chapter 5 deals with approximating the entire set of nondominated points for multicriteria integer programs. The main approach in this chapter is to consider the set of supported nondominated points, the set of poly-nondominated points, and their respective set union as a *representation* for the entire set of nondominated points. Regarding the set of supported nondominated points, we formalise the a priori *coverage error* that can be computed in the bicriteria case. For the tricriteria case we present an example that shows that the distance between any supported nondominated point and any *unsupported* nondominated point might become arbitrarily large. Regarding the set of poly-nondominated points, we formalise the a priori coverage error that can be computed for the general k-objective case. In the second part of this chapter, we present computational results for the tricriteria assignment problem and tricriteria knapsack problem regarding the representation quality. Our main contribution in this chapter is to show that by taking the set union of the set of poly-nondominated points and the set of supported nondominated points we can get a more *robust* representation.

Chapter 6 presents PolySCIP, a solver for multicriteria linear programs and multicriteria integer programs. PolySCIP was implemented over the course of several years and is based on the algorithmic approaches presented in Chapter 3 and Chapter 4. The first version was released in February 2016 as an official part of the constraint integer solving framework SCIP 3.2.1. In March 2017 a revised version, PolySCIP 2.0, was released as an official part of SCIP 4.0. It was used for most of the computational experiments presented in this thesis and is available in source code for interested users and researchers. Note that the content of this chapter is based on [17].

Chapter 7 applies multicriteria optimisation to sustainable manufacturing. In this chapter, a bicycle frame manufacturing problem is solved via bicriteria integer programming. The computational model considered in this chapter is based on available real-world data and was solved via PolySCIP. This chapter shows how the application of multicriteria optimisation tools can improve the decision making process by providing a set of nondominated points to a decision maker. Note that the content of this chapter is based on [70].

Chapter 8 concludes this thesis. In this chapter we examine the *Scenario Assignment Problem*, a design problem applied in sustainable manufacturing. In the first part of this chapter the problem is formulated and classified. In the second part of this chapter computational results for two real-world applications involving three objectives are presented. This chapter shows, similarly to Chapter 7, how mathematical formulations and multicriteria optimisation enables a decision maker to find quantifiable outcomes for sustainable manufacturing problems.

2

MULTICRITERIA LINEAR OPTIMISATION

> The beginning of wisdom is to call things by their proper name.
>
> *Confucius*

2.1 INTRODUCTION

In this chapter we establish a basis for what is presented in the subsequent chapters. In Section 2.2 we introduce basic definitions, clarify notation used throughout this thesis, and present basic but important results regarding multicriteria linear optimisation. In Section 2.3 we summarise several algorithmic approaches for solving multicriteria linear optimisation problems. Last but not least, in Section 2.4 we present two basic complexity results. In order to keep this chapter reasonably compact, we expect the reader to be familiar with the basics of linear algebra, linear programming, and integer programming. For an introduction to linear programming and aspects of integer programming see [12].

2.2 PRELIMINARIES

2.2.1 *Basic definitions*

The fundamental problem considered in this thesis can be summarised as follows: given a feasible domain denoted by $\mathcal{X}$ and k many linear objectives denoted by $c_1, \ldots, c_k$, find a solution that minimises all objectives simultaneously. We will denote this problem by

$$\min_{x \in \mathcal{X}} (c_1^\top x, \ldots, c_k^\top x). \tag{2.1}$$

Given (2.1), we have to clarify what we exactly mean by minimising the given objectives simultaneously and, moreover, what we understand to be an *optimal* solution of (2.1). For example, provided that we have a ranking on the objectives, an optimal solution of (2.1) could refer to a solution that minimises the objectives *lexicographically* according to our ranking. Another interpretation of (2.1), regardless of any ranking, could refer to some *min max* variant like $\min_{x \in \mathcal{X}} \max_{i=1,\ldots,k} (c_1^\top x, \ldots, c_k^\top x)$. In this thesis, we assume that all given objectives are equally important to us and our notion of optimality is given by the notion of *efficiency* and *nondominance*, respectively. For more details about different concepts of multicriteria optimisation see [29]. For what follows, we denote the set $\{1, \ldots, m\}$ by $[m]$ with $[0]$ corresponding to the empty set. We will denote the image

of $\mathcal{X}$ under $c_1, \ldots, c_k$ by $\mathcal{Y} = \{(c_1^\top x, \ldots, c_k^\top x) : x \in \mathcal{X}\} \subseteq \mathbb{R}^k$. Unless stated otherwise, we will assume that all given objectives are to be minimised.

Definition 2.1. (efficient solution, nondominated point) Let $\mathcal{X}$ be a given feasible domain and let $c_1, \ldots, c_k : \mathcal{X} \to \mathbb{R}$ be given linear objectives where $k \geq 2$. A solution $x^* \in \mathcal{X}$ is called *efficient* if there is no solution $x \in \mathcal{X}$ such that $c_i^\top x \leq c_i^\top x^*$ holds for all $i \in [k]$ and $c_j^\top x < c_j^\top x^*$ holds for some $j \in [k]$. The image $(c_1^\top x^*, \ldots, c_k^\top x^*) \in \mathcal{Y}$ of an efficient solution x^* is called *nondominated point (NDP)*.

Note that regarding a maximisation problem we would exchange $\leq$ with $\geq$ and $<$ with $>$ in Definition 2.1. A feasible solution that is not efficient is called *inefficient* and its corresponding image is called *dominated*. From here on, we will denote the entire set of nondominated points of a given problem instance by $\mathcal{N}$. See Figure 2.1 for an illustration of dominated points and nondominated points regarding a bicriteria optimisation problem.

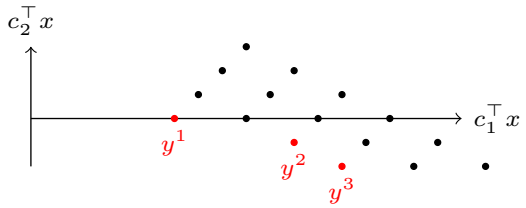

Figure 2.1: Image of a bicriteria minimisation problem with nondominated points y^1, y^2, y^3.

In a nutshell, considering a multicriteria optimisation problem

$$\min_{x \in \mathcal{X}} (c_1^\top x, \ldots, c_k^\top x)$$

we are interested in computing the set of nondominated points $\mathcal{N}$.

Let $m \in \mathbb{N}_+$ be some positive integer. Suppose that, for every $i \in [m]$, we are given an n-dimensional vector a_i and a scalar b_i. A multicriteria *linear program (LP)* can be expressed as

$$\begin{aligned} \min\ & (c_1^\top x, \ldots, c_k^\top x) \\ \text{s.t. } & a_i^\top x \geq b_i \ \text{ for all } i \in [m], \\ & x \in \mathbb{R}^n. \end{aligned} \tag{2.2}$$

Note that the feasible domain $\mathcal{X} = \{x \in \mathbb{R}^n : a_i^\top x \geq b_i \text{ for all } i \in [m]\}$ of (2.2) is a *polyhedron* [8, Definition 2.1]. For an illustration of a bicriteria linear program with a three-dimensional cube as feasible domain $\mathcal{X}$ see Figure 2.2.

In contrast to LPs, feasible solutions of an *integer program (IP)* need to be integral. Similarly to (2.2), a multicriteria IP can be expressed as

$$\begin{aligned} \min\ & (c_1^\top x, \ldots, c_k^\top x) \\ \text{s.t. } & a_i^\top x \geq b_i \ \text{ for all } i \in [m], \\ & x \in \mathbb{Z}^n. \end{aligned} \tag{2.3}$$

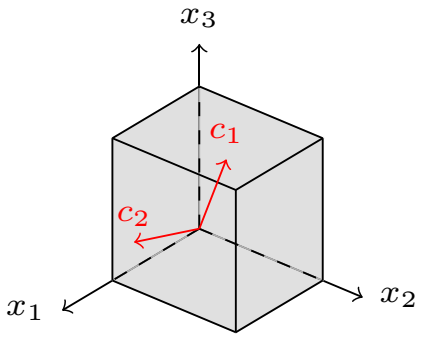

Figure 2.2: Bicriteria linear programming problem with three-dimensional cube as feasible domain $\mathcal{X}$.

Observe that the feasible domain $\mathcal{X} = \{x \in \mathbb{Z}^n : a_i^\top x \geq b_i \text{ for all } i \in [m]\}$ of an integer program is a discrete set of integer vectors. For an illustration of a bicriteria integer program with a two-dimensional feasible domain $\mathcal{X}$ containing finitely many feasible solutions see Figure 2.3.

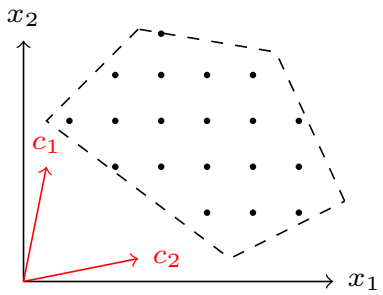

Figure 2.3: Bicriteria integer programming problem with two-dimensional bounded feasible domain $\mathcal{X}$.

From here on, unless stated otherwise, we will denote the minimisation problem $\min_{x\in\mathcal{X}}(c_1^\top x, \ldots, c_k^\top x)$ by $(c_1, \ldots, c_k, \mathcal{X})$. We will use the term multicriteria optimisation problem to subsume multicriteria linear programs and multicriteria integer programs for situations in which we do not want/need to distinguish between (2.2) and (2.3). Throughout this thesis we assume that none of the objectives and none of the constraints are redundant. For ease of presentation, we will sometimes subsume the considered objectives $c_1, \ldots, c_k \in \mathbb{R}^n$ in an objective matrix

$$\begin{pmatrix} c_{1_1} & \ldots & c_{1_n} \\ \vdots & \ddots & \vdots \\ c_{k_1} & \ldots & c_{k_n} \end{pmatrix}$$

denoted by C.

Definition 2.2. (ideal point, nadir point) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria optimisation problem. The point $y^N \in \mathbb{R}^k$ whose components, for all $i \in [k]$, are given by

$$y_i^N = \max_{y\in\mathcal{N}} y_i$$

is called *nadir point*. The point $y^I \in \mathbb{R}^k$ whose components, for all $i \in [k]$, are given by

$$y_i^I = \min_{y \in \mathcal{N}} y_i$$

is called *ideal point*.

Note that in order to compute y^I it suffices to solve $\min_{x \in \mathcal{X}} c_i^\top x$ for all $i \in [k]$.

Definition 2.3. (weakly nondominated point) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria optimisation problem. A solution $x^* \in \mathcal{X}$ is *weakly efficient* if there is no solution $x \in \mathcal{X}$ such that $c_i^\top x < c_i^\top x^*$ holds for all $i \in [k]$. The corresponding image $Cx^* \in \mathcal{Y}$ of a weakly efficient solution x^* is called *weakly nondominated point*.

By Definition 2.1, any nondominated point is weakly nondominated but not vice versa. For an illustration see Figure 2.4 in comparison to Figure 2.1.

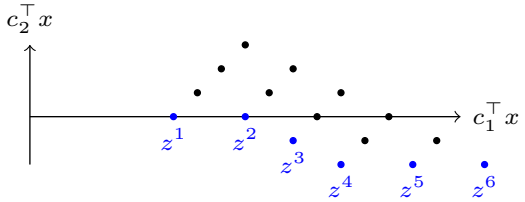

Figure 2.4: Image of a bicriteria minimisation problem with weakly nondominated points $z^1, \ldots, z^6$.

Let $u, v \in \mathbb{R}^k$ be two vectors. The set $\{u + \alpha v : \alpha \geq 0\}$ is called a *ray (emanating from u in the direction of v)*.

Definition 2.4. (ideal ray, efficient ray, nondominated ray) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria LP and let r be a feasible ray of $\mathcal{X}$. r is called *ideal* if $c_i^\top r \leq 0$ holds for all $i \in [k]$ with $c_j^\top r < 0$ for some $j \in [k]$.
r is called *efficient* if $c_i^\top r < 0$ for some $i \in [k]$ and $c_j^\top r > 0$ for some $j \in [k]$. The image $Cr \in \mathbb{R}^k$ of an efficient ray r is called *nondominated ray* of $\mathcal{Y}$.

Note that the existence of an ideal ray r implies that the corresponding set of nondominated points $\mathcal{N}$ is empty.

Remark 2.5. For ease of presentation we will generally assume that the feasible domain $\mathcal{X}$ of the considered multicriteria optimisation problem is bounded, i.e. $\mathcal{X}$ does not permit any feasible ray. However, we will indicate on how to deal with the unbounded case in situations where it might not be directly clear.

2.2.2 *Supported nondominated points*

Having defined nondominated points, how do we compute them? Many algorithmic approaches transform a given multicriteria optimisation problem into a single objective problem. For example, by taking a weight vector $\lambda \in \mathbb{R}^k$ we can transform a given multicriteria

optimisation problem $(c_1, \ldots, c_k, \mathcal{X})$ into a *weighted sum* single objective problem

$$\min_{x \in \mathcal{X}} \sum_{i=1}^{k} \lambda_i c_i^\top x \tag{2.4}$$

denoted, from here on, by $(\lambda^\top C, \mathcal{X})$. It is well-known that optimising $(\lambda^\top C, \mathcal{X})$ with regard to a positive weight vector $\lambda \in \mathbb{R}^k_+$ yields an efficient solution.

Theorem 2.6. ([29, Theorem 3.6]) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria optimisation problem. If $x^* \in \mathcal{X}$ is an optimal solution of $(\lambda^\top C, \mathcal{X})$ for some $\lambda \in \mathbb{R}^k_+$, then x^* is an efficient solution of $(c_1, \ldots, c_k, \mathcal{X})$.

Regarding Theorem 2.6, we distinguish between efficient solutions that can and cannot be found via a weighted sum objective.

Definition 2.7. (supported NDP, unsupported NDP) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria optimisation problem. An efficient solution $x^* \in \mathcal{X}$ and its corresponding image $Cx^* \in \mathcal{N}$ are called *supported* if x^* is an optimal solution of $(\lambda^\top C, \mathcal{X})$ for some $\lambda \in \mathbb{R}^k_+$.

An efficient solution $x^* \in \mathcal{X}$ that is not supported is called *unsupported*. From here on, we denote the set of supported nondominated points by $\mathcal{N}_s$ and the set of unsupported nondominated points by $\mathcal{N}_u$, respectively.

Remark 2.8. From a computational perspective (see also Section 2.3), the weighted sum problem $(\lambda^\top C, \mathcal{X})$ has the useful property that no additional constraints are added to the feasible domain $\mathcal{X}$ implying that structural properties, e.g. total unimodularity of the constraint matrix, are preserved.

Isermann was the first to show that for multicriteria LPs all efficient solutions are supported.

Theorem 2.9. ([44, Theorem 1]) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria LP. A solution $x^* \in \mathcal{X}$ is efficient if and only if x^* is an optimal solution of $(\lambda^\top C, \mathcal{X})$ for some $\lambda \in \mathbb{R}^k_+$.

Figure 2.5 indicates that Theorem 2.9 cannot be carried over to multicriteria IPs. If we want to compute the entire set of nondominated points for a multicriteria IP, then we have to take unsupported nondominated points into account.

2.2.3 *Weighted Chebyshev norm*

Definition 2.10. (weighted Chebyshev norm) The *weighted Chebyshev norm* $\|y\|_\beta$ of $y \in \mathbb{R}^k$ with regard to $\beta \in \mathbb{R}^k_+$ is defined by

$$\|y\|_\beta = \max\left(\beta_1|y_1|, \ldots, \beta_k|y_k|\right).$$

Bowman showed the following characterisation for efficient solutions:

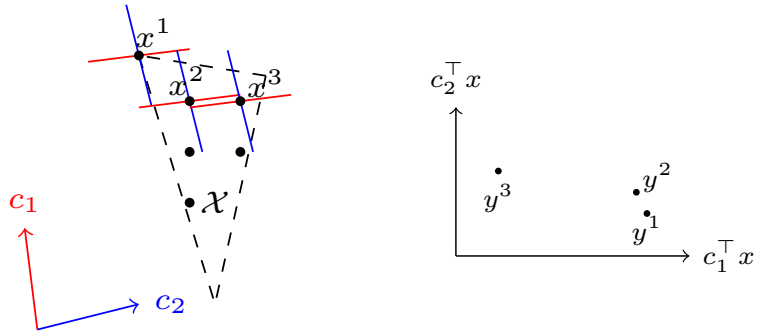

Figure 2.5: Left: A bicriteria integer program $(c_1, c_2, \mathcal{X})$ with efficient solutions $x^1, x^2, x^3 \in \mathcal{X}$. Right: Image of $(c_1, c_2, \mathcal{X})$ with nondominated points y^1, y^2, y^3. Note that y^2 is an unsupported nondominated point.

Theorem 2.11. ([18, Theorem 3]) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria optimisation problem and let $R \in \mathbb{R}^k$ be a *reference point*, i.e. $R_i < c_i^\top x$ for all $x \in \mathcal{X}$ and for all $i \in [k]$. Consider the problem

$$\min_{x \in \mathcal{X}} \|Cx - R\|_\beta \,. \tag{2.5}$$

The feasible solution $x^* \in \mathcal{X}$ is efficient only if x^* is an optimal solution to (2.5) for some $\beta \in \mathbb{R}^k_+$.

Observe that we can transform (2.5) into a problem with a linear objective by introducing a new variable $z \in \mathbb{R}$ and additional linear constraints as follows:

$$\begin{aligned} &\min z \\ \text{s.t. } &\beta_i(c_i^\top x - R_i) \leq z \quad \text{for all } i \in [k], \\ &x \in \mathcal{X}, \\ &z \in \mathbb{R} \,. \end{aligned} \tag{2.6}$$

2.2.4 *Lexicographic optimality*

We call a bijection $\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}$ a *permutation*.

Definition 2.12. ($<_{lex_\sigma}$) Let $y, z \in \mathbb{R}^k$ be two vectors and let $\sigma : [k] \to [k]$ with $\sigma([k]) = \{\sigma_1, \ldots, \sigma_k\}$ be a permutation. We have $y <_{lex_\sigma} z$ if there is an index $i \in [k]$ such that $y_{\sigma_j} = z_{\sigma_j}$ for all $j \in [i-1]$ and $y_{\sigma_i} < z_{\sigma_i}$.

Definition 2.13. (lexicographically optimal) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria optimisation problem and let $\sigma : [k] \to [k]$ be a permutation. The point $y^* \in \mathcal{Y}$ is called *lexicographically optimal* (with regard to σ) if $y^* <_{lex_\sigma} y$ for all $y \in \mathcal{Y} \setminus \{y^*\}$.

Hamacher and Ruhe were the first to show that in the context of multicriteria spanning trees lexicographically optimal points are supported nondominated. Their proof works one-to-one for the general bounded case.

Proposition 2.14. (cf. [41, Corollary to Theorem 4.1]) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria optimisation problem with bounded image $\mathcal{Y}$ and let $\sigma : [k] \to [k]$ be a permutation. If $y^* \in \mathcal{Y}$ is lexicographically optimal with regard to σ, then y^* is supported nondominated.

2.2.5 *Fundamental problems in combinatorial optimisation*

The *assignment problem* (see [49, Section 11.1]) and the *knapsack problem* (see [49, Chapter 17]) are two fundamental problems in combinatorial optimisation. Both of them often arise as subproblems in real-world applications. We will use their corresponding multicriteria versions for some of our computational evaluations, see Section 5.5.

2.2.5.1 *Multicriteria assignment problem*

The multicriteria assignment problem is given by

$$\begin{aligned} \min \ & \Big(\sum_{i=1}^{n}\sum_{j=1}^{n} c_{1_{ij}} x_{ij}, \ldots, \sum_{i=1}^{n}\sum_{j=1}^{n} c_{k_{ij}} x_{ij}\Big) \\ \text{s.t.} \ & \sum_{i=1}^{n} x_{ij} = 1 \ \text{ for all } j \in [n], \\ & \sum_{j=1}^{n} x_{ij} = 1 \ \text{ for all } i \in [n], \\ & x_{ij} \in \{0,1\}, \end{aligned} \tag{2.7}$$

where $k \geq 2$ is the number of considered objectives. Note that the assignment problem is often considered in the context of finding an optimal assignment of n agents to n jobs.

2.2.5.2 *Multicriteria knapsack problem*

The multicriteria knapsack problem is given by

$$\begin{aligned} \max \ & \Big(\sum_{i=1}^{n} c_{1_i} x_i, \ldots, \sum_{i=1}^{n} c_{k_i} x_i\Big) \\ \text{s.t.} \ & \sum_{i=1}^{n} w_i x_i \leq W, \\ & x_i \in \{0,1\}, \end{aligned} \tag{2.8}$$

where $k \geq 2$ is the number of considered objectives and $W \in \mathbb{R}$ is a given maximum weight capacity.

2.3 ALGORITHMIC APPROACHES IN THE LITERATURE

Research in multicriteria optimisation began several decades ago (e.g. see [45, 42]). It should come as no surprise that after several decades of research there is a huge amount of papers dealing with theoretical as well as practical aspects of multicriteria optimisation. However, the situation concerning available solvers for multicriteria optimisation problems is slightly different. ADBASE [74] by Steuer was (to the

best of our knowledge) the first program released to a wider audience aiming to analyse multicriteria linear programs. Only during the last years, after a long dry spell, several new *academic* solvers have been developed, e.g. Bensolve [50], inner [21], Symphony [66] and PolySCIP [17]. Note that this development might have been influenced by the amazing progress in (single objective) solver technology over the last two decades [13]. However, until now none of the available *commercial* solvers like Xpress [32], Cplex [43], and Gurobi [38] offer to compute nondominated points for multicriteria optimisation problems.

In this section we briefly review several algorithmic approaches based on linear programming and integer programming. For a recent and extensive review of multicriteria parametric algorithms and a review of different scalarisation methods we refer the reader to [22, Chapter 3] and [22, Section 2.3], respectively. Despite its popularity, we will not consider *evolutionary* multicriteria optimisation [26] as evolutionary algorithms do, in general, not come with any optimality guarantees on the computed solutions (cf. Section 5.2.1). We will also not consider any approaches for multicriteria *nonlinear* optimisation.

An important issue concerning the *outcome* of a multicriteria algorithm is the question whether the user is interested in the entire set of efficient solutions or rather in the entire set of nondominated points. It might happen that a nondominated point $y \in \mathcal{N}$ corresponds to different efficient solutions, i.e. there are $x^1, x^2 \in \mathcal{X}$ with $x^1 \neq x^2$ and $y = Cx^1 = Cx^2$. Hence, concerning the cardinality of the computed outcome of a multicriteria algorithm it can make a big difference whether for every nondominated point just a single preimage or all corresponding preimages need to be computed. In [25] it is noted that already a relatively small tricriteria LP with 21 nondominated vertices contained 344 efficient *bases* corresponding to 60 different efficient solutions. This disparity motivated the distinction between *feasible space methods* and *objective space methods* (e.g. see [10, Chapter 1]). As the names suggest, a feasible space method works solely in the feasible domain $\mathcal{X}$ and usually computes all efficient solutions (irrespective of having the same image) whereas an objective space method takes the objective space image $\mathcal{Y}$ into account and attempts to avoid the computation of efficient solutions which correspond to already computed nondominated points. We will further comment on this issue from the perspective of a decision maker in Section 6.5.2.

In this thesis we focus on solving multicriteria optimisation problems from a decision maker's perspective which means that, in addition to finding the entire set of nondominated points, we are also interested in partitioning approaches, in approximations of the entire set of nondominated points, and in comparing different subsets of nondominated points (see Chapters 4 and 5). In this regard, we often assume that we can solve multicriteria subproblems that we encounter, e.g. in an approach to partition the entire set of nondominated points, by some algorithm presented in this section. However, comparing the *performance* of different algorithms or arguing

that some algorithm is superior to some other algorithm is not part of this thesis.

2.3.1 *Parametric simplex method*

Bicriteria linear programs are well-investigated and can be solved via a parametric simplex method. Given a parametric LP

$$\min_{x \in \mathcal{X}} (c_1 + \lambda c_2)^\top x \tag{2.9}$$

where $c_1, c_2 \in \mathbb{R}^n$ are two given objectives and $\lambda \in \mathbb{R}$ is a scalar, let $g(\lambda)$ be the optimal objective value of (2.9) as a function of λ. It can be shown that $g(\lambda)$ is a piecewise linear function and its breakpoints can be obtained systematically. Just like in the single objective case, nonnegative reduced costs correspond to an optimal basis. A key observation is that nonnegative reduced costs are affine functions of λ. A basis, which is optimal for some $\lambda \in \mathbb{R}$, will be optimal for all λ values belonging to some closed interval $[\underline{\lambda}, \overline{\lambda}]$. For λ values smaller than $\underline{\lambda}$ or greater than $\overline{\lambda}$, respectively, the optimal basis will change. Thus, the entire range of possible values of λ can be traced by a finite number of iterations where each breakpoint of $g(\lambda)$ corresponds to an optimal basis change. For more details see [12, Section 5.5] or [29, Section 6.2].

Remark 2.15. Murty was the first to show that the number of breakpoints of $g(\lambda)$ can be exponential in the input size of (2.9) [55].

It can be shown that all efficient bases of a multicriteria LP are connected ([29, Theorem 7.10]). Steuer and Evans were among the first who tried to solve multicriteria LPs via a multicriteria simplex method that pivots among efficient bases [74].

2.3.2 *Weighted sum scalarisation method*

The fact that an optimal solution of $(\lambda^\top C, \mathcal{X})$ with positive weight vector $\lambda \in \mathbb{R}_+$ corresponds to an efficient solution of $(c_1, \dots, c_k, \mathcal{X})$ (see Theorem 2.6) is often used algorithmically.

In the bicriteria case, neighbouring supported nondominated points yield triangles in objective space in which remaining unsupported nondominated points can only be located. Let the set of supported nondominated points $\mathcal{N}_s$ be given by $\mathcal{N}_s = \{y^1, \dots, y^\ell\} \subset \mathbb{R}^2$ with $y^i_1 < y^{i+1}_1$ for $i \in [\ell - 1]$. Let $T_{i,i+1}$, for $i \in [\ell - 1]$, be the triangle given by the three points y^i, y^{i+1} and (y^{i+1}_1, y^i_2), see Figure 2.6. Then, any unsupported nondominated point $y \in \mathcal{N}_u$ is located in some triangle. Otherwise, y would either be supported nondominated or dominated.

In order to compute the entire set of nondominated points $\mathcal{N}$ of a bicriteria IP, the set of supported nondominated points $\mathcal{N}_s$ is computed in a first phase via a weighted sum method. Then, in a second phase, arising triangles which potentially contain unsupported nondominated points are investigated (e.g. see [79, 65]).

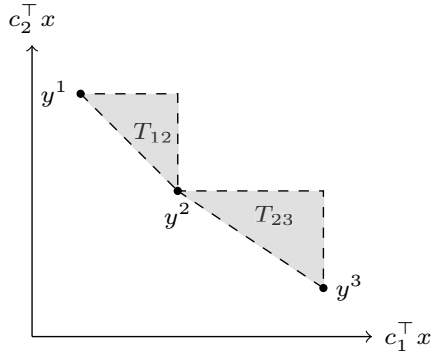

Figure 2.6: Supported nondominated points y^1, y^2, y^3 yielding triangles T_{12} and T_{23} where potential unsupported nondominated points can only be located.

2.3.3 *Weighted Chebyshev scalarisation method*

The weighted Chebyshev scalarisation approach (e.g. see [66]) makes use of Theorem 2.11 and transformation (2.6). In order to avoid the generation of weakly efficient solutions *augmented* weighted Chebyshev objectives might be considered [75, 23]. For a recent and detailed review of the weighted Chebyshev method and different variants see [22, Section 2.3].

2.3.4 *ϵ-constraint method*

The ϵ-constraint method goes back to Haimes et al. [40]. The multicriteria optimisation problem $(c_1, \ldots, c_k, \mathcal{X})$ is transformed into a single objective problem by imposing bounds with regard to some $\epsilon \in \mathbb{R}^k$ on all but one objective yielding the problem

$$\begin{aligned} \min\ & c_i^\top x \\ \text{s.t. } & c_j^\top x \leq \epsilon_j \ \text{ for all } j \in [k] \setminus \{i\}, \\ & x \in \mathcal{X}. \end{aligned} \tag{2.10}$$

It can be shown that an optimal solution of (2.10) is weakly efficient. Moreover, every efficient solution $x^* \in \mathcal{X}$ is an optimal solution of (2.10) for $\epsilon = Cx^*$ and for any $i \in [k]$.

Özlen et al. [58, 7] propose a recursive algorithm that can be considered to be a generalisation of the ϵ-constraint method.

2.3.5 *Branch-and-bound methods*

In single objective optimisation a branch of the enumeration tree is checked against computed bounds on the optimal objective value. A branch can be discarded if all solutions corresponding to it cannot produce a better objective value than the currently best one. In contrast, multicriteria branch-and-bound methods keep track of bound *sets* which yield upper and lower bounds on the values of nondominated points while attempting to discard regions of the enumeration

tree that contain only inefficient solutions. For a recent overview of multicriteria branch-and-bound methods see [63].

2.3.6 *Tricriteria integer programs*

Dächert et al. [24] present a theoretical upper bound on the number of scalarisations needed to solve discrete tricriteria optimisation problems. Independently, Boland et al. [16] present an L-shape search method that solves tricriteria problems efficiently from a practical point of view. The basic idea of the algorithm works as follows: The method maintains a priority queue of two-dimensional rectangles which might still contain projections of nondominated points. A rectangle is explored by finding all nondominated points with a projection in it. A nondominated point with a projection in the currently investigated rectangle induces an L-shape in this rectangle which is subsequently searched. Either the L-shape is shown to contain no projected nondominated point or a nondominated point with a projection in it is found. If the first case holds, then the L-shape can be discarded. If the latter case holds, the found nondominated point induces a new rectangle which is added to the priority queue. Boland et al. mention two important factors for achieving good computational results: Firstly, keeping the size of single objective subproblems small by working with rectangles and L-shapes only. Secondly, exploring different parts of the feasible domain early in the algorithm.

2.4 WORST-CASE NUMBER OF NONDOMINATED POINTS

Complexity results for single objective optimisation problems usually refer to the question in what *time* (in the input size of the given problem) can an optimal solution be found. Given that we are interested in a set of nondominated points, what is the *complexity* of solving a multicriteria optimisation problem? Should we refer to the size of the set of nondominated points which we are interested in or should we refer to the complexity to compute a single nondominated point or (a combination of) both?

For a recent discussion of *NP-hardness* in the context of multicriteria optimisation problems we refer the reader to [14].

For what follows we consider the size of the set of nondominated points as our basic *measure* of complexity. Provided that the set of points that we are interested in can be exponential in the input size of the given problem, we will deduce that we can generally not expect to solve our multicriteria optimisation problems efficiently.

We will distinguish between the IP case and the LP case. Since a multicriteria LP generally yields an infinite number of nondominated points which can be represented by the corresponding nondominated vertices (see Chapter 3), we are interested in the size of the set of nondominated vertices in the LP case. It will turn out that already in the bicriteria case the number of nondominated vertices (in the LP

case) and nondominated points (in the IP case), respectively, can be exponential in the input size of the problem.

2.4.1 *Bicriteria linear programs*

As noted in Remark 2.15, Murty [55] showed that the number of breakpoints of $\min_{x\in\mathcal{X}}(c_1+\lambda c_2)^\top x$ can be exponential in the input size. Combining this with Isermann's result that a solution is efficient if and only if it is supported (see Theorem 2.9), we get that the number of nondominated vertices of bicriteria linear programs can be exponential in the input size. Disser and Skutella [28] show an example in the context of network flows with an exponential number of nondominated points.

Aside from the above examples, in this subsection we present a family of bicriteria linear programs whose feasible domains are given by *Goldfarb* cubes. For more details about deformed products and, in particular, Goldfarb cubes see [2] and Section 4.3 therein.

Definition 2.16. (Goldfarb cube) Let $n \in \mathbb{N}_+$, $\epsilon < \frac{1}{2}$, and $\gamma < \frac{1}{4}\epsilon$ be given parameters. The n-dimensional Goldfarb cube $G(n)$ is defined by the following inequalities:

$$\begin{aligned} 0 \leq \quad & x_1 & & \leq 1 \\ \epsilon x_1 \leq \quad & x_2 & & \leq 1-\epsilon x_1 \\ \epsilon(x_i-\gamma x_{i-1}) \leq \quad & x_{i+1} & & \leq 1-\epsilon(x_i-\gamma x_{i-1}) \quad \text{for } 2\leq i\leq n. \end{aligned} \tag{2.11}$$

Goldfarb cubes have the property that their two-dimensional projection yields a polytope with exponentially many vertices.

Theorem 2.17. ([36],[2, Theorem 4.4]) The projection $\pi : G(n) \to \mathbb{R}^2$ given by $\pi(x) = (x_{n-1}, x_n)$ has 2^n vertices.

We can adapt the result and corresponding proof of Theorem 2.17 in order to show the next result.

Theorem 2.18. Let $n \in \mathbb{N}_+$ be a positive integer and let the projection $\pi : G(n) \to \mathbb{R}^2$ be given by $\pi(x) = (x_{n-1}, x_n)$. Then, the image $\mathcal{Y} \subset \mathbb{R}^2$ of

$$\max_{x\in G(n)} \pi(x)$$

has 2^{n-1} nondominated vertices.

Proof. Note that $G(n)$ is combinatorially equivalent to the n-dimensional cube $[0,1]^n$ (see [36, Theorem 1]). A vertex $x^* \in G(n)$ is a solution of (2.11) where each x_i^*, for $i = 1,\dots,n$, is either active at the lower bound or active at the upper bound of the corresponding i-th inequality. Let $U \subset G(n)$ be the set containing any vertex $x^* \in G(n)$ such that x_n^* is active at the upper bound of the corresponding inequality. Observe that U has cardinality 2^{n-1}. By following the proof of Theorem 4.4 in [2], we can show that every solution $x^* \in U$ corres-

ponds to a nondominated vertex of $\mathcal{Y}$. For any solution $x^* \in U$, let A_{x^*} be the $n \times n$-matrix whose rows are the facet normals at x^*:

$$A_{x^*} = \begin{pmatrix} \sigma_1 & & & & & & \\ \epsilon & \sigma_2 & & & & & \\ -\epsilon\gamma & \epsilon & \sigma_3 & & & & \\ & -\epsilon\gamma & \epsilon & \sigma_4 & & & \\ & & -\epsilon\gamma & \ddots & \ddots & & \\ & & & \ddots & \ddots & \sigma_{n-1} & \\ & & & & -\epsilon\gamma & \epsilon & \sigma_n \end{pmatrix}$$

where $\sigma_i = -1$ if x_i^* is active at the lower bound of the corresponding i-th inequality and $\sigma_i = 1$ otherwise. Note that $\sigma_n = 1$ for each $x^* \in U$. We get that

$$A_{x^*}x \leq A_{x^*}x^* \text{ is valid for all } x \in G(n)$$

with equality in all components if and only if $x = x^*$. Now let us define a row vector $\alpha^\top = (\alpha_1, \ldots, \alpha_n)$ recursively via

$$\alpha_1 = 1,\ \alpha_2 = \frac{2}{\epsilon},\ \alpha_{i+2} = \frac{1}{\gamma\epsilon}(\epsilon\alpha_{i+1} + \sigma_i\alpha_i).$$

We verify that $\alpha_{i+1} \geq \frac{2}{\epsilon}\alpha_i > 0$ for all $i \geq 1$. It holds that $\alpha_2 = \frac{2}{\epsilon}\alpha_1 = \frac{2}{\epsilon} > 0$ and

$$\alpha_3 = \frac{1}{\gamma\epsilon}(\epsilon\alpha_2 + \sigma_1\alpha_1) \geq \frac{1}{\gamma\epsilon}(\epsilon\alpha_2 - \alpha_1) = \frac{1}{\gamma\epsilon}(\epsilon\alpha_2 - \frac{\epsilon}{2}\alpha_2) = \frac{1}{2\gamma}\alpha_2 \geq \frac{2}{\epsilon}\alpha_2.$$

For $i \geq 3$, we get

$$\alpha_{i+2} \geq \frac{1}{\gamma\epsilon}(\epsilon\alpha_{i+1} - \alpha_i) \geq \frac{1}{\gamma\epsilon}(\epsilon\alpha_{i+1} - \frac{\epsilon}{2}\alpha_{i+1}) = \frac{1}{2\gamma}\alpha_{i+1} \geq \frac{2}{\epsilon}\alpha_{i+1} > 0$$

by using induction. Since $\alpha^\top$ is a positive vector, it holds that

$$\alpha^\top A_{x^*}x \leq \alpha^\top A_{x^*}x^* \text{ for all } x \in G(n)$$

with equality if and only if $x = x^*$. From

$$\alpha^\top A_{x^*} = (0, \ldots, 0, \sigma_{n-1}\alpha_{n-1} + \epsilon\alpha_n, \alpha_n)$$

we get that

$$(\sigma_{n-1}\alpha_{n-1} + \epsilon\alpha_n, \alpha_n)\pi(x) = \alpha^\top A_{x^*}x \leq \alpha^\top A_{x^*}x^*$$

with equality if and only if $x = x^*$. Since

$$\sigma_{n-1}\alpha_{n-1} + \epsilon\alpha_n \geq -\alpha_{n-1} + \epsilon\alpha_n \geq -\alpha_{n-1} + 2\alpha_{n-1} = \alpha_{n-1} > 0$$

and $\alpha_n > 0$, we get that $\pi(x^*)$ is a nondominated vertex of $\mathcal{Y}$ by Theorem 3.10 (in the version for maximisation problems). Since $x^* \in U$ is the only solution of $G(n)$ that projects to $\pi(x^*)$ and $|U| = 2^{n-1}$, the result follows. □

2.4.2 *Bicriteria integer programs*

Hansen was among the first who investigated different versions of multicriteria shortest s-t-path problems and showed the *intractability* of the bicriteria shortest s-t-path problem [42]. Recently, Bökler [14] argued that *NP-hardness* might not be well-suited for investigating the complexity status of multicriteria optimisation problems. As mentioned in the beginning of Section 2.4, we consider the size of $\mathcal{N}$ as our basic complexity measure.

Definition 2.19. (Shortest s-t-path problem) Let $G = (V, A)$ be a directed graph with vertex set V, arc set A, two distinguished vertices $s, t \in V$ and weights $c : A \to \mathbb{R}$. The *Shortest s-t-path problem* is to find an s-t-path of minimum weight.

Consider the instance of the bicriteria shortest s-t-path problem depicted in Figure 2.7. The graph $G = (V, A)$ has $|V| = 5$ vertices and $|A| = 2 \cdot 4$ arcs. Given $c_1, c_2 : A \to \mathbb{N}$ as depicted in Figure 2.7 we get that all of the 2^4 s-t-paths are efficient, see Figure 2.8. We can generalise this instance to a graph $G = (V, A)$ with $|V| = n + 1$ for $n \in \mathbb{N}_+$.

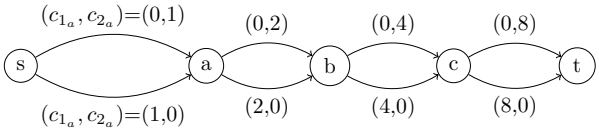

Figure 2.7: Instance of a bicriteria shortest s-t-path problem for which each feasible s-t-path is efficient.

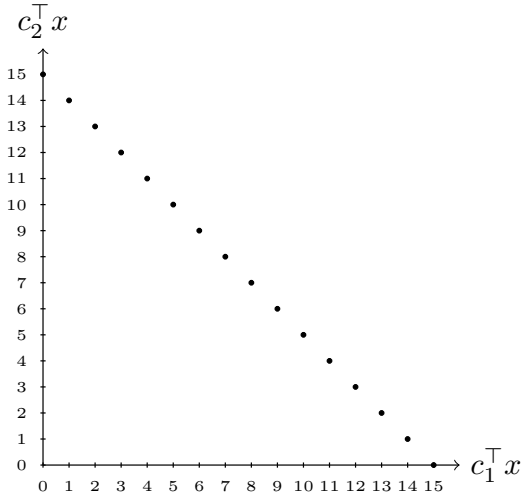

Figure 2.8: Image in objective space of the bicriteria shortest s-t-path problem instance depicted in Figure 2.7.

3

COMPUTING NONDOMINATED VERTICES FOR MULTICRITERIA LINEAR PROGRAMS

> [...] as with the swans that glide on the Thames, a serene surface conceals some frantic paddling underneath.
>
> *Buttonwood*, The Economist

3.1 INTRODUCTION

In this chapter we investigate how to compute nondominated vertices of $\mathcal{Y}$ via weight sets. Benson and Sun were the first to present and validate a basic weight set decomposition method where weight vectors that are used in a weighted sum objective are put into a one-to-one correspondence with nondominated vertices [9, 11].

Our contribution in this chapter is to show that by *lifting* the weight sets to a higher dimensional space the weight space decomposition method can be interpreted in the context of polyhedral combinatorics as a *cutting plane* approach. It can be elegantly presented using only basic results for polyhedra and profits from the possibility to use different vertex enumeration algorithms.

3.2 REPRESENTATION OF POLYHEDRA

3.2.1 *Resolution theorem*

Geometrically, a *vertex* of a polyhedron P is a unique optimal solution with regard to some linear objective and feasible domain P.

Definition 3.1. (vertex) Let $P \subseteq \mathbb{R}^k$ be a polyhedron. A point $y^* \in P$ is a *vertex* of P if there exists some $\omega \in \mathbb{R}^k$ such that

$$\omega^\top y^* < \omega^\top y \text{ for all } y \in P \setminus \{y^*\} .$$

In order to formulate the well-known resolution theorem for polyhedra we introduce the following basic definitions.

Definition 3.2. (convex combination) A point $y \in \mathbb{R}^k$ is called a *convex combination* of $y^1, \dots, y^\ell \in \mathbb{R}^k$ if

$$y = \sum_{i=1}^{\ell} \lambda_i y^i \text{ where } \lambda \in \mathbb{R}^\ell_{\geq 0} \text{ with } \mathbb{1}^\top \lambda = 1 .$$

The set of all convex combinations of $y^1, \dots, y^\ell \in \mathbb{R}^k$ is called *convex hull* of $y^1, \dots, y^\ell$ and denoted by $conv(y^1, \dots, y^\ell)$.

Definition 3.3. (conic combination) A point $y \in \mathbb{R}^k$ is called a *conic combination* of $y^1, \ldots, y^\ell \in \mathbb{R}^k$ if

$$y = \sum_{i=1}^{\ell} \omega_i y^i \text{ where } \omega \in \mathbb{R}^\ell_{\geq 0} .$$

The set of all conic combinations of $y^1, \ldots, y^\ell \in \mathbb{R}^k$ is called *conic hull* of $y^1, \ldots, y^\ell$ and denoted by $co(y^1, \ldots, y^\ell)$.

Definition 3.4. (Minkowski sum) Let $A, B \subseteq \mathbb{R}^k$ be two nonempty sets. The *Minkowski sum* $A + B \subseteq \mathbb{R}^k$ is defined as

$$A + B = \{a + b : a \in A,\ b \in B\} .$$

The resolution theorem for polyhedra, also often referred to as the *Minkowski-Weyl* theorem for convex polyhedra, states that every polyhedron is finitely generated and every finitely generated set is a polyhedron.

Theorem 3.5. ([12, Theorem 4.15]) Let $P = \{x \in \mathbb{R}^k : a_i^\top x \geq b_i \text{ for } i = 1, \ldots, m\}$ be a polyhedron with at least one vertex. Let $\{x^1, \ldots, x^s\}$ be the entire set of vertices of P and let $\{q^1, \ldots, q^t\}$ be a complete set of extreme rays of P. Then,

$$P = conv(x^1, \ldots, x^s) + co(q^1, \ldots, q^t) .$$

3.2.2 *Polyhedral cones*

Definition 3.6. (polyhedral cone) A polyhedron $C \subseteq \mathbb{R}^k$ is called *polyhedral cone* if $0 \in C$ and for every $x \in C$ and every $\lambda \geq 0$ we have $\lambda x \in C$.

Note that only the origin can possibly be a vertex of a cone. We can *homogenise* a general polyhedron $P \subseteq \mathbb{R}^k$ to a polyhedral cone $C_P \subseteq \mathbb{R}^{k+1}$ by mapping $x \in \mathbb{R}^k$ to $\binom{1}{x} \in \mathbb{R}^{k+1}$. If the polyhedron $P \subseteq \mathbb{R}^k$ is given by

$$P = \{x \in \mathbb{R}^k : a_i^\top x \geq b_i \text{ for } i = 1, \ldots, m\},$$

then define the polyhedral cone $C_P \subseteq \mathbb{R}^{k+1}$ by

$$C_P = \{\binom{z}{x} \in \mathbb{R}^{k+1} :\ z \geq 0,\ b_i z - a_i^\top x \leq 0 \text{ for } i = 1, \ldots, m\}.$$

It holds that $x \in P$ if and only if $\binom{1}{x} \in C_P$. If the polyhedron $P \subseteq \mathbb{R}^k$ is given by

$$P = conv(x^1, \ldots, x^s) + co(q^1, \ldots, q^t)$$

where $\{x^1, \ldots, x^s\}$ is the set of vertices of P and $\{q^1, \ldots, q^t\}$ is a complete set of extreme rays of P, then define the polyhedal cone $C_P \subseteq \mathbb{R}^{k+1}$ by

$$C_P = \{\begin{pmatrix} \mathbb{1}^\top \lambda \\ \sum_{i=1}^s \lambda_i x^i + \sum_{j=1}^t \omega_j q^j \end{pmatrix} :\ \lambda \in \mathbb{R}^s_{\geq 0},\ \omega \in \mathbb{R}^t_{\geq 0}\}.$$

Again, we obtain $x \in P$ if and only if $\binom{1}{x} \in C_P$. For the special case of polyhedral cones, Theorem 3.5 reads as follows:

Theorem 3.7. ([83, Theorem 1.3]) A polyhedral cone $C = co(r^1, \ldots, r^t) \subseteq \mathbb{R}^k$ is a finitely generated combination of vectors $r^1, \ldots, r^t \in \mathbb{R}^k$ if and only if it is a finite intersection of closed linear halfspaces $C = \{x \in \mathbb{R}^k : Ax \geq 0\}$ for some $A \in \mathbb{R}^{m \times k}$.

We call the set $\{r^1, \ldots, r^t\}$ a *generating set* and the matrix A a *representation matrix*.

3.2.3 *Vertex enumeration*

Theorem 3.7 in Section 3.2.2 tells us that a polyhedral cone $C = \{x \in \mathbb{R}^k : Ax \geq 0\}$ given via a finite intersection of closed half-spaces can simultaneously be represented as $C = co(r^1, \ldots, r^t)$ by a finite generating set $R = \{r^1, \ldots, r^t\}$. Motzkin et al. introduced the term *double description (DD)* pair for (A, R) [53]. Given a finite intersection of half-spaces representing the lower hull of $\mathcal{Y}$, our motivation for the computation of a DD pair stems from the interest to compute the set of nondominated vertices of $\mathcal{Y}$.

Vertex enumeration, i.e. the computation of a generating set given a representation matrix, is an important problem in computational geometry. There are several implementations and codes available for computing DD pairs, e.g. ppl [20], Normaliz [57], lrs [4]. Most of them are based on either of the following approaches: pivoting using reverse search [5] or the Fourier-Motzkin double description method [34]. The difficulty of solving vertex enumeration problems can vary enormously depending (among other things) on the size of the polyhedron and the *level of degeneracy*. Avis and Jordan report computational results showing that all investigated codes perform differently on different polytopes [6]. However, the availability of different codes which we can use as a subroutine in order to compute vertices of a polytope is the important point for us in this thesis.

In the remainder of this subsection we present a primitive version for computing a DD pair based on the Fourier-Motzkin method. Note that we base our presentation on [34]. Given a representation matrix $A \in \mathbb{R}^{m \times k}$ the DD method is an incremental algorithm to construct a finite generating set $R \subseteq \mathbb{R}^k$ such that (A, R) is a DD pair. For what follows we assume that the polyhedral cone $C = \{x \in \mathbb{R}^k : Ax \geq 0\}$ is pointed, i.e. it contains no line. We also assume that $Ax \geq 0$ is irredundant.

Let L be a subset of the row indices $\{1, \ldots, m\}$ of A and let A_L denote the submatrix of A consisting of the rows in L. Assuming that we have already found a generating set R_L for $C(A_L) = \{x \in \mathbb{R}^k : A_L x \geq 0\}$, we select any row index $i \notin L$ and aim to construct a DD pair $(A_{L \cup \{i\}}, R_{L \cup \{i\}})$.

Selecting an index $i \notin L$, the corresponding inequality $a_i^\top x \geq 0$ partitions $\mathbb{R}^k$ into three parts:

$$\begin{aligned} H_i^+ &= \{x \in \mathbb{R}^k : a_i^\top x > 0\} \\ H_i^0 &= \{x \in \mathbb{R}^k : a_i^\top x = 0\} \\ H_i^- &= \{x \in \mathbb{R}^k : a_i^\top x < 0\}. \end{aligned} \tag{3.1}$$

Similarly, the generating set R_L is partitioned into three sets:

$$\begin{aligned} R_L^+ &= \{r \in R_L : a_i^\top r > 0\} \\ R_L^0 &= \{r \in R_L : a_i^\top r = 0\} \\ R_L^- &= \{r \in R_L : a_i^\top r < 0\}. \end{aligned} \tag{3.2}$$

To construct a generating set $R_{L\cup\{i\}}$ from R_L, we generate new rays lying on the i-th hyperplane H_i^0 by using an appropriate combination of each *positive* ray $q \in R_L^+$ and each *negative* ray $r \in R_L^-$.

Lemma 3.8. ([34, Lemma 3]) Let (A_L, R_L) be a DD pair and let $i \in [m]$ be a row index of $A \in \mathbb{R}^{m\times k}$ such that $i \notin L$. Then, the pair $(A_{L\cup\{i\}}, R_{L\cup\{i\}})$ is a DD pair where $R_{L\cup\{i\}} = R_L^+ \cup R_L^0 \cup \tilde{R}$ with

$$\tilde{R} = \{(a_i^\top q)r - (a_i^\top r)q : q \in R_L^+,\ r \in R_L^-\}.$$

By applying Lemma 3.8 consecutively, we can formulate a primitive algorithm for computing a DD pair. Note that it is not difficult to obtain an initial DD pair when $|L| = 1$.

Algorithm 3.1 Primitive version of the double description method

Obtain initial DD pair (A_L, R_L)
while $L \neq [m]$ **do**
  Select $i \in [m] \setminus L$
  Construct $(A_{L\cup\{i\}}, R_{L\cup\{i\}})$ from (A_L, R_L) using Lemma 3.8
  Set $L := L \cup \{i\}$

Note that a straightforward implementation of Algorithm 3.1 will not be practical as the set $\tilde{R}$ in Lemma 3.8 might easily grow to an intractable size. Practical codes for the vertex enumeration problem usually incorporate more structural insights of the underlying mathematical problem, e.g. many of the rays in $\tilde{R}$ generated in intermediate steps can be neglected, as well as parallel processing capabilities of modern central processing units. For more practical versions of the double description method see [34, Chapter 3].

3.3 REPRESENTATION OF NONDOMINATED POINTS

Given a multicriteria linear program, the set of nondominated points $\mathcal{N}$ might comprise an infinite number of points due to the convexity of $\mathcal{Y}$, see Figure 3.1. Analogous to Theorem 3.5 in Section 3.2.1, we show in this section that the entire set of nondominated points can be represented by the set of nondominated vertices of $\mathcal{Y}$ and a complete set of nondominated extreme rays.

Theorem 3.9. Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria LP and assume the corresponding image $\mathcal{Y} \subset \mathbb{R}^k$ has at least one vertex. Let $\{y^1, \ldots, y^n\}$ be the entire set of nondominated vertices of $\mathcal{Y}$ and let $\{q^1, \ldots, q^m\}$ be a complete set of nondominated extreme rays of $\mathcal{Y}$. If $y \in \mathcal{Y}$ is a nondominated point, then y can be written as

$$y = \sum_{i=1}^{n} \lambda_i y^i + \sum_{j=1}^{m} \omega_j q^j \quad \text{where } \lambda \in \mathbb{R}^n_{\geq 0},\ \omega \in \mathbb{R}^m_{\geq 0}, \text{ and } \mathbb{1}^\top \lambda = 1.$$

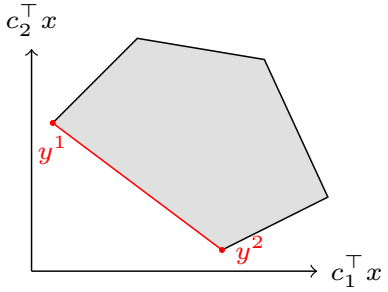

Figure 3.1: Image of a bicriteria linear programming problem where any of the infinitely many nondominated points can be obtained by a convex combination of the nondominated vertices y^1 and y^2.

Proof. Let $\bar{y}^1, \dots, \bar{y}^s$ be the vertices (not necessarily all nondominated) of $\mathcal{Y}$ and let $\{\bar{q}^1, \dots, \bar{q}^t\}$ be a complete set of extreme rays (not necessarily all nondominated) of $\mathcal{Y}$. Note that there cannot be an ideal ray, otherwise the set of nondominated points would be empty. By Theorem 3.5, y can be written as

$$y = \sum_{i=1}^{s} \bar{\lambda}_i \bar{y}^i + \sum_{j=1}^{t} \bar{\omega}_j \bar{q}^j \tag{3.3}$$

where $\bar{\lambda} \in \mathbb{R}^s_{\geq 0}$, $\bar{\omega} \in \mathbb{R}^t_{\geq 0}$ and $\mathbb{1}^\top \bar{\lambda} = 1$. Suppose that y cannot be represented by considering only nondominated vertices and nondominated extreme rays in (3.3). Let us assume w.l.o.g. that the dominated vertices used in (3.3) are given by $\bar{y}^1, \dots, \bar{y}^{\ell_1}$ with $\ell_1 \leq s$. Then, for all $i \in [\ell_1]$, there is $z^i \in \mathcal{Y}$ such that $\bar{y}^i$ is dominated by z^i. Similarly, assume w.l.o.g. that the dominated extreme rays used in (3.3) are given by $\bar{q}^1, \dots, \bar{q}^{\ell_2}$ with $\ell_2 \leq t$. Now, consider the point

$$z = \sum_{i=1}^{\ell_1} \bar{\lambda}_i z^i + \sum_{i=\ell_1+1}^{s} \bar{\lambda}_i \bar{y}^i + \sum_{j=\ell_2+1}^{t} \bar{w}_j \bar{q}^j$$

where $\bar{\lambda}$, $\bar{\omega}_{\ell_2+1}, \dots, \bar{\omega}_t$ are chosen exactly as in (3.3). It holds that $z \in \mathcal{Y}$. Furthermore, z dominates y contradicting the assumption that y is a nondominated point. □

In general, the set of nondominated points $\mathcal{N}$ will not be a convex set. In other words, taking the Minkowski sum of the convex hull of nondominated vertices of $\mathcal{Y}$ and the conic hull of nondominated extreme rays of $\mathcal{Y}$ will generally not coincide with $\mathcal{N}$. However, for what follows we assume that after computing the nondominated vertices of $\mathcal{Y}$ (and nondominated extreme rays) computing the *lower envelope* of the Minkowski sum in order to represent the entire set of nondominated points $\mathcal{N}$ can be done by some post-processing algorithm.

3.4 COMPUTING NONDOMINATED VERTICES

In the previous section we have shown that any nondominated point of a multicriteria LP can be represented by an appropriate combina-

tion of nondominated vertices and nondominated rays. In this section we show how nondominated vertices can be computed.

3.4.1 *Characterisation of nondominated vertices*

Given a multicriteria LP, the corresponding image $\mathcal{Y}$ is a polyhedron since it is a linear transformation of the feasible domain $\mathcal{X}$ [8, Theorem 3.1]. In Definition 3.1 in Section 2.2.1, a vertex of a polyhedron was characterised as the unique minimiser with regard to some weight vector. The following result shows that a nondominated vertex is the unique minimiser with regard to some positive weight vector.

Theorem 3.10. Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria LP and let $\mathcal{Y} \subseteq \mathbb{R}^k$ be the corresponding image. The point $y^* \in \mathcal{Y}$ is a nondominated vertex of $\mathcal{Y}$ if and only if there exists some $\omega \in \mathbb{R}^k_+$ such that

$$\omega^\top y^* < \omega^\top y \text{ for all } y \in \mathcal{Y} \setminus \{y^*\}.$$

Proof. "⇒" Let $y^* \in \mathcal{Y}$ be a nondominated vertex of $\mathcal{Y}$ and let $x^* \in \mathcal{X}$ be an efficient preimage corresponding to y^*. By Theorem 2.9, there exists $\lambda \in \mathbb{R}^k_+$ such that $\sum_{i=1}^k \lambda_i c_i^\top x^* \leq \sum_{i=1}^k \lambda_i c_i^\top x$ for all $x \in \mathcal{X}$. The latter implies that $\lambda^\top y^* \leq \lambda^\top y$ for all $y \in \mathcal{Y}$. Moreover, by Definition 3.1, there exists $\gamma \in \mathbb{R}^k$ such that $\gamma^\top y^* < \gamma^\top y$ for all $y \in \mathcal{Y} \setminus \{y^*\}$. Now, if $\gamma \in \mathbb{R}^k_{\geq 0}$, then we set $\omega = \gamma + \lambda$ and get that $\omega \in \mathbb{R}^k_+$ with $\omega^\top y^* < \omega^\top y$ for all $y \in \mathcal{Y} \setminus \{y^*\}$. Otherwise, we set $\omega = \gamma + \alpha\lambda$ where α is given by $\alpha = \max_{i:\gamma_i<0} \frac{1-\gamma_i}{\lambda_i} > 0$. Then, for all $i \in [k]$,

$$\omega_i = \gamma_i + \max_{i:\gamma_i<0} \frac{1-\gamma_i}{\lambda_i}\lambda_i \geq \gamma_i + (\frac{1-\gamma_i}{\lambda_i})\lambda_i = 1$$

holds. In other words, we get $\omega \in \mathbb{R}^k_+$. Moreover, for all $y \in \mathcal{Y} \setminus \{y^*\}$,

$$\omega^\top y^* = \gamma^\top y^* + \alpha\lambda^\top y^* < \gamma^\top y + \alpha\lambda^\top y = \omega^\top y$$

holds.
"⇐" Definition 3.1 and Theorem 2.9 are fulfilled. □

The next result is well-known and straightforward.

Theorem 3.11. ([11, Theorem 3]) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria LP. If $x^* \in \mathcal{X}$ is a unique optimal solution to $(\lambda^\top C, \mathcal{X})$ for some $\lambda \in \mathbb{R}^k_+$, then the corresponding image $y^* = Cx^*$ is a nondominated vertex of $\mathcal{Y}$.

3.4.2 *The bounded case*

Given an instance $(c_1, \ldots, c_k, \mathcal{X})$ and a weight vector $\lambda \in \mathbb{R}^k_+$ such that $\lambda^\top C$ is bounded over $\mathcal{X}$, we can compute a nondominated vertex $y^* \in \mathcal{Y}$ such that $\lambda^\top y^* \leq \lambda^\top y$ for all $y \in \mathcal{Y}$ by applying Algorithm 3.2. The first step is to solve $(\lambda^\top C, \mathcal{X})$. If the computed optimal solution x^* is not a unique optimal solution, then the constraint $\lambda^\top Cx = \lambda^\top Cx^*$ is added to $\mathcal{X}$ and the algorithm proceeds by

optimising c_1 over $\mathcal{X}$. Let x^1 be the corresponding computed optimal solution. If x^1 is a unique optimal solution, we are done. Otherwise, we add the constraint $c_1^\top x = c_1^\top x^1$ to $\mathcal{X}$ and proceed by optimising c_2 over $\mathcal{X}$. This procedure is repeated until a unique optimal solution is found or until the objectives are exhausted.

Algorithm 3.2 Computing a nondominated vertex

Input: $(c_1, \ldots, c_k, \mathcal{X})$, weight vector $\lambda \in \mathbb{R}^k_+$ such that $(\lambda^\top C, \mathcal{X})$ is bounded
Output: nondominated vertex $y^* \in \mathcal{Y}$ with $\lambda^\top y^* \leq \lambda^\top y$ for all $y \in \mathcal{Y}$

$x^* \leftarrow \arg\min_{x \in \mathcal{X}} \lambda^\top C x$
if x^* is unique optimal solution **then**
    return Cx^*
else
    add constraint $\lambda^\top C x = \lambda^\top C x^*$ to $\mathcal{X}$
    for $i = 1, \ldots, k-1$ **do**
        $x^* \leftarrow \arg\min_{x \in \mathcal{X}} c_i^\top x$
        if x^* is unique optimal solution **then**
            return Cx^*
        else
            add constraint $c_i^\top x = c_i^\top x^*$ to $\mathcal{X}$
    $x^* \leftarrow \arg\min_{x \in \mathcal{X}} c_k^\top x$
    return Cx^*

The next result will be useful for proving the correctness of Algorithm 3.2.

Proposition 3.12. Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria LP and let $\mathcal{Y}$ be the corresponding image. Let $\lambda \in \mathbb{R}^k_+$ and assume that $\bar{x} \in \mathcal{X}$ is an optimal solution to $(\lambda^\top C, \mathcal{X})$. Define $\mathcal{X}_0 = \{x \in \mathcal{X} : \lambda^\top C x = \lambda^\top C \bar{x}\}$ and $\mathcal{X}_i = \{x \in \arg\min_{x \in \mathcal{X}_{i-1}} c_i^\top x\}$ for all $i \in [k]$.
a) If $i \in [k]$ such that $|\mathcal{X}_i| = 1$, then the image Cx^* of $x^* \in \mathcal{X}_i$ corresponds to a nondominated vertex of $\mathcal{Y}$.
b) If $x^* \in \mathcal{X}_k$, then Cx^* corresponds to a nondominated vertex of $\mathcal{Y}$.

Proof. Note that $\mathcal{X}_k \subseteq \ldots \subseteq \mathcal{X}_0 \subseteq \mathcal{X}$. Hence, $x^* \in \mathcal{X}_i$ for some $i \in [k]$ implies $Cx^* \in \mathcal{Y}$. For what follows let $y^* \in \mathcal{Y}$ be given by $y^* = Cx^*$. $x^* \in \mathcal{X}_i$ for some $i \in [k]$ implies that x^* is an optimal solution to $(\lambda^\top C, \mathcal{X})$. Hence, by Theorem 2.9, y^* is nondominated in a) and in b). What remains to be shown is that y^* is a vertex of $\mathcal{Y}$. We will prove a) and b) in similar ways as follows: We suppose, to the contrary, that y^* is a convex combination of two distinct points $Cx^1, Cx^2 \in \mathcal{Y}$ and show in this case that the preimages x^1 and x^2 are both contained in $\mathcal{X}_0$. Provided the latter, we proceed by considering two mutually exclusive cases regarding the containment of x^1 and x^2 in $\mathcal{X}_i$ for any $i \in [k]$ and show that both cases lead to contradictions.
a) Assume we have $i \in [k]$ such that $|\mathcal{X}_i| = 1$. Let x^* be the unique solution in $\mathcal{X}_i$ and suppose that y^* is not a vertex of $\mathcal{Y}$. Then, by the equivalence of vertices and extreme points, there are distinct $y^1, y^2 \in \mathcal{Y}$ such that

$$y^* = \alpha y^1 + (1 - \alpha) y^2 \tag{3.4}$$

for some $\alpha \in (0,1)$. From (3.4) we get $\lambda^\top y^* = \alpha\lambda^\top y^1 + (1-\alpha)\lambda^\top y^2$. We show next that $\lambda^\top y^1 = \lambda^\top y^2$ holds. Suppose $\lambda^\top y^1 \neq \lambda^\top y^2$ and assume w.l.o.g. that $\lambda^\top y^1 < \lambda^\top y^2$. Then,

$$\lambda^\top y^* = \alpha\lambda^\top y^1 + (1-\alpha)\lambda^\top y^2 > \alpha\lambda^\top y^1 + (1-\alpha)\lambda^\top y^1 = \lambda^\top y^1 \quad (3.5)$$

holds. Since $y^1, y^2 \in \mathcal{Y}$ there is at least one feasible solution $x^1 \in \mathcal{X}$ with $y^1 = Cx^1$ and at least one feasible solution $x^2 \in \mathcal{X}$ with $y^2 = Cx^2$. Then, (3.5) implies that $\lambda^\top Cx^* > \lambda^\top Cx^1$ which contradicts the optimality of x^* for $(\lambda^\top C, \mathcal{X})$. Hence, the claim $\lambda^\top y^1 = \lambda^\top y^2$ is established. The latter equality and (3.4) imply

$$\lambda^\top y^* = \lambda^\top y^1 = \lambda^\top y^2. \quad (3.6)$$

From (3.6) we get $x^1, x^2 \in \mathcal{X}_0$. Now consider the following two mutually exclusive cases: 1) There is some $j \in [i]$ such that $x^1 \notin \mathcal{X}_j$ or $x^2 \notin \mathcal{X}_j$ or both. 2) $x^1, x^2 \in \mathcal{X}_i$ (implying $x^1, x^2 \in \mathcal{X}_\ell$ for $\ell = i, i-1, \dots, 0$).
Suppose 1) holds. Then, let $j \in [i]$ be the smallest index such that $x^1 \notin \mathcal{X}_j$ or $x^2 \notin \mathcal{X}_j$ and assume w.l.o.g. that $x^1 \notin \mathcal{X}_j$. Then, $x^1 \in \mathcal{X}_{j-1}$ but $x^1 \notin \mathcal{X}_j$ implies $c_j^\top x^* < c_j^\top x^1$. The latter and $y_j^* = \alpha y_j^1 + (1-\alpha)y_j^2$ given by (3.4) implies $y_j^* > y_j^2$. In other words, we get $c_j x^* > c_j x^2$ with $x^2 \in \mathcal{X}_{j-1}$ which implies $x^* \notin \mathcal{X}_j$ which is a contradiction to $x^* \in \mathcal{X}_i$.
Now suppose 2) holds. Since $Cx^1 = y^1 \neq y^2 = Cx^2$ implies $x^1 \neq x^2$, we get from $x^1, x^2 \in \mathcal{X}_i$ that $|\mathcal{X}_i| > 1$ holds. This is a contradiction to $|\mathcal{X}_i| = 1$.
Both cases, 1) and 2), lead to contradictions. Hence, our initial supposition that y^* is not a vertex cannot hold.
b) Suppose y^* is not a vertex of $\mathcal{Y}$. Then, by the equivalence of vertices and extreme points, there are distinct $y^1, y^2 \in \mathcal{Y}$ such that $y^* = \alpha y^1 + (1-\alpha)y^2$ for some $\alpha \in (0,1)$. By exactly the same reasoning as done in the proof of a), we can show that there are solutions $x^1, x^2 \in \mathcal{X}$ with $y^1 = Cx^1$, $y^2 = Cx^2$ and $x^1, x^2 \in \mathcal{X}_0$. Now consider the following two cases: 1) There is some $j \in [k]$ such that $x^1 \notin \mathcal{X}_j$ or $x^2 \notin \mathcal{X}_j$ or both. 2) $x^1, x^2 \in \mathcal{X}_k$.
Suppose 1) holds. Then, let $j \in [k]$ be the smallest index such that $x^1 \notin \mathcal{X}_j$ or $x^2 \notin \mathcal{X}_j$ (or both) and assume w.l.o.g. that $x^1 \notin \mathcal{X}_j$. Then, $x^1 \in \mathcal{X}_{j-1}$ but $x^1 \notin \mathcal{X}_j$ implies $c_j^\top x^* < c_j^\top x^1$. The latter and $c_j^\top x^* = \alpha c_j^\top x^1 + (1-\alpha)c_j^\top x^2$ given by (3.4) imply $c_j^\top x^* > c_j^\top x^2$. Since $x^2 \in \mathcal{X}_{j-1}$, we get $x^* \notin \mathcal{X}_j$ which is a contradiction to $x^* \in \mathcal{X}_k$.
Now suppose 2) holds. Since $x^1, x^2, x^* \in \mathcal{X}_k$, we get $x^1, x^2, x^* \in \mathcal{X}_i$ for $i = 0, \dots, k$. This implies that $c_i^\top x^* = c_i^\top x^1$ and $c_i^\top x^* = c_i^\top x^2$ for all $i \in [k]$ leading to a contradiction that y^1 and y^2 are distinct. Hence, our initial supposition that y^* is not a vertex cannot hold. □

Theorem 3.11 and Proposition 3.12 will serve as the two main ingredients for the proof of the following statement.

Theorem 3.13. Let $(c_1, \dots, c_k, \mathcal{X})$ be a multicriteria LP and let $\mathcal{Y}$ be the corresponding image. Let $\lambda \in \mathbb{R}^k_+$ be a positive vector and assume that $(\lambda^\top C, \mathcal{X})$ is feasible and bounded. Then, Algorithm 3.2 yields a nondominated vertex $y^* \in \mathcal{Y}$ such that $\lambda^\top y^* \leq \lambda^\top y$ for all $y \in \mathcal{Y}$.

Proof. If x^* is a unique solution in line 2 of Algorithm 3.2, then the returned image Cx^* corresponds to a nondominated vertex by Theorem 3.11. If x^* is a unique optimal solution in line 8, then the returned image Cx^* corresponds to a nondominated vertex by Proposition 3.12 a). If x^* is an optimal solution in line 12, then Cx^* corresponds to a nondominated vertex by Proposition 3.12 b). □

3.4.3 *The unbounded case*

How do we compute a nondominated vertex if $\mathcal{X}$ is not bounded and $\min_{x\in\mathcal{X}} c_i^\top x$ might be unbounded for some $i \in [k]$? Begin by solving $\min_{x\in\mathcal{X}} \mathbb{1}^\top Cx$. If the latter is bounded, then we can compute a nondominated vertex by calling Algorithm 3.2 with $\lambda = (1,\dots,1)$. Otherwise, let $r \in \mathbb{R}^n$ be a corresponding ray with $\mathbb{1}^\top Cr < 0$ and let $q = Cr \in \mathbb{R}^k$ be the image. If $q_i \leq 0$ for all $i \in [k]$ with $q_j < 0$ for some $j \in [k]$, then r is an ideal ray and $\mathcal{Y}$ does not contain any nondominated point. Otherwise, consider the set $\Lambda = \{\lambda \in \mathbb{R}^k_+ : \lambda^\top q > 0 \text{ for all } q \in Q\}$ where Q is the set of nondominated rays computed so far. The set Λ contains all potential weight vectors λ for which $(\lambda^\top C, \mathcal{X})$ might still achieve an optimal solution. By systematically choosing $\lambda \in \Lambda$, checking $(\lambda^\top C, \mathcal{X})$ for boundedness and extending Λ in case of a newly computed nondominated ray, we either find $\lambda \in \Lambda$ such that $(\lambda^\top C, \mathcal{X})$ is bounded and for which Algorithm 3.2 yields a nondominated vertex or we conclude with $\Lambda = \emptyset$.

3.5 WEIGHT SPACE DECOMPOSITION METHOD

In this section we deal with the challenge of computing all nondominated vertices for a given multicriteria LP (or computing the nondominated vertices of $conv(\mathcal{Y})$ for a given multicriteria IP, respectively). Note that we cannot directly apply a vertex enumeration algorithm (see Section 3.2.3) to $\mathcal{Y}$ since we do generally not have a representation matrix for the polyhedron $\mathcal{Y} = \{Cx : x \in \mathcal{X}\}$. Benson was the first to present an *outer approximation* algorithm for generating all nondominated vertices of a multicriteria LP [10]. The algorithm constructs a *simplex* S that contains the lower hull of $\mathcal{Y}$ throughout the algorithm and consecutively refines S until it coincides with the lower hull of $\mathcal{Y}$. In this regard, Benson's algorithm is an objective space method (see discussion in Section 2.3). Ehrgott et al. present a *dual* variant of Benson's algorithm [30]. In the remainder of this section we briefly review two weight space decompositin approaches (Section 3.5.1 and Section 3.5.2) and conclude this section with the presentation of a novel weight space polyhedron method (Section 3.5.3).

3.5.1 *Benson's and Sun's weight space decomposition method*

Instead of working directly in the objective space, Benson and Sun consider a weight space approach for computing the nondominated vertices of a multicriteria linear program [9, 11]. For $y \in \mathcal{Y}$ they consider the set $W(y)$ given by

$$W(y) = \{w \in \mathbb{R}^k : w^\top y \leq w^\top \bar{y} \text{ for all } \bar{y} \in \mathcal{Y}\}.$$

They show that

$$\mathbb{R}^k_+ = \bigcup_{i=1}^{\ell} (\mathbb{R}^k_+ \cap W(y^i))$$

where $\{y^1, \ldots, y^\ell\}$ is the entire set of nondominated vertices of $\mathcal{Y}$. Furthermore, they show that $W(y^i)$ has a nonempty *relative interior* for each vertex y^i and that the relative interior of $W(y^i)$ and the relative interior of $W(y^j)$ do not intersect for distinct vertices y^i and y^j.

Their basic approach is iterative and involves finding a new weight vector $w^i \in \mathbb{R}^k_+$ corresponding to an unexplored nondominated vertex $y^i \in \mathcal{Y}$ at each iteration i. Let W^i represent the subset of $\mathbb{R}^k_+$ remaining at the beginning of iteration i, i.e. $W^i = \mathbb{R}^k_+ \setminus \bigcup_{j=1}^{i-1} W(y^j)$. After a new weight w^* is found and a corresponding new nondominated vertex y^* is computed, $W(y^*) \cap W^i$ is removed from W^i. A crucial point is how to identify W^{i+1}. For multicriteria LPs, Benson and Sun provide a necessary and sufficient condition for a weight vector $w \in \mathbb{R}^k_+$ to satisfy $w \notin W(y)$ for some $y \in \mathcal{Y}$ by considering feasible directions of $\mathcal{X}$. Let $x \in \mathcal{X}$ be an extreme point of $\mathcal{X}$ and let $y = Cx$ be the corresponding image. Moreover, let $S(x)$ denote the set of all extreme points of $\mathcal{X}$ which are adjacent to x. They show that a vector $w \in \mathbb{R}^k$ satisfies $w \notin W(y)$ if and only if there exists a point $\bar{x} \in S(x)$ such that $w^\top C(\bar{x} - x) < 0$.

3.5.2 *Weight space decomposition methods for multicriteria integer programs*

Przybylski et al. use a weight space decomposition method to compute nondominated vertices of $conv(\mathcal{Y})$ for multicriteria IPs [64]. They show that the considered weight set $W(y^*) = \{\lambda \in \mathbb{R}^k_+ : \lambda^\top y^* = \min_{y \in conv(\mathcal{Y})} \lambda^\top y,\ \mathbb{1}^\top \lambda = 1\}$ of a nondominated vertex $y^* \in conv(\mathcal{Y})$ is a polytope of dimension $k-1$ and that the intersection $W(y^1) \cap W(y^2)$ of two *adjacent* nondominated vertices y^1 and y^2 is a polytope of dimension $k-2$. They find new nondominated vertices by considering weights located on the boundary of intermediate weight sets. They present results for tricriteria assignment problems and tricriteria knapsack problems based on an implementation that works recursively.

Özpeynirci and Köksalan also use a weight space decomposition method to compute nondominated vertices for multicriteria IPs [59]. They introduce *dummy points* in order to deal with issues related to the boundary of a weight set, i.e. weight vector components that are (close to) zero. The primary effect of these dummy points on the

extended space is to ensure that every nondominated vertex is adjacent to at least k points and that the intersection of the weight space boundary and $W(y^*)$ is empty. They present computational results for integer problems with 3 and 4 objectives.

3.5.3 *Weight space polyhedron method*

In this section we present a variant of the weight space decomposition method that *lifts* the k-dimensional weight sets to $k+1$-dimensional space and allows us to interpret an update step of incorporating a newly found nondominated vertex as an additional cutting plane regarding a weight space polyhedron.

As shown in Theorem 3.10 in Section 3.4.1, any nondominated vertex $y^* \in \mathcal{Y}$ corresponds to at least one positive weight vector λ for which $\lambda^\top y^*$ achieves a strictly better value than any $y \in \mathcal{Y} \setminus \{y^*\}$. Note that we can assume w.l.o.g. that λ is *normalized*, i.e. $\mathbb{1}^\top \lambda = 1$. For computational reasons, we will consider nonnegative instead of strictly positive weight vectors in the following definition.

Definition 3.14. (weight space) The k-dimensional *weight space* W is given by $W = \{w \in \mathbb{R}^k_{\geq 0} : \mathbb{1}^\top w = 1\}$.

Definition 3.15. (weight space polyhedron) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria LP and let $\mathcal{Y} \subseteq \mathbb{R}^k$ be the corresponding image with at least one nondominated vertex. Let $\bar{Y}$ be the entire set of nondominated vertices of $\mathcal{Y}$. For $Y \subseteq \bar{Y}$, we define the *weight space polyhedron* P_Y by

$$P_Y = \{(w, a) \in W \times \mathbb{R} : w^\top y \geq a \text{ for all } y \in Y\}. \tag{3.7}$$

Note that the weight space polyhedron P_Y is a subset of $\mathbb{R}^{k+1}$. For $Y \subseteq \bar{Y}$ the set P_Y is indeed a polyhedron since it is given as the intersection of finitely many inequalities:

$$\begin{aligned} w^\top y - a &\geq 0 \text{ for } y \in Y, \\ \mathbb{1}^\top w &\geq 1, \\ -\mathbb{1}^\top w &\geq -1, \\ w_j &\geq 0 \text{ for } j \in [k]. \end{aligned}$$

Remark 3.16. For ease of presentation, we will consider multicriteria optimisation problems which do not comprise nondominated rays in the remainder of this section. In order to deal with nondominated rays we could add inequalities of the form $w^\top q \geq 0$ to (3.7) for all found nondominated rays $q \in \mathbb{R}^k$. The constraint $w^\top q \geq 0$ cuts off weight vectors w for which the problem $(\lambda^\top C, \mathcal{X})$ is unbounded with regard to q.

The following result states that there is a subset chain between weight space polyhedra.

Theorem 3.17. Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria LP and let $\mathcal{Y} \subseteq \mathbb{R}^k$ be the corresponding image with at least one nondominated vertex. Let $\bar{Y}$ be the entire set of nondominated vertices of $\mathcal{Y}$ and let $Y_1, Y_2 \subseteq$

$\bar{Y}$ be two subsets of nondominated vertices with $Y_1 \subseteq Y_2$. Then, $P_{Y_2} \subseteq P_{Y_1}$ and $P_{Y_2} = P_{Y_1}$ if and only if $Y_1 = Y_2$.

Proof. We have $P_{Y_2} \subseteq P_{Y_1}$ for $Y_1 \subseteq Y_2$ since the inequalities describing P_{Y_1} are a subset of the inequalities describing P_{Y_2}.
If $Y_1 = Y_2$, then the inequalities describing P_{Y_1} coincide with the inequalities describing P_{Y_2} implying $P_{Y_2} = P_{Y_1}$. If $Y_1 \neq Y_2$, then there is $y^* \in Y_2 \setminus Y_1$. By Theorem 3.10 in Section 3.4.1, there is a weight vector $w \in \mathbb{R}^k_+$ such that $w^\top y^* < w^\top y$ for all $y \in Y_2 \setminus \{y^*\}$. Assume w.l.o.g. that w is normalized and let a be given by $a = w^\top y^*$. Let δ be given by $\delta = \min_{y \in Y_1}(w^\top y - a)$ and note that $\delta > 0$ holds. We get that $(w, a + \delta) \in P_{Y_1}$ since $w^\top y \geq a + \delta$ holds for all $y \in Y_1$. However, we have $w^\top y^* < a + \delta$ implying $(w, a + \delta) \notin P_{Y_2}$. Hence, $P_{Y_1} \neq P_{Y_2}$. □

The following two results characterise the structure of a weight space polyhedron in more detail. Firstly, we will show that a complete set of extreme rays for a weight space polyhedron is given by a singleton. Secondly, we show that a weight space polyhedron has at least one vertex.

Proposition 3.18. Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria LP and let $\mathcal{Y} \subseteq \mathbb{R}^k$ be the corresponding image with at least one nondominated vertex. Let $\bar{Y}$ be the entire set of nondominated vertices of $\mathcal{Y}$ and let $Y \subseteq \bar{Y}$ be a nonempty subset of nondominated vertices. Then, a complete set of extreme rays of P_Y is given by $\{(0, \ldots, 0, -1)\}$.

Proof. For a nonempty polyhedron P of the form $P = \{x \in \mathbb{R}^n : Ax \geq b\}$ the recession cone is the set $\{d \in \mathbb{R}^n : Ad \geq 0\}$ (see [12, Section 4.8]). Assume Y is given by $Y = \{y^1, \ldots, y^\ell\} \subset \mathbb{R}^k$. Then, the representation matrix $A \in \mathbb{R}^{(\ell+2+k)\times(k+1)}$ corresponding to P_Y is given by

$$A = \begin{array}{c} \\ r_0 \\ r_1 \\ r_2 \\ \vdots \\ r_k \end{array} \begin{pmatrix} y^1_1 & y^1_2 & \ldots & y^1_k & -1 \\ y^2_1 & y^2_2 & \ldots & y^2_k & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ y^\ell_1 & y^\ell_2 & \ldots & y^\ell_k & -1 \\ 1 & 1 & \ldots & 1 & 0 \\ -1 & -1 & \ldots & -1 & 0 \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & 0 \end{pmatrix}. \tag{3.8}$$

Let $\bar{d} \in \mathbb{R}^{k+1}$ be the vector given by $\bar{d} = (0, \ldots, 0, -1)^\top$. Then, $A\bar{d} \geq 0$ holds. Moreover, $\bar{d}$ is an extreme ray of the recession cone since the rows $r_1, \ldots, r_k$ in (3.8) correspond to k linearly independent active constraints at $\bar{d}$. Now assume that there is a feasible ray $d \in \mathbb{R}^{k+1}$ with $d_i \neq 0$ for at least one $i \in [k]$. If $Ad \geq 0$ holds, then the rows $r_1, \ldots, r_k$ in (3.8) imply that $d_i > 0$ for all $i \in [k]$ with $d_i \neq 0$. However, in this case, the r_0 constraint $-(d_1 + \ldots + d_k) \geq 0$ in (3.8) is violated. □

Proposition 3.19. Let $(c_1, \dots, c_k, \mathcal{X})$ be a multicriteria LP and let $\mathcal{Y} \subseteq \mathbb{R}^k$ be the corresponding image with at least one nondominated vertex. Let $\bar{Y}$ be the entire set of nondominated vertices of $\mathcal{Y}$ and let $Y \subseteq \bar{Y}$ be a nonempty subset of nondominated vertices. Then, P_Y has at least one vertex.

Proof. Define $y^* = \arg\min_{y \in Y} y_1$ and let $(w, a) \in W \times \mathbb{R}$ be given by $(w, a) = (1, 0, \dots, 0, y_1^*)$. It follows that $w^\top y = y_1 \geq y_1^*$ for all $y \in Y$ implying $(w, a) \in P_Y$. By the equivalence of vertices and basic feasible solutions (see [12, Theorem 2.3]) it suffices to show that there are $k+1$ linearly independent constraints which are active at (w, a). We get that $\mathbb{1}^\top w \geq 1,\ w_2 \geq 0,\ \dots,\ w_k \geq 0$ are k linearly independent constraints active at (w, a). Furthermore, we have that $w^\top y^* - a \geq 0$ is another active constraint which is linearly independent to the previous constraints. □

The next result yields one of the main ingredients of our computational approach. By Theorem 3.17 we know that $P_{Y_2} \subseteq P_{Y_1}$ for $Y_1 \subseteq Y_2$ and $P_{Y_2} = P_{Y_1}$ if and only if $Y_1 = Y_2$. Given a subset Y of the entire set of nondominated vertices $\bar{Y}$, we can check whether any vertex (w, a) of P_Y is also a vertex of $P_{\bar{Y}}$ by solving the single objective problem $\min_{x \in \mathcal{X}} w^\top Cx$.

Theorem 3.20. Let $(c_1, \dots, c_k, \mathcal{X})$ with bounded $\mathcal{X}$ be a multicriteria LP and let $\mathcal{Y} \subseteq \mathbb{R}^k$ be the corresponding image with at least one nondominated vertex. Let $\bar{Y}$ be the entire set of nondominated vertices of $\mathcal{Y}$ and let $(w, a) \in P_Y$ be a vertex of P_Y for some $Y \subseteq \bar{Y}$. Then, $(w, a) \notin P_{\bar{Y}}$ if and only if $\min_{x \in \mathcal{X}} w^\top Cx < a$.

Proof. "⇒" If $(w, a) \notin P_{\bar{Y}}$, then there exists $y^* \in \bar{Y}$ such that $w^\top y^* - a < 0$ by Definition 3.15. Since $y^* \in \mathcal{Y}$, there is a corresponding preimage $x^* \in \mathcal{X}$ with $y^* = Cx^*$. Then, $\min_{x \in \mathcal{X}} w^\top Cx \leq w^\top Cx^* < a$. "⇐" Let $x^* \in \arg\min_{x \in \mathcal{X}} w^\top Cx$ and let $y^* = Cx^*$ be the corresponding image. By Theorem 3.9, y^* can be represented as $y^* = \sum_{i=1}^{\ell} \lambda_i y^i$ where $\lambda \in \mathbb{R}^{\ell}_{\geq 0}$ with $\mathbb{1}^\top \lambda = 1$ and $y^1, \dots, y^\ell$ are the nondominated vertices of $\mathcal{Y}$. Then, $w^\top (\sum_{i=1}^{\ell} \lambda_i y^i) = w^\top y^* < a$. Let $J \subseteq [\ell]$ be the set of indices such that $\lambda_j > 0$ for all $j \in J$. Then, $w^\top (\sum_{j \in J} \lambda_j y^j) < a$. Now suppose that $w^\top y^j \geq a$ for all $j \in J$. Then, we get

$$w^\top \Big(\sum_{j \in J} \lambda_j y^j\Big) = \sum_{j \in J} \lambda_j (w^\top y^j) \geq \sum_{j \in J} \lambda_j a = a$$

yielding a contradiction. Hence, there exists $j \in J$ such that $w^\top y^j < a$. Since $y^j \in \bar{Y}$, we get $(w, a) \notin P_{\bar{Y}}$. □

In Theorem 3.20 we assume that $\mathcal{X}$ is bounded in order to ease the presentation of our computational approach. By assuming that $\mathcal{X}$ is bounded (and nonempty) we do not change the general approach, but ensure that $\min_{x \in \mathcal{X}} w^\top Cx$ always yields an optimal solution in Algorithm 3.3 and in Algorithm 3.2, respectively. See Section 3.4.3 for a summary of how to deal with the case where $\min_{x \in \mathcal{X}} w^\top Cx$ might be unbounded for some $w \in \mathbb{R}^k_+$.

Our algorithmic approach for computing the entire set of nondominated vertices $\bar{Y}$ can be summarised as follows: Provided that an initial nondominated vertex $y^* \in \mathcal{Y}$ was established (see Algorithm 3.2), the weight space polyhedron $P_{\{y^*\}}$ is initialised, i.e. we compute the vertices of $P_{\{y^*\}}$. Let P_Y denote the currently considered weight space polyhedron. In order to find out whether P_Y coincides with $P_{\bar{Y}}$ we iterate over the vertices of P_Y and apply Theorem 3.20. If we find a vertex $(w, a) \in P_Y$ for which $\min_{x \in \mathcal{X}} w^\top Cx < a$ holds, we compute a new nondominated vertex $\bar{y}$ based on the weight vector w given by (w, a) via Algorithm 3.2 and update the weight space polyhedron to $P_{Y \cup \{\bar{y}\}}$. If we do not find any such vertex, P_Y coincides with $P_{\bar{Y}}$ and we are done. Provided that we are not done yet, we proceed by iterating over the vertices of the updated weight space polyhedron $P_{Y \cup \{\bar{y}\}}$. Note that computation of the vertex representation of the encountered weight space polyhedra is done by a vertex enumeration algorithm (see Section 3.2.3).

Algorithm 3.3 Computing all nondominated vertices of a multicriteria linear program

```
1: function V-REPRESENTATION(P)
2:     compute vertices V of polyhedron P via vertex enumeration algorithm
3:     return V
```

Input: $(c_1, \ldots, c_k, \mathcal{X})$ with bounded $\mathcal{X}$, nondominated vertex $y^* \in \mathcal{Y}$
Output: Entire set of nondominated vertices of $\mathcal{Y}$

```
4: Y ← {y*}
5: finished ← false
6: while finished ≠ true do
7:     V ← V-REPRESENTATION(P_Y)
8:     finished ← true
9:     for (λ, a) ∈ V do
10:        opt ← min(λ^⊤ C, X)
11:        if opt < a then
12:            compute nondom. vertex y* wrt λ via Algorithm 3.2
13:            Y ← Y ∪ {y*}
14:            finished ← false
15:            break out of for-loop
16: return Y
```

The correctness of Algorithm 3.3 stems from the correctness of Algorithm 3.2 (see Theorem 3.13) which is applied in line 12 and application of Theorem 3.20 in line 6.

3.6 SUMMARY

In this chapter we considered the problem of computing nondominated vertices for multicriteria linear programs. We showed that in the multicriteria linear programming case any nondominated point can be represented by a combination of nondominated vertices and

nondominated extreme rays. We provided a practical algorithm for computing a nondominated vertex given an initial weight vector. We presented a novel weight space polyhedron method based on the concept of a weight space decomposition. We conclude this chapter by noting that the presented weight space polyhedron method using a practically efficient variant of the double description method for the vertex enumeration step is successfully implemented in the solver POLYSCIP (see Chapter 6).

4

PARTITIONING THE SET OF NONDOMINATED POINTS FOR MULTICRITERIA INTEGER PROGRAMS

> Nothing is particularly hard if you divide it into small jobs.
>
> *Henry Ford*

4.1 INTRODUCTION

As we know from Definition 2.7 in Section 2.2.2, the set of nondominated points can be partitioned into supported nondominated points and unsupported nondominated points. Furthermore, we know from Section 2.3.2 that in the bicriteria case the objective values of supported nondominated points can be used to find the potential location of unsupported nondominated points. In this chapter we consider a different partitioning approach of the set of nondominated points based on projections. Given a k-criteria integer program, we will partition the set of nondominated points into the set of points whose corresponding projections are nondominated for at least one corresponding $k-1$-criteria subproblem and the set of points whose corresponding projections are dominated for any such subproblem. Based on this partitioning approach we will see that the objective values of the points in the latter set can be bounded from above and from below by the objective values of the points in the first partition. Furthermore, we will see that this partitioning approach generally leads to different subsets than the classification into supported and unsupported nondominated points. In Chapter 5 we will consider the first partition, i.e. set of nondominated points whose corresponding projections are nondominated, as an *approximation* for the entire set of nondominated points.

4.2 POLY-NONDOMINANCE AND MONO-NONDOMINANCE

Tenfelde-Podehl [76] considers a recursive approach to solve a k-criteria integer program based on computing efficient solutions for the corresponding k many $k-1$-criteria subproblems where one of the given objectives is discarded. Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria integer program. Tenfelde-Podehl defines a solution $x \in \mathcal{X}$ to be $Q-1$-*Pareto* (where Q corresponds to the number of objectives) if there is an $i \in [k]$ such that x is efficient for $(c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_k, \mathcal{X})$ and investigates how $Q-1$-Pareto solutions can be used to compute supported and unsupported nondominated points.

In this section we use a very similar definition (considering the objective space) in order to partition the entire set of nondominated points into nondominated points whose preimages are basically given by $Q-1$-Pareto solutions and nondominated points whose preimages do not correspond to any $Q-1$-Pareto solution. In the remaining sections of this chapter we will then further investigate the resulting partitions.

For what follows we use the following projection.

Definition 4.1. (projection π_i) Let $k \geq 2$ and $i \in [k]$ be two positive integers. By $\pi_i : \mathbb{R}^k \to \mathbb{R}^{k-1}$ we denote the projection $\pi_i(y_1, \ldots, y_k) = (y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_k)$.

Definition 4.2. (poly-nondominated point) Let $(c_1, \ldots, c_k, \mathcal{X})$ with $k \geq 3$ be a multicriteria integer program and let $\mathcal{N}$ be the corresponding entire set of nondominated points. For $i \in [k]$ the set of nondominated points $\mathcal{N}_i$ is given by

$$\mathcal{N}_i = \{y \in \mathcal{N} : \pi_i(y) \text{ is a NDP of } (c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_k, \mathcal{X})\}.$$

By $\mathcal{N}_p$ we denote the union $\mathcal{N}_p = \bigcup_{i=1}^{k} \mathcal{N}_i$. A point $y \in \mathcal{N}_p$ is called *poly-nondominated point (PNDP)*.

Note that for $i \neq j$ the sets $\mathcal{N}_i$ and $\mathcal{N}_j$ are generally not disjoint, see Example 4.6. The prefix *poly* in poly-nondominated is to convey that not only the point $y^* \in \mathcal{N}_p$ itself is nondominated but at least one of its projections $\pi_i(y^*)$ is, too. Note that a $Q-1$-Pareto solution does not necessarily correspond to a preimage of a poly-nondominated point. However, the next result shows that computing the preimage of a poly-nondominated point from a $Q-1$-Pareto solution is straightforward.

Proposition 4.3. Let $(c_1, \ldots, c_k, \mathcal{X})$ with $k \geq 3$ be a multicriteria integer program with a nonempty set of nondominated points $\mathcal{N}$. Let $i \in [k]$ be an integer. If $\bar{x} \in \mathcal{X}$ is an efficient solution of $(c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_k, \mathcal{X})$, then a poly-nondominated point $y^* \in \mathcal{N}_i$ can be computed by solving the single objective problem:

$$\begin{aligned} \min\ & c_i^\top x \\ \text{s.t. } & c_j^\top x = c_j^\top \bar{x} \ \text{ for all } j \in [k] \setminus \{i\}, \\ & x \in \mathcal{X}. \end{aligned} \tag{4.1}$$

Proof. Since $\bar{x} \in \mathcal{X}$ is a feasible solution, (4.1) is either unbounded or bounded. If (4.1) is unbounded, then there is a feasible ray r with $c_i^\top r < 0$ and $c_j^\top r = 0$ for all $j \in [k] \setminus \{i\}$ implying $\mathcal{N} = \emptyset$ which contradicts our assumption that $\mathcal{N} \neq \emptyset$. Now it is a direct consequence of the constraints of (4.1) that the image y^* of an optimal solution x^* of (4.1) is a nondominated point of $(c_1, \ldots, c_k, \mathcal{X})$ whose projection $\pi_i(y^*)$ coincides with a nondominated point of $(c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_k, \mathcal{X})$. □

The next definition considers the set given by the nondominated points which are not poly-nondominated points.

Definition 4.4. (mono-nondominated point) Let $(c_1, \ldots, c_k, \mathcal{X})$ with $k \geq 3$ be a multicriteria integer program and let $\mathcal{N}$ be the corresponding entire set of nondominated points. The set $\mathcal{N}_m$ is given by

$$\mathcal{N}_m = \{y \in \mathcal{N} : \forall\, i \in [k]\ \pi_i(y) \text{ is dominated w.r.t. } (c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_k, \mathcal{X})\}.$$

A point $y \in \mathcal{N}_m$ is called *mono-nondominated point (MNDP)*.

Observe that $\mathcal{N} = \mathcal{N}_m \cup \mathcal{N}_p$ and $\mathcal{N}_m \cap \mathcal{N}_p = \emptyset$. The prefix *mono* in mono-nondominated is to convey that only the point $y^* \in \mathcal{N}_m$ itself is nondominated, but none of its projections $\pi_i(y^*)$ is.

Remark 4.5. We assume in Definition 4.2 and Definition 4.4, respectively, that the number of given objectives k is greater than 2. For $k = 2$ the sets $\mathcal{N}_1$ and $\mathcal{N}_2$ are not well-defined. In this case $\mathcal{N}_1$ and $\mathcal{N}_2$ could be defined to contain the *optimal* outcome (provided that there is one) of $\min_{x\in\mathcal{X}} c_2^\top x$ and $\min_{x\in\mathcal{X}} c_1^\top x$, respectively. However, although such a singleton definition for $k = 2$ leads to a result equivalent to Theorem 4.7, the bounds that we would get in this case are generally inferior than the bounds we get from supported nondominated points.

Example 4.6. Consider the tricriteria assignment problem $(c_1, c_2, c_3, \mathcal{X})$ given by

$$\begin{aligned} \min\Big(\sum_{i=1}^{4}\sum_{j=1}^{4} c_{1_{ij}} x_{ij},\ \sum_{i=1}^{4}\sum_{j=1}^{4} c_{2_{ij}} x_{ij},\ \sum_{i=1}^{4}\sum_{j=1}^{4} c_{3_{ij}} x_{ij}\ \Big) \\ \text{s.t.} \quad \sum_{i=1}^{4} x_{ij} = 1 \ \text{ for all } j \in [4], \\ \sum_{j=1}^{4} x_{ij} = 1 \ \text{ for all } i \in [4], \\ x_{ij} \in \{0,1\}, \end{aligned} \tag{4.2}$$

where

$$c_1 = \begin{pmatrix} 3,6,4,5 \\ 2,3,5,4 \\ 3,5,4,2 \\ 4,5,3,6 \end{pmatrix},\ c_2 = \begin{pmatrix} 2,3,5,4 \\ 5,3,4,3 \\ 5,2,6,4 \\ 4,5,2,5 \end{pmatrix},\ c_3 = \begin{pmatrix} 4,2,4,2 \\ 4,2,4,6 \\ 4,2,6,3 \\ 2,4,5,3 \end{pmatrix}.$$

The corresponding image $\mathcal{Y}$ is illustrated in Figure 4.1. The entire set of nondominated points $\mathcal{N}$ is given by $\mathcal{N} = \{y^1, y^2, y^3, y^4, y^5, y^6, y^7\}$ where $y^1 = (11, 11, 14)$, $y^2 = (15, 9, 17)$, $y^3 = (19, 14, 10)$, $y^4 = (13, 16, 11)$, $y^5 = (15, 13, 13)$, $y^6 = (17, 15, 11)$ and $y^7 = (14, 14, 13)$. The set of poly-nondominated points $\mathcal{N}_p = \mathcal{N}_1 \cup \mathcal{N}_2 \cup \mathcal{N}_3$ is given by $\mathcal{N}_p = \{y^1, y^2, y^3, y^4, y^5\}$ where $\mathcal{N}_1 = \{y^1, y^2, y^3, y^5\}$, $\mathcal{N}_2 = \{y^1, y^3, y^4\}$ and $\mathcal{N}_3 = \{y^1, y^2\}$. The set of mono-nondominated points $\mathcal{N}_m$ is given by $\mathcal{N}_m = \{y^6, y^7\}$. See also Figure 4.2.

The next result tells us that the objective values of any mono-nondominated point can be bounded from below and from above by the objective values of some poly-nondominated points.

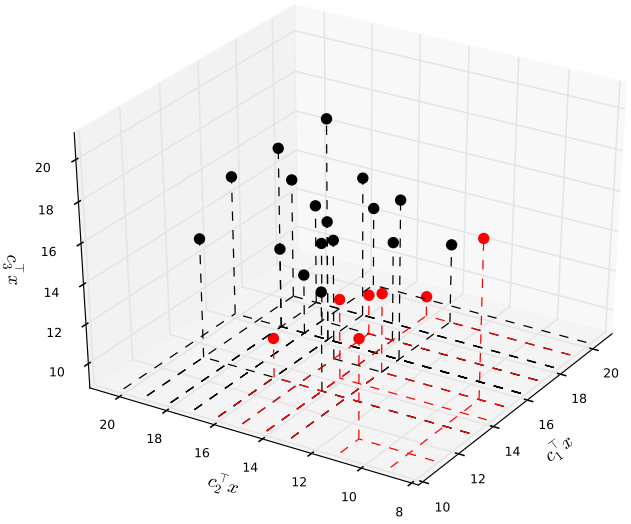

Figure 4.1: Image $\mathcal{Y}$ of the tricriteria integer program given in Example 4.6 with dominated points in black and nondominated points in red.

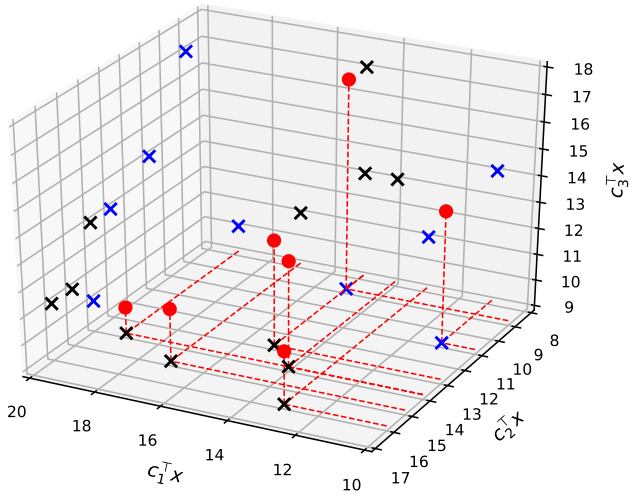

Figure 4.2: Nondominated points of the tricriteria integer program given in Example 4.6 with nondominated projections in blue and dominated projections in black.

Theorem 4.7. Let $(c_1, \ldots, c_k, \mathcal{X})$ with $k \geq 3$ be a multicriteria integer program. If $z \in \mathcal{N}_m$, then there are $y^1 \in \mathcal{N}_1, \ldots, y^k \in \mathcal{N}_k$ such that

$$\max_{i \in [k] \setminus \{j\}} y_j^i \leq z_j < y_j^j \tag{4.3}$$

for all $j \in [k]$.

Proof. By Definition 4.4, for all $i \in [k]$ it holds that $\pi_i(z)$ is a dominated point of $(c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_k, \mathcal{X})$. Hence, for all $i \in [k]$ there is $y^i \in \mathcal{N}_i$ with $y^i \neq z$ such that $y^i_j \leq z_j$ for all $j \in [k] \setminus \{i\}$ and $y^i_i > z_i$. The latter must hold since $z \in \mathcal{N}_m$ is a nondominated point of $(c_1, \ldots, c_k, \mathcal{X})$. Combining these inequalities we get

$$\begin{aligned} \max(y^2_1, \ldots, y^k_1) &\leq z_1 < y^1_1 \\ \max(y^1_2, y^3_2, \ldots, y^k_2) &\leq z_2 < y^2_2 \\ &\vdots \\ \max(y^1_k, \ldots, y^{k-1}_k) &\leq z_k < y^k_k \end{aligned}$$

□

Theorem 4.7 is not written down in such a general form in [76], but it is conceived and implicitly used by Tenfelde-Podehl for her recursive approach (compare Section 3.2.4 in [76]).

Example 4.8. Let $(c_1, c_2, c_3, \mathcal{X})$ be the tricriteria assignment problem given in (4.2). The objective values of $y^6 = (17, 15, 11) \in \mathcal{N}_m$ are bounded from below and from above by $y^3 = (19, 14, 10) \in \mathcal{N}_1$, $y^4 = (13, 16, 11) \in \mathcal{N}_2$ and $y^2 = (15, 9, 17) \in \mathcal{N}_3$ since

$$\begin{aligned} \max(y^4_1, y^2_1) = \max(13, 15) &\leq y^6_1 = 17 < y^3_1 = 19, \\ \max(y^3_1, y^2_2) = \max(14, 9) &\leq y^6_2 = 15 < y^4_2 = 16, \\ \max(y^3_3, y^4_3) = \max(10, 11) &\leq y^6_3 = 11 < y^2_3 = 17. \end{aligned}$$

The objective values of $y^7 = (14, 14, 13) \in \mathcal{N}_m$ are bounded from below and from above by $y^5 = (15, 13, 13) \in \mathcal{N}_1$, $y^4 = (13, 16, 11) \in \mathcal{N}_2$ and $y^1 = (11, 11, 14) \in \mathcal{N}_3$ since

$$\begin{aligned} \max(y^4_1, y^1_1) = \max(13, 11) &\leq y^7_1 = 14 < y^5_1 = 15, \\ \max(y^5_2, y^1_2) = \max(13, 11) &\leq y^7_2 = 14 < y^4_2 = 16, \\ \max(y^5_3, y^4_3) = \max(13, 11) &\leq y^7_3 = 13 < y^1_3 = 14. \end{aligned}$$

For an illustration see also Figure 4.3.

Remark 4.9. Note that Definition 4.2, Definition 4.4 and Theorem 4.7 could be generalised to multicriteria integer programs with nonlinear objectives or a feasible domain given by nonlinear constraints. In other words, the set of nondominated points $\mathcal{N}$ of such a nonlinear multicriteria integer program could be partitioned into the set of poly-nondominated points $\mathcal{N}_p$ and the set of mono-nondominated points $\mathcal{N}_m$. Moreover, Theorem 4.7 would remain valid since its corresponding proof considers only objective values of points in $\mathcal{N}_p$ and $\mathcal{N}_m$, but does not assume any linearity on the objectives or on the constraints of the feasible domain.

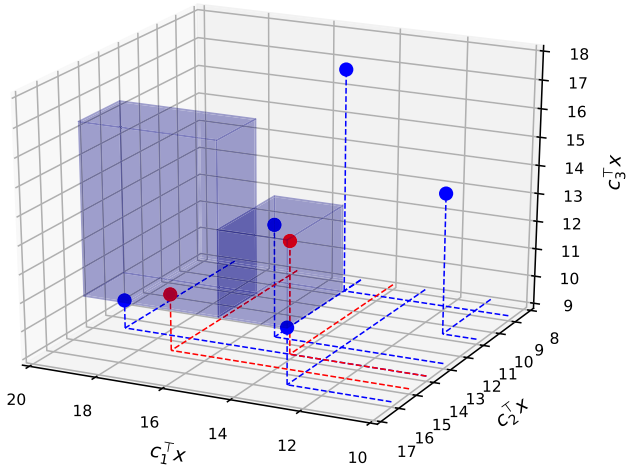

Figure 4.3: Objective values of mono-nondominated points (in red) bounded from above and from below by objective values of poly-nondominated points (in blue).

4.3 SOME CHARACTERISTICS OF POLY-NONDOMINATED POINTS

In this section we present some properties and characteristics of poly-nondominated points in relation to other *types* of nondominated points.

4.3.1 *Relation to lexicographically nondominated points*

The next result shows that any lexicographically optimal point is a poly-nondominated point.

Proposition 4.10. ([31, 76]) Let $(c_1, \ldots, c_k, \mathcal{X})$ with $k \geq 3$ be a multicriteria integer program. If $y^* \in \mathcal{Y}$ is lexicographically optimal, then $y^* \in \mathcal{N}_p$.

4.3.2 *Relation to the nadir point*

Definition 2.2 in Section 2.2.1 introduced the nadir point as the point which achieves the componentwise maximum values taken over all nondominated points. To quote a line from [15]: 'Note that computing the nadir point is not easy in general, but trivial once the nondominated frontier is known.' The next result shows that it suffices to consider only the set of poly-nondominated points in order to compute the nadir point.

Theorem 4.11. (cf. [31, 76]) Let $(c_1, \ldots, c_k, \mathcal{X})$ with $k \geq 3$ be a multicriteria integer program and let $\mathcal{N}_p$ be the corresponding set of poly-

nondominated points. Then, the i-th component of the nadir point $y^N \in \mathbb{R}^k$ is given by

$$y_i^N = \max_{y \in \mathcal{N}_p} y_i$$

for all $i \in [k]$.

4.3.3 *Relation to (un)supported nondominated points*

Section 4.3.1 has shown us that the set of lexicographically optimal points is a subset of the set of poly-nondominated points. In this subsection we consider the relation of the set of poly-nondominated points $\mathcal{N}_p$ to the set of supported nondominated points $\mathcal{N}_s$. We will see that the two sets have a nonempty intersection, but do generally neither coincide nor yield a subset relation.

Proposition 4.12. Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria integer program with a nonempty and finite set of nondominated points $\mathcal{N}$. Then, $\mathcal{N}_p \cap \mathcal{N}_s \neq \emptyset$ holds.

Proof. Since $0 < |\mathcal{N}| < \infty$, there is at least one lexicographically optimal point $y^* \in \mathcal{N}$. We get $y^* \in \mathcal{N}_s \cap \mathcal{N}_p$ by Proposition 2.14 and Proposition 4.10. □

The next example shows that a poly-nondominated point can be unsupported nondominated.

Example 4.13. Let $(c_1, c_2, c_3, \mathcal{X})$ be the tricriteria assignment problem given by (4.2). Consider the three poly-nondominated points $y^1 = (11, 11, 14)$, $y^4 = (13, 16, 11)$ and $y^5 = (15, 13, 13) \in \mathcal{N}_p$ (see also Example 4.6). We will show that y^5 is unsupported nondominated by showing that for any $\lambda \in \mathbb{R}^3_+$ with $\mathbb{1}^\top \lambda = 1$ either $\lambda^\top y^1 < \lambda^\top y^5$ or $\lambda^\top y^4 < \lambda^\top y^5$ holds. If $\lambda^\top y^1 \geq \lambda^\top y^5$ ought to hold, then $\lambda_2 \leq -\frac{5}{3}\lambda_1 + \frac{1}{3}$ must hold. The latter implies $\lambda_2 < \frac{1}{3}$ since $\lambda_1 \in (0, 1)$. Moreover, if $\lambda^\top y^4 \geq \lambda^\top y^5$ ought to hold, then $\lambda_2 \geq \frac{2}{5}$ must hold. The latter and $\lambda_2 < \frac{1}{3}$ are mutually exclusive. In other words, there is no $\lambda \in \mathbb{R}^3_+$ with $\mathbb{1}^\top \lambda$ such that $\lambda^\top y^5 \leq \lambda^\top y^1$ and $\lambda^\top y^5 \leq \lambda^\top y^4$. Hence, $y^5 \in \mathcal{N}_p$ cannot be supported nondominated.

The following example shows that a supported nondominated point can be mono-nondominated.

Example 4.14. Consider the tricriteria assignment problem

$$\begin{aligned} \min \ & \Big(\sum_{i=1}^{5}\sum_{j=1}^{5} c_{1_{ij}} x_{ij},\ \sum_{i=1}^{5}\sum_{j=1}^{5} c_{2_{ij}} x_{ij},\ \sum_{i=1}^{5}\sum_{j=1}^{5} c_{3_{ij}} x_{ij}\Big) \\ \text{s.t.} \ & \sum_{i=1}^{5} x_{ij} = 1 \ \text{ for all } j \in [5], \\ & \sum_{j=1}^{5} x_{ij} = 1 \ \text{ for all } i \in [5], \\ & x_{ij} \in \{0, 1\}, \end{aligned}$$

where

$$c_1 = \begin{pmatrix} 6,1,20,2,3 \\ 2,6,9,10,18 \\ 1,6,20,5,9 \\ 6,8,6,9,6 \\ 7,10,10,6,2 \end{pmatrix}, \; c_2 = \begin{pmatrix} 17,20,8,8,20 \\ 10,13,1,10,15 \\ 4,11,1,13,1 \\ 19,13,7,18,17 \\ 15,3,5,1,11 \end{pmatrix}, \; c_3 = \begin{pmatrix} 10,7,1,19,12 \\ 2,15,12,10,3 \\ 11,20,16,12,9 \\ 10,15,20,11,7 \\ 1,9,20,7,6 \end{pmatrix}.$$

The entire set of nondominated points $\mathcal{N}$ comprises the following nondominated points:

$$y^1 = (23,43,44),\ y^2 = (28,33,58),\ y^3 = (35,38,56),$$
$$y^4 = (39,43,41),\ y^5 = (18,47,67),\ y^6 = (28,66,39),$$
$$y^7 = (29,29,59),\ y^8 = (35,49,39),\ y^9 = (38,33,53),$$
$$y^{10} = (22,54,47),\ y^{11} = (16,61,47),\ y^{12} = (24,39,45),$$
$$y^{13} = (20,52,54),\ y^{14} = (50,40,32),\ y^{15} = (22,37,63),$$
$$y^{16} = (17,43,71),\ y^{17} = (45,33,34),\ y^{18} = (43,51,31)$$
$$y^{19} = (40,47,37),\ y^{20} = (37,55,36),\ y^{21} = (34,60,42).$$

It holds that $y^{12} = (24,39,45)$ is supported nondominated as y^{12} is the (unique) minimiser of $\arg\min_{y\in\mathcal{N}} \lambda^\top y$ for $\lambda = (1, 0.3518, 0.2222) \in \mathbb{R}^3_+$. Furthermore, we have that $\pi_1(y^{12}) = (39,45)$ is dominated by $\pi_1(y^{17}) = (33,34)$, $\pi_2(y^{12}) = (24,45)$ is dominated by $\pi_2(y^1) = (23,44)$, and $\pi_3(y^{12}) = (24,39)$ is dominated by $\pi_3(y^{15}) = (22,37)$. Hence, y^{12} is supported mono-nondominated.

4.4 NUMBER OF (POLY | MONO)-NONDOMINATED POINTS

Given a multicriteria integer program what is the number of poly-nondominated points and what is the number of mono-nondominated points? Is the cardinality of the first set generally larger than the cardinality of the latter set or vice versa? Tenfelde-Podehl computes several small test instances and notes that in many of the instances (almost) the entire set of nondominated points $\mathcal{N}$ is given by the set of poly-nondominated points $\mathcal{N}_p$ [76]. However, our results show that this observation is due to the small size of considered test instances. In contrast to the observation of Tenfelde-Podehl, our computational evaluation for the tricriteria assignment problem (see Table 1 on page 46) and for the tricriteria knapsack problem (see Table 2 on page 48) shows that the number of mono-nondominated points can be much larger than the number of poly-nondominated points.

The following example constructs the case where the number of mono-nondominated points grows quadratically whereas the number of poly-nondominated points remains constant.

Example 4.15. Consider the space $\mathbb{R}^3$ and, for $k \in \mathbb{N}_+$, let the hyperplane $h(k) \subset \mathbb{R}^3$ be given by the equation $x_1 + x_2 + x_3 = k$,

see Figure 4.4. Suppose that, for $k \in \mathbb{N}_+$, we are given a multicriteria integer program $(c_1, c_2, c_3, \mathcal{X}_k)$ whose corresponding image $\mathcal{Y}_k \subset \mathbb{N}^3$ coincides with $\mathbb{N}^3 \cap h(k)$. Then, we get that $|\mathcal{N}| \approx k^2$ as any point on $\mathbb{N}^3 \cap h(k)$ will be nondominated. In other words, the number of nondominated points of $(c_1, c_2, c_3, \mathcal{X}_k)$ grows quadratically in k. Now consider the bicriteria subproblem $(c_1, c_2, \mathcal{X}_k)$ for some $k \in \mathbb{N}_+$. We get that the projection $\pi_3((0,0,k)) = (0,0)$ dominates any other projection $\pi_3(y)$ where $y \in \mathcal{N} \setminus \{(0,0,k)\}$. Hence, $\mathcal{N}_3$ is given by $\mathcal{N}_3 = \{(0,0,k)\}$. Equivalently, we have $\mathcal{N}_1 = \{(k,0,0)\}$ and $\mathcal{N}_2 = \{(0,k,0)\}$. In other words, the number of poly-nondominated points $\mathcal{N}_p$ is constant compared to a quadratically growing number of mono-nondominated points $\mathcal{N}_m$.

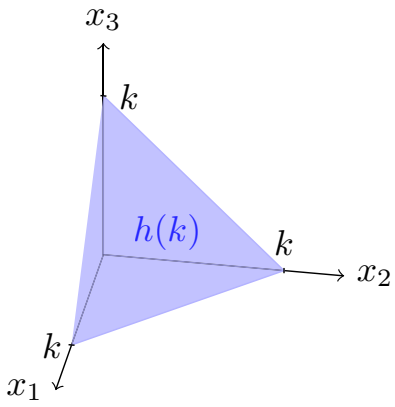

Figure 4.4: Hyperplane $h(k) \subset \mathbb{R}^3$ defined by $x_1 + x_2 + x_3 = k$

4.5 LOCATING POTENTIAL MONO-NONDOMINATED POINTS

Theorem 4.7 tells us that the objective values of any mono-nondominated point are bounded from below and from above by the objective values of some poly-nondominated points. Based on this result it might seem reasonable to try to compute the entire set of nondominated points by computing the set of poly-nondominated points in a first step and then, based on the lower and upper bounds provided by different combinations of poly-nondominated points, to find the remaining set of mono-nondominated points in a second step. In this section we investigate such an approach in more detail by investigating the number of *boxes* that needed to be investigated in order to find potential mono-nondominated points.

4.5.1 *Feasible boxes*

As a first insight, not all of the $\prod_{i=1}^{k} |\mathcal{N}_i|$ theoretically possible combinations of poly-nondominated points yield feasible bounds regarding Theorem 4.7.

Example 4.16. Let $(c_1, c_2, c_3, \mathcal{X})$ be the tricriteria assignment problem given in (4.2) and consider the three poly-nondominated points $y^1 =$

$(11,11,14) \in \mathcal{N}_1$, $y^4 = (13,16,11) \in \mathcal{N}_2$ and $y^2 = (15,9,17) \in \mathcal{N}_3$. Considering the first components y_1^1, y_1^4 and y_1^2, we get that

$$\max(y_1^4, y_1^2) = \max(13,15) \quad > \quad y_1^1 = 11.$$

which implies infeasible bounds regarding Theorem 4.7.

We introduce the term *feasible box* in order to distinguish between feasible bounds and infeasible bounds given by poly-nondominated points.

Definition 4.17. (feasible box) Let $(c_1, \ldots, c_k, \mathcal{X})$ with $k \geq 3$ be a multicriteria integer program and let $y^1 \in \mathcal{N}_1, \ldots, y^k \in \mathcal{N}_k$ be poly-nondominated points. By $B(y^1, \ldots, y^k)$ we denote the k-ary Cartesian product

$$B(y^1, \ldots, y^k) = \bigtimes_{j=1}^{k} [\max_{i \in [k] \setminus \{j\}} y_j^i,\ y_j^j). \tag{4.4}$$

$B(y^1, \ldots, y^k)$ is called *feasible box* if $B(y^1, \ldots, y^k)$ is nonempty, i.e. if $\max_{i \in [k] \setminus \{j\}} y_j^i < y_j^j$ holds for all $j \in [k]$. By $\mathcal{B}$ we denote the set of all feasible boxes

$$\mathcal{B} = \{B(y^1, \ldots, y^k) \text{ is feasible box} : \ y^1 \in \mathcal{N}_1, \ldots, y^k \in \mathcal{N}_k\}.$$

The next result shows that a feasible box does not contain any poly-nondominated points.

Theorem 4.18. Let $(c_1, \ldots, c_k, \mathcal{X})$ with $k \geq 3$ be a multicriteria integer program and let $B(y^1, \ldots, y^k)$ be a feasible box given by $y^1 \in \mathcal{N}_1, \ \ldots, \ y^k \in \mathcal{N}_k$. Then, $y \notin B(y^1, \ldots, y^k)$ for all $y \in \mathcal{N}_p$.

Proof. $B(y^1, \ldots, y^k)$ is a k-ary Cartesian product of half-open intervals excluding $y_1^1, \ldots, y_k^k$ by definition. Hence, $y^1, \ldots, y^k$ are not contained in $B(y^1, \ldots, y^k)$. Now suppose that $\mathcal{N}_p \setminus \{y^1, \ldots, y^k\} \cap B(y^1, \ldots, y^k) \neq \emptyset$ and let $z \in \mathcal{N}_p \setminus \{y^1, \ldots, y^k\} \cap B(y^1, \ldots, y^k)$. Then, for $j = 1, \ldots, k$,

$$\max_{i \in [k] \setminus \{j\}} y_j^i \leq z_j < y_j^j \tag{4.5}$$

holds. Now consider the relation of z to y^i for all $i = 1, \ldots, k$. Starting with $i = 1$ we get from (4.5) that

$$y_j^1 \leq z_j \text{ for all } j \in [k] \setminus \{1\}.$$

It follows that either $y_j^1 = z_j$ for all $j \in [k] \setminus \{1\}$ or $y_\ell^1 < z_\ell$ for at least one $\ell \in [k] \setminus \{1\}$. Firstly, assume that $y_j^1 = z_j$ for all $j \in [k] \setminus \{1\}$. Then, since $z_1 < y_1^1$ by (4.5), z dominates y^1 contradicting $y^1 \in \mathcal{N}_p$. Secondly, assume that $y_j^1 \leq z_j$ for all $j \in [k] \setminus \{1\}$ with at least one strict inequality. In other words, $\pi_1(y^1)$ dominates $\pi_1(z)$ implying that $z \notin \mathcal{N}_1$. By considering the relation to the remaining points $y^2, \ldots, y^k$ we get that $z \notin \mathcal{N}_j$ for $j = 2, \ldots, k$ which implies that $z \notin \mathcal{N}_p$. □

Remark 4.19. In order to algorithmically use Theorem 4.7 and Theorem 4.18, respectively, we need to be able to adequately handle the strict $<$-inequalities in (4.3). Since we assume that the input data of $(c_1, \ldots, c_k, \mathcal{X})$ is given by rational numbers, we can turn the input data into integers by scaling it appropriately. Then, strict inequalities of the form $z_j < y_j^j$ can be transformed into $z_j \leq y_j^j - 1$ for $j \in [k]$.

4.5.2 *Computational results regarding redundant and irredundant feasible boxes*

In this section we investigate whether every feasible box (see Definition 4.17) should be included in a search for mono-nondominated points or whether some of the boxes can be discarded. Our computational results will show that many feasible boxes might be discarded as they are contained in other feasible boxes.

4.5.2.1 *Redundant and irredundant feasible boxes*

Definition 4.20. (redundant feasible box) Let $\mathcal{N}_p$ be the entire set of poly-nondominated points of a multicriteria integer program and let $\mathcal{B}$ be the corresponding set of feasible boxes given by the points in $\mathcal{N}_p$. We call $B \in \mathcal{B}$ *redundant* if there is $\bar{B} \in \mathcal{B}$ such that $B \subset \bar{B}$. We call a box *irredundant* if it is not redundant.

The next example shows that Definition 4.20 is not just theoretical. Furthermore, it shows that the difference between the number of redundant and the number of irredundant feasible boxes can be quite significant.

Example 4.21. Consider the set of poly-nondominated points $\mathcal{N}_p$ given by $\mathcal{N}_p = \{(19, 32, 39),\ (56, 15, 49),\ (31, 32, 24),\ (27, 20, 44),\ (36, 16, 41),\ (24, 41, 34), (39, 20, 29)\}$ corresponding to a tricriteria assignment problem instance. The set of feasible boxes $\mathcal{B}$ is given by $\mathcal{B} = \{B_1, \ldots, B_{12}\}$ where

$$B_1 = [24, 39) \times [32, 41) \times [34, 39),$$

$$B_2 = [24, 31) \times [32, 41) \times [34, 39),$$

$$B_3 = [27, 36) \times [20, 32) \times [41, 44),$$

$$B_4 = [27, 39) \times [20, 32) \times [39, 44),$$

$$B_5 = [27, 36) \times [20, 41) \times [41, 44),$$

$$B_6 = [27, 39) \times [20, 41) \times [34, 44),$$

$$B_7 = [27, 31) \times [32, 41) \times [34, 44),$$

$$B_8 = [31, 36) \times [20, 32) \times [41, 44),$$

$$B_9 = [31, 39) \times [20, 32) \times [29, 44),$$

$$B_{10} = [36, 39) \times [20, 32) \times [39, 41),$$

$$B_{11} = [36, 39) \times [20, 41) \times [34, 41),$$

$$B_{12} = [36, 39) \times [20, 32) \times [29, 41).$$

We get that $B_2 \subset B_1$, $B_3 \subset B_6$, $B_4 \subset B_6$, $B_5 \subset B_6$, $B_7 \subset B_6$, $B_8 \subset B_6$, $B_{10} \subset B_9$, $B_{11} \subset B_6$, and $B_{12} \subset B_9$. In other words, we have 9 redundant feasible boxes $B_2, B_3, B_4, B_5, B_7, B_8, B_{10}, B_{11}, B_{12}$ and 3 irredundant feasible boxes B_1, B_6, B_9.

Przybylski et al. compute the number of feasible boxes (called *remaining intervals*) for some instances of the tricriteria assignment problem [3]. Based on their computational experiments, they consider the resulting number of feasible boxes as being too large for any practical algorithm. However, they do not distinguish between redundant and irredundant boxes. Regarding Example 4.21, we would have to investigate only the 3 irredundant feasible boxes from initially 12 feasible boxes in order to find the set of mono-nondominated points. Given such a difference in the number of redundant versus irredundant feasible boxes it should be clear that an algorithm that searches for mono-nondominated points only in the set of irredundant feasible boxes might be much more efficient compared to an algorithm that searches in all feasible boxes. In the remainder of this section we consider the tricriteria assignment problem and the tricriteria knapsack problem, respectively. All considered instances of the tricriteria assignment problem and tricriteria knapsack problem were generated by Kirlik and Sayın [47] and are available online [77]. Our computational experiments shows that the number of redundant feasible boxes can become large and that distinguishing between irredundant and redundant feasible boxes would be crucial when it comes to evaluating the practicality of an algorithm that computes mono-nondominated points via feasible boxes.

We will use the following abbreviations:

AP - average number of poly-nondominated points,

AM - average number of mono-nondominated points,

AR - average number of redundant feasible boxes,

AI - average number of irredundant feasible boxes.

4.5.2.2 *Tricriteria assignment problem*

Kirlik and Sayın generated 100 instances of the tricriteria assignment problem [77]. The objective function coefficients are integers which are randomly generated from the interval $[1, 20]$. The number of agents (and jobs, respectively) of the generated test instances ranges from $n = 5$ to $n = 50$ with a step size of 5. For each $n \in \{5, 10, 15, \ldots, 50\}$ there are 10 instances. Note that the following computational results consider the average number of poly-nondominated points, mono-nondominated points, redundant feasible boxes, and irredundant feasible boxes taken over all 10 instances.

The computational results in Table 1 show that except for very small instances (i.e. for $n = 5$) the number of mono-nondominated

n	AP	AM	AR	AI	AR/AI	AI/AP	AM/AI
5	10	4.1	28.1	11.6	2.4	1.2	0.4
10	53.3	123.5	4823.4	157.2	30.7	2.9	0.8
15	98.4	576.5	30839.2	327.9	94.1	3.3	1.8
20	164.8	1695.7	156691.0	787.1	199.1	4.8	2.2
25	195.2	3372.6	273803.2	830.2	329.8	4.3	4.1
30	265.5	5915.8	661082.7	1492.6	442.9	5.6	4.0
35	308.8	8663.5	1060644.4	1817.6	583.5	5.9	4.8
40	356.2	14323.5	1648610.2	1959.8	841.2	5.5	7.3
45	386.9	17315.3	2111083.8	2509.8	841.1	6.5	6.9
50	424.6	24492.2	2799022.3	2578.7	1085.4	6.1	9.5

Table 1: Average number of poly-nondominated points (AP), mono-nondominated points (AM), redundant feasible boxes (AR), and irredundant feasible boxes (AI) for the tricriteria assignment problem.

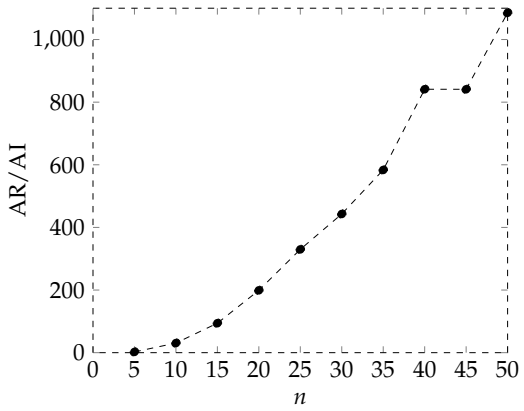

Figure 4.5: Average number of redundant feasible boxes per irredundant feasible box for the tricriteria assignment problem.

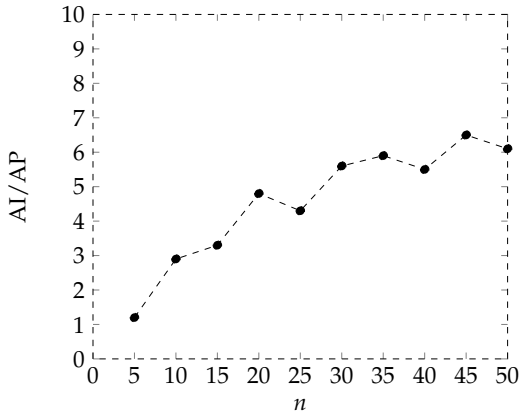

Figure 4.6: Average number of irredundant feasible boxes per poly-nondominated point for the tricriteria assignment problem.

points gets much larger than the number of poly-nondominated points, up to a factor of about 57 for $n = 50$. Equivalently, the number of redundant feasible boxes gets much larger than the number of irredund-

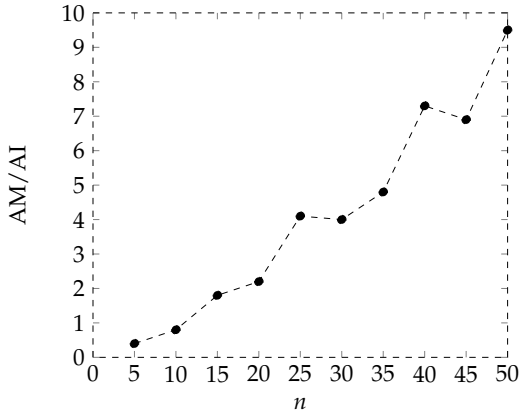

Figure 4.7: Average number of mono-nondominated points per irredundant feasible box for the tricriteria assignment problem.

ant feasible boxes. Considering the ratio AR/AI between the number of redundant boxes and the number of irredundant boxes we see that for each irredundant feasible box we have a growing number of redundant feasible boxes with increasing instance size, see Figure 4.5. Considering $n = 50$, the number of redundant feasible boxes is on average more than a 1000 times larger than the number of irredundant feasible boxes. This implies that the running time of an algorithm that iterates over the set of feasible boxes can be greatly improved by discarding redundant feasible boxes. Moreover, the number of irredundant boxes does not become intractable for the considered instances in contrast to the number of redundant feasible boxes. Considering the ratio AI/AP between the number of irredundant feasible boxes and the number of poly-nondominated points (see Figure 4.6), we also have a qualitative growth in the number of irredundant boxes per poly-nondominated point with increasing instance size. Considering the ratio AM/AI between the number of mono-nondominated points and the number of irredundant boxes (see Figure 4.7), we again have a qualitative growth with values similar to the values of AI/AP. For $n = 50$, on average each irredundant feasible box contains roughly 10 mono-nondominated points.

4.5.2.3 *Tricriteria knapsack problem*

Kirlik and Sayın generated 100 instances of the tricriteria knapsack problem [77]. The objective coefficients and weight coefficients are random integers drawn from the interval $[1, 1000]$. The capacity W of the knapsack is given by $W = 0.5 \sum_{i=1}^{n} w_i$ where n corresponds to the number of knapsack items. The number of knapsack items of the generated test instances ranges from $n = 10$ to $n = 100$ with a step size of 10. For each $n \in \{10, 20, \ldots, 100\}$ there are 10 instances. Note that the following computational results consider the average number of poly-nondominated points, mono-nondominated points, redundant feasible boxes, and irredundant feasible boxes taken over all 10 instances.

See page 45 for an explanation of the used abbreviations.

n	AP	AM	AR	AI	AR/AI	AI/AP	AM/AI
10	8.3	1.5	27.7	9.7	2.9	1.2	0.2
20	25.8	12.2	310.1	54.0	5.7	2.1	0.2
30	51.0	64.8	3659.8	168.7	21.7	3.3	0.4
40	94.0	217.2	27669.3	391.8	70.6	4.2	0.6
50	127.3	316.9	52551.1	589.2	89.2	4.6	0.5
60	178.2	738.9	162942.3	955.7	170.5	5.4	0.8
70	231.7	1411.7	357862.1	1476.3	242.4	6.4	1.0
80	318.8	1977.0	868945.4	2183.3	398.0	6.8	0.9
90	351.9	2755.9	1167100.3	2796.3	417.4	7.9	1.0
100	460.8	5388.2	2973864.7	3946.0	753.6	8.6	1.4

Table 2: Average number of poly-nondominated points (AP), mono-nondominated points (AM), redundant feasible boxes (AR), and irredundant feasible boxes (AI) for the tricriteria knapsack problem.

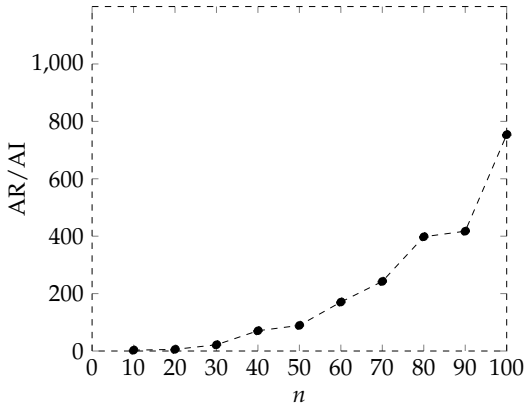

Figure 4.8: Average number of redundant feasible boxes per irredundant feasible box for the tricriteria knapsack problem.

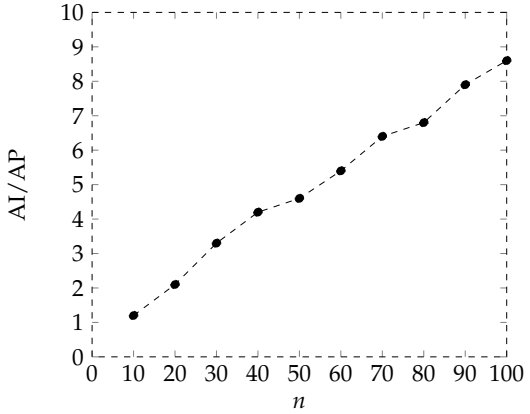

Figure 4.9: Average number of irredundant feasible boxes per poly-nondominated point for the tricriteria knapsack problem.

The computational results in Table 2 show that except for small instances (i.e. for $n \in \{10, 20\}$) the number of mono-nondominated

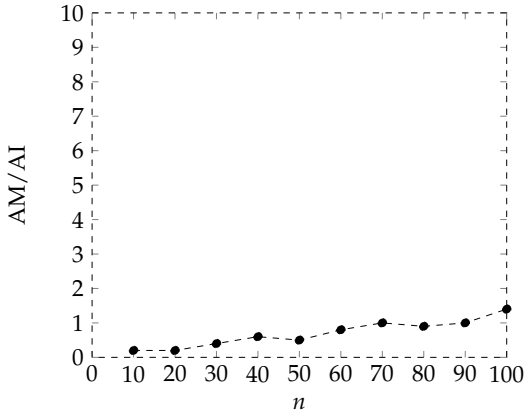

Figure 4.10: Average number of mono-nondominated points per irredundant feasible box for the tricriteria knapsack problem.

points is larger than the number of poly-nondominated points, up to a factor of 10 for $n = 100$. The number of redundant feasible boxes gets much larger than the number of irredundant feasible boxes, up to a factor of 753 for $n = 100$ (see Figure 4.8). This implies that the running time of an algorithm that iterates over the set of feasible boxes can be greatly improved by discarding redundant feasible boxes. Furthermore, the number of irredundant boxes does not become intractable for the considered instances. Considering the ratio AI/AP (see Figure 4.9) we get a growing number of irredundant boxes per poly-nondominated point with increasing instance size. In contrast to the tricriteria assignment problem, the ratio AM/AI between the number of mono-nondominated points and the number of irredundant feasible boxes grows relatively slowly, see Figure 4.10. For $n = 100$ every irredundant box contains on average 1.4 mono-nondominated points.

4.5.3 *Disjoint feasible boxes*

In Section 4.5.2 we have seen that the number of feasible boxes can be significantly reduced by discarding redundant feasible boxes. However, the remaining irredundant feasible boxes will generally not be disjoint.

Example 4.22. Let $(c_1, c_2, c_3, \mathcal{X})$ be the tricriteria assignment problem given in (4.2). Then, the two irredundant feasible boxes $B_1 = [13, 19) \times [14, 16) \times [11, 14)$ and $B_2 = [13, 15) \times [13, 16) \times [13, 14)$ given by the set of poly-nondominated points have a common intersection given by $[13, 15) \times [14, 16) \times [13, 14)$.

Example 4.22 implies that an algorithmic approach that computes the set of poly-nondominated points $\mathcal{N}_p$ in a first step, followed by the generation of all irredundant feasible boxes based on $\mathcal{N}_p$, followed by the search for mono-nondominated points in each irredundant feasible box explores some regions repeatedly. In the remainder of this section we show that computing a *partition* of the set union of feasible boxes is straight-forward.

Observation 4.23. Let $A = \bigtimes_{i=1}^{k} A_i$, $B = \bigtimes_{i=1}^{k} B_i \subset R^k$ be two feasible boxes where A_i, $B_i \subset \mathbb{R}$ are half-open intervals for all $i \in [k]$. Then, $A \cap B = \emptyset$ if and only $A_i \cap B_i = \emptyset$ for some $i \in [k]$, see Figure 4.11 and Figure 4.12.

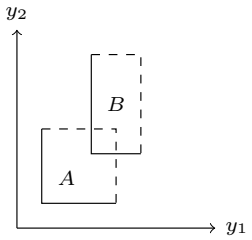

Figure 4.11: Overlapping boxes.

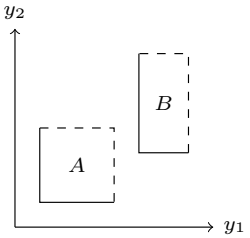

Figure 4.12: Non-overlapping boxes.

Note that $B_i = (B_i \setminus A_i) \cup (B_i \cap A_i)$ for $A_i, B_i \subset \mathbb{R}$. If $A_i = [a_i, \bar{a}_i)$, $B_i = [b_i, \bar{b}_i) \subset \mathbb{R}$ with $a_i < \bar{a}_i$, $b_i < \bar{b}_i$ and $A_i \cap B_i \neq \emptyset$, then $B_i \cap A_i$ is again a half-open interval and the result of $B_i \setminus A_i$ is either empty (see Figure 4.13), the half-open interval $[\bar{a}_i, \bar{b}_i)$ (see Figure 4.14), the half-open interval $[b_i, a_i)$ (see Figure 4.15) or the two intervals $[\bar{a}_i, \bar{b}_i)$ and $[b_i, a_i)$ (see Figure 4.16).

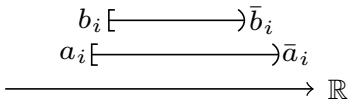

Figure 4.13: Interval difference $[b_i, \bar{b}_i) \setminus [a_i, \bar{a}_i)$ resulting in an empty set.

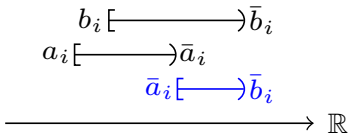

Figure 4.14: Interval difference $[b_i, \bar{b}_i) \setminus [a_i, \bar{a}_i)$ resulting in $[\bar{a}_i, \bar{b}_i)$.

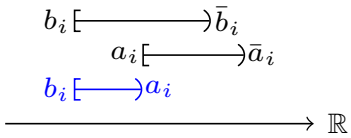

Figure 4.15: Interval difference $[b_i, \bar{b}_i) \setminus [a_i, \bar{a}_i)$ resulting in $[b_i, a_i)$.

The next definition considers the resulting boxes given by the intersection of two feasible boxes.

Definition 4.24. (intersection box) Let $A = \bigtimes_{j=1}^{k} [a_j, \bar{a}_j) \subset \mathbb{R}^k$ and $B = \bigtimes_{j=1}^{k} [b_j, \bar{b}_j) \subset \mathbb{R}^k$ be two feasible boxes with $A \cap B \neq \emptyset$ and let

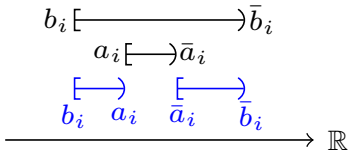

Figure 4.16: Interval difference $[b_i, \bar{b}_i) \setminus [a_i, \bar{a}_i)$ resulting in $[b_i, a_i)$ and $[\bar{a}_i, \bar{b}_i)$.

$i \in [k]$ be an integer. The i-th *lower intersection box* $\underline{\square}_i(B, A) \subset \mathbb{R}^k$ is given by the k-ary Cartesian product

$$\underline{\square}_i(B, A) = \bigtimes_{\ell=1}^{i-1} ([a_\ell, \bar{a}_\ell) \cap [b_\ell, \bar{b}_\ell)) \times [b_i, a_i) \times \bigtimes_{\ell=i+1}^{k} [b_\ell, \bar{b}_\ell).$$

We consider $\underline{\square}_i(B, A) = \emptyset$ if $b_i \geq a_i$.
Equivalently, the i-th *upper intersection box* $\overline{\square}_i(B, A) \subset \mathbb{R}^k$ is given by the k-ary Cartesian product

$$\overline{\square}_i(B, A) = \bigtimes_{\ell=1}^{i-1} ([a_\ell, \bar{a}_\ell) \cap [b_\ell, \bar{b}_\ell)) \times [\bar{a}_i, \bar{b}_i) \times \bigtimes_{\ell=i+1}^{k} [b_\ell, \bar{b}_\ell).$$

We consider $\overline{\square}_i(B, A) = \emptyset$ if $\bar{a}_i \geq \bar{b}_i$.

The next result shows that different intersection boxes are disjoint.

Proposition 4.25. Let $A = \bigtimes_{i=1}^k [a_i, \bar{a}_i)$, $B = \bigtimes_{i=1}^k [b_i, \bar{b}_i) \subset \mathbb{R}^k$ be feasible boxes with $A \cap B \neq \emptyset$. Then,

$$\underline{\square}_i(B, A) \cap \underline{\square}_j(B, A) = \emptyset \quad \text{and} \quad \overline{\square}_i(B, A) \cap \overline{\square}_j(B, A) = \emptyset$$

for all $i, j \in [k]$ with $i \neq j$. Furthermore,

$$\underline{\square}_i(B, A) \cap \overline{\square}_j(B, A) = \emptyset$$

for all $i, j \in [k]$.

Proof. By Observation 4.23 it suffices to show that there is an index $\ell \in [k]$ such that the intersection of the corresponding ℓ-th intervals is empty. For the first part of the proposition assume w.l.o.g. that $i < j$. The i-th component of $\underline{\square}_j(B, A)$ is given by $[a_i, \bar{a}_i) \cap [b_i, \bar{b}_i)$ which is disjoint to (b_i, a_i), the i-th component of $\underline{\square}_i(B, A)$. Equivalently, the i-th component of $\overline{\square}_j(B, A)$ is given by $[a_i, \bar{a}_i) \cap [b_i, \bar{b}_i)$ which is disjoint to $[\bar{a}_i, \bar{b}_i)$, the i-th component of $\overline{\square}_j$. Note that the same argument as above can be used to show $\underline{\square}_i(B, A) \cap \overline{\square}_j(B, A) = \emptyset$ for $i \neq j$. In order to see that $\underline{\square}_i(B, A) \cap \overline{\square}_i(B, A) = \emptyset$ for all $i \in [k]$ observe that $[b_i, a_i) \cap [\bar{a}_i, \bar{b}_i) = \emptyset$ holds for all $i \in [k]$. □

Given two overlapping feasible boxes A and B with a nonempty common intersection (see Figure 4.17) we can partition the set union $A \cup B$ via lower and upper intersection boxes (see Figure 4.18).

Proposition 4.26. Let $A = \bigtimes_{i=1}^k [a_i, \bar{a}_i)$, $B = \bigtimes_{i=1}^k [b_i, \bar{b}_i) \subset \mathbb{R}^k$ be two feasible boxes with $A \cap B \neq \emptyset$. Then

$$\mathcal{D} = \{\underline{\square}_1(B, A), \overline{\square}_1(B, A), \ldots, \underline{\square}_k(B, A), \overline{\square}_k(B, A)\}$$

is a partition of $B \setminus A$ with $|\mathcal{D}| \leq 2k$.

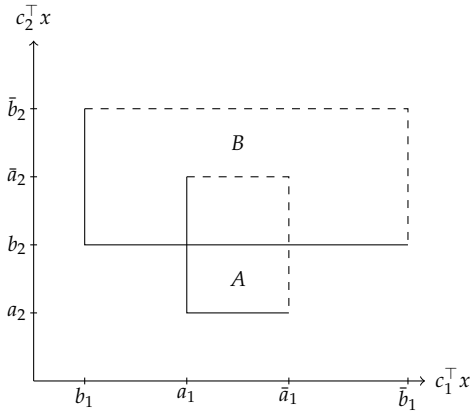

Figure 4.17: Overlapping feasible boxes A and B.

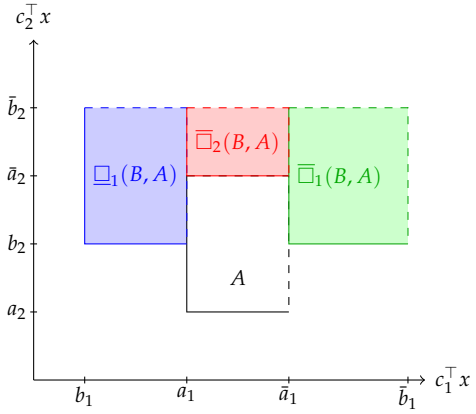

Figure 4.18: Partition of $A \cup B$ via upper and lower intersection boxes.

Proof. All boxes in $\mathcal{D}$ are pairwise disjoint by Proposition 4.25. By Definition 4.24, $\underline{\square}_i(B, A) \subset B \setminus A$ and $\overline{\square}_i(B, A) \subset B \setminus A$ hold for any $i \in [k]$. Hence, $\bigcup_{i\in[k]} \underline{\square}_i(B, A) \cup \overline{\square}_i(B, A) \subseteq B \setminus A$. Next, we show that $\bigcup_{i\in[k]} \underline{\square}_i(B, A) \cup \overline{\square}_i(B, A) \supseteq B \setminus A$ holds. If $x \in B \setminus A$, then there is some $i \in [k]$ such that

$$x \in \bigtimes_{j=1}^{i-1} ([a_j, \bar{a}_j) \cap [b_j, \bar{b}_j)) \times ([b_i, \bar{b}_i) \setminus [a_i, \bar{a}_i)) \times \bigtimes_{j=i+1}^{k} [b_j, \bar{b}_j).$$

The latter is equivalent to $x \in \underline{\square}_i(B, A) \cup \overline{\square}_i(B, A)$. □

Given two feasible boxes A and B, Algorithm 4.1 computes a partition of $A \cup B$ based on Proposition 4.26. Note that the correctness of Algorithm 4.1 follows from Proposition 4.26.

Given more than two feasible boxes $A_1, A_2, \ldots, A_n$ we can repeatedly apply Algorithm 4.1 in order to compute a partition $\mathcal{D}$ whose set union coincides with the set union of the given feasible boxes $A_1, \ldots, A_n$, see Algorithm 4.2.

4.6 SUMMARY AND DISCUSSION

In this chapter a partition of the set of nondominated points given by the set of poly-nondominated points and the set of mono-nondomin-

Algorithm 4.1 Computing a partition for the set union of two feasible boxes

Input: two feasible boxes $\emptyset \neq A \subset \mathbb{R}^k$, $\emptyset \neq B \subset \mathbb{R}^k$
Output: Partition $\mathcal{D}$ of $A \cup B$

set $\mathcal{D} = \{A\}$
if $A \cap B = \emptyset$ **then** add B to $\mathcal{D}$
else
    for $i = 1, \ldots, k$ **do**
        if $\underline{\square}_i(B, A) \neq \emptyset$ **then** add $\underline{\square}_i(B, A)$ to $\mathcal{D}$
        if $\overline{\square}_i(B, A) \neq \emptyset$ **then** add $\overline{\square}_i(B, A)$ to $\mathcal{D}$
return $\mathcal{D}$

Algorithm 4.2 Computing a partition for the set union of finitely many feasible boxes

function BOX-PARTITION(A, B)
    Return partition of $A \cup B$ based on Algorithm 4.1
Input: Finite number of feasible boxes $A_1, \ldots, A_n$ with $A_i \neq \emptyset$ for $i \in [n]$ and $A_i \not\subseteq A_j$ for $i \neq j$
Output: Partition $\mathcal{D}$ of $\bigcup_{i=1}^{n} A_i$
set $\mathcal{A} = \{A_1, \ldots, A_n\}$
set $\mathcal{D} = \{\}$
while $\mathcal{A} \neq \emptyset$ **do**
    set $A = \mathcal{A}$.lastElement()
    set $\mathcal{C} = \{\}$
    for D in $\mathcal{D}$ **do**
        if $D \cap A = \emptyset$ **then**
            add D to $\mathcal{C}$
        else if $D \subseteq A$ **then**
            continue
        else
            set $\mathcal{B} =$ BOX-PARTITION(A, D)
            for B in $\mathcal{B} \setminus \{A\}$ **do** add B to $\mathcal{C}$
    set $\mathcal{D} = \mathcal{C} \cup \{A\}$
    remove A from $\mathcal{A}$
return $\mathcal{D}$

ated points for general multicriteria integer programs was presented. It was shown that this partition does generally not coincide with the partition given by the set of supported and unsupported nondominated points. In Section 4.4 and in Section 4.5, respectively, it was shown that the number of mono-nondominated points can vastly exceed the number of poly-nondominated points. Moreover, all our computational results for the tricriteria assignment problem in Section 4.5.2.2 and for the tricriteria knapsack problem in Section 4.5.2.3 resulted in a vast number of redundant feasible boxes. We would like to highlight that this insight has not been previously touched upon by other researchers in this field. Note that having a vast number of redundant feasible boxes and a relatively small number of irredund-

ant feasible boxes sheds a different light on the overall practicality of an algorithm that computes mono-nondominated points via feasible boxes. Finally, in Section 4.5.3 an algorithm to compute a partition for overlapping feasible boxes was presented.

As indicated in Chapter 1 our perspective on multicriteria optimisation is twofold. The availability of solvers that can efficiently compute the entire set of nondominated points is fundamental. However, regarding the computational results in Section 4.5.2.2, we see that a medium-sized tricriteria assignment problem that assigns 50 agents to 50 jobs might easily result in about 25000 different nondominated points. Even if some solver computes this set of nondominated points in a short amount of time what is a decision maker supposed to do with 25000 different nondominated points? In this regard, computing disjoint subsets of nondominated points with different characteristics is a useful approach to tackle multicriteria integer programs resulting in a large number of nondominated points.

Our computational results (disregarding very small test instances) shows that the number of mono-nondominated points can vastly exceed the number of poly-nondominated points for assignment as well as for knapsack problems. But there is still a factor of 5 regarding the number of mono-nondominated points between assignment problems and knapsack problems. Whereas for the largest computed instances the number of poly-nondominated points is about 460 and 425, respectively, the number of mono-nondominated points is around 25000 for assignment problems but *only* about 5400 for knapsack problems. Since the number of resulting irredundant boxes in both problems has similar magnitude we get that an irredundant box contains on average about 10 mono-nondominated points for assignment problems and about 1 mono-nondominated point for knapsack problems for the largest investigated test instances. This indicates that for some problem classes considering a feasible box approach is a useful way to compute mono-nondominated points whereas for other problem classes less so.

Finally, note that there is more potential in improving a feasible box approach than just discarding redundant boxes followed by computing non-overlapping subboxes for overlapping boxes. First of all, in its presented version the size of the returned partition of Algorithm 4.1 is not independent on the input order. In Figure 4.17, the set union $A \cup B$ can be partitioned into two non-overlapping boxes B and $\square_2(B, A)$ instead of the four boxes in Figure 4.18.

More importantly, we do generally not know in advance which of the feasible boxes contain mono-nondominated points and which do not. However, if we find a nondominated point in some box, then this point will generally dominate certain parts of other feasible boxes which could be discarded. Hence, it could have a big impact to iterate first over boxes that would make (regions of) other previously irredundant boxes redundant, see Figure 4.19.

In Chapter 5 we consider the set of poly-nondominated points as an *approximation* for the entire set of nondominated points given the

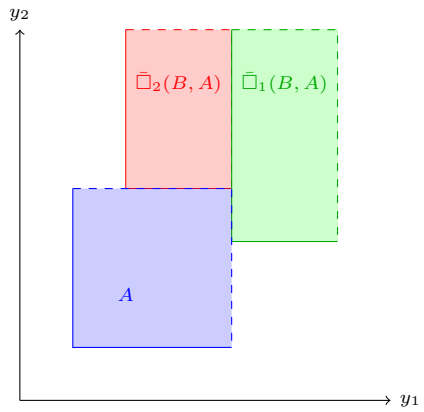

Figure 4.19: Non-overlapping box partition $\{A, \bar{\square}_1(B, A), \bar{\square}_2(B, A)\}$ of $A \cup B$ in which any point in A makes parts of $\bar{\square}_1(B, A) \cup \bar{\square}_2(B, A)$ redundant.

big difference in the number of poly-nondominated points and the number of mono-nondominated points.

5

APPROXIMATING MULTICRITERIA INTEGER PROGRAMS

> It is generally better to be approximately right than precisely wrong.
>
> *John Kay*

5.1 INTRODUCTION

In Chapter 4 we introduced poly-nondominated points and mono-nondominated points. We observed that for the considered tricriteria assignment problems and tricriteria knapsack problems the average number of mono-nondominated points generally exceeded the average number of poly-nondominated points (see Section 4.5.2). To overcome the case where the computation of the entire set of nondominated points $\mathcal{N}$ might be too computationally expensive, we can restrict the computation to a subset of nondominated points. The question is then which subset of nondominated points should be computed. Based on the insight that the set of poly-nondominated points yields boxes that contain the set of mono-nondominated points (see Theorem 4.7) and on the observation that the number of poly-nondominated points remained relatively small in our computational experiments the main idea in this chapter is to consider the set of poly-nondominated points as an *approximation* for the entire set of nondominated points.

In Section 5.2 we briefly summarise different approximation concepts in the context of multicriteria optimisation. In Section 5.3 we consider the set of supported nondominated points in the context of representations and in Section 5.4 we consider the set of poly-nondominated points in the context of representations. Section 5.5 presents computational results for the tricriteria assignment problem and for the tricriteria knapsack problem. Section 5.6 concludes this chapter with a summary and discussion.

5.2 NOTIONS OF APPROXIMATION

In this section we present three notions of approximation with regard to multicriteria optimisation. For a more comprehensive survey about approximation methods in linear and nonlinear multiobjective programming we refer the reader to [67].

5.2.1 *Approximation sets*

In [84] an *approximation set* for a given multicriteria problem instance is defined as a set containing feasible points which do not dominate each other. It is often considered in evolutionary multicriteria optimisation [26].

Note that even a subset of dominated points would establish an approximation set as long as the contained points do not dominate each other. In this regard, an approximation set is a relatively *weak* notion of approximation since it might contain dominated points and does not offer any information on the entire set of nondominated points $\mathcal{N}$ unless the latter is known. An advantage, however, is that for a given multicriteria integer program it is relatively easy to compute an approximation set. Note that a widely used quality measure to evaluate (different) approximation sets is the hypervolume indicator introduced by Zitzler and Thiele [85]. It represents the volume of the region dominated by the considered approximation set up to an additionally considered upper bound vector.

5.2.2 *ϵ-approximate Pareto sets*

Definition 5.1. (ϵ-approximate Pareto set) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria optimisation problem with nonnegative image $\mathcal{Y} \subseteq \mathbb{R}^k_{\geq 0}$ and let $\epsilon > 0$ be positive. A set $Y_\epsilon \subseteq \mathcal{Y}$ is an *ϵ-approximate Pareto set* if for all nondominated points $y^* \in \mathcal{N}$ there is $y \in Y_\epsilon$ such that $y_i \leq (1+\epsilon) y_i^*$ for all $i \in [k]$.

Hansen [42] was the first to show that for the bicriteria shortest s-t-path problem (see Definition 2.19) an ϵ-approximate Pareto set can be constructed in polynomial time by considering a pseudopolynomial dynamic programming generalisation of Dijkstra's algorithm. Safer [68] gave necessary and sufficient conditions for the existence of fast approximation schemes and showed the existence of fast approximation schemes for a variety of flow, knapsack, and lot-sizing problems. Papadimitriou and Yannakakis [61] showed that ϵ-approximate Pareto sets generally exist and that the question whether they can be constructed in polynomial time reduces to the problem whether, for a given point $y \in \mathcal{Y}$ and $\epsilon > 0$, $y^* \in \mathcal{Y}$ with $y^* \leq y$ can be found in polynomial time or the answer that there is no point $\bar{y} \in \mathcal{Y}$ with $\bar{y} \leq (1+\epsilon) y$ can be given in polynomial time. Vassilvitskii and Yannakakis [80] extended the results given in [61] aiming to compute smallest possible ϵ-approxmiate Pareto sets. Diakonikolas and Yannakakis [27] consider and extend previous results to the convex case by proposing ϵ-convex Pareto sets, i.e. sets whose convex hull contains ϵ-approximate Pareto sets.

5.2.3 *Representations*

In contrast to approximation sets and ϵ-approximate Pareto sets a *representation* is defined as a finite set of nondominated points.

Definition 5.2. (Representation) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a given multicriteria optimisation problem. A finite set of nondominated points $Y \subseteq \mathcal{N}$ is called a *representation* of $\mathcal{N}$.

In [69] *coverage*, *uniformity* and *cardinality* are considered as three quality attributes for a representation. Coverage refers to the property that all points in $\mathcal{N}$ should be well-represented by the representation, i.e. for all nondominated points $y^* \in \mathcal{N}$ there should be a point y in the representation such that y is *close* to y^*. Uniformity refers to the property that any redundancies or clustered points, respectively, should be avoided in a representation. Cardinality refers to the property that the number of points in a representation should be kept (as) small (as possible). Is it not hard to see that these three attributes might be in conflict to each other.

In the remainder of this chapter we will consider representations focussing on the coverage attribute.

Definition 5.3. (ϵ-representation) Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria optimisation problem and let $\epsilon > 0$ be positive. A set $Y_\epsilon \subseteq \mathcal{N}$ is an *ϵ-representation* if Y_ϵ is a representation of $\mathcal{N}$ and for all $y^* \in \mathcal{N}$ there is $y \in Y_\epsilon$ such that

$$\|y^* - y\|_\infty \leq \epsilon. \tag{5.1}$$

ϵ is the *coverage error* of Y_ϵ.

Note that ϵ-representations can be defined in more general terms via a *metric* $d : \mathcal{N} \times \mathcal{N} \to \mathbb{R}_{\geq 0}$, see [69]. Regarding Definition 5.3, given a representation we would like to be able to compute a corresponding coverage error without knowing the entire set of nondominated points.

The relation between ϵ-representations and ϵ-approximate Pareto sets can be characterised as follows.

Proposition 5.4. Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria integer program with a finite set of positive nondominated points $\mathcal{N} \subset \mathbb{R}^k_+$ and let $y^I \in \mathbb{R}^k_+$ be the corresponding ideal point. If Y_ϵ is an ϵ-representation with coverage error $\epsilon > 0$, then Y_ϵ is an $\bar{\epsilon}$-approximate Pareto set where $\bar{\epsilon} = \epsilon / \min_{i \in [k]} y^I_i$.

Proof. Let $y^* \in \mathcal{N}$ be arbitrary but fixed. By Definition 5.3, there is $y \in Y_\epsilon$ such that $\|y^* - y\|_\infty \leq \epsilon$. The latter implies that $y_i \leq y^*_i + \epsilon$ for all $i \in [k]$. Define $\bar{\epsilon} = \epsilon / \min_{i \in [k]} y^I_i$. We get

$$y_i \leq y^*_i + \epsilon = y^*_i + \epsilon \frac{y^*_i}{y^*_i} \leq y^*_i + \epsilon \frac{y^*_i}{y^I_i} \leq y^*_i + \bar{\epsilon} y^*_i = (1 + \bar{\epsilon}) y^*_i$$

for all $i \in [k]$. □

In general, given an ϵ-approximate Pareto set Y_ϵ we cannot calculate a coverage error for (5.1) without knowing the entire set of nondominated points $\mathcal{N}$. Consider a nondominated point $y^* \in \mathcal{N}$ and its approximation $y \in Y_\epsilon$ in the ϵ-approximate Pareto set, i.e. $y_i \leq (1 + \epsilon) y^*_i$ holds for all $i \in [k]$. There might be an index $j \in [k]$ such that $y_j < y^*_j$. In this case $|y_j - y^*_j|$ can be arbitrarily large without contradicting $y_i \leq (1 + \epsilon) y^*_i$ for any $i \in [k]$.

5.3 REPRESENTATION VIA SUPPORTED NONDOMINATED POINTS

In this section we consider the set of supported nondominated points $\mathcal{N}_s$ in the context of ϵ-representations (see Definition 5.3). In Section 5.3.1 we show that for bicriteria integer programs the set of supported nondominated points $\mathcal{N}_s$ yields an ϵ-representation Y_ϵ with a coverage error ϵ which can be computed by only considering $\mathcal{N}_s$ without the need to know the entire set of nondominated points. The main ingredient for this result is the property that the values of any unsupported nondominated point are bounded from below and from above by the values of two supported nondominated points. Subsequently, in Section 5.3.2 we provide an example for the tricriteria case that shows that the set of supported nondominated points $\mathcal{N}_s$ does not yield upper bounds on the values of unsupported nondominated points like in the bicriteria case. This indicates that in the tricriteria case the coverage error of the ϵ-representation that is given by the set of supported nondominated points $\mathcal{N}_s$ cannot be computed by only considering $\mathcal{N}_s$ itself.

5.3.1 *The bicriteria case*

In order to establish a coverage error we will make use of the following straightforward result which establishes a bound on the distance of the defining points of a rectangle and any point located in the rectangle.

Proposition 5.5. Let $y, z \in \mathbb{R}^2$ be two points with $y_1 < z_1$ and $y_2 > z_2$. Let $R(y,z)$ be the rectangle given by the four vertices y, (z_1, y_2), z, $(y_1, z_2) \in \mathbb{R}^2$ and let $\ell_1 = z_1 - y_1$ and $\ell_2 = y_2 - z_2$ be the side lengths of the rectangle $R(y,z)$. Then, for every $u \in R(y,z)$ we have

$$\min(\|y-u\|_\infty, \|z-u\|_\infty) \leq \max(\frac{\max(\ell_1,\ell_2)}{2}, \min(\ell_1,\ell_2)).$$

Proof. Split $R(y,z)$ into two subrectangles R^1, R^2 as follows: If $\ell_1 \geq \ell_2$, then R^1 is given by the vertices y, $(y_1 + \frac{\ell_1}{2}, z_2)$, $(y_1 + \frac{\ell_1}{2}, z_2)$, (y_1, z_2) and R^2 is given by the vertices $(y_1 + \frac{\ell_1}{2}, y_2)$, (z_1, y_2), z, $(y_1 + \frac{\ell_1}{2}, z_2)$, see Figure 5.1. Otherwise, if $\ell_1 < \ell_2$, then R^1 is given by the vertices y, (z_1, y_2), $(z_1, z_2 + \frac{\ell_2}{2})$, $(y_1, z_2 + \frac{\ell_2}{2})$ and R^2 is given by the vertices $(y_1, z_2 + \frac{\ell_2}{2})$, $(z_1, z_2 + \frac{\ell_2}{2})$, z, (y_1, z_2), see Figure 5.2. Then, if $\ell_1 \geq \ell_2$, it holds that for every $u \in R^1$ we have $\|y-u\|_\infty \leq \max(\frac{\ell_1}{2}, \ell_2)$ and for every $u \in R^2$ we have $\|z-u\|_\infty \leq \max(\frac{\ell_1}{2}, \ell_2)$. In other words, if $\ell_1 \geq \ell_2$, then for every $u \in R(y,z)$ we get $\min(\|y-u\|_\infty, \|z-u\|_\infty) \leq \max(\frac{\ell_1}{2}, \ell_2)$. Equivalently, if $\ell_1 < \ell_2$, then for every $u \in R(y,z)$ we get $\min(\|y-u\|_\infty, \|z-u\|_\infty) \leq \max(\frac{\ell_2}{2}, \ell_1)$. Combining both cases we get

$$\min(\|y-u\|_\infty, \|z-u\|_\infty) \leq \max(\frac{\max(\ell_1,\ell_2)}{2}, \min(\ell_1,\ell_2)).$$

□

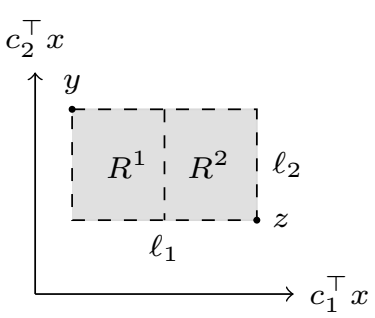

Figure 5.1: Rectangle splitting if $\ell_1 \geq \ell_2$.

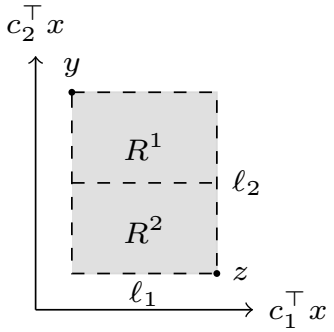

Figure 5.2: Rectangle splitting if $\ell_1 < \ell_2$.

As described in Section 2.3.2, for bicriteria integer programs the set of supported nondominated points $\mathcal{N}_s$ yields lower bounds as well as upper bounds on the values of any unsupported nondominated point. This implies that $\mathcal{N}_s$ yields an ϵ-representation for the entire set of nondominated points with a coverage error that can be computed from $\mathcal{N}_s$ itself.

Theorem 5.6. Let $(c_1, c_2, \mathcal{X})$ be a bicriteria integer program with a nonempty and finite set of nondominated points $\mathcal{N}$. Let $\mathcal{N}_s = \{y^1, \ldots, y^n\}$ be the corresponding set of supported nondominated points where $y_1^i < y_1^{i+1}$ for all $i \in [n-1]$. Furthermore, for $i \in [n-1]$, let $\ell_1^i = y_1^{i+1} - y_1^i$ and $\ell_2^i = y_2^i - y_2^{i+1}$ be the componentwise distance of neighbouring supported nondominated points. Then, $\mathcal{N}_s$ yields an ϵ-representation with (maximal) coverage error

$$\epsilon = \max_{i \in [n-1]} \left(\frac{\max(\ell_1^i, \ell_2^i)}{2},\ \min(\ell_1^i, \ell_2^i)\right).$$

Proof. By Definition 2.1 and by Definition 2.7, it holds that for every $u \in \mathcal{N}_u$ there are $y^i, y^{i+1} \in \mathcal{N}_s$ for some $i \in [n-1]$ such that $y_1^i < u_1 < y_1^{i+1}$ and $y_2^i > u_2 > y_2^{i+1}$. Then, Proposition 5.5 implies that for every $u \in \mathcal{N}_u$ there are $y^i, y^{i+1} \in \mathcal{N}_s$ for some $i \in [n-1]$ such that

$$\min(\|y^i - u\|_\infty, \|y^{i+1} - u\|_\infty) \leq \max\left(\frac{\max(\ell_1^i, \ell_2^i)}{2}, \min(\ell_1^i, \ell_2^i)\right).$$

It follows that for all $y^* \in \mathcal{N}$ there is $y \in \mathcal{N}_s$ such that

$$\|y^* - y\|_\infty \leq \max_{i \in [n-1]} \left(\frac{\max(\ell_1^i, \ell_2^i)}{2},\ \min(\ell_1^i, \ell_2^i)\right).$$

□

Note that the coverage error in Theorem 5.6 is only tight in the worst-case, i.e. there is an unsupported nondominated point which regarding the i-th component is located exactly halfway between two neighbouring supported nondominated points and the distance in the i-th component between this neighbouring pair corresponds to the maximal distance of any neighbouring supported pair taken over

all components, see Figure 5.3. For example, if the unsupported nondominated point $u^3 \in \mathcal{N}_u$ in Figure 5.3 were located closer to either $y^3 \in \mathcal{N}_s$ or $y^4 \in \mathcal{N}_s$, then the theoretically given coverage error would not be tight anymore. In other words, the *practical* coverage error given by the set of supported nondominated points $\mathcal{N}_s$ might be lower than the one given in Theorem 5.6. However, the latter value can be computed by only considering the set of supported nondominated points $\mathcal{N}_s$ whereas we are generally not able to compute the practical coverage error without knowing the entire set of nondominated points $\mathcal{N}$.

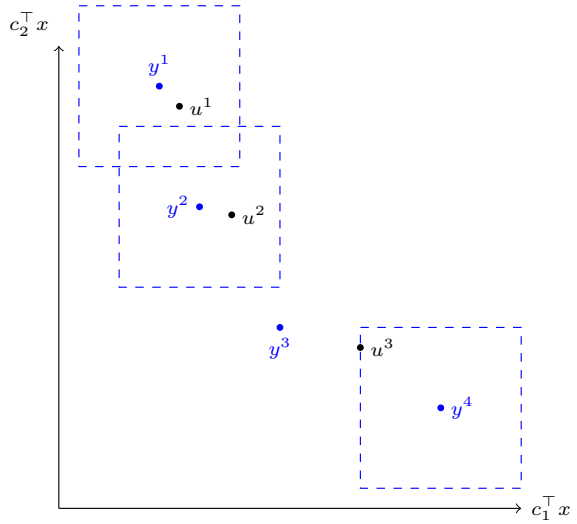

Figure 5.3: Set of supported nondominated points $\mathcal{N}_s = \{y^1, y^2, y^3, y^4\}$ yielding an ϵ-representation Y_ϵ with coverage error $\epsilon = \frac{y_1^4 - y_1^3}{2}$.

5.3.2 *The tricriteria case*

In Section 5.3.1 we saw that in the bicriteria case the set of supported nondominated points $\mathcal{N}_s$ yields boxes which contain any unsupported nondominated point. These boxes enable us to compute a coverage error of the corresponding ϵ-representation given by $\mathcal{N}_s$ without knowing the entire set of nondominated points. In the tricriteria case such potential boxes resulting from supported nondominated points might need to be constructed in a more complicated fashion (e.g. see the construction of the feasible boxes in Chapter 4) such that it is not clear whether a nontrivial coverage error of the ϵ-representation given by the set of supported nondominated points $\mathcal{N}_s$ could be computed by only considering $\mathcal{N}_s$. In the remainder of this section we provide an example that shows that the Chebyshev distance between any supported nondominated point and any unsupported nondominated point might become arbitrarily large in the tricriteria case such that any box whose values are based solely on the values of supported nondominated points does not suffice to contain any unsupported nondominated point.

Example 5.7. Let

$$c_1 = \begin{pmatrix} -4,6,4,5 \\ 2,3,-1,4 \\ 3,5,4,2 \\ 4,5,3,6 \end{pmatrix},\ c_2(n) = \begin{pmatrix} 2,n,5,4 \\ 5,3,3,3 \\ 5,2,6,4 \\ 4,5,2,5 \end{pmatrix} \text{ and } c_3 = \begin{pmatrix} 4,2,4,2 \\ 4,2,4,6 \\ 4,2,6,3 \\ 2,4,5,3 \end{pmatrix}$$

be coefficient matrices and consider, for $n \in \mathbb{N}$, the tricriteria assignment problem instance

$$\begin{aligned} \min\ & (\sum_{i=1}^{4}\sum_{j=1}^{4} c_{1_{ij}} x_{ij},\ \sum_{i=1}^{4}\sum_{j=1}^{4} c_2(n)_{ij} x_{ij},\ \sum_{i=1}^{4}\sum_{j=1}^{4} c_{3_{ij}} x_{ij}) \\ \text{s.t.}\ & \sum_{i=1}^{4} x_{ij} = 1 \ \text{ for all } j \in [4], \\ & \sum_{j=1}^{4} x_{ij} = 1 \ \text{ for all } i \in [4], \\ & x_{ij} \in \{0,1\}. \end{aligned} \tag{5.2}$$

Let $\mathcal{N}_s(n)$ and $\mathcal{N}_u(n)$, respectively, be the corresponding set of supported nondominated points and unsupported nondominated points of (5.2) for $n \in \mathbb{N}$. We get that, for all $n \geq 2$,

$$\max_{y,z \in \mathcal{N}_s(n)} \|y - z\|_\infty = 11 \tag{5.3}$$

holds. Moreover, for $n \geq 15$, we have

$$\max_{y,z \in \mathcal{N}_s(n)} \|y - z\|_\infty < \min_{\substack{y \in \mathcal{N}_s(n) \\ u \in \mathcal{N}_u(n)}} \|y - u\|_\infty \tag{5.4}$$

and, in particular,

$$\min_{\substack{y \in \mathcal{N}_s(n) \\ u \in \mathcal{N}_u(n)}} \|y - u\|_\infty = n - 3. \tag{5.5}$$

In order to confirm (5.3), (5.4) and (5.5) let us consider the feasible domain $\mathcal{X}$ and the corresponding image $\mathcal{Y}$ of (5.2). The feasible domain $\mathcal{X}$ is given by the feasible solutions $x^1, \ldots, x^{24}$ and $\mathcal{Y}$ is given by the feasible points $y^1, \ldots, y^{24} \in \mathcal{Y}$ with the following values:

- $x^1 = (0,0,1,0,0,1,0,0,0,0,0,1,1,0,0,0),\ y^1 = (13,16,11)$
- $x^2 = (1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,1),\ y^2 = (6,12,13)$
- $x^3 = (0,0,0,1,0,1,0,0,1,0,0,0,0,0,1,0),\ y^3 = (14,14,13)$
- $x^4 = (0,1,0,0,1,0,0,0,0,0,0,1,0,0,1,0),\ y^4 = (13,11+n,14)$
- $x^5 = (0,1,0,0,0,0,0,1,0,0,1,0,1,0,0,0),\ y^5 = (18,13+n,16)$
- $x^6 = (0,1,0,0,0,0,1,0,0,0,0,1,1,0,0,0),\ y^6 = (11,11+n,11)$
- $x^7 = (0,1,0,0,0,0,1,0,1,0,0,0,0,0,0,1),\ y^7 = (14,13+n,13)$
- $x^8 = (0,0,0,1,0,1,0,0,0,0,1,0,1,0,0,0),\ y^8 = (16,17,12)$

- $x^9 = (0,0,1,0,1,0,0,0,0,0,0,1,0,1,0,0)$, $y^9 = (13,19,15)$
- $x^{10} = (0,0,0,1,1,0,0,0,0,0,0,1,0,0,1,0,0)$, $y^{10} = (16,20,16)$
- $x^{11} = (1,0,0,0,0,0,0,1,0,1,0,0,0,0,1,0)$, $y^{11} = (8,9,17)$
- $x^{12} = (0,1,0,0,0,0,0,1,1,0,0,0,0,0,1,0)$, $y^{12} = (16,10,17)$
- $x^{13} = (0,0,0,1,1,0,0,0,0,1,0,0,0,0,1,0)$, $y^{13} = (15,13,13)$
- $x^{14} = (1,0,0,0,0,1,0,0,0,0,0,1,0,0,1,0)$, $y^{14} = (4,11,14)$
- $x^{15} = (0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,1)$, $y^{15} = (18,16,15)$
- $x^{16} = (1,0,0,0,0,0,1,0,0,0,0,1,0,1,0,0)$, $y^{16} = (2,14,15)$
- $x^{17} = (0,0,0,1,0,0,1,0,1,0,0,0,0,1,0,0)$, $y^{17} = (12,17,14)$
- $x^{18} = (0,0,1,0,1,0,0,0,0,1,0,0,0,0,0,1)$, $y^{18} = (17,17,13)$
- $x^{19} = (0,0,1,0,0,0,0,1,0,1,0,0,1,0,0,0)$, $y^{19} = (17,14,14)$
- $x^{20} = (0,0,1,0,0,0,0,1,1,0,0,0,0,1,0,0)$, $y^{20} = (16,18,18)$
- $x^{21} = (1,0,0,0,0,0,0,1,0,0,1,0,0,1,0,0)$, $y^{21} = (9,16,20)$
- $x^{22} = (1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,1)$, $y^{22} = (9,16,15)$
- $x^{23} = (0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,1)$, $y^{23} = (16,18,13)$
- $x^{24} = (0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0)$, $y^{24} = (13,13,10)$

Firstly, observe that the only feasible points whose values depend on n are y^4, y^5, y^6 and y^7. Secondly, a closer look reveals that y^4, y^5, y^7 are always dominated by y^6 irrespective of the choice of $n \in \mathbb{N}$. Further inspection reveals that the entire set of nondominated points $\mathcal{N}$ is always given by $\mathcal{N} = \{y^2, y^6, y^{11}, y^{14}, y^{16}, y^{24}\}$ for all $n \in \mathbb{N}$. Next, we show that $y^2, y^{11}, y^{14}, y^{16}$ and y^{24} are supported nondominated points irrespective of the choice of $n \in \mathbb{N}$. Regarding Theorem 2.6, to show that $y^* \in \mathcal{N}$ is supported we need to find a positive weight vector $\lambda \in \mathbb{R}^3_+$ such that $y^* \in \arg\min_{y \in \mathcal{N}} \lambda^\top y$ holds. As $y^{11}, y^{16}, y^{24} \in \mathcal{N}$ are the lexicographically optimal points it is not difficult to find such weight vectors by considering $(1,\delta,\delta), (\delta,1,\delta), (\delta,\delta,1) \in \mathbb{R}^3_+$ with a sufficiently small $\delta > 0$. In order to see that $y^2 \in \mathcal{N}$ is supported nondominated we can take $\lambda = (44,1,100) \in \mathbb{R}^3_+$. Furthermore, to see that $y^{14} \in \mathcal{N}$ is supported nondominated we can take $\lambda = (1,1,1) \in \mathbb{R}^3_+$. Next, we argue that for all $n \geq 2$ the set of unsupported nondominated points $\mathcal{N}_u$ is the singleton $\mathcal{N}_u = \{y^6\}$. Consider $\lambda^\top y^6 \leq \lambda^\top y^2$. We get $11\lambda_1 + (11+n)\lambda_2 + 11\lambda_3 \leq 6\lambda_1 + 12\lambda_2 + 13\lambda_3$. The latter holds if and only if

$$2.5\lambda_1 + (0.5n - 0.5)\lambda_2 \leq \lambda_3. \tag{5.6}$$

Moreover, consider $\lambda^\top y^6 \leq \lambda^\top y^{24}$. We get $11\lambda_1 + (11+n)\lambda_2 + 11\lambda_3 \leq 13\lambda_1 + 13\lambda_2 + 10\lambda_3$. The latter holds if and only if

$$2\lambda_1 + (2-n)\lambda_2 \geq \lambda_3. \tag{5.7}$$

A closer look reveals that for all $n \geq 2$ there is no $\lambda \in \mathbb{R}^3_+$ which fulfils (5.6) and (5.7) simultaneously. Hence, we have $\mathcal{N}_u(n) = \{y^6\}$ and $\mathcal{N}_s(n) = \{y^2, y^{11}, y^{14}, y^{16}, y^{24}\}$ for all $n \geq 2$. Then, we get that

$$\max_{y,z \in \mathcal{N}_s(n)} \|y - z\|_\infty = \|y^{16} - y^{24}\|_\infty = 11$$

holds for all $n \geq 2$. Moreover, for all $n \geq 15$, we get that

$$\min_{\substack{y \in \mathcal{N}_s(n) \\ u \in \mathcal{N}_u(n)}} \|y - u\|_\infty = \min_{y \in \mathcal{N}_s(n)} \|y - y^6\|_\infty = \|y^{16} - y^6\|_\infty = n - 3$$

holds which implies (5.4).

5.4 REPRESENTATION VIA POLY-NONDOMINATED POINTS

In this section we show that the set of poly-nondominated points $\mathcal{N}_p$ yields an ϵ-representation with coverage error that can be computed solely from $\mathcal{N}_p$ for general multicriteria integer programs.

Before we state the next proposition, we remind the reader of a *feasible box* defined in Definition 4.17 in Section 4.5.1.

Proposition 5.8. Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria integer program and let $B(y^1, \ldots, y^k)$ be a feasible box given by the poly-nondominated points $y^1 \in \mathcal{N}_1, \ldots, y^k \in \mathcal{N}_k$. Then, for any $z \in B(y^1, \ldots, y^k)$

$$\min_{i \in [k]} \|y^i - z\|_\infty \leq \min_{i \in [k]} (\max(y^i_i - \max_{j \in [k] \setminus \{i\}} y^j_i \, , \max_{j \in [k] \setminus \{i\}} |y^i_j - y^j_j|)) \quad (5.8)$$

holds.

Proof. For $\ell \in [k]$ we have

$$\begin{aligned} \|y^\ell - z\|_\infty &= \max(|y^\ell_\ell - z_\ell|, \max_{j \in [k] \setminus \{\ell\}} |y^\ell_j - z_j|) \\ &\leq \max(y^\ell_\ell - \max_{j \in [k] \setminus \{\ell\}} y^j_\ell \, , \max_{j \in [k] \setminus \{\ell\}} |y^\ell_j - y^j_j|) \end{aligned}$$

since $\max_{j \in [k] \setminus \{i\}} y^j_i \leq z_i < y^i_i$ for all $i \in [k]$. □

From here on for ease of presentation we denote the right hand side of (5.8) by

$$\delta(B(y^1, \ldots, y^k)) = \min_{i \in [k]} (\max(y^i_i - \max_{j \in [k] \setminus \{i\}} y^j_i \, , \max_{j \in [k] \setminus \{i\}} |y^i_j - y^j_j|)).$$

The next result shows us that the set of poly-nondominated points $\mathcal{N}_p$ yields an ϵ-representation with a coverage error that can be computed solely from $\mathcal{N}_p$.

Theorem 5.9. Let $(c_1, \ldots, c_k, \mathcal{X})$ be a multicriteria integer program with a finite set of nondominated points $\mathcal{N}$. Let $\mathcal{B}$ be the set of all feasible boxes given by $\mathcal{N}_{-1}, \ldots, \mathcal{N}_{-k}$. Then, $\mathcal{N}_p$ yields an ϵ-representation with a (maximal) coverage error

$$\epsilon = \max_{B(y^1, \ldots, y^k) \in \mathcal{B}} \delta(B(y^1, \ldots, y^k)).$$

Proof. Let $z \in \mathcal{N} \setminus \mathcal{N}_p$ be arbitrary but fixed. By Theorem 4.7 and Definition 4.17, there are $y^1 \in \mathcal{N}_1, \dots, y^k \in \mathcal{N}_k$ yielding a feasible box $B(y^1, \dots, y^k)$ such that $z \in B(y^1, \dots, y^k)$. By Proposition 5.8, we get $\min_{i \in [k]} \|y^i - z\|_\infty \leq \delta(B(y^1, \dots, y^k))$. Taking the maximum over all feasible boxes yields the result. □

Example 5.10. Consider the tricriteria assignment problem (cf. Example 4.6)

$$\begin{aligned} \min\ & \Big(\sum_{i=1}^{4}\sum_{j=1}^{4} c_{1_{ij}} x_{ij},\ \sum_{i=1}^{4}\sum_{j=1}^{4} c_{2_{ij}} x_{ij},\ \sum_{i=1}^{4}\sum_{j=1}^{4} c_{3_{ij}} x_{ij}\Big) \\ \text{s.t.}\ & \sum_{i=1}^{4} x_{ij} = 1 \ \text{ for all } j \in [4], \\ & \sum_{j=1}^{4} x_{ij} = 1 \ \text{ for all } i \in [4], \\ & x_{ij} \in \{0,1\}, \end{aligned} \tag{5.9}$$

where

$$c_1 = \begin{pmatrix} 3,6,4,5 \\ 2,3,5,4 \\ 3,5,4,2 \\ 4,5,3,6 \end{pmatrix},\ c_2 = \begin{pmatrix} 2,3,5,4 \\ 5,3,4,3 \\ 5,2,6,4 \\ 4,5,2,5 \end{pmatrix} \text{ and } c_3 = \begin{pmatrix} 4,2,4,2 \\ 4,2,4,6 \\ 4,2,6,3 \\ 2,4,5,3 \end{pmatrix}.$$

We have $\mathcal{N}_1 = \{y^1, y^2, y^3, y^5\}$, $\mathcal{N}_2 = \{y^1, y^3, y^4\}$, $\mathcal{N}_3 = \{y^1, y^2\}$ where $y^1 = (11, 11, 14)$, $y^2 = (15, 9, 17)$, $y^3 = (19, 14, 10)$, $y^4 = (13, 16, 11)$, $y^5 = (15, 13, 13)$. Overall, there are three feasible boxes given by $B^1 = B(y^5, y^4, y^1)$, $B^2 = B(y^3, y^4, y^2)$ and $B^3 = B(y^3, y^4, y^1)$, see Figure 5.4. We get

$$\begin{aligned} \delta(B^1) &= \min(\max(2,3),\ \max(3,3),\ \max(1,5)) = 3, \\ \delta(B^2) &= \min(\max(4,7),\ \max(2,6),\ \max(6,7)) = 6, \\ \delta(B^3) &= \min(\max(6,4),\ \max(2,6),\ \max(3,8)) = 6 \end{aligned}$$

and conclude that the set of poly-nondominated points $\mathcal{N}_p = \{y^1, y^2, y^3, y^4, y^5\}$ yields an ϵ-representation for the entire set of nondominated points $\mathcal{N}$ with a (maximal) coverage error $\epsilon = \max(3, 6, 6) = 6$.

5.5 COMPUTATIONAL RESULTS

In this section we present computational results based on test instances of the tricriteria assignment problem and the tricriteria knapsack problem, respectively. All instances were generated by Kirlik and Sayın [47] and are available online [77]. To get a broader picture of the representation qualities of the set of poly-nondominated points $\mathcal{N}_p$ and the set of supported nondominated points $\mathcal{N}_s$, respectively, we consider the corresponding cardinality and the corresponding coverage error. For the set of poly-nondominated points $\mathcal{N}_p$ the maximum coverage error can be computed by Theorem 5.9. However,

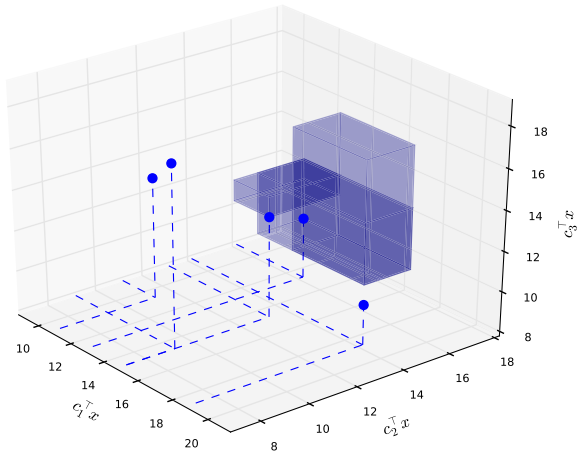

Figure 5.4: Entire set of feasible boxes for the tricriteria integer program given in (5.9).

this bound might not be tight and might differ from the *practically* given coverage error of $\mathcal{N}_p$, i.e. the value given by

$$\max_{y^* \in \mathcal{N}} \min_{y \in \mathcal{N}_p} \|y^* - y\|_\infty$$

Similarly, the practically given coverage error of the set of supported nondominated points $\mathcal{N}_s$ is given by $\max_{y^* \in \mathcal{N}} \min_{y \in \mathcal{N}_s} \|y^* - y\|_\infty$. Note that both values can (only) be computed if the entire set of nondominated points $\mathcal{N}$ is available. An interesting question is how the practically given coverage error of set of poly-nondominated points compares to the practically given coverage error of the set of supported nondominated points.

Regarding the computational experiments we will consider the following (average) values and abbreviations:

AN the average number of nondominated points,

AS the average number of supported nondominated vertices,

AP the average number of poly-nondominated points,

ASP the average number of the set union of supported nondominated vertices and poly-nondominated points.

S the average spread, i.e. the average Chebyshev distance between the ideal and nadir point,

TCP the average theoretically given coverage error of the set of poly-nondominated points (see Theorem 5.9),

PCP the average practically given coverage error of poly-nondominated points,

PCS the average practically given coverage error of the set of supported nondominated vertices,

PCSP the average practically given coverage error of the set union of supported nondominated vertices and poly-nondominated points.

Note that by supported nondominated vertex we refer to a supported nondominated point which is a vertex of $conv(\mathcal{Y})$.

5.5.1 *Tricriteria assignment problem*

There are 100 tricriteria assignment problem instances available at [77]. The objective function coefficients are integers which are randomly generated from the interval $[1, 20]$. The number of agents (and jobs, respectively) of the generated test instances ranges from $n = 5$ to $n = 50$ with a step size of 5. For each $n \in \{5, 10, 15, \ldots, 50\}$ there are 10 instances. Note that for each n the average value is taken over all 10 instances.

5.5.1.1 *Average number of different nondominated point sets*

Table 3 shows the different average number of points AN, AS, AP, ASP (see page 66 for a definition of the used abbreviations) and different corresponding ratios AS/AN, AP/AN, ASP/AN computed for $n = 5, 10, \ldots, 50$. Figure 5.5 illustrates the average number of points for $n = 5, 10, \ldots, 25$ and Figure 5.6 illustrates the average number of points for $n = 30, 35, \ldots, 50$.

Table 3: Average number of different nondominated point sets for the tricriteria assignment problem.

n	AN	AS	AS/AN	AP	AP/AN	ASP	ASP/AN
5	14.1	7.5	0.532	10	0.709	10.2	0.723
10	176.8	37.9	0.214	53.3	0.301	64.5	0.365
15	674.9	78.5	0.116	98.4	0.146	134.2	0.199
20	1860.5	152.5	0.082	164.8	0.0886	255.1	0.137
25	3567.8	222.2	0.0623	195.2	0.0547	344.5	0.0966
30	6181.3	346.3	0.056	265.5	0.043	509.4	0.0824
35	8972.3	454.9	0.0507	308.8	0.0344	649.7	0.0724
40	14679.7	627.7	0.0428	356.2	0.024	844.3	0.0575
45	17702.2	703.6	0.0397	386.9	0.0219	942.6	0.0532
50	24916.8	919.4	0.0369	424.6	0.017	1180.7	0.0474

5.5.1.2 *Average representation*

Table 4 shows the average values S, TCP, PCP, PCS, PCSP (see page 66 for a definition of the used abbreviations) computed for $n = 5, 10, \ldots, 50$. Figure 5.7 illustrates these values for $n = 5, 10, \ldots, 25$ and Figure 5.8 illustrates them for $n = 30, 35, \ldots, 50$.

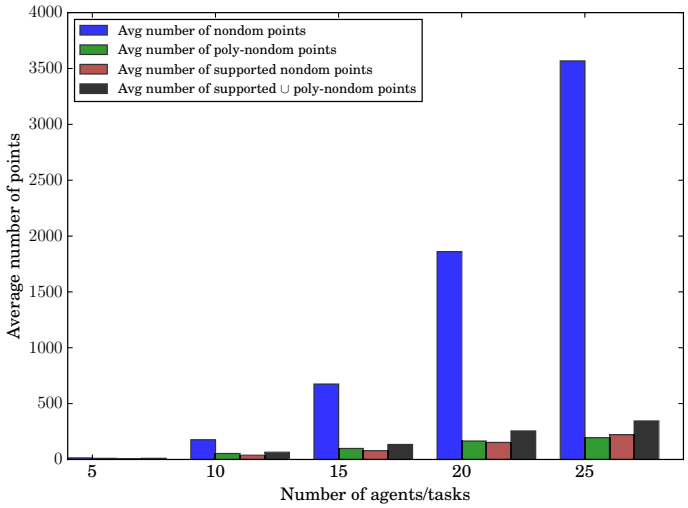

Figure 5.5: Average number of different nondominated point sets of the tricriteria assignment problem with $n = 5, 10, \ldots, 25$ agents assigned to n tasks.

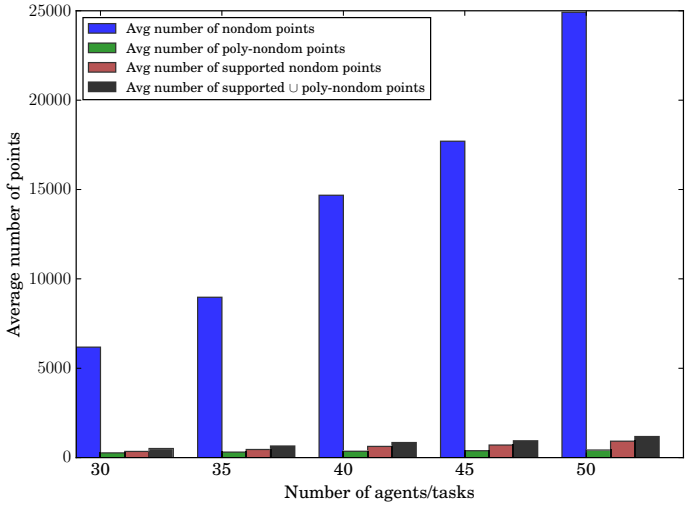

Figure 5.6: Average number of different nondominated point sets of the tricriteria assignment problem with $n = 30, 35, \ldots, 50$ agents assigned to n tasks.

Table 4: Average representation values of different nondominated point sets for the tricriteria assignment problem.

n	S	TCP	PCP	PCS	PCSP
5	43.9	37.1	7.2	11.6	7.2
10	106.3	104.6	21.8	23.4	14.3
15	164.6	164	41.9	25.6	20.3
20	229.2	227.9	69.2	27.9	20.9
25	275	274.4	102.2	32.5	27.3
30	339.3	339.2	124.7	42	31
35	386.5	386.3	153.2	39.2	35.8
40	445.4	445.2	183	44.4	41.5

45	488.5	488.5	217	58.3	56.9
50	549.4	549.2	246.8	51.3	48.3

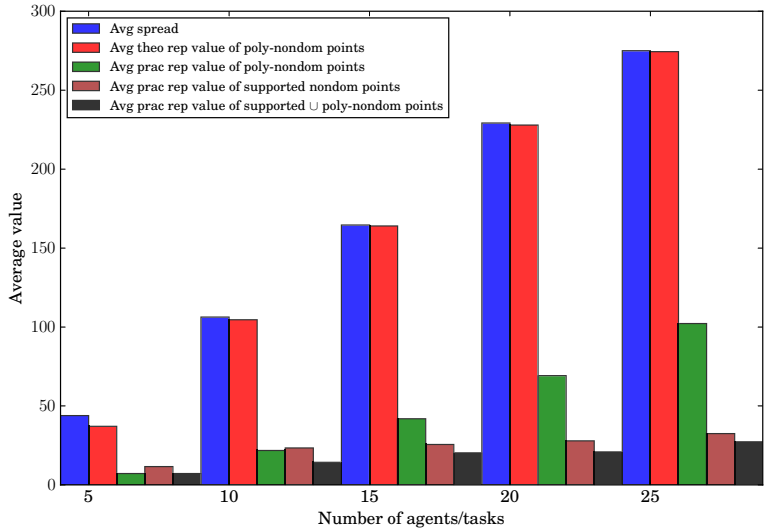

Figure 5.7: Average representation values S, TCP, PCP, PCS, PCSP for the tricriteria assignment problem with $n = 5, 10, \ldots, 25$ agents assigned to n tasks.

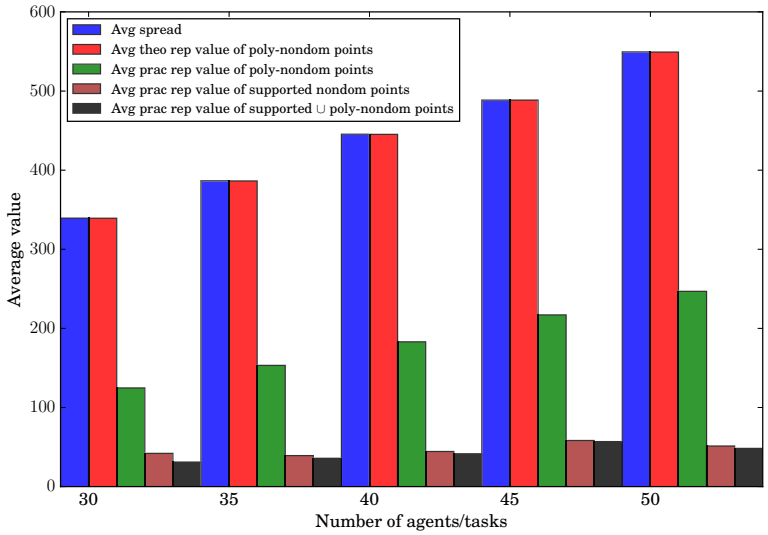

Figure 5.8: Average representation values S, TCP, PCP, PCS, PCSP for the tricriteria assignment problem with $n = 30, 35, \ldots, 50$ agents assigned to n tasks.

5.5.1.3 *Discussion*

For the considered tricriteria assignment problem, disregarding small instances, the average number of poly-nondominated points and the average number of supported nondominated vertices remain relatively small compared to the large overall number of nondominated

points. Furthermore, the average number of poly-nondominated points is generally smaller than the average number of supported nondominated vertices. Taken over all instances the average theoretically given coverage error coincides with the average spread given by the nadir and ideal point. This means that the largest side length of a box given by the nadir point and ideal point coincides with the largest side length of some feasible box given some poly-nondominated points. In other words, regarding the considered instances the theoretically given coverage error which we can compute without knowing $\mathcal{N}$ is a weak bound. For all instance sizes the best average coverage error is given by the set union of poly-nondominated points and the set of nondominated vertices. However, the set of supported nondominated vertices itself is not far off. The set of poly-nondominated points achieves with half as many points as the set of supported nondominated vertices a coverage error which is roughly four times larger than the one given by the set of supported nondominated vertices.

5.5.2 *Tricriteria knapsack problem*

There are 100 tricriteria knapsack problem instances available at [77]. The objective coefficients and weight coefficients are random integers drawn from the interval $[1, 1000]$. The capacity W of the knapsack is given by $W = 0.5 \sum_{i=1}^{n} w_i$ where n is the number of knapsack items. The number of knapsack items n of the generate test instances ranges from $n = 10$ to $n = 100$ with a step size of 10. For each $n \in \{10, 20, \ldots, 100\}$ there are 10 instances. Note that for each n the average value is taken over all 10 instances.

5.5.2.1 *Average number of different nondominated point sets*

Table 5 shows the different average number of points AN, AS, AP, ASP (see page 66 for a definition of the used abbreviations) and different corresponding ratios AS/AN, AP/AN, ASP/AN computed for $n = 10, 20, \ldots, 100$. Figure 5.9 illustrates the average number of points for $n = 10, 20, \ldots, 50$ and Figure 5.10 illustrates the average number of points for $n = 60, 70, \ldots, 100$.

Table 5: Average number of different nondominated point sets for instances of the tricriteria knapsack problem.

n	AN	AS	AS/AN	AP	AP/AN	ASP	ASP/AN
10	9.8	5	0.5102	8.3	0.847	8.3	0.847
20	38.0	14.1	0.371	25.8	0.679	26.9	0.708
30	115.8	26.1	0.225	51.0	0.440	53.9	0.465
40	311.2	35.9	0.115	94.0	0.302	102.1	0.328
50	444.2	48.7	0.11	127.3	0.287	139.2	0.313
60	917.1	69.7	0.076	178.2	0.194	200.0	0.218
70	1643.4	92.4	0.0562	231.7	0.141	268.8	0.164
80	2295.8	114.1	0.05	318.8	0.139	367.4	0.16
90	3107.8	141.8	0.046	351.9	0.113	418.4	0.135

100	5849.0	176.7	0.03	460.8	0.079	550.9	0.094

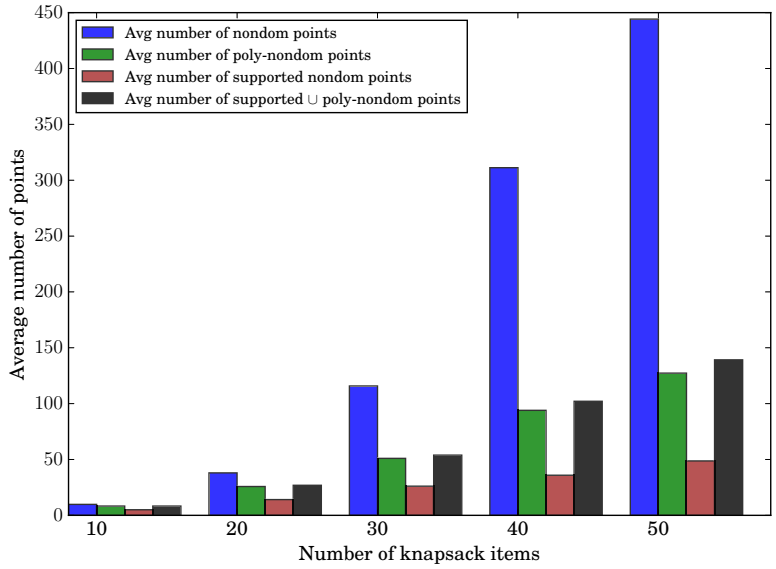

Figure 5.9: Average number of different nondominated point sets of the tricriteria knapsack problem with $n = 10, 20, \ldots, 50$ knapsack items.

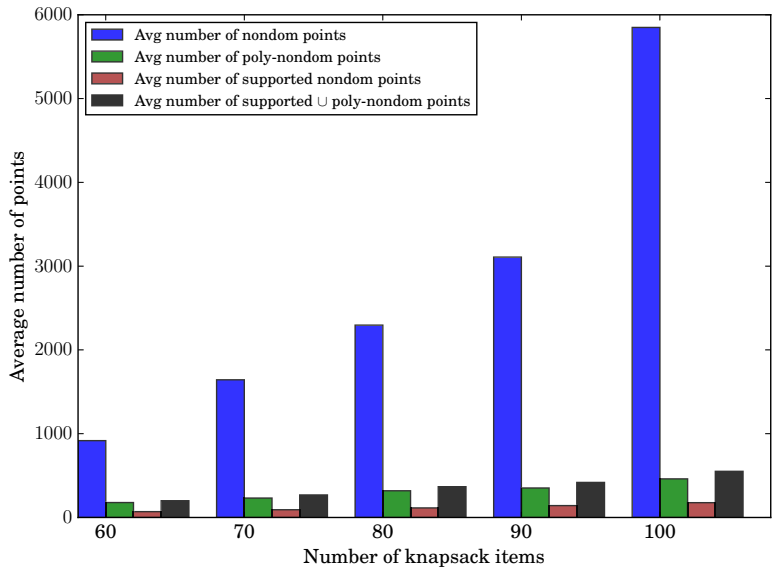

Figure 5.10: Average number of different nondominated point sets of the tricriteria knapsack problem with $n = 60, 70, \ldots, 100$ knapsack items.

5.5.2.2 *Average representation*

Table 6 shows the average values S, TCP, PCP, PCS, PCSP (see page 66 for a definition of the used abbreviations) computed for $n = 10, 20, \ldots, 100$. Figure 5.11 illustrates the average values for $n = 5, 10, \ldots, 25$ and Figure 5.12 illustrates the average values for $n = 30, 35, \ldots, 50$.

Table 6: Average representation values of different nondominated point sets for the tricriteria knapsack problem.

n	S	TCP	PCP	PCS	PCSP
10	1572.8	1189.5	166.0	541.1	166.0
20	2500.5	2104.9	342.3	634.1	327.6
30	4082.9	3674.7	552.4	835.3	505.8
40	4962.0	4649.3	805.9	1060.6	587.6
50	5952.8	5411.4	891.1	1095.1	543.3
60	7543.0	7429.4	1265.7	1248.9	717.2
70	7615.4	7437.8	1617.2	1209.2	699.5
80	9461.1	9154.5	1912.3	1405.6	786.0
90	10315.2	9739.5	2097.9	1249.7	736.2
100	11850.3	11607.8	2721.3	1433.9	847.0

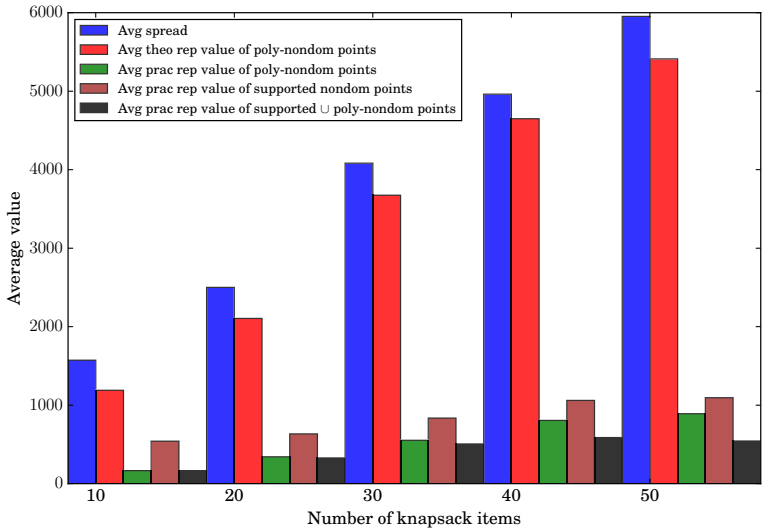

Figure 5.11: Average representation values S, TCP, PCP, PCS, PCSP for the tricriteria knapsack problem with $n = 10, 20, \ldots, 50$ knapsack items.

5.5.2.3 *Discussion*

For the considered tricriteria knapsack problem, disregarding small instances, the average number of poly-nondominated points and the average number of supported nondominated vertices remain relatively small compared to the large overall number of nondominated points. Moreover, in contrast to the considered assignment problem instances, the number of supported nondominated vertices is generally less than half the number of poly-nondominated points. Taken over all instances the average theoretically given coverage error is far from the practically given one, but strictly smaller than the average spread between the ideal point and nadir point. This was not the case for the considered assignment problem instances. For all instance sizes the best average coverage error is given by the set union of poly-

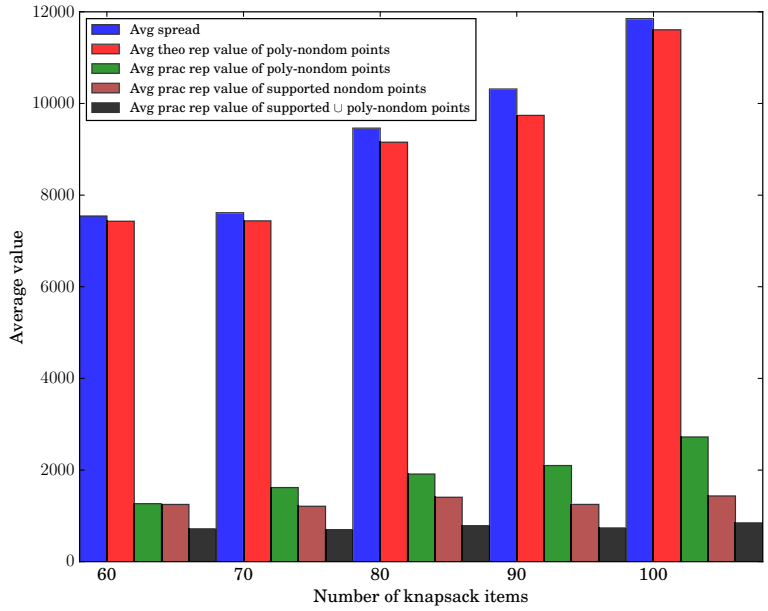

Figure 5.12: Average representation values S, TCP, PCP, PCS, PCSP for the tricriteria knapsack problem with $n = 60, 70, \ldots, 100$ knapsack items.

nondominated points and the set of nondominated vertices. Interestingly, for the larger instances the set of supported nondominated vertices achieves a better representation than the set of poly-nondominated points although the latter comprises more points. Considering the representation quality of the set of poly-nondominated points, it achieves a better representation than the set of supported nondominated vertices for instance sizes up to $n = 50$.

5.6 SUMMARY AND DISCUSSION

This chapter focussed on the idea of considering the set of supported nondominated points $\mathcal{N}_s$ and the set of poly-nondominated points $\mathcal{N}_p$, respectively, as an ϵ-representation for the entire set of nondominated points $\mathcal{N}$. Regarding the set of poly-nondominated points $\mathcal{N}_p$, we showed that a maximal coverage error can always be computed without knowing $\mathcal{N}$ entirely. We computed the average number of points and the average coverage error for instances of the tricriteria assignment problem and instances of the tricriteria knapsack problem. We showed that the set union of $\mathcal{N}_p$ and $\mathcal{N}_s$ achieves a better coverage error than $\mathcal{N}_p$ or $\mathcal{N}_s$ individually. Moreover, as the latter two sets are generally not disjoint taking the union results in a relatively slight increase in the overall number of points.

In the bicriteria case the set of supported nondominated points $\mathcal{N}_s$ is often used as an approximation for the entire set of nondominated points. Our computational experiments showed that also in the tricriteria case the set of supported nondominated points $\mathcal{N}_s$ can achieve a convenient representation quality. However, it does generally not

enable the computation of the spread between the ideal point and the nadir point or a maximum coverage error without knowing $\mathcal{N}$. In contrast, the set of poly-nondominated points enables the computation of the nadir point (see Theorem 4.11) and, thus, the spread as well as a maximum coverage error can be computed without knowing the entire set of nondominated points $\mathcal{N}$. By taking the set union of $\mathcal{N}_s$ and $\mathcal{N}_p$ we can achieve a more *robust* representation of the entire set of nondominated points $\mathcal{N}$ as $\mathcal{N}_s$ and $\mathcal{N}_p$ can be considered to represent different *traits* (see Definition 2.7 and Definition 4.2). As seen in Section 5.5.2, the set of poly-nondominated points $\mathcal{N}_p$ achieves a smaller coverage error for smaller instance sizes whereas the set of supported nondominated points $\mathcal{N}_s$ achieves a smaller coverage error for larger instance sizes. Hence, by considering the set union of $\mathcal{N}_p$ and $\mathcal{N}_s$ we can merge the respective advantages of both representations.

6 POLYSCIP

> On two occasions, I have been asked, "Pray, Mr. Babbage, if you put into the machine wrong figures, will the right answers come out?"
>
> *Charles Babbage*

6.1 INTRODUCTION

In this chapter we present PolySCIP, a solver for computing nondominated vertices of general multicriteria linear programs and nondominated points for {bi,tri}criteria integer programs. The development of PolySCIP was motivated by the need to solve multicriteria optimisation problems in sustainable manufacturing (see Chapters 7 and 8). Note that solving problems of this kind is not supported by commercial solvers. The work presented in this chapter is joint work with Ralf Borndörfer, Martin Skutella and Timo Strunk [17].

6.2 HIGH-LEVEL OVERVIEW

The name PolySCIP is composed of the Greek word $\pi o \lambda \acute{\upsilon} \varsigma$ which stands for *many* and SCIP [1], a widely used noncommercial constraint integer programming framework. The first version of PolySCIP was released in February 2016 as an official part of SCIP 3.2.1. In March 2017 a revised version, PolySCIP 2.0, was released as an official part of SCIP 4.0.
PolySCIP can be considered as a framework on top of SCIP, see Figure 6.1. Given a multicriteria optimisation problem, PolySCIP computes the set of nondominated points by consecutively transforming the given multicriteria optimisation problem into an appropriate single objective problem which is given to the single objective solver SCIP. The corresponding outcome of SCIP is processed and incorporated by PolySCIP. Depending on whether the outcome corresponds to a new nondominated point and whether the feasible domain still contains unexplored regions, the processing step adds new constraints to the feasible domain and yields a new single objective problem. When the search for nondominated points is exhausted, the set of nondominated points is returned to the user. Depending on the user input, i.e. depending on whether a multicriteria linear program or a multicriteria integer program is to be solved, the internal solution approach differs. For a multicriteria LP the implementation is based on the theory presented in Chapter 3. In this case the transformed single objective problem corresponds to a weighted

sum objective problem (see (2.4)) and the search for unexplored nondominated points is facilitated by using the weight space polyhedron method (see Section 3.5.3). For a multicriteria IP the implementation is based on the theory presented in Chapter 4. In this case the single objective problem corresponds to a transformed problem with weighted Chebyshev norm and additional constraints (see (2.6)). For tricriteria integer problems the set of poly-nondominated points is computed in a first phase. Then, based on the feasible boxes (see Definition 4.17) given by the set of poly-nondominated points, the set of mono-nondominated points is computed in a second phase.

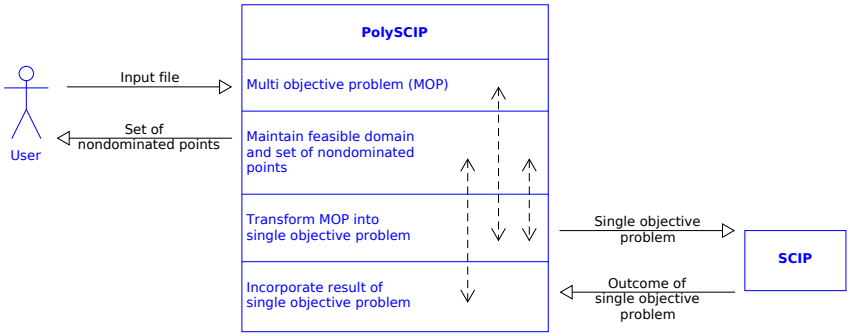

Figure 6.1: High-level overview of PolySCIP.

6.3 INPUT FILE FORMAT

Unlike the LP file format [51] or the MPS file format [54] for single objective problems, there is no widely recognised input format for multicriteria optimisation problems. The file format used in PolySCIP is based on the MPS format. MPS is column-oriented and all model components, i.e. variables, constraints, objectives, receive a name. An objective in MPS is indicated in the ROWS section via an N followed by the name of the objective. Equivalently, for the multicriteria case we indicate any objective via an N followed by the corresponding name, see Example 6.1. The two main reasons to base the file format of PolySCIP on MPS are the above mentioned simple extension towards several objectives and the wide availability of MPS parsers in other integer programming software packages. The latter might facilitate an extension of some of these software packages to accept a similar multicriteria file format based on MPS in order to reach the goal of a widely used file format for multicriteria optimisation problems.

Example 6.1. The following bicriteria integer program

$$
\begin{array}{lll}
\text{minimize} & \texttt{Obj1:} & 3x_1 + 2x_2 - 4x_3 \\
& \texttt{Obj2:} & x_1 + x_2 + 2x_3 \\
\text{subject to} & & \\
& \texttt{Eqn:} & x_1 + x_2 + x_3 = 2 \\
& \texttt{Lower:} & x_1 + 0.4x_2 \leq 1.5 \\
& & x_1,\ x_2,\ x_3 \geq 0 \\
& & x_1,\ x_2,\ x_3 \in \mathbb{Z}
\end{array}
$$

can be written in the MPS format as

```
NAME          BICRIT
OBJSENSE
 MIN
ROWS
 N  Obj1
 N  Obj2
 E  Eqn
 L  Lower
COLUMNS
    x#1       Lower             1
    x#1       Eqn               1
    x#1       Obj2              1
    x#1       Obj1              3
    x#2       Lower           0.4
    x#2       Eqn               1
    x#2       Obj2              1
    x#2       Obj1              2
    x#3       Eqn               1
    x#3       Obj2              2
    x#3       Obj1             -4
RHS
    RHS       Eqn               2
    RHS       Lower           1.5
BOUNDS
 LI BOUND     x#1               0
 LI BOUND     x#2               0
 LI BOUND     x#3               0
ENDATA
```

6.4 PROBLEM FILE GENERATION

Regardless of the solver, an important aspect for any user is the ability to easily generate input files for different problems and data. Having to write MPS files by hand would be time-consuming and error-prone. ZIMPL [48] is a modelling language that can be used to translate a mathematical model of an optimisation problem into a mathematical program in MPS format. POLYSCIP comes with a script (named

`mult_zimpl_to_mop.py`) that takes an *extended* ZIMPL file containing several objectives (see Examples 6.2 and 6.3) and turns it into a multicriteria `MPS` file that can be read by POLYSCIP. The script rewrites all but the first objective into constraints, executes ZIMPL on the rewritten file and changes all constraints corresponding to original objectives back into objectives in the generated `MPS` file. In this way we can effectively use the capabilities of ZIMPL in order to easily translate a mathematical model into a multicriteria input file.

Example 6.2. The bicriteria minimisation problem of Example 6.1 expressed in ZIMPL syntax:

```
set I := {1..3};
param obj1[I] := <1> 3, <2> 2, <3> -4;
param obj2[I] := <3> 2 default 1;
param low[I] := <1> 1, <2> 0.4, <3> 0;
var x[I] integer >= 0;

minimize Obj1: sum <i> in I: obj1[i]*x[i];
Obj2: sum <i> in I: obj2[i]*x[i];

subto Eqn: sum <i> in I: x[i] == 2;
subto Lower: sum <i> in I: low[i]*x[i] <= 1.5;
```

Example 6.3. Generating the input file for a tricriteria assignment problem given via the following data:

```
3
5
6, 1, 20, 2, 3,
2, 6, 9, 10, 18,
1, 6, 20, 5, 9,
6, 8, 6, 9, 6,
7, 10, 10, 6, 2
,
17, 20, 8, 8, 20,
10, 13, 1, 10, 15,
4, 11, 1, 13, 1,
19, 13, 7, 18, 17,
15, 3, 5, 1, 11
,
10, 7, 1, 19, 12,
2, 15, 12, 10, 3,
11, 20, 16, 12, 9,
10, 15, 20, 11, 7,
1, 9, 20, 7, 6
```

The first line specifies the number of objectives. The second line specifies the number of variables and the following three 5×5 matrices contain the coefficients of the three objectives. Provided that the above data is stored in a file named `my_data.txt` the tricriteria assignment problem can be written in ZIMPL syntax as follows:

```
param prob_file := "my_data.txt";
param no_objs := read prob_file as "1n" use 1;
param no_vars := read prob_file as "1n" use 1 skip 1;

set I := {1..no_vars};
set T := {1..no_objs*no_vars*no_vars};
param coeffs[T] := read prob_file as "n+" match "[0-
9]+" skip 2;
param offset := no_vars*no_vars;
param obj1[<i,j> in I*I] := coeffs[(i-1)*no_vars + j];
param obj2[<i,j> in I*I] := coeffs[(i-1)*no_vars + j + offset];
param obj3[<i,j> in I*I] := coeffs[(i-1)*no_vars + j + 2*offset];

var x[I*I] binary;

minimize Obj1: sum <i,j> in I*I: obj1[i,j]*x[i,j];
Obj2: sum <i,j> in I*I: obj2[i,j]*x[i,j];
Obj3: sum <i,j> in I*I: obj3[i,j]*x[i,j];

subto row: forall <i> in I do
      sum <j> in I: x[i,j] == 1;
subto col: forall <i> in I do
      sum <j> in I: x[j,i] == 1;
```

6.5 DISCUSSION

In the following Sections 6.5.1-6.5.4 we discuss some insights we learned by implementing and by using PolySCIP.

6.5.1 *Solution querying*

Integer programming solvers generally allow a user to explore an optimal solution by displaying the values of the solution variables. In the multicriteria case, however, exploring values of a particular solution might not be sufficient for the final decision process. The capability to effectively compare solution values of different (efficient) solutions will be an important aspect, too. For instance, let us consider a bicriteria production problem $(c_1, c_2, \mathcal{X})$ whose underlying solution variables partly correspond to the acquisition of new machinery. Assume we have computed the corresponding set of nondominated points $\{y^1, \dots, y^5\}$ depicted in Figure 6.2. A decision maker who considers only the objective values might be likely to choose the solution corresponding to y^3 as this is the most *balanced* nondominated point. However, it might turn out that the preimage of y^3 corresponds to the acquisition of unique machinery that is not shared by any solution corresponding to the other nondominated points y^1, y^2, y^4, y^5. Moreover, it might be the case that the solutions corresponding to y^2 and y^4 would share most of the same machinery. In other words, by choosing y^2 and by additionally investing into

the (not overly expensive) machinery of y^4 that is not shared with y^2 a decision maker might be able to better adapt to different market conditions by being able to realise both scenarios without much overhead. In contrast, by choosing y^3 this would not be possible. In this regard, a decision maker might not just be interested in the solution values corresponding to particular nondominated points but in the possibility to reoptimise and to find similarities between different efficient solutions. As the set of nondominated points is generally large, this is not easily doable if the solver only offers to display values of a particular solution as currently most single objective solvers do.

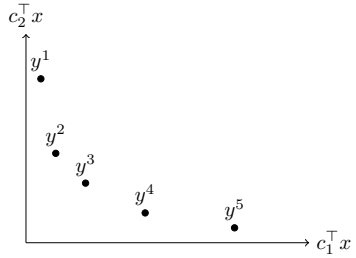

Figure 6.2: Simple front of nondominated points $y^1, \ldots, y^5$ for a bicriteria integer problem.

6.5.2 *Availability of the entire set of efficient solutions*

As already indicated in Section 2.3, for a multicriteria optimisation problem it is not obvious what a solver is supposed to compute as an outcome. Should the solver compute the entire set of efficient solutions or the entire set of nondominated points with a single preimage for each nondominated point? From a computational perspective it seems reasonable to avoid the computation of different efficient solutions that correspond to the same nondominated point. However, as the solution values correspond to actual *decisions* and a decision maker might be interested in similarities to other solutions in the context of real-world applications (as discussed in Section 6.5.1), it seems useful to offer the possibility to compute the entire set of efficient solutions in multicriteria solvers.

6.5.3 *Numerical stability*

In general, the number of subproblems that need to be solved in order to get the desired set of nondominated points is considered as the most important factor for the performance of a multicriteria solver. However, a generated subproblem used to compute a particular nondominated point might confront the solver with coefficients that are difficult to handle numerically. For instance, transformation (2.6) based on Bowman's theorem (see Theorem 2.11) might become difficult to solve due to numerical issues resulting from certain weights. In this regard, the question whether a lower number of subproblems which are numerically instable should be considered *superior* to a lar-

ger number of subproblems which are numerically stable does not offer an easy and general answer. However, it indicates that focussing solely on the number of subproblems solved by a multicriteria solver as the main performance factor might be disputable.

6.5.4 *Parameter tuning for subproblems*

Many single objective integer program solvers allow the user to control and fine-tune the solution process via a wide range of parameters, e.g. there are, in general, parameters affecting applied heuristics, used cutting planes, presolving et cetera. Given a large-scale problem, the right mixture of parameters can enable a solver to compute the optimum (within a certain time horizon) where default settings do not. A multicriteria solver like PolySCIP needs to solve many subproblems in order to compute the set of nondominated points. Using the same parameter settings for different subproblems might not guarantee to compute the optimum for each subproblem. For instance, it might be the case that different subproblems given by a multicriteria instance need different parameter settings, e.g. different frequencies of calling a certain cut separator, in order to be solved optimally (within a certain time horizon). In this regard, being able to control the *local* solution process and access detailed information for a subproblem generated in the course of the *global* solution process might be a useful feature for a multicriteria solver.

6.6 SUMMARY

In this chapter we presented PolySCIP, a solver for multicriteria linear optimisation problems. It can be used to compute the entire set of nondominated vertices for multicriteria linear programs and the entire set of nondominated points for {bi,tri}criteria integer programs. As an official part of SCIP (the source code of) PolySCIP is freely available for academic purposes. We consider the achievement to make PolySCIP an official part of SCIP as an important aspect to keep PolySCIP long-term available to interested users and researchers. Moreover, PolySCIP can be applied to a wide variety of problems by using a freely available algebraic modelling language to generate multicriteria input files.

7

DECISION SUPPORT IN SUSTAINABLE MANUFACTURING

> Nothing compares to the simple pleasures of riding a bike.
>
> *John F. Kennedy*

7.1 INTRODUCTION

Sustainability for humankind can be seen as the ability to meet its needs without a disruption to nature and society. A long-term goal of sustainability is to abandon a system where only a small proportion of the world's population benefits from the global resource and where access to social support, knowledge and well-being is granted to the entire world's population. Taking into account that sustainability comprises environmental, economic and social aspects which generally conflict each other, we can consider sustainability problems as multicriteria decision problems, see Figure 7.1. Sustainability manufacturing is the translation of sustainability into modern manufacturing approaches. For instance, a environmental goal for sustainability manufacturing is that once non-renewable raw materials have been transformed into products, they may not be discarded but will have to be regained in product and material cycles. An alternative to reduce the use of non-renewable raw materials is to substitute them with renewable raw materials. This substitution could be carried out as long as it does not exceed the renewal rate. For more information about sustainability in modern manufacturing, we refer the reader to [72].

Figure 7.1: The environmental, economic and social dimension interpreted as different objectives.

Sustainability assessment is a field which is still under development and frameworks for assessing sustainability of products and processes still struggle with a lack of data, best practices or well-established indicator sets [56].

By interpreting conflicting aspects of sustainability as different conflicting objectives that need to be optimised simultaneously, we can approach problems of sustainable manufacturing as multicriteria optimisation problems, see Figure 7.2. This approach does not solve above mentioned data problems, but it offers the possibility to apply mathematical optimisation tools to quantitative sustainable manufacturing problems. Furthermore, by interpreting dominated points as unsustainable and nondominated points as sustainable candidates, respectively, we are able to compute valuable information for a decision maker without the risk of being too conservative or biased towards certain outcomes.

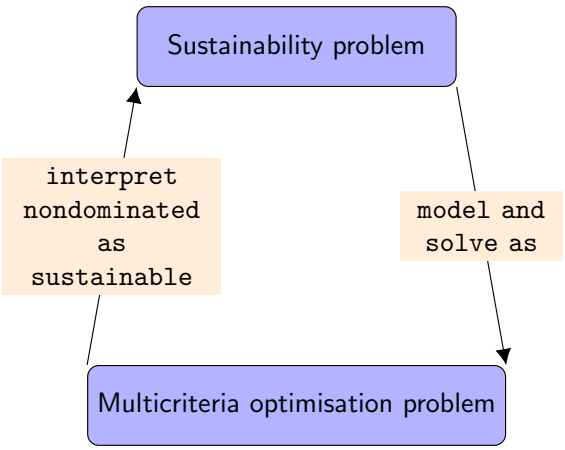

Figure 7.2: Solving sustainability manufacturing problems as multicriteria optimisation problems.

The main goal of this chapter is to show the applicability and usefulness of multicriteria optimisation for the decision making process for problems in sustainability manufacturing. In the following, we consider a bicycle frame manufacturing problem where different available materials and different manufacturing procedures are modelled via integer programming. Based on the available data, two objectives are to be optimised simultaneously. The presented results are based on [70] which is joint work with Jón Garðar Steingrímsson, Ralf Borndörfer and Günther Seliger. The nondominated fronts presented in Section 7.2.4 were computed by PolySCIP.

7.2 MODELLING BICYCLE FRAME MANUFACTURING VIA BICRITERIA INTEGER PROGRAMMING

7.2.1 *Overview*

For a product like a common bicycle, a holistic view on the different life cycle phases has to be taken into account in order to ensure that resources are utilised in a sustainable manner. Different preferences on the economic, environmental or social dimension might lead to different selections of materials, used equipment or required staff

training. The life cycle of a product usually starts with the idea of how the product should look like, the product design. The realisation of a product traditionally begins with raw material extraction, raw material processing, manufacturing, usage. At the point in time when the product is obsolete, an end-of-life strategy has to be exploited in order to reclaim the materials in the obsolete product. Attempts to manufacture in a sustainable manner require focusing equally on the three dimensions of sustainability.

7.2.2 *Modern bicycle manufacturing*

Equipment and tools are required for manufacturing procedures. They come in various shapes, sizes and capabilities depending on the process requirements. For instance, automated stamp presses, CNC machine centres, automated laser cutting machines or automated coating systems provide high manufacturing output, high level of quality and high level of precision. However, this kind of equipment tends to be capital intensive. Furthermore, automated equipment usually requires highly qualified workers due to the complexity of operating the equipment. Small handheld tools, e.g. files, hammers, hand drills and hand saws, are labour intensive. They do not yield the same output, lower level of quality and lower level of precision but often have little requirements on qualification. The common bicycle contains around 200 parts that are globally produced by different types of manufacturers. Subassemblies are jointed from individual parts and assemblies are jointed from subassemblies, see Figure 7.3. To a large extent, the supply chain of a bicycle is push driven, i.e. component groups such as the rear wheel assembly, pedal set, front wheel assembly and break set are manufactured based on forecast in high volumes. The frame is the main assembly of the bicycle acting as the main branding association force for the whole assembly. It can be made from various materials, e.g. steel, stainless steel, aluminium alloys, titanium, wood and metal inserts, or bamboo and metal inserts composite. The manufacturing of the bicycle frame ranges from bespoke single made items to items that are carried out under preconditions of mass manufacturing, i.e. single design is manufactured in thousands of units annually. The front set contains a fork set, a head set, a handle bar and a grip. Frames are often handmade independently of the fork, i.e. a final assembly of a bicycle could have a custom made frame but standard assemblies [82].

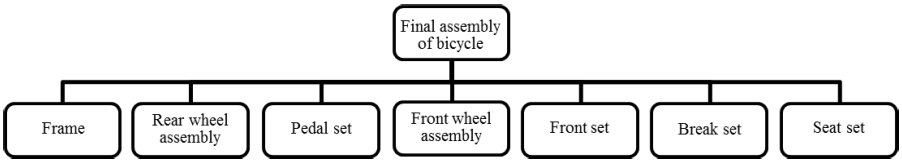

Figure 7.3: Bill of materials for a bicycle assembly.

7.2.3 *Available data and mathematical model*

Our model takes a middle-sized bicycle factory into account which aims at manufacturing a number of bicycle frames in the range of a few hundred to a few thousand. The manufacturing model comprises two alternatives in raw material (bamboo or aluminium) and for procedures regarding aluminium frames a *binary* degree of automation (manually executed or automatically executed). The transformation from a *raw* frame into a *sellable* frame is given by an ordered sequence of different manufacturing procedures, see Figure 7.4. Note that the sequences for bamboo and aluminium do not overlap even if some procedures are labelled identically since different tools are needed for the respective material yielding different costs, scrap rates, et cetera. The procedures applied to aluminium frames might be performed in an automatic manner via machines or in a mechanised/manual fashion yielding two alternatives for each procedure (and indicated by two arrows between consecutive procedures in Figure 7.4). Based on the available data, the procedures applied to bamboo frames are considered to be executed in a manual way only. Depending on the material and the degree of automation, each manufacturing procedure induces a scrap rate representing the amount of waste (in percentage) that is incurred by executing it. Data on required processes, e.g. processing time, equipment cost, was gathered through several workshops collected by students and experts in the bicycle frame building industry. On-the-job data was collected for the construction of the two mechanised alternatives.

Let $P_A = \{\text{cutting, bending, mitering, deburring, pre-cleaning, fixing, welding, straightening, miling, deburring, cleaning, coating}\}$ be the multiset of procedures used for aluminium frames and let $P_B = \{\text{cutting, filing, mitering, deburring, pre-cleaning, fixing, adhesion, deburring, grinding, sanding, cleaning, coating}\}$ be the multiset of procedures used for bamboo frames, see Figure 7.4. Let $I_A = \{1, \ldots, |P_A|\}$ and $I_B = \{1, \ldots, |P_B|\}$, respectively, be two index sets corresponding to the above multisets. Since procedures need to be executed in a chronological order, we identify a procedure in P_A with a corresponding index in I_A, e.g. index $1 \in I_A$ corresponds to cutting, index $2 \in I_A$ corresponds to bending, et cetera. Equivalently, we identify a procedure in P_B with a corresponding index in I_B. We introduce nonnegative integer variables x_i^a, x_i^m for $i \in I_A$ and y_i^m for $i \in I_B$. x_i^a for $i \in I_A$ represents the number of aluminium frames that are treated in an automatic manner with regard to procedure i. x_i^m for $i \in I_A$ represents the number of aluminium frames that are treated in a manual way with regard to procedure i. Moreover, y_i^m for $i \in I_B$ represents the number of bamboo frames that are treated in a manual way with regard to procedure i. To adjust to the granularity of the available data for the objective coefficients, we also introduce nonnegative integer variables x_i^{a100}, x_i^{m100} for $i \in I_A$ and y_i^{m100} for $i \in I_B$. The values of x_i^{a100}, x_i^{m100} for $i \in I_A$ and y_i^{m100} for $i \in I_B$, respectively, will be multiples of hundred depending on the values of x_i^a, x_i^m and y_i^m, respectively.

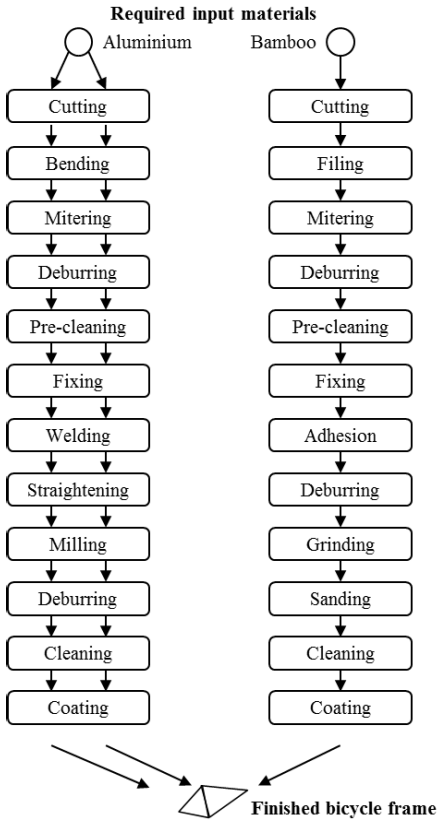

Figure 7.4: Procedural diagram for the manufacturing of bicycle frames.

As mentioned above, each manufacturing procedure induces a scrap rate depending on the material and the degree of automation. Let $\text{scrap}_i^{\text{alu}_a} \in \mathbb{R}_+$ for $i \in I_A$ be the given scrap rate for an automatically executed procedure i applied to aluminium. Let $\text{scrap}_i^{\text{alu}_m} \in \mathbb{R}_+$ for $i \in I_A$ be the given scrap rate for a manually executed procedure i applied to an aluminium frame and let $\text{scrap}_i^{\text{bam}_m} \in \mathbb{R}_+$ for $i \in I_B$ be the given scrap rate for a manually executed procedure i with regard to a bamboo frame. Then, we consider the following *yield* constraints:

$$y_i^m \leq (1 - \text{scrap}_{i-1}^{\text{bam}_m}) y_{i-1}^m \quad \text{for } i \in I_B \setminus \{1\} \tag{7.1}$$

$$x_i^a + x_i^m \leq (1 - \text{scrap}_{i-1}^{\text{alu}_a}) x_{i-1}^a + (1 - \text{scrap}_{i-1}^{\text{alu}_m}) x_{i-1}^m \text{ for } \; i \in I_A \setminus \{1\} \tag{7.2}$$

(7.1) states that the number of bamboo frames that will be executed by procedure i must be less than or equal to the number of bamboo frames that were executed by procedure $i-1$. Similarly, (7.2) states that the number of aluminium frames that will be executed by procedure i must be less than or equal to the number of aluminium frames that were executed by procedure $i-1$.

Provided that we are given a parameter $n \in \mathbb{N}_+$ representing the number of frames that need to be manufactured, we consider the *outcome* constraint:

$$(1 - \text{scrap}^{\text{bam}_m}_{|I_B|})y^m_{|I_B|} + (1 - \text{scrap}^{\text{alu}_a}_{|I_A|})x^a_{|I_A|} + (1 - \text{scrap}^{\text{alu}_m}_{|I_A|})x^m_{|I_A|} \geq n \tag{7.3}$$

Note that (7.3) does not strictly enforce that the final number of frames (i.e. bamboo frames plus aluminium frames after executing the last procedure) is at least n since the left hand side of (7.3) is given by a sum of fractional numbers. To be strict we needed to make sure that, by taking the floor of each addend, the sum of the integral values of each addend is at least n. However, we will work with the relaxation (7.3) and assume that a final outcome of either $n-2$, $n-1$ or n frames is adequate for our considered scenario provided that n is sufficiently large.

For each combination of manufacturing procedure, execution method, and material alternative, we are given a maximal production capacity. Let $\text{cap}^{\text{alu}_a}_i$ for $i \in I_A$ be the maximal production capacity of a machine which automatically executes procedure i with regard to aluminium frames. Let $\text{cap}^{\text{alu}_m}_i \in I_A$ be the maximal production capacity of a worker who manually executes procedure i with regard to aluminium frames. Let $\text{cap}^{\text{bam}_m}_i \in I_B$ be the maximal production capacity of a worker who manually executes procedure i with regard to bamboo frames. In relation to the maximal production capacities we introduce positive integer variables u^a_i, u^m_i for $i \in I_A$ and v^m_i for $i \in I_B$ which represent the number of machines and workers, respectively, which are needed to accomplish the manufacturing task. For instance, if the number of frames that are to be automatically executed with regard to procedure i is greater than the maximal production capacity of the respective machine, then an investment in a second machine (or additional workers) is needed. Similarly, a manual worker is able to execute a procedure a certain number of times. If the number of frames that need to be handled is greater, then additional workers (or an additional machine) is needed. In other words, the following *capacity* constraints express that for each procedure, depending on the material and degree of automation, the number of handled frames must not exceed the corresponding production capacity.

$$\text{cap}^{\text{alu}_a}_i u^a_i \geq x^a_i \qquad \text{for } i \in I_A \tag{7.4}$$

$$\text{cap}^{\text{alu}_m}_i u^m_i \geq x^m_i \qquad \text{for } i \in I_A \tag{7.5}$$

$$\text{cap}^{\text{bam}_m}_i v^m_i \geq y^m_i \qquad \text{for } i \in I_B \tag{7.6}$$

To stick to the scenario of a middle-sized bicycle manufacturing company having a restricted investment budget, we will bound to number of available machines, which generally incur large investment costs, by a given upper bound $m \in \mathbb{N}$.

$$\sum_{i \in I_A} u^a_i \leq m \tag{7.7}$$

The first objective will be given by the overall manufacturing cost which accumulates fixed costs (including initial investment costs for acquiring machines and tools), consumables costs and labour costs. Let $\text{fix}_i^{\text{alu}_a}$, $\text{fix}_i^{\text{alu}_m}$ for $i \in I_A$ and $\text{fix}_i^{\text{bam}_m}$ for $i \in I_B$ be given fixed costs. $\text{fix}_i^{\text{alu}_a}$ for $i \in I_A$ represents the fixed costs caused by a machine and worker to automatically execute procedure i on an aluminium frame. $\text{fix}_i^{\text{alu}_m}$ for $i \in I_A$ represents the fixed costs caused by a mechanized tool and worker to manually execute procedure i on an aluminium frame. Equivalently, $\text{fix}_i^{\text{bam}_m}$ for $i \in I_B$ represents the fixed costs caused by a mechanized tool and worker to manually execute procedure i with regard to bamboo frames. Let $\text{lab}_i^{\text{alu}_a}$, $\text{con}_i^{\text{alu}_a}$ for $i \in I_A$ and $\text{lab}_i^{\text{bam}_m}$, $\text{con}_i^{\text{bam}_m}$ for $i \in I_B$ be given labour costs and consumables costs, respectively. $\text{lab}_i^{\text{alu}_a}$ for $i \in I_A$ represents the labour costs of an automatically exectued procedure i applied to 100 aluminium frames. Similarly, $\text{con}_i^{\text{alu}_a}$ for $i \in I_A$ represents the consumables costs of an automatically executed procedure i applied to 100 aluminium frames. Similarly, $\text{lab}_i^{\text{alu}_m}$, $\text{con}_i^{\text{alu}_m}$ for $i \in I_A$ and $\text{lab}_i^{\text{bam}_m}$, $\text{con}_i^{\text{bam}_m}$ for $i \in I_B$ represent the labour costs and consumables costs of a manually executed procedure i applied to an aluminium frame and bamboo frame, respectively.

The first objective representing overall manufacturing costs is then given by:

$$\begin{aligned}
&\sum_{i \in I_A} ((\text{lab}_i^{\text{alu}_a} + \text{con}_i^{\text{alu}_a}) x_i^{a100} + \text{fix}_i^{\text{alu}_a} u_i^a) + \\
&\sum_{i \in I_A} ((\text{lab}_i^{\text{alu}_m} + \text{con}_i^{\text{alu}_m}) x_i^{m100} + \text{fix}_i^{\text{alu}_m} u_i^m) + \\
&\sum_{i \in I_B} ((\text{lab}_i^{\text{bam}_m} + \text{con}_i^{\text{bam}_m}) y_i^{m100} + \text{fix}_i^{\text{bam}_m} v_i^m)
\end{aligned} \tag{7.8}$$

Note that fixed costs incur per unit of machinery and worker whereas consumables costs and labour costs incur per 100 units of frames.

The second objective will represent the overall processing time. Let $\text{time}_i^{\text{alu}_a}$, $\text{time}_i^{\text{alu}_m}$ for $i \in I_A$ and $\text{time}_i^{\text{bam}_m}$ for $i \in I_B$ be given processing times. $\text{time}_i^{\text{alu}_a}$ for $i \in I_A$ represents the processing time of an automatically executed procedure i applied to an aluminium frame. $\text{time}_i^{\text{alu}_m}$ for $i \in I_A$ represents the processing time of a manually executed procedure i applied to an aluminium frame and $\text{time}_i^{\text{bam}_m}$ for $i \in I_B$ represents the processing time of a manually executed procedure i applied to a bamboo frame.

The second objective is then given by:

$$\sum_{i \in I_A} (\text{time}_i^{\text{alu}_a} x_i^a + \text{time}_i^{\text{alu}_m} x_i^m) + \sum_{i \in I_B} \text{time}_i^{\text{bam}_m} y_i^m \tag{7.9}$$

Regarding the objectives it would have been desirable to take the social or environmental dimension of sustainability more into account. However, we decided against creating randomized or fake data and keep the considered objectives close to the available real data.

7.2.4 *Computational results*

We will denote the previously described bicriteria frame manufacturing problem

$$\begin{aligned} &\min (7.8), (7.9) \\ &\text{s.t. } (7.1) - (7.7), \\ 100y_i^{m100} &\geq y_i^m \quad \text{for } i \in I_B, \\ 100x_i^{a100} &\geq x_i^a \quad \text{for } i \in I_A, \\ 100x_i^{m100} &\geq x_i^m \quad \text{for } i \in I_A, \\ x_i^a, x_i^{a100}, x_i^m, x_i^{m100}, u_i^a, u_i^m &\in \mathbb{N} \quad \text{for } i \in I_A, \\ y_i^m, y^{m100}, v_i^m &\in \mathbb{N} \quad \text{for } i \in I_B, \end{aligned}$$

by $Bike(m, n)$ where $m \in \mathbb{N}$ used in (7.7) is a given upper bound on the number of automatically executed procedures and $n \in \mathbb{N}$ used in (7.3) is the (approximate) number of frames that are to be manufactured.

For what follows we consider the manufacturing of $n = 1000$ bicycle frames and a possible usage of $m = 1, 2$ automatically executed procedures.

7.2.4.1 *Computational results for Bike(1,1000)*

The bicriteria integer program $\text{Bike}(1, 1000)$ results in three supported nondominated points y^6, y^7, y^2 and seven unsupported nondominated points $y^0, y^1, y^3, y^4, y^5, y^8, y^9$, see Figure 7.5.

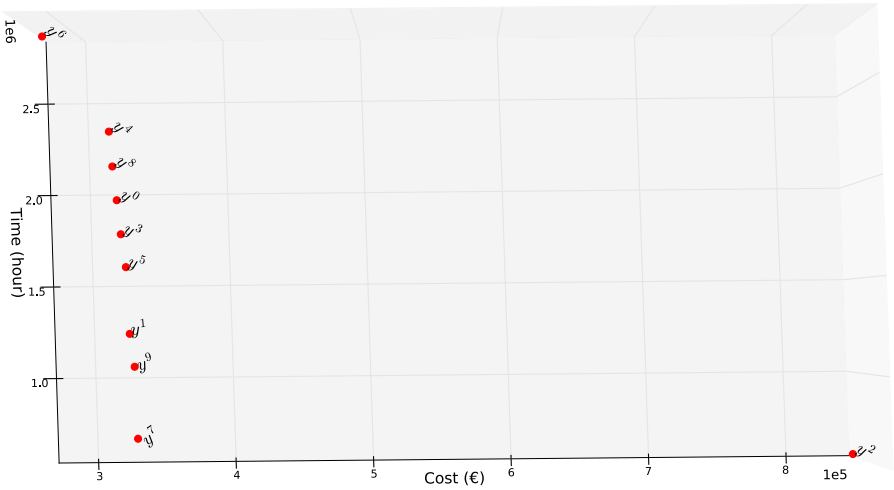

Figure 7.5: Computed nondominated points of $\text{Bike}(1, 1000)$.

The supported nondominated point $y^6 = (282710, 2.793e + 06)$ corresponds to the efficient solution $x^6 = (x^{6^a}_1, \ldots, x^{6^a}_{12}, x^{6^m}_1, \ldots, x^{6^m}_{12}, u^{6^a}_1, \ldots, u^{6^a}_{12}, u^{6^m}_1, \ldots, u^{6^m}_{12}, y^{6^m}_1, \ldots, y^{6^m}_{12}, v^{6^m}_1, \ldots, v^{6^m}_{12})$ with nonzero solution values $y_{12}^{m100} = 11$, $y_{11}^{m100} = 11$, $y_{10}^{m100} = 11$, $y_9^{m100} = 11$, $y_8^{m100} = 12$, $y_7^{m100} = 12$, $y_6^{m100} = 12$, $y_5^{m100} = 12$, $y_4^{m100} = 12$, $y_3^{m100} = 13$, $y_2^{m100} = 13$, $y_1^{m100} = 14$, $v_{12}^m = 1$, $v_{11}^m = 1$, $v_{10}^m = 1$, $v_9^m = 1$, $v_8^m = 1$, $v_7^m = 17$, $v_6^m = 1$, $v_5^m = 1$, $v_4^m = 1, v_3^m = 1$, $v_2^m = 1$, $v_1^m =$

1, $y^m_{12} = 1021$, $y^m_{11} = 1023$, $y^m_{10} = 1044$, $y^m_9 = 1077$, $y^m_8 = 1105$, $y^m_7 = 1107$, $y^m_6 = 1114$, $y^m_5 = 1116$, $y^m_4 = 1145$, $y^m_3 = 1206$, $y^m_2 = 1244$, $y^m_1 = 1310$. This solution corresponds to the sole manufacturing of bamboo frames and represents the most inexpensive manufacturing solution with regard to (7.8). Due to the large investment costs of the machinery no automatically executed procedure is used.
The supported nondominated point $y^2 = (834208, 585209)$ corresponds to the efficient solution $x^2 = (x^{2^a}_1, \ldots, x^{2^a}_{12}, x^{2^m}_1, \ldots, x^{2^m}_{12}, u^{2^a}_1, \ldots, u^{2^a}_{12}, u^{2^m}_1, \ldots, u^{2^m}_{12}, y^{2^m}_1, \ldots, y^{2^m}_{12}, v^{2^m}_1, \ldots, v^{2^m}_{12})$ with nonzero variables $x^{m100}_{11} = 11$, $x^{m100}_{10} = 11$, $x^{m100}_9 = 11$, $x^{m100}_8 = 11$, $x^{m100}_6 = 11$, $x^{m100}_5 = 11$, $x^{m100}_4 = 11$, $x^{m100}_3 = 11$, $x^{m100}_2 = 12$, $x^{m100}_1 = 12$, $x^{a100}_7 = 11$, $u^m_{12} = 1$, $u^m_{11} = 1$, $u^m_{10} = 1$, $u^m_9 = 1$, $u^m_8 = 1$, $u^m_6 = 1$, $u^m_5 = 1$, $u^m_4 = 1$, $u^m_3 = 1$, $u^m_2 = 1$, $u^m_1 = 1$, $u^a_7 = 1$, $x^m_{12} = 1042$, $x^m_{11} = 1044$, $x^m_{10} = 1051$, $x^m_9 = 1062$, $x^m_8 = 1065$, $x^m_6 = 1078$, $x^m_5 = 1080$, $x^m_4 = 1087$, $x^m_3 = 1098$, $x^m_2 = 1104$, $x^m_1 = 1116$, $x^a_7 = 1076$, $x^{m100}_{12} = 11$. This solution corresponds to the sole manufacturing of aluminium frames and represents the quickest manufacturing solution with regard to (7.9) via an automatically executed *welding* procedure.
The supported nondominated point $y^7 = (340456, 699110)$ corresponds to the efficient solution $x^7 = (x^{7^a}_1, \ldots, x^{7^a}_{12}, x^{7^m}_1, \ldots, x^{7^m}_{12}, u^{7^a}_1, \ldots, u^{7^a}_{12}, u^{7^m}_1, \ldots, u^{7^m}_{12}, y^{7^m}_1, \ldots, y^{7^m}_{12}, v^{7^m}_1, \ldots, v^{7^m}_{12})$ with nonzero solution values $x^{m100}_{11} = 11$, $x^{m100}_{10} = 11$, $x^{m100}_9 = 11$, $x^{m100}_8 = 11$, $x^{m100}_7 = 11$, $x^{m100}_5 = 12$, $x^{m100}_4 = 12$, $x^{m100}_3 = 12$, $x^{m100}_2 = 12$, $x^{m100}_1 = 12$, $x^{a100}_6 = 11$, $u^m_{12} = 1$, $u^m_{11} = 1$, $u^m_{10} = 1$, $u^m_9 = 1$, $u^m_8 = 1$, $u^m_7 = 2$, $u^m_5 = 1$, $u^m_4 = 1$, $u^m_3 = 1$, $u^m_2 = 1$, $u^m_1 = 1$, $u^a_6 = 1$, $x^m_{12} = 1042$, $x^m_{11} = 1044$, $x^m_{10} = 1051$, $x^m_9 = 1062$, $x^m_8 = 1064$, $x^m_7 = 1097$, $x^m_5 = 1101$, $x^m_4 = 1108$, $x^m_3 = 1120$, $x^m_2 = 1126$, $x^m_1 = 1200$, $x^a_6 = 1099$, $x^{m100}_{12} = 11$. This solution corresponds to the sole manufacturing of aluminium frames and uses an automatically executed *fixing* procedure.
All unsupported efficient solutions correspond to the manufacturing bamboo and aluminium frames, respectively, and use and an automatically executed *fixing* procedure, see Table 7.

Table 7: Unsupported nondominated points of Bike(1,1000).

point	obj (7.8)	obj (7.9)	var x^a_6	var x^m_1	var y^m_1
y^4	326923	2295030	208	219	1100
y^8	328564	2113540	385	400	948
y^0	330881	1938050	483	500	843
y^3	332973	1761060	579	600	700
y^5	335743	1590070	609	632	600
y^1	336991	1243590	801	830	400
y^9	339683	1072590	900	1000	300

7.2.4.2 *Computational results for Bike(2,1000)*

The bicriteria integer program Bike$(2, 1000)$ results in five supported nondominated points $y^9, y^{10}, y^4, y^1, y^3$ and seven unsupported nondominated points $y^6, y^{11}, y^0, y^5, y^8, y^2, y^{12}, y^7$, see Figure 7.6.

The supported nondominated point $y^9 = (282710, 2.793e + 06)$ corresponds to the efficient solution $x^9 = (x^{9^a}_1, \ldots, x^{9^a}_{12}, x^{9^m}_1, \ldots, x^{9^m}_{12},$

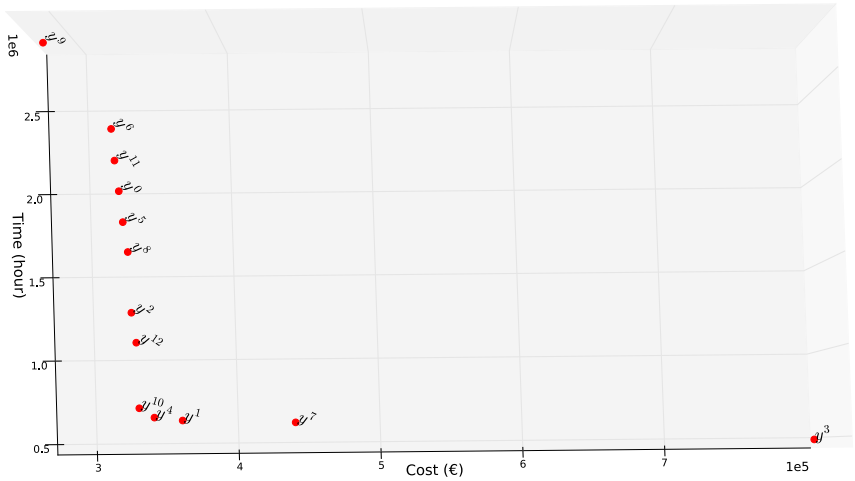

Figure 7.6: Computed nondominated points of Bike(2, 1000).

$u^{9^a}{}_1, \ldots, u^{9^a}{}_{12}, u^{9^m}{}_1, \ldots, u^{9^m}{}_{12}, y^{9^m}{}_1, \ldots, y^{9^m}{}_{12}, v^{9^m}{}_1, \ldots, v^{9^m}{}_{12}$) with nonzero values $y_{12}^{m100} = 11$, $y_{11}^{m100} = 11$, $y_{10}^{m100} = 11$, $y_9^{m100} = 11$, $y_8^{m100} = 12$, $y_7^{m100} = 12$, $y_6^{m100} = 12$, $y_5^{m100} = 12$, $y_4^{m100} = 12$, $y_3^{m100} = 13$, $y_2^{m100} = 13$, $y_1^{m100} = 14$, $v_{12}^m = 1$, $v_{11}^m = 1$, $v_{10}^m = 1$, $v_9^m = 1$, $v_8^m = 1$, $v_7^m = 17$, $v_6^m = 1$, $v_5^m = 1$, $v_4^m = 1, v_3^m = 1$, $v_2^m = 1$, $v_1^m = 1$, $y_{12}^m = 1021$, $y_{11}^m = 1023$, $y_{10}^m = 1044$, $y_9^m = 1077$, $y_8^m = 1105$, $y_7^m = 1107$, $y_6^m = 1114$, $y_5^m = 1116$, $y_4^m = 1145$, $y_3^m = 1206$, $y_2^m = 1244$, $y_1^m = 1310$. This solution corresponds to the sole manufacturing of bamboo frames and represents the most inexpensive manufacturing solution with regard to (7.8). Due to the relatively slow processing times of manually executed procedures (compared to automatically executed procedures) it is the slowest solution with regard to (7.9) among all efficient solutions.

The supported nondominated point $y^3 = (792158, 486319)$ corresponds to the efficient solution $x^3 = (x^{3^a}{}_1, \ldots, x^{3^a}{}_{12}, x^{3^m}{}_1, \ldots, x^{3^m}{}_{12}, u^{3^a}{}_1, \ldots, u^{3^a}{}_{12}, u^{3^m}{}_1, \ldots, u^{3^m}{}_{12}, y^{3^m}{}_1, \ldots, y^{3^m}{}_{12}, v^{3^m}{}_1, \ldots, v^{3^m}{}_{12})$ with nonzero variables $x_{11}^{m100} = 11$, $x_{10}^{m100} = 11$, $x_9^{m100} = 11$, $x_8^{m100} = 11$, $x_5^{m100} = 11$, $x_4^{m100} = 11$, $x_3^{m100} = 11$, $x_2^{m100} = 12$, $x_1^{m100} = 12$, $x_7^{a100} = 11$, $x_6^{a100} = 11$, $u_{12}^m = 1$, $u_{11}^m = 1$, $u_{10}^m = 1$, $u_9^m = 1$, $u_8^m = 1$, $u_5^m = 1$, $u_4^m = 1$, $u_3^m = 1$, $u_2^m = 1$, $u_1^m = 1$, $u_7^a = 1$, $u_6^a = 1$, $x_{12}^m = 1042$, $x_{11}^m = 1047$, $x_{10}^m = 1054$, $x_9^m = 1065$, $x_8^m = 1067$, $x_5^m = 1082$, $x_4^m = 1089$, $x_3^m = 1100$, $x_2^m = 1107$, $x_1^m = 1119$, $x_7^a = 1078$, $x_6^a = 1080$, $x_{12}^{m100} = 11$. This solution corresponds to the sole manufacturing of aluminium frames and represents the quickest manufacturing solution with regard to (7.9) via two automatically executed procedures, *fixing* and *welding*. Due to the large investment costs for automats it is the most expensive solution with regard to (7.9) among all efficient solutions.

The supported nondominated point $y^{10} = (340456, 699110)$ corresponds to the efficient solution $x^{10} = (x^{10^a}{}_1, \ldots, x^{10^a}{}_{12}, x^{10^m}{}_1, \ldots, x^{10^m}{}_{12}, u^{10^a}{}_1, \ldots, u^{10^a}{}_{12}, u^{10^m}{}_1, \ldots, u^{10^m}{}_{12}, y^{10^m}{}_1, \ldots, y^{10^m}{}_{12}, v^{10^m}{}_1, \ldots, v^{10^m}{}_{12})$ with nonzero solution values $x_{11}^{m100} = 11$, $x_{10}^{m100} = 11$, $x_9^{m100} = 11$, $x_8^{m100} = 11$, $x_7^{m100} = 11$, $x_5^{m100} = 12$, $x_4^{m100} = 12$, $x_3^{m100} = 12$, $x_2^{m100} = 12$, $x_1^{m100} = 12$, $x_6^{a100} = 11$, $u_{12}^m = 1$, $u_{11}^m = 1$, $u_{10}^m = 1$, $u_9^m = 1$, $u_8^m = 1$, $u_7^m = 2$, $u_5^m = 1$, $u_4^m = 1$, $u_3^m = 1$, $u_2^m = 1$, $u_1^m = 1$, $u_6^a =$

1, $x_{12}^m = 1042$, $x_{11}^m = 1045$, $x_{10}^m = 1052$, $x_9^m = 1063$, $x_8^m = 1065$, $x_7^m = 1098$, $x_5^m = 1162$, $x_4^m = 1170$, $x_3^m = 1182$, $x_2^m = 1188$, $x_1^m = 1200$, $x_6^a = 1100$, $x_{12}^{m100} = 11$. This solution corresponds to the sole manufacturing of aluminium frames and uses an automatically executed *fixing* procedure.

The supported nondominated point $y^4 = (350435, 645165)$ corresponds to the efficient solution $x^4 = ({x^4}_1^a, \ldots, {x^4}_{12}^a, {x^4}_1^m, \ldots, {x^4}_{12}^m, {u^4}_1^a, \ldots, {u^4}_{12}^a, {u^4}_1^m, \ldots, {u^4}_{12}^m, {y^4}_1^m, \ldots, {y^4}_{12}^m, {v^4}_1^m, \ldots, {v^4}_{12}^m)$ with nonzero solution values $x_{11}^{m100} = 11$, $x_9^{m100} = 11$, $x_8^{m100} = 11$, $x_7^{m100} = 11$, $x_5^{m100} = 11$, $x_4^{m100} = 12$, $x_3^{m100} = 12$, $x_2^{m100} = 12$, $x_1^{m100} = 12$, $x_{10}^{a100} = 11$, $x_6^{a100} = 11$, $u_{12}^m = 1$, $u_{11}^m = 1$, $u_9^m = 1$, $u_8^m = 1$, $u_7^m = 2$, $u_5^m = 1$, $u_4^m = 1$, $u_3^m = 1$, $u_2^m = 1$, $u_1^m = 1$, $u_{10}^a = 1$, $u_6^a = 1$, $x_{12}^m = 1042$, $x_{11}^m = 1048$, $x_9^m = 1061$, $x_8^m = 1063$, $x_7^m = 1096$, $x_5^m = 1100$, $x_4^m = 1109$, $x_3^m = 1121$, $x_2^m = 1127$, $x_1^m = 1139$, $x_{10}^a = 1050$, $x_6^a = 1098$, $x_{12}^{m100} = 11$. It uses two automatically executed procedures, *fixing* and *deburring*.

The supported nondominated point $y^1 = (369283, 627182)$ corresponds to the efficient solution $x^1 = ({x^1}_1^a, \ldots, {x^1}_{12}^a, {x^1}_1^m, \ldots, {x^1}_{12}^m, {u^1}_1^a, \ldots, {u^1}_{12}^a, {u^1}_1^m, \ldots, {u^1}_{12}^m, {y^1}_1^m, \ldots, {y^1}_{12}^m, {v^1}_1^m, \ldots, {v^1}_{12}^m)$ with nonzero values $x_{11}^{m100} = 11$, $x_{10}^{m100} = 11$, $x_9^{m100} = 11$, $x_8^{m100} = 11$, $x_7^{m100} = 11$, $x_5^{m100} = 12$, $x_4^{m100} = 12$, $x_3^{m100} = 12$, $x_2^{m100} = 12$, $x_6^{a100} = 11$, $x_1^{a100} = 12$, $u_{12}^m = 1$, $u_{11}^m = 1$, $u_{10}^m = 1$, $u_9^m = 1$, $u_8^m = 1$, $u_7^m = 2$, $u_5^m = 1$, $u_4^m = 1$, $u_3^m = 1$, $u_2^m = 1$, $u_6^a = 1$, $u_1^a = 1$, $x_{12}^m = 1042$, $x_{11}^m = 1044$, $x_{10}^m = 1051$, $x_9^m = 1062$, $x_8^m = 1064$, $x_7^m = 1097$, $x_5^m = 1101$, $x_4^m = 1108$, $x_3^m = 1120$, $x_2^m = 1126$, $x_6^a = 1099$, $x_1^a = 1128$, $x_{12}^{m100} = 11$. It uses two automatically executed procedures, *fixing* and *cutting*.

The unsupported efficient solution corresponding to y^7 is the only unsupported efficient solution using two automatically executed procedures, *fixing* and *coating*. All other unsupported efficient solutions use only an automatically executed *fixing* procedure, see Table 8.

Table 8: Unsupported nondominated points of Bike(2,1000).

point	obj (7.8)	obj (7.9)	var x_6^a	var x_{12}^a	var x_1^m	var y_1^m	$\sum_{i=1}^{12} u_i^a$
y^6	326923	2295030	209	0	220	1100	1
y^{11}	328564	2113540	311	0	324	948	1
y^0	330881	1938050	479	0	496	900	1
y^5	332973	1761060	517	0	537	700	1
y^8	335743	1590070	615	0	638	600	1
y^2	336991	1243590	801	0	831	400	1
y^{12}	339683	1072590	900	0	947	300	1
y^7	444998	610676	1058	1002	1095	0	2

8 THE SCENARIO ASSIGNMENT PROBLEM

> It's difficult to make predictions, especially about the future.
>
> *attributed to diverse individuals*

8.1 INTRODUCTION

As written in the introduction of Chapter 7, sustainability for humankind can be seen as the ability to meet its needs without disrupting nature and society. Taking into account that sustainability consists of environmental, economic and social aspects which generally conflict each other, we can consider sustainability problems as multicriteria decision problems. Scenario analysis is a method for dealing with uncertainties which aims at finding *coherent*, i.e. logical and consistent, scenarios of reasonable future outcomes. Scenario analysis was first applied in military planning in the 1950s [19] and the term *scenario* (in the context of predictions) was coined by Kahn and Wiener [46]. In sustainable manufacturing the application of scenario analysis aims at finding sustainable scenarios based on technological developments [35]. For example, it deals with the question which future scenarios of urban mobility are coherent (or desirable) for a large city where environmental and political constraints might forbid an ever growing number of cars and curb car-related mobility. Note that the pros and cons of this predictive method or the significance of forecasting are not part of this chapter. (For a well-received book on forecasting see [78].)

In Section 8.2 we formulate the mathematical problem that is posed by scenario analysis as a zero-one problem, show its *hardness* and describe related design problems. In Section 8.3 we consider three conflicting criteria based on a sustainable manufacturing perspective and compute nondominated scenarios for a power generation analysis and for a bus related mobility analysis. The considered *technology pools*, i.e. the input sets and corresponding objective coefficients, were provided by Pia Gausemeier as part of her research on sustainable technology pathways [35] within the collaborative research centre 1026.

8.2 PROBLEM FORMULATION AND CLASSIFICATION

The single criteria version of the *Scenario Assignment Problem* occurring in scenario analysis can be described as follows: Given $m \geq 2$ disjoint sets $T_1, \ldots, T_m$ and an objective function $c : T_1 \times \ldots \times T_m \to \mathbb{N}$; find a tuple $(t_1, \ldots, t_m) \in T_1 \times \ldots \times T_m$ maximising c.

8.2.1 *Single criteria zero-one formulation*

Let $T_1, \dots, T_m$ for $m \geq 2$ be given nonempty disjoint sets and let $n_i = |T_i|$ be the cardinality of T_i for $i \in [m]$. Define binary variables x_{ij} for $i = 1, \dots, m$ and $j = 1, \dots, n_i$ with

$$x_{ij} = \begin{cases} 1 & \text{if the } j\text{-th element of } T_i \text{ is chosen,} \\ 0 & \text{otherwise.} \end{cases}$$

The single criteria Scenario Assignment Problem can be formulated as follows:

$$\max \sum_{i=1}^{m-1} \sum_{k=i+1}^{m} \sum_{j=1}^{n_i} \sum_{\ell=1}^{n_k} c_{ijk\ell} x_{ij} x_{k\ell} \tag{8.1a}$$

$$\text{s.t.} \sum_{j=1}^{n_i} x_{ij} = 1 \quad \text{for all } i \in [m], \tag{8.1b}$$

$$x_{ij} \in \{0,1\} \quad \text{for all } i \in [m], \text{ for all } j \in [n_i], \tag{8.1c}$$

where $c_{ijk\ell}$ $(1 \leq i < k \leq m,\ j \in [n_i],\ \ell \in [n_k])$ is a *c*onsistency coefficient describing how coherent, e.g. from a technological perspective, it is to assign the j-th element of T_i to the ℓ-th element of T_k.

Let $\bar{n} = \sum_{i=1}^{m} n_i$ and let $C \in \mathbb{N}^{\bar{n} \times \bar{n}}$ be the (strictly) upper triangular matrix given by

$$C = \begin{pmatrix} 0_{11} & C_{12} & C_{13} & \cdots & C_{1m} \\ 0_{21} & 0_{22} & C_{23} & \cdots & C_{2m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0_{m-1,1} & 0_{m-1,2} & 0_{m-1,3} & \cdots & C_{m-1,m} \\ 0_{m1} & 0_{m2} & 0_{m3} & \cdots & 0_{mm} \end{pmatrix}$$

where $0_{ik} \in \mathbb{R}^{n_i \times n_k}$ consists only of zeros for $1 \leq k \leq i \leq m$ and the submatrices $C_{ik} \in \mathbb{N}_+^{n_i \times n_k}$ for $1 \leq i < k \leq m$ are given by

$$C_{ik} = \begin{pmatrix} c_{i1k1} & \cdots & c_{i1kn_k} \\ \vdots & \ddots & \vdots \\ c_{in_ik1} & \cdots & c_{in_ikn_k} \end{pmatrix}.$$

Then the above single criteria Scenario Assignment Problem can be formulated alternatively as

$$\max\{x^T C x : x \text{ satisfies (8.1b) and (8.1c)}\}. \tag{8.2}$$

Example 8.1. Let $T_1 = \{\text{pump storage, compressed air reservoir}\}$, $T_2 = \{\text{generator, fuel cell, photovoltaic cell}\}$, $T_3 = \{\text{water turbine, steam turbine}\}$ be given scenarios sets. Let $x = (x_{11}, x_{12}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}) \in \{0,1\}^{2+3+2}$ be a binary vector and let

$$C = \begin{pmatrix} 0 & 0 & 3 & 5 & 1 & 2 & 4 \\ 0 & 0 & 2 & 4 & 2 & 5 & 3 \\ 0 & 0 & 0 & 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 0 & 0 & 3 & 2 \\ 0 & 0 & 0 & 0 & 0 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

be a given consistency value matrix. Observe that based on the entries in the first row of C assigning a pump storage to a fuel cell has a consistency value of 5 whereas assigning a pump storage to a photovoltaic cell has a consistency value of 1, i.e. the latter assignment is considered much less technologically coherent/consistent than the first. Regarding (8.2), we get the following instance

$$\begin{aligned}\max\ (&3x_{11}x_{21} + 5x_{11}x_{22} + x_{11}x_{23} + 2x_{11}x_{31} + 4x_{11}x_{32} + 2x_{12}x_{21} + \\ &4x_{12}x_{22} + 2x_{12}x_{23} + 5x_{12}x_{31} + 3x_{12}x_{32} + 2x_{21}x_{31} + 3x_{21}x_{32} + \\ &3x_{22}x_{31} + 2x_{22}x_{32} + 4x_{23}x_{31} + x_{23}x_{32})\end{aligned}$$

subject to

$$\begin{aligned} x_{11} + x_{12} &= 1 \\ x_{21} + x_{22} + x_{23} &= 1 \\ x_{31} + x_{32} &= 1 \\ x_{11}, x_{12}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32} &\in \{0,1\}. \end{aligned}$$

Setting $x_{12} = 1,\ x_{22} = 1,\ x_{31} = 1$ which corresponds to the selection of a compressed air reservoir, a generator, and water turbine yields a unique optimal scenario assignment with an overall consistency value of 12.

8.2.2 *NP-Hardness*

Next we show that the Scenario Assignment Problem is NP-hard. One way to show this is by reduction from the *Maximum Cut Problem*. A nonempty node set $U \subset V$ of a graph $G = (V, E)$ induces a cut $\delta(U) \subseteq E$ consisting of all edges $e \in E$ having one endpoint in U and the other endpoint in $V \setminus U$. Given an undirected graph $G = (V, E)$ and positive edge weights $w : E \to \mathbb{R}_+$, the Maximum Cut Problem asks for a cut of maximal total weight. Even for the special case with unit edge weights $w_e = 1$ for all edges $e \in E$ the Maximum Cut Problem is known to be NP-hard [49, Theorem 16.6].

In order to reduce the Maximum Cut Problem (with unit weights) we encode whether a node $v \in V$ is in the node set $U \subset V$ or its complement by introducing a set $T_v = \{\bar{v}, \hat{v}\}$ for each node $v \in V$ in the Scenario Assignment Problem instance.

Proposition 8.2. The Scenario Assignment Problem is NP-hard.

Proof. Given an instance of the Maximum Cut Problem $G = (V, E)$ with $V = \{1, \ldots, m\}$ and unit edge weights $w : E \to 1$, construct a corresponding instance of the Scenario Assignment Problem with m disjoint sets $T_1, \ldots, T_m$ as follows: for each $u \in V$ introduce the set T_u containing 2 elements $\hat{u}$ and $\bar{u}$. The binary consistency coefficients are given as follows: If $(u, v) \in E$, then $c_{\hat{u}\bar{v}} = 1$ and $c_{\bar{u}\hat{v}} = 1$. Otherwise $c_{\hat{u}\bar{v}} = c_{\bar{u}\hat{v}} = 0$. Furthermore, $c_{\hat{u}\hat{v}} = 0$ and $c_{\bar{u}\bar{v}} = 0$ for all $u, v \in V$ with $u \neq v$. Now consider a feasible solution $(t_1, \ldots, t_m) \in T_1 \times \ldots \times T_m$ of the constructed Scenario Assignment Problem instance with positive consistency value $w \in \mathbb{N}_+$. Let the nodes in the cut $U \subset V$ be

given by those t_u which correspond to the second element $\bar{u} \in T_u$, i.e. $U = \bigcup_{u=1}^{m} \{u : t_u = \bar{u}\}$. Regarding the overall consistency value $w \in \mathbb{N}_+$ of $(t_1, \ldots, t_m)$ only nonzero consistency coefficients $c_{t_u t_v} \in \{0, 1\}$ contribute to the value w which, by construction, implies that either $t_u = \bar{u}$ and $t_v = \hat{v}$ or $t_u = \hat{u}$ and $t_v = \bar{v}$. In either case, the corresponding edge $(u, v) \in E$ is in the cut $\delta(U)$, i.e. has one endpoint in U and the other endpoint in $V \setminus U$. Hence, the value of an optimal scenario assignment coincides with the value of a maximum cut. □

Remark 8.3. The number of possible scenario assignments equals $\prod_{i=1}^{m} |T_i|$. If the number of given sets $T_1, \ldots, T_m$ is considered to be fixed, then the optimal solution can be found in polynomial time in the input by simple enumeration.

See Figure 8.1 for a simple Maximum Cut Problem instance with unit edge weights encoded as an instance of the Scenario Assignment Problem with binary consistency coefficients. The optimal scenario assignment $(\bar{u}, \hat{v}, \hat{w}) \in T_u \times T_v \times T_w$ corresponds to a maximum cut with node set $U = \{u\}$.

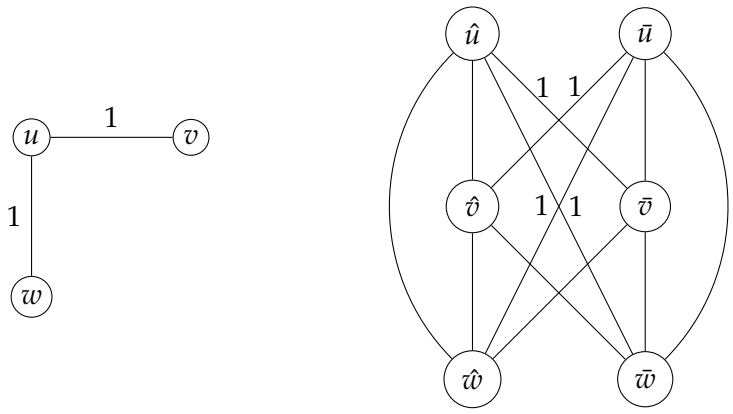

Figure 8.1: Instance of the Maximum Cut Problem (with unit edge weights) reduced to an instance of the Scenario Assignment Problem.

8.2.3 *Related Problems*

Grötschel [37] as well as Padberg and Rijal [60, Chapter 2] consider several quadratic zero-one problems with one set of assignment type constraints (8.1b) having applications in manufacturing, scheduling and clustering. Let $C \in \mathbb{R}^{mn \times mn}$ be the upper triangular matrix given by

$$C = \begin{pmatrix} \mathbf{0} & C_{12} & C_{13} & \cdots & C_{1m} \\ \mathbf{0} & \mathbf{0} & C_{23} & \cdots & C_{2m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & C_{m-1,m} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{pmatrix}$$

where $\mathbf{0} \in \mathbb{R}^{n\times n}$ consists only of zeros and $C_{ik} \in \mathbb{R}^{n\times n}$ for $1 \leq i < k \leq m$. Different design problems are characterised by differences in the structure of the submatrices C_{ik}.

8.2.3.1 *Circuit Layout Design Problem*

The *Circuit Layout Design Problem* (CLDP) occurs in the design of electronic circuits where a set of logic cells $M = \{1, \ldots, m\}$ needs to be assigned to a set of base cells $N = \{1, \ldots, n\}$ with $m \geq n$ involving assignment costs. Defining

$$x_{ij} = \begin{cases} 1 & \text{if logic cell } i \in M \text{ is assigned to base cell } j \in N, \\ 0 & \text{otherwise,} \end{cases}$$

the CLDP is the zero-one optimisation problem

$$\min\{x^T C x : x \text{ satisfies (8.1b) and (8.1c)}\}$$

where $n_i = n$ for all $i \in [m]$ in (8.1b) and in (8.1c) and where the submatrices $C_{ik} \in \mathbb{R}^{n\times n}$ for $1 \leq i < k \leq m$ are given by

$$C_{ik} = \begin{pmatrix} 0 & c_{i1k2} & \cdots & c_{i1kn} \\ c_{i2k1} & 0 & \cdots & c_{i1kn} \\ \vdots & \vdots & \ddots & \vdots \\ c_{ink1} & c_{ink2} & \cdots & 0 \end{pmatrix}.$$

8.2.3.2 *Boolean Quadratic Problem*

Given a set $V = \{1, \ldots, v\}$ and a vector $x \in \mathbb{R}^v$, the *constrained Boolean Quadratic Problem* (cBQP) is the zero-one optimisation problem

$$\begin{aligned} &\min\ q^T x + x^T C x \\ \text{s.t. } &\sum_{i\in S_j} x_i = b_j \ \text{ for } j \in [k], \\ &x_i \in \{0,1\} \ \text{ for } 1 \leq i \leq v, \end{aligned} \tag{8.3}$$

where $q \in \mathbb{R}^v$, $C \in \mathbb{R}^{v\times v}$ is an upper triangular matrix with zero diagonal, $S_j \subseteq V$ for $j \in [k]$ are nonempty subsets of V and $\bigcup_{j=1}^{k} S_j = V$ for some $k \geq 0$. Note for $k = 0$ we get the so-called *unconstrained* BQP [60, Chapter 2.9].

The cBQP subsumes (among several others) the Circuit Layout Design Problem and the Scenario Assignment Problem. Padberg and Rijal classify many quadratic zero-one problems based on the number of classes of assignment type constraints and structure of the submatrices [60, Figure 2.10].

8.3 COMPUTATIONAL RESULTS FOR THE TRICRITERIA CASE

The first computational study with respect to the Scenario Assignment Problem in sustainable manufacturing was done by Fügenschuh et al. [33]. Therein a formulation via a single criteria linear program

with integrality constraints on the variables is considered and solved with the focus on generating all feasible scenarios. In this section we consider a tricriteria Scenario Assignment Problem and are interested in the set of nondominated points. Let $T_1, \dots, T_m$ for $m \geq 2$ be nonempty disjoint sets and let $n_i = |T_i|$ be the cardinality for $i \in [m]$. Let $\mathbf{con}_{ijkl}$ $(1 \leq i < k \leq m,\ j \in [n_i],\ \ell \in [n_k])$ be given consistency coefficients describing how coherent/consistent it is to assign the j-th element of T_i to the ℓ-th element of T_k. Let $\mathbf{eco}_{ij}$ $(i \in [m],\ j \in [n_i])$ be given objective values describing how *economic* it is to select the j-th element in T_i. Similarly, let $\mathbf{env}_{ij}$ $(i \in [m], j \in [n_i])$ be given objective values describing how *environmentally friendly* it is to select the j-th element in T_i.

For what follows, we consider the following linearised tricriteria integer programming formulation

$$
\begin{aligned}
\max \Big(& \sum_{i=1}^{m-1} \sum_{k=i+1}^{m} \sum_{j=1}^{n_i} \sum_{\ell=1}^{n_k} \mathbf{con}_{ijk\ell} y_{ij}^{k\ell}, \quad \sum_{i=1}^{m} \sum_{j=1}^{m_i} \mathbf{eco}_{ij} x_{ij}, \quad \sum_{i=1}^{m} \sum_{j=1}^{m_i} \mathbf{env}_{ij} x_{ij} \Big) \\
\text{s.t.} \quad & \sum_{j=1}^{n_i} x_{ij} = 1 && \text{for } i \in [m], \\
& x_{ij} + x_{k\ell} - y_{ij}^{k\ell} \leq 1 && \text{for } 1 \leq i < k \leq m,\ j \in [n_i],\ \ell \in [n_k], \\
& -x_{ij} - x_{k\ell} + 2y_{ij}^{k\ell} \leq 0 && \text{for } 1 \leq i < k \leq m,\ j \in [n_i],\ \ell \in [n_k], \\
& x_{ij} \in \{0,1\} && \text{for } i \in [m],\ j \in [n_i], \\
& y_{ij}^{k\ell} \in \{0,1\} && \text{for } 1 \leq i < k \leq m,\ j \in [n_i],\ \ell \in [n_k].
\end{aligned}
$$

Remark 8.4. We have linearised the quadratic terms $x_{ij}x_{k\ell}$ in (8.1a) by introducing binary variables $y_{ij}^{k\ell} \in \{0,1\}$ and adding the constraints $x_{ij} + x_{k\ell} - y_{ij}^{k\ell} \leq 1$ and $-x_{ij} - x_{k\ell} + 2y_{ij}^{k\ell} \leq 0$ for $1 \leq i < k \leq m,\ j \in [n_i],\ \ell \in [n_k]$. The added constraints enforce that $y_{ij}^{k\ell} = 1$ if and only if $x_{ij} = 1$ and $x_{k\ell} = 1$. Note that there are, in general, different linearisations possible. Different linearisations might result in a different number of fractional points in the underlying polytope. Padberg considers *locally ideal LP formulations* for several quadratic zero-one problems [60, Chapter 4,5]. However, in this section we do not focus on the underlying polytope. Since the available input data was not large enough to get our solver into computational trouble we were satisfied with the above formulation.

8.3.1 *Power generation scenario analysis*

The following Scenario Assignment Problem instance considers self-sufficient power generation in the developing world. The sets $T_i = \{t_{i1}, \dots, t_{i|T_i|}\}$ for $i \in [4]$ consist of the following elements:

> $T_1 = \{$monokristalline Silizium-Solarzellen, polykristalline Silizium-Solarzellen, Dünnschicht-Solarzellen aus amorphem Silizium, III-V-Halbleiter-Solarzellen, II-VI-Halbleiter-Solarzellen, I-III-VI-Halbleiter-Solarzellen, Farbstoff-Solarzellen, organische Solarzellen$\}$,

$T_2 = \{$Serienregler, Parallelregler, Maximum-Power-Tracker$\}$,

$T_3 = \{$rotierende Umformer, Rechteckwechselrichter, Trapezwechselrichter, Sinuswechselrichter, netzgeführte Wechselrichter, pulsweitenmodulierte Wechselrichter ohne Trafo, pulsweitenmodulierte Wechselrichter mit HF-Zwischenkreis, Zentralrichterwechselstation mit Direkteinspeisung$\}$,

$T_4 = \{$Bleibatterie, Nickel-Kadmium-Batterie, Nickel-Zink-Batterie, Nickel-Metall-Hybrid-Batterie, Lithium-Ionen-Batterie, Wasserstoffspeicher, Redox-Flow-Zelle, Pumpspeicherkraftwerk, Druckluftspeicher, Kondensator, Supraleitender magnetischer Energiespeicher$\}$.

All considered consistency objective coefficients, economic objective coefficients and environmental objective coefficients were established within the collaborative research centre *Sustainable Manufacturing* [73]. The nondominated points were computed with PolySCIP, see Chapter 6.

We compute 9 nondominated points $y^0, \ldots, y^8$ for the considered instance, see Table 9 and Figure 8.2.

point	**con**	**eco**	**env**	T_1	T_2	T_3	T_4
y^5	27	291	366	t_{12}	t_{23}	t_{38}	t_{49}
y^4	14	308	353	t_{12}	t_{22}	t_{37}	t_{47}
y^6	23	301	353	t_{12}	t_{23}	t_{37}	t_{47}
y^7	25	297	366	t_{12}	t_{23}	t_{37}	t_{49}
y^1	26	293	360	t_{12}	t_{23}	t_{37}	t_{46}
y^3	17	304	350	t_{12}	t_{22}	t_{37}	t_{43}
y^0	16	304	366	t_{12}	t_{22}	t_{37}	t_{49}
y^8	19	300	360	t_{12}	t_{22}	t_{37}	t_{46}
y^2	18	298	366	t_{12}	t_{22}	t_{38}	t_{49}

Table 9: Nondominated points of the Scenario Assignment Problem instance regarding self-sufficient power generation.

The set of supported nondominated points $\mathcal{N}_s$ is given by $\mathcal{N}_s = \{y^0, y^4, y^5, y^6, y^7\}$ and the set of unsupported nondominated points $\mathcal{N}_u$ is given by $\mathcal{N}_u = \{y^1, y^2, y^3, y^8\}$. The set of poly-nondominated points $\mathcal{N}_p$ is given by $\mathcal{N}_p = \{y^0, y^1, y^2, y^3, y^4, y^5, y^6, y^7\}$ and the set of mono-nondominated points $\mathcal{N}_m$ is the singleton $\mathcal{N}_m = \{y^8\}$. Interestingly, the mono-nondominated point y^8 is characterised by *median* objective values among the set of nondominated points, i.e. the value 19 for the consistency objective **con**, the value 300 for the economic objective **eco** and the value 366 for the environmental objective **env**. Observe that all efficient solutions contain *polykristalline Silizium-Solarzellen*, i.e. $t_{12} \in T_1$. Moreover, the only selected elements of the set T_2 in any efficient solution are either *Parallelregler* or *Maximum-Power-Tracker*, i.e. t_{22} or t_{23}. The only elements of the set T_3 in any efficient solution are either *Wechselrichter mit HF-Zwischenkreis* or *Zentralrichterwechselstation mit Direkteinspeisung*, i.e. t_{37} or t_{38}. Regarding

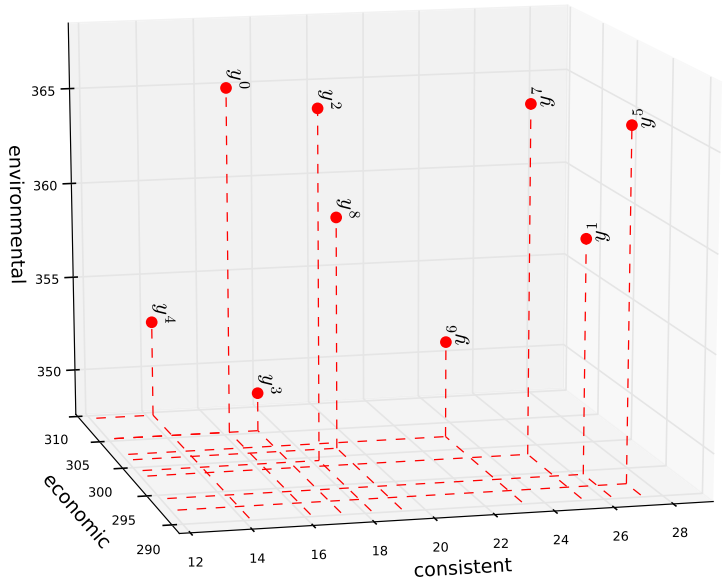

Figure 8.2: Nondominated points of the Scenario Assignment Problem instance for self-sufficient power generation.

the set T_4, four different elements, i.e. *Nickel-Zink-Batterie*, *Wasserstoffspeicher*, *Redox-Flow-Zelle*, *Druckluftspeicher*, are used in the set of nondominated points.

8.3.2 *Bus related mobility scenario analysis*

The following Scenario Assignment Problem instance considers bus related mobility in emerging markets. The sets $T_i = \{t_{i1}, \ldots, t_{i|T_i|}\}$ for $i \in [8]$ consist of the following elements:

$T_1 = \{$4-Takt-Diesel, 4-Takt-Otto, Elektromotor, Radnabenmotor, Druckluftantrieb, Vielstoffmotor, Diesel-Otto-Motor$\}$,

$T_2 = \{$Solarzelle, Brennstoffzelle, Gasturbine, Motor mit äußerer Verbrennung, Wankelmotor$\}$,

$T_3 = \{$fossile Kraftstoffe, Biokraftstoffe, Druckluft, Wasserstoff, flüssiger Wasserstoff, flüssiger Ammoniak, Lithium-Borhydrid Carbazol$\}$,

$T_4 = \{$Lithium-Eisen, Zink-Luft, Nickel-Metallhydrid, Lithium-Schwefel, Lithium-Luft$\}$,

$T_5 = \{$Rekuperation, Segeln, Turbosteamer, Schwungradspeicher, Turboaufladung, Nebenaggregate, Start-Stopp$\}$,

$T_6 = \{$Druckbetankung, Austausch der gesamten Speichereinheit, Aufnahme der Sonnenstrahlung, Induktion, Oberleitung$\}$,

$T_7 = \{$Stahl-Karosserie, Aluminium, Karosserie aus Kunststoff, Magnesium$\}$,

$T_8 = \{$intelligente Tanksysteme, Damping-Kontrolle, Bremsregelsysteme, Fahrerassistenzsysteme, autonomes Fahrzeug$\}$.

All considered consistency objective coefficients, economic objective coefficients and environmental objective coefficients were established within the collaborative research centre *Sustainable Manufacturing* [73]. The nondominated points were computed with PolySCIP, see Chapter 6.

We compute 53 nondominated points for the considered instance, see Figure 8.3 and Table 10.

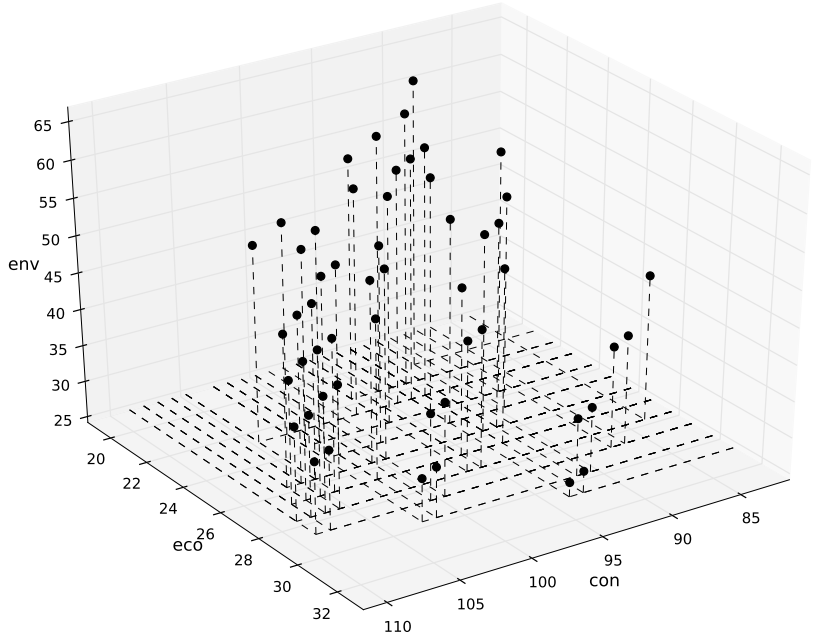

Figure 8.3: Bus related mobility Scenario Assignment Problem instance with 53 nondominated points.

The set of supported nondominated points $\mathcal{N}_s$ contains 23 points, see Figure 8.4. Moreover, the set of poly-nondominated points $\mathcal{N}_p$ contains 23 points and the set of mono-nondominated points $\mathcal{N}_m$ contains 30 points, see Figure 8.5. We get that 16 of the 23 poly-nondominated points are supported nondominated. The Chebyshev distance between the ideal point $y^I = (-109, -31, -64)$ and the nadir point $y^N = (-84, -21, -27)$ is $\|y^I - y^N\|_\infty = 37$. The maximum Chebyshev distance $\max_{u,v \in \mathcal{N}_m} \|u - v\|_\infty$ among the mono-nondominated points is 26. We get that the theoretically given coverage error given by the poly-nondominated points has the value 32 and the practically given coverage error has the value 7. Moreover, we have that the practically given coverage error of the set of supported nondominated points has the value 6. (See Chapter 5 for more details about these indicators.)

Table 10: Nondominated points of the Scenario Assignment Problem instance regarding bus related mobility.

con	**eco**	**env**	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8
109	29	35	t_{14}	t_{22}	t_{34}	t_{42}	t_{51}	t_{61}	t_{74}	t_{81}
93	31	28	t_{14}	t_{21}	t_{36}	t_{42}	t_{56}	t_{63}	t_{74}	t_{81}
91	21	64	t_{16}	t_{22}	t_{32}	t_{43}	t_{52}	t_{61}	t_{71}	t_{85}
103	30	31	t_{13}	t_{22}	t_{35}	t_{42}	t_{56}	t_{61}	t_{74}	t_{81}
94	31	27	t_{13}	t_{22}	t_{35}	t_{42}	t_{56}	t_{63}	t_{74}	t_{81}

103	23	54	t_{14}	t_{21}	t_{32}	t_{45}	t_{51}	t_{64}	t_{71}	t_{85}
106	26	48	t_{14}	t_{22}	t_{34}	t_{45}	t_{51}	t_{61}	t_{71}	t_{85}
107	26	46	t_{14}	t_{22}	t_{34}	t_{45}	t_{51}	t_{61}	t_{73}	t_{85}
108	27	42	t_{14}	t_{22}	t_{35}	t_{45}	t_{51}	t_{61}	t_{73}	t_{81}
109	28	38	t_{14}	t_{22}	t_{35}	t_{45}	t_{51}	t_{61}	t_{74}	t_{81}
105	23	52	t_{14}	t_{21}	t_{32}	t_{45}	t_{51}	t_{64}	t_{73}	t_{85}
97	22	58	t_{17}	t_{22}	t_{32}	t_{45}	t_{51}	t_{61}	t_{71}	t_{85}
98	23	56	t_{14}	t_{21}	t_{32}	t_{43}	t_{51}	t_{64}	t_{71}	t_{85}
93	22	62	t_{12}	t_{22}	t_{32}	t_{43}	t_{52}	t_{61}	t_{71}	t_{85}
95	22	60	t_{12}	t_{22}	t_{32}	t_{45}	t_{52}	t_{61}	t_{71}	t_{85}
91	30	34	t_{14}	t_{22}	t_{38}	t_{42}	t_{56}	t_{63}	t_{71}	t_{81}
87	29	40	t_{13}	t_{21}	t_{38}	t_{42}	t_{52}	t_{63}	t_{71}	t_{81}
84	28	45	t_{13}	t_{21}	t_{36}	t_{43}	t_{52}	t_{63}	t_{71}	t_{81}
93	27	49	t_{14}	t_{22}	t_{38}	t_{43}	t_{52}	t_{61}	t_{71}	t_{81}
90	25	54	t_{14}	t_{22}	t_{32}	t_{43}	t_{52}	t_{61}	t_{71}	t_{81}
92	26	53	t_{14}	t_{22}	t_{38}	t_{43}	t_{52}	t_{61}	t_{71}	t_{85}
89	24	58	t_{14}	t_{22}	t_{32}	t_{43}	t_{52}	t_{61}	t_{71}	t_{85}
93	23	59	t_{16}	t_{22}	t_{38}	t_{43}	t_{52}	t_{61}	t_{71}	t_{85}
94	23	58	t_{16}	t_{22}	t_{35}	t_{43}	t_{52}	t_{61}	t_{71}	t_{85}
95	23	57	t_{17}	t_{22}	t_{32}	t_{42}	t_{52}	t_{61}	t_{71}	t_{85}
94	24	57	t_{12}	t_{22}	t_{36}	t_{43}	t_{52}	t_{61}	t_{71}	t_{85}
97	24	56	t_{14}	t_{22}	t_{32}	t_{43}	t_{51}	t_{61}	t_{71}	t_{85}
102	24	54	t_{14}	t_{22}	t_{32}	t_{45}	t_{51}	t_{61}	t_{71}	t_{85}
94	25	53	t_{17}	t_{22}	t_{38}	t_{43}	t_{52}	t_{61}	t_{71}	t_{83}
92	30	33	t_{13}	t_{22}	t_{35}	t_{42}	t_{56}	t_{63}	t_{71}	t_{81}
102	30	32	t_{14}	t_{22}	t_{38}	t_{42}	t_{56}	t_{61}	t_{74}	t_{81}
93	26	52	t_{14}	t_{22}	t_{34}	t_{43}	t_{52}	t_{61}	t_{71}	t_{85}
103	25	50	t_{14}	t_{22}	t_{32}	t_{45}	t_{51}	t_{61}	t_{71}	t_{81}
100	26	51	t_{14}	t_{22}	t_{38}	t_{43}	t_{51}	t_{61}	t_{71}	t_{85}
103	24	52	t_{14}	t_{22}	t_{32}	t_{45}	t_{51}	t_{61}	t_{73}	t_{85}
101	26	50	t_{14}	t_{22}	t_{34}	t_{43}	t_{51}	t_{61}	t_{71}	t_{85}
99	25	52	t_{14}	t_{22}	t_{32}	t_{43}	t_{51}	t_{61}	t_{71}	t_{81}
102	25	51	t_{14}	t_{22}	t_{32}	t_{42}	t_{51}	t_{61}	t_{71}	t_{85}
105	26	49	t_{14}	t_{22}	t_{38}	t_{45}	t_{51}	t_{61}	t_{71}	t_{85}
102	27	47	t_{14}	t_{22}	t_{38}	t_{43}	t_{51}	t_{61}	t_{71}	t_{81}
96	27	48	t_{14}	t_{22}	t_{38}	t_{42}	t_{52}	t_{61}	t_{71}	t_{85}
105	27	46	t_{14}	t_{22}	t_{38}	t_{42}	t_{51}	t_{61}	t_{71}	t_{85}
107	27	44	t_{14}	t_{22}	t_{35}	t_{44}	t_{51}	t_{61}	t_{71}	t_{81}
106	27	45	t_{14}	t_{22}	t_{34}	t_{42}	t_{51}	t_{61}	t_{71}	t_{85}
97	28	43	t_{13}	t_{22}	t_{38}	t_{43}	t_{56}	t_{61}	t_{71}	t_{81}
96	28	44	t_{13}	t_{22}	t_{38}	t_{42}	t_{52}	t_{61}	t_{71}	t_{81}
107	28	41	t_{14}	t_{22}	t_{35}	t_{42}	t_{51}	t_{61}	t_{71}	t_{81}
106	28	42	t_{14}	t_{22}	t_{38}	t_{42}	t_{51}	t_{61}	t_{71}	t_{81}
108	28	39	t_{14}	t_{22}	t_{35}	t_{42}	t_{51}	t_{61}	t_{73}	t_{81}
88	29	39	t_{13}	t_{21}	t_{36}	t_{43}	t_{56}	t_{63}	t_{71}	t_{81}
108	29	36	t_{14}	t_{22}	t_{38}	t_{42}	t_{51}	t_{61}	t_{74}	t_{81}
100	29	38	t_{13}	t_{22}	t_{38}	t_{42}	t_{56}	t_{61}	t_{71}	t_{81}
101	29	37	t_{13}	t_{22}	t_{34}	t_{42}	t_{56}	t_{61}	t_{71}	t_{81}

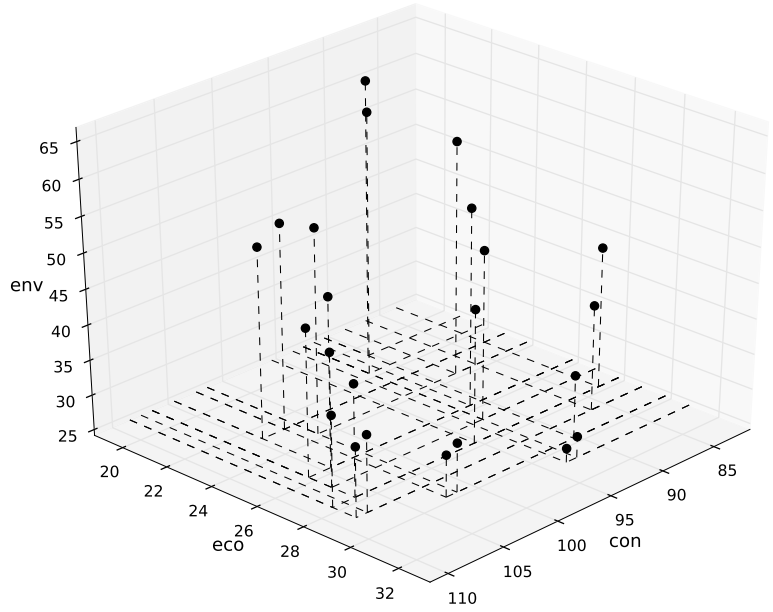

Figure 8.4: All 23 supported nondominated points of the bus related mobility Scenario Assignment Problem instance.

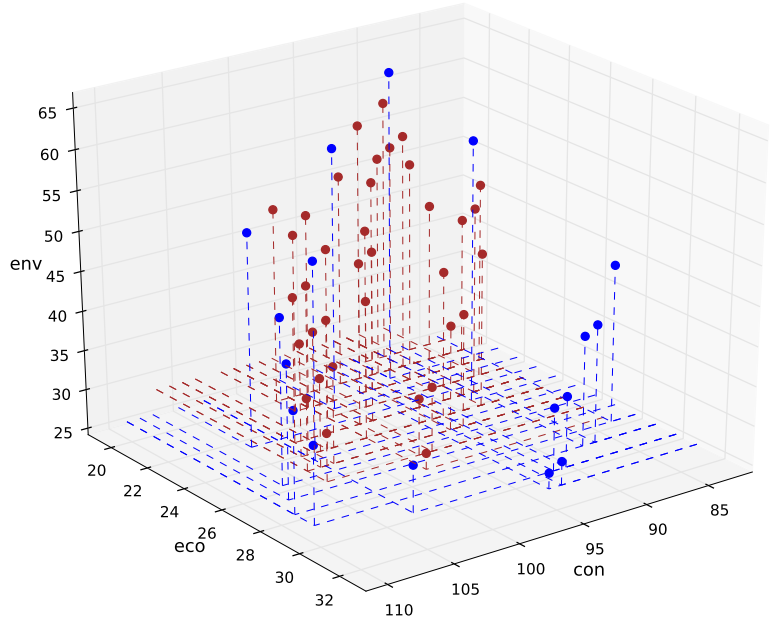

Figure 8.5: The set of poly-nondominated points (in blue) and the set of mono-nondominated points (in brown) of the bus related mobility Scenario Assignment Problem instance.

8.4 SUMMARY AND OUTLOOK

In this chapter the Scenario Assignment Problem was described and investigated. It was shown that the general version of the Scenario Assignment Problem belongs to the class of *hard* design problems which can be modelled via a zero-one formulation. The set of nondominated points was computed for two different tricriteria problem instances based on real-world data. This chapter showed exemplarily that mathematical formulations and multicriteria optimisation meth-

ods are powerful tools to solve quantitative problems in sustainable manufacturing.

For some design problems there are special cases for which polynomial algorithms are known, e.g. the Maximum Cut Problem is polynomially solvable for planar graphs [39]. Regarding the Scenario Assignment Problem, the objective coefficients used in scenario analysis in the framework of sustainable manufacturing are established based on data related to the considered scenario sets $T_1, \ldots, T_m$. Hence, it is not far-fetched to assume that the consistency coefficients could be computed by vectors v_{ij} containing characteristic *core* values of the j-th element of the set T_i. In other words, the submatrices C_{ik} for $1 \leq i < k \leq m$ might be given or approximated by

$$C_{ik} = \begin{pmatrix} f(v_{i1}, v_{k1}) & \cdots & f(v_{i1}, v_{kn_k}) \\ \vdots & \ddots & \vdots \\ f(v_{in_i}, v_{k1}) & \cdots & f(v_{in_i}, v_{kn_k}) \end{pmatrix}$$

where $f(x, y)$ is some function, e.g. the scalar product of x and y. In this regard, it might be possible to find special cases for which the problem structure can be further exploited. This might be an interesting direction for further research.

BIBLIOGRAPHY

[1] Tobias Achterberg. 'SCIP: Solving constraint integer programs'. In: *Mathematical Programming Computation* 1.1 (July 2009), pp. 1–41.

[2] Nina Amenta and Günter M. Ziegler. 'Deformed Products and Maximal Shadows of Polytopes'. In: *Advances In Discrete And Computational Geometry, Amer. Math. Soc., Providence, Contemporary Mathematics 223*. 1996, pp. 57–90.

[3] Przybylski Anthony, Gandibleux Xavier and Matthias Ehrgott. 'Computational Results for Four Exact Methods to Solve the Three-Objective Assignment Problem'. In: *Multiobjective Programming and Goal Programming*. Ed. by Vincent Barichard, Matthias Ehrgott, Xavier Gandibleux and Vincent T'Kindt. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 79–88. ISBN: 978-3-540-85646-7.

[4] David Avis. *lrs*. [Online; accessed 14-September-2013]. URL: `http://cgm.cs.mcgill.ca/~avis/C/lrs.html`.

[5] David Avis and Komei Fukuda. 'Reverse Search for Enumeration'. In: *Discrete Applied Mathematics* 65 (1993), pp. 21–46.

[6] David Avis and Charles Jordan. 'Comparative computational results for some vertex and facet enumeration codes'. In: *CoRR* abs/1510.02545 (2015). URL: `http://arxiv.org/abs/1510.02545`.

[7] Melih Azizoğlu, Benjamin A. Burton and Cameron A. G. MacRae. 'Multi-Objective Integer Programming: An Improved Recursive Algorithm'. In: *Journal of Optimization Theory and Applications* 160.2 (Feb. 2014), pp. 470–482. ISSN: 1573-2878.

[8] A. Barvinok. *Integer Points in Polyhedra*. Contemporary mathematics. European Mathematical Society, 2008. ISBN: 9783037190524.

[9] H. P. Benson and E. Sun. 'Outcome Space Partition of the Weight Set in Multiobjective Linear Programming'. In: *Journal of Optimization Theory and Applications* 105.1 (Apr. 2000), pp. 17–36. ISSN: 1573-2878.

[10] Harold P. Benson. 'An Outer Approximation Algorithm for Generating All Efficient Extreme Points in the Outcome Set of a Multiple Objective Linear Programming Problem'. In: *Journal of Global Optimization* 13.1 (Jan. 1998), pp. 1–24. ISSN: 1573-2916.

[11] Harold P. Benson and Erjiang Sun. 'A weight set decomposition algorithm for finding all efficient extreme points in the outcome set of a multiple objective linear program'. In: *European Journal of Operational Research* 139.1 (2002), pp. 26–41. ISSN: 0377-2217.

[12] D. Bertsimas and J.N. Tsitsiklis. *Introduction to linear optimization*. Athena Scientific, 1997. ISBN: 1-886529-19-1.

[13] Robert E. Bixby. 'Solving Real-World Linear Programs: A Decade and More of Progress'. In: *Operations Research* 50 (2002), pp. 3–15.

[14] Fritz Bökler. 'The Multiobjective Shortest Path Problem Is NP-Hard, or Is It?' In: *9th International Conference on Evolutionary Multi-Criterion Optimization - Volume 10173*. EMO 2017. Springer-Verlag New York, 2017, pp. 77–87. ISBN: 978-3-319-54156-3.

[15] Natashia Boland, Hadi Charkhgard and Martin Savelsbergh. 'A Criterion Space Search Algorithm for Biobjective Mixed Integer Programming: The Triangle Splitting Method'. In: *INFORMS Journal on Computing* 27.4 (2015), pp. 597–618.

[16] Natashia Boland, Hadi Charkhgard and Martin Savelsbergh. 'The L-shape search method for triobjective integer programming'. In: *Mathematical Programming Computation* 8.2 (2016), pp. 217–251. ISSN: 1867-2957.

[17] Ralf Borndörfer, Sebastian Schenker, Martin Skutella and Timo Strunk. 'PolySCIP'. In: *Mathematical Software – ICMS 2016, 5th International Congress, Proceedings*. Ed. by G.-M. Greuel, T. Koch, P. Paule and A. Sommese. Vol. 9725. LNCS. Berlin, Germany: Springer, 2016. ISBN: 978-3-319-42431-6.

[18] V. Joseph Bowman. 'On the Relationship of the Tchebycheff Norm and the Efficient Frontier of Multiple-Criteria Objectives'. In: *Multiple Criteria Decision Making: Proceedings of a Conference Jouy-en-Josas, France May 21–23, 1975*. Ed. by Hervé Thiriez and Stanley Zionts. Springer Berlin Heidelberg, 1976, pp. 76–86. ISBN: 978-3-642-87563-2.

[19] Seyom Brown. 'Systems Analysis and Policy Planning: Applications in Defence'. In: ed. by E.S. Quade and W.I. Boucher. New York: American Elsevier Publishing, 1968. Chap. Scenarios in Systems Analysis.

[20] Bugseng. *Parma Polyhedral Library*. [Online; accessed 12-April-2019]. URL: `http://bugseng.com/products/ppl`.

[21] László Csirmaz. *inner*. [Online; accessed 12-April-2019]. URL: `https://github.com/csirmaz/inner`.

[22] Kerstin Dächert. 'Adaptive Parametric Scalarizations in Multicriteria Optimization'. PhD thesis. Bergische Universität Wuppertal, 2014.

[23] Kerstin Dächert, Jochen Gorski and Kathrin Klamroth. 'An augmented weighted Tchebycheff method with adaptively chosen parameters for discrete bicriteria optimization problems'. In: *Computers & Operations Research* 39.12 (2012), pp. 2929–2943. ISSN: 0305-0548.

[24] Kerstin Dächert and Kathrin Klamroth. 'A linear bound on the number of scalarizations needed to solve discrete tricriteria optimization problems'. In: *Journal of Global Optimization* 61.4 (2015), pp. 643–676. ISSN: 1573-2916.

[25] J. Dauer and Y.H. Liu. 'Solving multiple objective linear programs in objective space'. In: *European Journal of Operational Research* 46 (1990), pp. 350–357.

[26] Kalyanmoy Deb and Deb Kalyanmoy. *Multi-Objective Optimization Using Evolutionary Algorithms*. New York, NY, USA: John Wiley & Sons, Inc., 2001. ISBN: 047187339X.

[27] Ilias Diakonikolas and Mihalis Yannakakis. 'Succinct Approximate Convex Pareto Curves'. In: *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '08. San Francisco, California, 2008, pp. 74–83.

[28] Yann Disser and Martin Skutella. 'The Simplex Algorithm is NP-mighty'. In: *Proceedings of the Twenty-sixth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '15. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2015, pp. 858–872.

[29] Matthias Ehrgott. *Multicriteria Optimization*. 2nd. Springer-Verlag Berlin Heidelberg, 2005. ISBN: 978-3-540-27659-3.

[30] Matthias Ehrgott, Andreas Löhne and Lizhen Shao. 'A dual variant of Benson's outer approximation algorithm for multiple objective linear programming'. In: *Journal of Global Optimization* 52.4 (Apr. 2012), pp. 757–778. ISSN: 1573-2916.

[31] Matthias Ehrgott and Dagmar Tenfelde-Podehl. 'Computation of ideal and Nadir values and implications for their use in MCDM methods'. In: *European Journal of Operational Research* 151.1 (2003), pp. 119–139. ISSN: 0377-2217.

[32] FICO. *Xpress*. [Online; accessed 12-April-2019]. URL: `www.fico.com/en/products/fico-xpress-optimization-suite`.

[33] Armin Fügenschuh, Pia Gausemeier, Günther Seliger and Semih Severengiz. 'Advanced Manufacturing and Sustainable Logistics: 8th International Heinz Nixdorf Symposium, IHNS 2010, Paderborn, Germany, April 21-22, 2010. Proceedings'. In: ed. by Wilhelm Dangelmaier, Alexander Blecken, Robin Delius and Stefan Klöpfer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. Chap. Scenario Technique with Integer Programming for Sustainability in Manufacturing, pp. 320–331. ISBN: 978-3-642-12494-5.

[34] Komei Fukuda and Alain Prodon. 'Double description method revisited'. In: *Combinatorics and Computer Science: 8^{th} Franco-Japanese and 4^{th} Franco-Chinese Conference Brest, France, July 3–5, 1995 Selected Papers*. Ed. by Michel Deza, Reinhardt Euler and Ioannis Manoussakis. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 91–111. ISBN: 978-3-540-70627-4.

[35] Pia Gausemeier, Günther Seliger, Sebastian Schenker and Ralf Borndörfer. 'Nachhaltige Technologiepfade für unterschiedliche Entwicklungsniveaus mithilfe mehrkriterieller Entscheidungsfindung. (German) [Sustainable technology pathways for different

development levels by means of multi-criteria decision support]'. In: *Vorausschau und Technologieplanung*. Vol. 334. 2014, pp. 65–94.

[36] Donald Goldfarb. 'On the Complexity of the Simplex Method'. In: *Advances in Optimization and Numerical Analysis*. Ed. by Susana Gomez and Jean-Pierre Hennart. Dordrecht: Springer Netherlands, 1994, pp. 25–38. ISBN: 978-94-015-8330-5.

[37] Martin Grötschel. 'Discrete Mathematics in Manufacturing'. In: *Proceedings of the Second International Conference on Industrial and Applied Mathematics*. ICIAM 91. Washington, D.C., USA: Society for Industrial and Applied Mathematics, 1992, pp. 119–145. ISBN: 0-89871-302-1.

[38] Inc. Gurobi Optimization. *GUROBI Optimization*. [Online; accessed 12-April-2019]. URL: `www.gurobi.com`.

[39] F. Hadlock. 'Finding a Maximum Cut of a Planar Graph in Polynomial Time.' In: *SIAM J. Comput.* 4.3 (1975), pp. 221–225.

[40] Y.Y. Haimes, L.S. Lasdon and D.A. Wismer. 'On a bicriterion formation of the problems of integrated system identification and system optimization'. In: *IEEE Transactions on Systems, Man and Cybernetics* SMC-1.3 (July 1971), pp. 296–297. ISSN: 0018-9472.

[41] H.W. Hamacher and G. Ruhe. 'On spanning tree problems with multiple objectives'. In: *Annals of Operations Research* 52.4 (1994), pp. 209–230. ISSN: 1572-9338.

[42] Pierre Hansen. 'Bicriterion Path Problems'. In: *Multiple Criteria Decision Making Theory and Application*. Ed. by T. Gal G. Fandel. Springer Berlin Heidelberg, 1980, pp. 109–127.

[43] IBM. *ILOG CPLEX*. [Online; accessed 12-April-2019]. URL: `https://www.ibm.com/uk-en/analytics/cplex-optimizer`.

[44] H. Isermann. 'Proper Efficiency and the Linear Vector Maximum Problem'. In: *Operations Research* 22.1 (1974), pp. 189–191.

[45] H. Isermann. 'The enumeration of the set of all efficient solutions for a linear multiple objective program'. In: *Operations Research Quaterly* 28.3 (1977), pp. 711–725.

[46] H. Kahn and A.J. Wiener. *The Year 2000: A Framework for Speculations on the next Thirty-Three Years*. New York: Macmillan, 1967.

[47] Gokhan Kirlik and Serpil Sayin. 'A new algorithm for generating all nondominated solutions of multiobjective discrete optimization problems'. In: *European Journal of Operational Research* 232.3 (2014), pp. 479–488. ISSN: 0377-2217.

[48] Thorsten Koch. 'Rapid Mathematical Programming'. PhD thesis. TU Berlin, 2004.

[49] Bernhard Korte and Jens Vygen. *Combinatorial Optimization: Theory and Algorithms*. 4th. Springer Publishing Company, Incorporated, 2007. ISBN: 9783540718437.

[50] Andreas Löhne and Benjamin Weißing. *BENSOLVE 2*. [Online; accessed 12-April-2019]. URL: https://www.optimierung-loehne.uni-jena.de/bensolve.html.

[51] *LP File Format*. [Online; accessed 12-April-2019]. URL: http://www.gurobi.com/documentation/8.1/refman/lp_format.html.

[52] K. Marsh et al. 'Multiple Criteria Decision Analysis for Health Care Decision Making - Emerging Good Practices: Report 2 of the ISPOR/MCDA Emerging Good Practices Task Force'. In: *Value in Health* 19.2 (2016), pp. 125–137. ISSN: 1098-3015.

[53] T.S. Motzkin, H. Raiffa, G.L. Thompson and R.M. Thrall. 'Contributions to the Theory of Games - Volume II'. In: ed. by H.W. Kuhn and A.W. Tucker. Princeton: Princeton University Press, 1953. Chap. The double description method, pp. 51–73.

[54] *MPS (format) — Wikipedia, The Free Encyclopedia*. [Online; accessed 08-October-2017]. URL: https://en.wikipedia.org/wiki/MPS_(format).

[55] Katta G. Murty. 'Computational complexity of parametric linear programming'. In: *Mathematical Programming* 19.1 (1980), pp. 213–219. ISSN: 1436-4646.

[56] Sabrina Neugebauer, Julia Martinez-Blanco, René Scheumann and Matthias Finkbeiner. 'Enhancing the practical implementation of life cycle sustainability assessment - Proposal of a Tiered approach'. In: *Journal of Cleaner Production* 102.Supplement C (2015), pp. 165–176. ISSN: 0959-6526.

[57] Normaliz. [Online; accessed 12-April-2019]. URL: www.normaliz.uni-osnabrueck.de.

[58] Melih Özlen and Meral Azizoğlu. 'Multi-objective integer programming: A general approach for generating all non-dominated solutions'. In: *European Journal of Operational Research* 199.1 (2009), pp. 25–35. ISSN: 0377-2217.

[59] Özgür Özpeynirci and Murat Köksalan. 'An Exact Algorithm for Finding Extreme Supported Nondominated Points of Multiobjective Mixed Integer Programs'. In: *Management Science* 56.12 (2010), pp. 2302–2315.

[60] Manfred W. Padberg and Minendra P. Rijal. *Location, Scheduling, Design and Integer Programming*. 1st. Kluwer Academic Publishers, 1996. ISBN: 978-0-7923-9715-1.

[61] Christos H. Papadimitriou and Mihalis Yannakakis. 'On the Approximability of Trade-offs and Optimal Access of Web Sources'. In: *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*. 2000, pp. 86–92.

[62] Vilfredo Pareto. *Manual of political economy (manuale di economia politica)*. Trans. by Ann S. Schwier and Alfred N. Page. Translated by Ann S. Schwier and Alfred N. Page. New York: Kelley, 1971 (1906), xii, 504 p.

[63] Anthony Przybylski and Xavier Gandibleux. 'Multi-objective branch and bound'. In: *European Journal of Operational Research* 260.3 (2017), pp. 856–872. ISSN: 0377-2217.

[64] Anthony Przybylski, Xavier Gandibleux and Matthias Ehrgott. 'A Recursive Algorithm for Finding All Nondominated Extreme Points in the Outcome Set of a Multiobjective Integer Problem'. In: *INFORMS J. on Computing* 22.3 (2010), pp. 371–386. ISSN: 1526-5528.

[65] Anthony Przybylski, Xavier Gandibleux and Matthias Ehrgott. 'Two phase algorithms for the bi-objective assignment problem'. In: *European Journal of Operational Research* 185.2 (2008), pp. 509–533. ISSN: 0377-2217.

[66] Ted K. Ralphs, Matthew J. Saltzman and Margaret M. Wiecek. 'An improved algorithm for solving biobjective integer programs'. In: *Annals of Operations Research* 147.1 (2006), pp. 43–70. ISSN: 1572-9338.

[67] S. Ruzika and M. M. Wiecek. 'Approximation Methods in Multiobjective Programming'. In: *Journal of Optimization Theory and Applications* 126.3 (Sept. 2005), pp. 473–501. ISSN: 1573-2878.

[68] Hershel M. Safer. 'Fast Approximation Schemes for Multicriteria Combinatorial Optimization'. AAI0572177. PhD thesis. Cambridge, MA, USA, 1992.

[69] Serpil Sayin. 'Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming'. In: *Mathematical Programming* 87.3 (May 2000), pp. 543–560. ISSN: 1436-4646.

[70] S. Schenker, J.G. Steingrimsson, R. Borndörfer and G. Seliger. 'Modelling of Bicycle Manufacturing via Multi-criteria Mixed Integer Programming'. In: *Procedia CIRP* 26 (2015). 12th Global Conference on Sustainable Manufacturing – Emerging Potentials, pp. 276–280. ISSN: 2212-8271.

[71] T. Schlechte and R. Borndörfer. 'Balancing Efficiency and Robustness - A Bi-Criteria Optimization Approach to Railway Track Allocation'. In: *Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems*. Ed. by M. Ehrgott, B. Naujoks, T. J. Stewart and J. Wallenius. Vol. 634. Springer Berlin Heidelberg, 2010.

[72] Günther Seliger. *Sustainability in Manufacturing*. Springer Berlin Heidelberg, 2007. ISBN: 978-3-540-49870-4.

[73] R. Stark, G. Seliger and J. Bonvoisin. *Sustainable Manufacturing: Challenges, Solutions and Implementation Perspectives*. Sustainable Production, Life Cycle Engineering and Management. Springer International Publishing, 2017. ISBN: 9783319485133.

[74] R. E. Steuer. 'ADBASE: A Program for Analyzing Multiple Objective Linear Programming Problems'. In: *Journal of Marketing Research* 12 (1975), pp. 453–455.

[75] Ralph E. Steuer and Eng-Ung Choo. 'An interactive weighted Tchebycheff procedure for multiple objective programming'. In: *Mathematical Programming* 26.3 (1983), pp. 326–344. ISSN: 1436-4646.

[76] D. Tenfelde-Podehl. *A recursive algorithm for multiobjective combinatorial optimization problems with Q criteria*. Tech. rep. Technische Universität Graz, 2003.

[77] *Test instances for the multicriteria assignment and the multicriteria knapsack problem*. [Online; accessed 10-December-2017]. URL: `http://home.ku.edu.tr/~moolibrary/`.

[78] P.E. Tetlock and D. Gardner. *Superforecasting: The Art and Science of Prediction*. Crown, 2015. ISBN: 978-0804136693.

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

[80] Sergei Vassilvitskii and Mihalis Yannakakis. 'Efficiently Computing Succinct Trade-off Curves'. In: *Theor. Comput. Sci.* 348.2 (Dec. 2005), pp. 334–356. ISSN: 0304-3975.

[81] A. G. S. Ventre, A. Maturo, S. Hoskova-Mayerova and J. Kacprzyk, eds. *Multicriteria and Multiagent Decision Making with Applications to Economics and Social Sciences*. Rand Corporation Research Study. Springer Berlin Heidelberg, 2013.

[82] David Gordon Wilson. *Bicycling Science*. MIT Press, 2004. ISBN: 9780262731546.

[83] Günter M. Ziegler. *Lectures on Polytopes*. Springer New York, 1995. ISBN: 978-0-387-94365-7.

[84] E. Zitzler et al. 'Performance Assessment of Multiobjective Optimizers: An Analysis and Review'. In: *Trans. Evol. Comp* 7.2 (Apr. 2003), pp. 117–132. ISSN: 1089-778X.

[85] Eckart Zitzler and Lothar Thiele. 'Multiobjective optimization using evolutionary algorithms — A comparative case study'. In: *Parallel Problem Solving from Nature — PPSN V: 5th International Conference Amsterdam, The Netherlands September 27–30, 1998 Proceedings*. Ed. by Agoston E. Eiben, Thomas Bäck, Marc Schoenauer and Hans-Paul Schwefel. Springer Berlin Heidelberg, 1998, pp. 292–301. ISBN: 978-3-540-49672-4.